


Z czego składa się strona internetowa

PAWEŁ DOMAŃSKI



Copyright © 2025 DexterLab Paweł Domański

Wydane przez DexterLab

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metoda kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji. Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli. Autor dołożył wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor nie ponosi również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

DexterLab Paweł Domański,
Brzóz, Polska
email: kontakt@dexterlab.pl
www: <https://dexterlab.dev>

Kochany czytelniku! Jeśli podobała się, tobie ta książka to zajrzyj pod adres
<https://dexterlab.dev/support-portal/> masz tam możliwość wpisać swoje uwagi, spostrzeżenia lub recenzję.

SPIS TREŚCI

Szczegółowe opracowanie elementów strony internetowej	9
Materiał szkoleniowy dla początkujących twórców internetowych	9
1. Kod źródłowy	9
HTML (HyperText Markup Language)	9
CSS (Cascading Style Sheets)	10
JavaScript	11
2. Główne elementy strukturalne	12
Nagłówek (Header)	12
Treść główna (Main content)	13
Stopka (Footer)	14
Menu nawigacyjne	16
Sidebar (panel boczny)	18
3. Elementy wizualne i multimedialne	21
Obrazy i grafiki	21
Filmy	23
Animacje	26
Czcionki	30
4. Elementy interaktywne	34
Formularze	34
Przyciski	40
Linki	46
Menu rozwijane	51


Galerie.....	59
5. Elementy techniczne (niewidoczne dla użytkownika)	74
Meta tagi.....	74
Skrypty	79
Pliki cookie	85
API (Application Programming Interface)	96
6. Responsywność.....	111
Media queries	111
Elastyczne siatki (grid)	121
Obrazy responsywne	132
7. Warstwa backendowa (zaplecze).....	140
Serwer	140
Baza danych	144
Języki backendowe.....	156
CMS (Content Management System).....	166
8. Funkcje i usługi	180
System logowania	180
Koszyk zakupowy	208
Wyszukiwarka	249
Komentarze	283
Regulamin	343
Informacje o plikach cookie	360
Dostępność (WCAG)	375
1.2. Media zależne od czasu	376

1.3. Adaptowalność	377
1.4. Rozróżnialność	377
2. Funkcjonalność	377
2.1. Dostępność z klawiatury	377
2.2. Wystarczająca ilość czasu	378
2.3. Ataki padaczki	378
2.4. Nawigowanie.....	379
2.5. Metody wprowadzania danych	379
3. Zrozumiałość	380
3.1. Czytelność	380
3.2. Przewidywalność	380
3.3. Pomoc przy wprowadzaniu informacji.....	381
4. Solidność	381
4.1. Kompatybilność.....	381
10. Optymalizacja	388
SEO (Search Engine Optimization).....	388
1.2. Struktura URL	389
1.3. Struktura nagłówków	389
1.4. XML Sitemap	390
1.5. Plik robots.txt	390
2. Optymalizacja treści	391
2.1. Badanie słów kluczowych	391
2.2. Tworzenie treści przyjaznej dla SEO	391
2.3. Przyjazne dla użytkownika i SEO formatowanie	391

2.4. Linkowanie wewnętrzne	391
2.5. Regularne aktualizacje	392
3. Optymalizacja techniczna	392
3.1. Szybkość ładowania strony	392
3.2. Responsywny design.....	392
3.3. Bezpieczeństwo (HTTPS)	393
3.4. Dane strukturalne (Schema.org)	393
3.5. Optymalizacja dla Core Web Vitals	394
4. Budowa linków i autorytetu.....	394
4.1. Naturalne pozyskiwanie linków	394
4.2. Miejsca do pozyskiwania linków	394
4.3. Budowanie autorytetu (E-A-T).....	395
4.4. Monitorowanie profilu linków	395
5. Monitorowanie i analityka	395
5.1. Konfiguracja podstawowych narzędzi	395
5.2. Kluczowe metryki do śledzenia.....	396
5.3. Regularne audyty SEO	396
5.4. Dostosowywanie strategii	396
6. SEO dla różnych typów stron	396
6.1. Sklepy internetowe	396
6.2. Blogi.....	397
6.3. Strony lokalne.....	397
6.4. Strony firmowe	397
1. Optymalizacja serwera	399

1.1. Hosting	399
1.2. Konfiguracja serwera	399
1.3. Przykładowa konfiguracja Apache (.htaccess)	399
1.4. Przykładowa konfiguracja Nginx	400
2. Optymalizacja ładowania zasobów	400
2.1. Minimalizacja żądań HTTP	400
2.2. Optymalizacja CSS i JavaScript.....	401
2.3. Optymalizacja obrazów.....	401
2.4. Wykorzystanie CDN	402
3. Monitorowanie i testowanie wydajności	402
3.1. Narzędzia do testowania	402
3.2. Kluczowe metryki wydajnościowe.....	402
3.3. Waterfall analiza.....	403
3.4. Ciągłe monitorowanie	403
4. Optymalizacja kodu frontendowego.....	403
4.1. Optymalizacja HTML	403
4.2. Optymalizacja CSS	404
4.3. Optymalizacja JavaScript	404
4.4. Optymalizacja renderowania.....	404
1. Podstawowe zabezpieczenia	405
1.1. Protokół HTTPS.....	405
1.2. Bezpieczne hasła i uwierzytelnianie	406
2. Zabezpieczenia aplikacji webowej	406
2.1. Ochrona przed typowymi atakami	406

2.2. Implementacja CSP (Content Security Policy)	407
2.3. Bezpieczna konfiguracja cookie.....	407
2.4. Walidacja danych	407
3. Zarządzanie i monitoring.....	408
3.1. Regularnie aktualizacje	408
3.2. Zarządzanie uprawnieniami.....	408
3.3. Audyty i skanowanie bezpieczeństwa	408
3.4. Zapasowe kopie danych	408
4. Ochrona infrastruktury	409
4.1. Bezpieczeństwo serwera.....	409
4.2. Ochrona przed atakami DDoS	409
4.3. Konfiguracja DNS.....	409
4.4. Zabezpieczenia bazy danych	409
5. Zgodność z przepisami i standardy.....	410
5.1. Regulacje dotyczące ochrony danych	410
5.2. Standardy bezpieczeństwa.....	410
5.3. Dokumentacja bezpieczeństwa.....	410
Podsumowanie.....	416
Kluczowe elementy udanej strony internetowej:	416
Wskazówki dla początkujących twórców stron internetowych:	417
Zasoby do dalszej nauki	419
Dokumentacja i poradniki	419
Narzędzia	419
Społeczności	419



Kursy online	419
Newslettery	419

SZCZEGÓŁOWE OPRACOWANIE ELEMENTÓW STRONY INTERNETOWEJ

Materiał szkoleniowy dla początkujących
twórców internetowych

1. Kod źródłowy

HTML (HYPERTEXT MARKUP LANGUAGE)

HTML to podstawowy język używany do tworzenia struktury stron internetowych. Jest to język znaczników, który definiuje elementy strony za pomocą tagów.

Kluczowe cechy HTML: - Tworzy szkielet strony internetowej -
Definiuje nagłówki, akapity, listy, tabele, formularze i inne elementy
- Pozwala na osadzanie obrazów, filmów i innych multimediów -
Jest interpretowany przez przeglądarki i wyświetlany użytkownikom
- Najnowsza wersja to HTML5, która wprowadziła nowe tagi semantyczne

Podstawowe tagi HTML:

```
<!DOCTYPE html> <!-- Deklaracja typu dokumentu -->
<html> <!-- Korzeń dokumentu HTML -->
  <head> <!-- Zawiera metadane, linki do CSS, tytuł s
trony -->
    <title>Tytuł strony</title>
  </head>
  <body> <!-- Zawiera widoczną zawartość strona -->
    <h1>Nagłówek pierwszego poziomu</h1>
    <p>Paragraf tekstu</p>
    <a href="https://przyklad.pl">Link</a>
    
  </body>
</html>
```

Dlaczego jest ważny dla początkujących: - Jest fundamentem każdej strony internetowej - Nawet korzystając z systemów CMS (jak WordPress), podstawowa znajomość HTML pozwala na dostosowanie wyglądu i funkcjonalności - Zrozumienie struktury HTML ułatwia zarządzanie treścią i formatowanie

CSS (CASCADING STYLE SHEETS)

CSS to język arkuszy stylów odpowiedzialny za wygląd i formatowanie strony internetowej.

Kluczowe cechy CSS: - Kontroluje kolor, czcionkę, marginesy, odstępy, układ i inne aspekty wizualne - Umożliwia stosowanie różnych stylów dla różnych urządzeń (responsywność) - Pozwala na animacje i efekty wizualne - Może być zapisany bezpośrednio w HTML (inline), w sekcji head (internal) lub w oddzielnym pliku (external)

Podstawowa składnia CSS:

```
selektor {  
  właściwość: wartość;  
  właściwość2: wartość2;  
}
```

/ Przykłady: */*

```
body {  
  background-color: #f5f5f5;  
  font-family: 'Arial', sans-serif;  
  margin: 0;  
  padding: 0;  
}
```

```
h1 {  
  color: #333;  
  font-size: 2.5em;  
  text-align: center;  
}
```

Dlaczego jest ważny dla początkujących: - Umożliwia przekształcenie prostego dokumentu HTML w atrakcyjną wizualnie stronę - Oddzielenie stylów od struktury ułatwia zarządzanie stronami - Znajomość CSS jest niezbędna do tworzenia stron responsywnych (dostosowanych do urządzeń mobilnych)

JAVASCRIPT

JavaScript to język programowania umożliwiający dodanie interaktywności do stron internetowych.

Kluczowe cechy JavaScript: - Działa po stronie klienta (w przeglądarce użytkownika) - Umożliwia dynamiczną modyfikację zawartości strony - Pozwala na reagowanie na działania użytkownika (kliknięcia, przewijanie, wprowadzanie danych) - Umożliwia walidację formularzy przed wysłaniem danych - Jest podstawą dla wielu frameworków (React, Angular, Vue.js)

Przykład prostego skryptu JavaScript:

```
// Prosta funkcja zmieniająca tekst po kliknięciu przycisku
function zmienTekst() {
    document.getElementById("demo").innerHTML = "Tekst został zmieniony!";
}

// Nastuchiwanie na zdarzenie kliknięcia
document.getElementById("przycisk").addEventListener(
    "click", zmienTekst);
```

Dlaczego jest ważny dla początkujących: - Nawet podstawowa znajomość JavaScript znacząco zwiększa możliwości tworzenia stron - Pozwala na tworzenie lepszych doświadczeń użytkownika (UX) - Jest niezbędny do tworzenia nowoczesnych, interaktywnych aplikacji webowych

2. Główne elementy strukturalne

NAGŁÓWEK (HEADER)

Nagłówek to górna część strony zawierająca kluczowe elementy nawigacyjne i identyfikacyjne.

Kluczowe cechy nagłówka: - Zazwyczaj zawiera logo, nazwę marki lub tytuł strony - Mieści główne menu nawigacyjne - Często zawiera wyszukiwarkę, przyciski logowania, ikony mediów społecznościowych - Jest spójny na wszystkich podstronach, zapewniając jednolite doświadczenie - W HTML5 reprezentowany przez tag `<header>`

Przykładowy kod:

```
<header>
  <div class="logo">
    
  </div>
  <nav>
    <ul>
      <li><a href="index.html">Strona główna</a></li>
      <li><a href="o-nas.html">O nas</a></li>
      <li><a href="uslugi.html">Usługi</a></li>
      <li><a href="kontakt.html">Kontakt</a></li>
    </ul>
  </nav>
  <div class="header-right">
    <form class="search">
      <input type="text" placeholder="Szukaj...">
      <button type="submit">🔍</button>
    </form>
    <a href="login.html" class="login-btn">Zaloguj się</a>
  </div>
</header>
```

Dlaczego jest ważny dla początkujących: - Stanowi wizytówkę strony i pierwszy punkt kontaktu z użytkownikiem - Właściwie zaprojektowany nagłówek ułatwia nawigację i buduje markę - Konsekwentny design nagłówka buduje zaufanie i profesjonalizm

TREŚĆ GŁÓWNA (MAIN CONTENT)

Treść główna to zasadniczy obszar strony zawierający główne informacje, artykuły lub produkty.

Kluczowe cechy treści głównej: - Zawiera najważniejsze informacje dla użytkownika - Jest zorganizowana w sekcje, artykuły, nagłówki i paragrafy - Może zawierać różne typy multimediów (obrazy, wideo, infografiki) - W HTML5 reprezentowana przez tag `<main>` - Często podzielona na sekcje z wykorzystaniem tagów `<section>` i `<article>`

Przykładowy kod:

```
<main>
  <section class="hero">
    <h1>Witamy na naszej stronie</h1>
    <p>Oferujemy najlepsze rozwiązania dla Twojego bi
znesu</p>
    <a href="oferta.html" class="cta-button">Zobacz o
fertę</a>
  </section>

  <section class="features">
    <h2>Nasze usługi</h2>
    <div class="feature-grid">
      <article class="feature">
        
        <h3>Konsultacje</h3>
        <p>Profesjonalne doradztwo w zakresie strateg
ii internetowej.</p>
      </article>
      <article class="feature">
        
```

```

        <h3>Projektowanie</h3>
        <p>Nowoczesne projekty stron zgodne z najnowszymi trendami.</p>
    </article>
    <!-- Więcej elementów... -->
</div>
</section>
</main>

```

Dlaczego jest ważna dla początkujących: - Stanowi serce strony internetowej - Jakość i organizacja treści głównej wpływa na skuteczność przekazu - Właściwe wykorzystanie tagów semantycznych poprawia SEO i dostępność

STOPKA (FOOTER)

Stopka to dolna część strony zawierająca dodatkowe informacje, linki i dane kontaktowe.

Kluczowe cechy stopki: - Zawiera informacje o prawach autorskich, dane kontaktowe, mapę strony - Często zawiera dodatkowe menu z ważnymi linkami - Może zawierać formularze newslettera, linki do mediów społecznościowych - W HTML5 reprezentowana przez tag `<footer>` - Jest obecna na wszystkich podstronach, zapewniając spójność

Przykładowy kod:

```

<footer>
  <div class="footer-columns">
    <div class="footer-column">
      <h4>O firmie</h4>
      <ul>
        <li><a href="historia.html">Historia</a></li>
        <li><a href="zespól.html">Zespół</a></li>
        <li><a href="kariera.html">Kariera</a></li>
      </ul>
    </div>
    <div class="footer-column">

```

```

<h4>Wsparcie</h4>
<ul>
  <li><a href="faq.html">FAQ</a></li>
  <li><a href="pomoc.html">Centrum pomocy</a></li>
  <li><a href="kontakt.html">Kontakt</a></li>
</ul>
</div>
<div class="footer-column">
  <h4>Kontakt</h4>
  <address>
    ul. Przykładowa 123<br>
    00-000 Warszawa<br>
    Tel: +48 123 456 789<br>
    Email: kontakt@firma.pl
  </address>
</div>
<div class="footer-column newsletter">
  <h4>Newsletter</h4>
  <form>
    <input type="email" placeholder="Twój email">
    <button type="submit">Zapisz się</button>
  </form>
</div>
</div>
<div class="footer-bottom">
  <p>&copy; 2025 Nazwa Firmy. Wszelkie prawa zastrzeżone.</p>
  <div class="legal-links">
    <a href="prywatnosc.html">Polityka prywatności</a>
    <a href="regulamin.html">Regulamin</a>
    <a href="cookies.html">Polityka cookies</a>
  </div>
</div>
</footer>

```


Dlaczego jest ważna dla początkujących: - Zapewnia ważne informacje prawne i kontaktowe - Pomaga użytkownikom w nawigacji po stronie - Buduje wiarygodność i profesjonalizm strony

MENU NAWIGACYJNE

Menu nawigacyjne to element umożliwiający użytkownikom poruszanie się po różnych sekcjach i podstronach witryny.

Kluczowe cechy menu nawigacyjnego: - Zazwyczaj umieszczone w nagłówku, choć może występować też z boku lub jako menu “hamburger” na urządzeniach mobilnych - Zawiera linki do najważniejszych sekcji i podstron - Może być wielopoziomowe (z podmenu) - W HTML5 reprezentowane przez tag `<nav>` - Powinno być intuicyjne i łatwe w obsłudze

Przykładowy kod:

```
<nav class="main-navigation">
  <ul>
    <li><a href="index.html">Strona główna</a></li>
    <li class="dropdown">
      <a href="produkty.html">Produkty</a>
      <ul class="submenu">
        <li><a href="kategoria1.html">Kategoria 1</a>
      </li>
        <li><a href="kategoria2.html">Kategoria 2</a>
      </li>
        <li><a href="kategoria3.html">Kategoria 3</a>
      </li>
      </ul>
    </li>
    <li><a href="uslugi.html">Usługi</a></li>
    <li><a href="blog.html">Blog</a></li>
    <li><a href="o-nas.html">O nas</a></li>
    <li><a href="kontakt.html">Kontakt</a></li>
  </ul>
</nav>
```

```

<script>
  // Prosty skrypt do obsługi menu rozwijanego
  document.querySelectorAll('.dropdown').forEach(drop
down => {
    dropdown.addEventListener('mouseenter', () => {
      dropdown.querySelector('.submenu').style.displa
y = 'block';
    });
    dropdown.addEventListener('mouseleave', () => {
      dropdown.querySelector('.submenu').style.displa
y = 'none';
    });
  });
</script>

```

CSS dla responsywnego menu:

```

/* Desktop */
.main-navigation ul {
  display: flex;
  list-style: none;
  margin: 0;
  padding: 0;
}

.main-navigation li {
  position: relative;
}

.main-navigation a {
  display: block;
  padding: 15px 20px;
  text-decoration: none;
  color: #333;
}

.submenu {
  display: none;
  position: absolute;
  background: white;

```

```

    box-shadow: 0 2px 5px rgba(0,0,0,0.2);
    min-width: 200px;
}

/* Mobile */
@media (max-width: 768px) {
    .main-navigation ul {
        flex-direction: column;
    }

    .submenu {
        position: static;
        box-shadow: none;
    }
}

```

Dlaczego jest ważne dla początkujących: - Jest kluczowym elementem UX (User Experience) - Dobrze zaprojektowane menu znacząco wpływa na łatwość korzystania ze strony - Różne style menu (poziome, pionowe, hamburger) odpowiadają różnym potrzebom projektowym

SIDEBAR (PANEL BOCZNY)

Sidebar to boczny panel zawierający dodatkowe informacje, menu lub elementy interaktywne.

Kluczowe cechy sidebara: - Zazwyczaj umieszczony po lewej lub prawej stronie treści głównej - Może zawierać dodatkową nawigację, popularne artykuły, reklamy, widżety - Na urządzeniach mobilnych często jest ukrywany lub przenoszony pod treść główną - W HTML5 często reprezentowany przez tag `<aside>`

Przykładowy kod:

```

<div class="layout">
  <main>
    <!-- Treść główna strony -->
  </main>

```

```

<aside class="sidebar">
  <div class="widget search">
    <h3>Wyszukiwarka</h3>
    <form>
      <input type="text" placeholder="Szukaj...">
      <button type="submit">🔍</button>
    </form>
  </div>

  <div class="widget categories">
    <h3>Kategorie</h3>
    <ul>
      <li><a href="#">Technologia (12)</a></li>
      <li><a href="#">Biznes (8)</a></li>
      <li><a href="#">Marketing (15)</a></li>
      <li><a href="#">Projektowanie (6)</a></li>
    </ul>
  </div>

  <div class="widget popular-posts">
    <h3>Popularne artykuły</h3>
    <ul>
      <li>
        <a href="#">
          
          <span>Jak stworzyć skuteczną stronę inter
netową</span>
        </a>
      </li>
      <li>
        <a href="#">
          
          <span>10 zasad dobrego designu</span>
        </a>
      </li>
    </ul>
  </div>

```

```

</div>

<div class="widget newsletter">
  <h3>Newsletter</h3>
  <p>Zapisz się, aby otrzymywać najnowsze aktuali
zacje.</p>
  <form>
    <input type="email" placeholder="Twój email">
    <button type="submit">Zapisz się</button>
  </form>
</div>
</aside>
</div>

```

CSS dla responsywnego układu z sidebarem:

```

/* Desktop */
.layout {
  display: flex;
  gap: 30px;
  max-width: 1200px;
  margin: 0 auto;
}

main {
  flex: 1;
}

.sidebar {
  width: 300px;
}

.widget {
  margin-bottom: 30px;
  padding: 20px;
  background: #f5f5f5;
  border-radius: 5px;
}

/* Mobile */

```

```
@media (max-width: 768px) {  
  .layout {  
    flex-direction: column;  
  }  
  
  .sidebar {  
    width: 100%;  
  }  
}
```

Dlaczego jest ważny dla początkujących: - Pozwala na dodanie funkcjonalności bez zaśmiecania treści głównej - Umożliwia lepszą organizację informacji i nawigacji - Responsywne podejście do sidebara jest ważnym aspektem mobilnego designu

3. Elementy wizualne i multimedialne

OBRAZY I GRAFIKI

Obrazy i grafiki są kluczowymi elementami wizualnymi stron internetowych, które przyciągają uwagę, ilustrują koncepcje i budują estetykę witryny.

Kluczowe cechy: - Różne formaty dostosowane do potrzeb (JPEG, PNG, WebP, SVG) - Optymalizacja pod kątem rozmiaru pliku i jakości - Responsywność (dostosowanie do różnych rozmiarów ekranu) - Znaczenie atrybutu alt dla dostępności i SEO

Formaty obrazów: - **JPEG** - najlepszy dla fotografii i obrazów z wieloma kolorami - **PNG** - idealny dla obrazów wymagających przezroczystości - **WebP** - nowoczesny format oferujący lepszą kompresję niż JPEG i PNG - **SVG** - format wektorowy, idealny dla logo, ikon i ilustracji, skaluje się bez utraty jakości

Przykładowy kod HTML:

```
<!-- Podstawowy obraz -->  

```

```

<!-- Responsywny obraz z różnymi rozmiarami -->


<!-- Obrazy SVG -->

<!-- lub bezpośrednio w kodzie -->
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40" stroke="green" strok
e-width="4" fill="yellow" />
</svg>

```

CSS dla obrazów:

```

/* Podstawowe style dla obrazów */
.image-container img {
  max-width: 100%; /* Zapewnia responsywność */
  height: auto;
  display: block; /* Usuwa niepotrzebne przestrzenie
pod obrazem */
}

/* Efekty dla obrazów */
.image-hover {
  transition: transform 0.3s ease;
}

.image-hover:hover {
  transform: scale(1.05);
}

/* Ramki i cienie */
.image-framed {
  border: 5px solid white;
  box-shadow: 0 4px 8px rgba(0,0,0,0.2);
}

```

```

}

/* Obrazy tła */
.hero-section {
  background-image: url('tlo.jpg');
  background-size: cover;
  background-position: center;
  min-height: 500px;
}

```

Optymalizacja obrazów: - Kompresja obrazów przed umieszczeniem na stronie - Używanie odpowiednich wymiarów (nie skalowanie dużych obrazów CSS-em) - Lazy loading dla poprawy wydajności strony: `` - Używanie formatu WebP z fallbackiem dla starszych przeglądarek

Dlaczego są ważne dla początkujących: - Są kluczowym elementem przyciągającym uwagę użytkowników - Wpływają na szybkość ładowania strony (kluczowy czynnik SEO i UX) - Prawidłowe wdrożenie responsywnych obrazów jest niezbędne dla mobilnych stron

FILMY

Filmy są potężnym narzędziem angażującym użytkowników, które pozwala na przekazanie złożonych informacji w przystępny sposób.

Kluczowe cechy: - Mogą być hostowane lokalnie lub osadzone z platform jak YouTube, Vimeo - Wymagają odpowiedniego formatu i kompresji - Powinny mieć opcje kontroli odtwarzania - Mogą wpływać na wydajność strony

Przykładowy kod HTML:

```

<!-- Natywny odtwarzacz HTML5 -->
<video width="640" height="360" controls>
  <source src="film.mp4" type="video/mp4">

```



```

<source src="film.webm" type="video/webm">
  Twoja przeglądarka nie obsługuje tagu video.
</video>

<!-- Osadzony film z YouTube -->
<iframe
  width="560"
  height="315"
  src="https://www.youtube.com/embed/ID_FILMU"
  frameborder="0"
  allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture"
  allowfullscreen>
</iframe>

<!-- Osadzony film z Vimeo -->
<iframe
  src="https://player.vimeo.com/video/ID_FILMU"
  width="640"
  height="360"
  frameborder="0"
  allow="autoplay; fullscreen; picture-in-picture"
  allowfullscreen>
</iframe>

```

CSS dla filmów:

```

/* Responsywne osadzanie filmów */
.video-container {
  position: relative;
  padding-bottom: 56.25%; /* Proporcje 16:9 */
  height: 0;
  overflow: hidden;
  max-width: 100%;
}

.video-container iframe,
.video-container video {
  position: absolute;
  top: 0;

```

```

left: 0;
width: 100%;
height: 100%;
}

/* Style dla odtwarzacza HTML5 */
video {
  max-width: 100%;
  height: auto;
}

/* Tło wideo */
.video-background {
  position: relative;
  overflow: hidden;
  height: 500px;
}

.video-background video {
  position: absolute;
  min-width: 100%;
  min-height: 100%;
  width: auto;
  height: auto;
  z-index: -1;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}

```

Dobre praktyki: - Nie ustawiaj automatycznego odtwarzania z dźwiękiem (irytuje użytkowników) - Zawsze zapewniaj kontrolki do sterowania odtwarzaniem - Używaj atrybutu poster dla filmów (miniatura wyświetlana przed odtworzeniem) - Rozważ dodanie napisów dla lepszej dostępności - Używaj lazy loading dla filmów znajdujących się niżej na stronie

Dlaczego są ważne dla początkujących: - Zwiększają czas spędzony na stronie i zaangażowanie użytkowników - Pozwalają na

efektywne prezentowanie produktów lub usług - Wymagają odpowiedniego wdrożenia, aby nie wpływały negatywnie na wydajność strony

ANIMACJE

Animacje to dynamiczne elementy strony, które przyciągają uwagę, ilustrują procesy lub urozmaicają interakcje użytkownika.

Kluczowe cechy: - Mogą być tworzone za pomocą CSS, JavaScript lub bibliotek jak GSAP - Powinny być subtelne i celowe, nie rozpraszać - Ważne dla sygnalizowania akcji, przejść między stanami i informacji zwrotnej - Wpływają na wrażenia użytkownika (UX)

Typy animacji: - Animacje przejścia (hover, focus) - Animacje ładowania (loaders, spinners) - Animacje wejścia (przy scrollovaniu) - Animacje ilustrujące procesy - Mikroanimacje (drobne elementy interaktywne)

Przykładowy kod CSS:

```
/* Proste przejście */
.button {
  background-color: blue;
  color: white;
  padding: 10px 20px;
  transition: background-color 0.3s ease, transform 0
.2s ease;
}

.button:hover {
  background-color: darkblue;
  transform: scale(1.05);
}

/* Animacja z keyframes */
@keyframes fadeIn {
  from {
```

```

    opacity: 0;
    transform: translateY(20px);
  }
  to {
    opacity: 1;
    transform: translateY(0);
  }
}

.fade-in-element {
  opacity: 0;
  animation: fadeIn 1s ease forwards;
}

/* Animacja ładowania */
@keyframes rotate {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}

.loader {
  width: 40px;
  height: 40px;
  border: 4px solid #f3f3f3;
  border-top: 4px solid #3498db;
  border-radius: 50%;
  animation: rotate 1s linear infinite;
}

```

Przykład animacji przy scrollowaniu z JavaScript:

```

<div class="scroll-animation" data-animation="fadeIn">
  <div>
    To pojawi się podczas scrollowania.
  </div>
</div>

```

```

<script>
  // Prosty skrypt do animacji przy scrollowaniu
  document.addEventListener('DOMContentLoaded', () => {
    const scrollElements = document.querySelectorAll(
      '.scroll-animation');

    const elementInView = (el, percentageScroll = 100)
    => {
      const elementTop = el.getBoundingClientRect().top;
      return (
        elementTop <=
          ((window.innerHeight || document.documentElement.clientHeight) * (percentageScroll/100))
      );
    };

    const displayScrollElement = (element) => {
      element.classList.add('scrolled');
    };

    const hideScrollElement = (element) => {
      element.classList.remove('scrolled');
    };

    const handleScrollAnimation = () => {
      scrollElements.forEach((el) => {
        if (elementInView(el, 90)) {
          displayScrollElement(el);
        } else {
          hideScrollElement(el);
        }
      })
    }

    window.addEventListener('scroll', () => {
      handleScrollAnimation();
    });
  });

```

```
// Initialize  
handleScrollAnimation();  
});  
</script>
```

CSS dla animacji przy scrollowaniu:

```
.scroll-animation {  
  opacity: 0;  
  transition: all 1s ease;  
}  
  
.scroll-animation.scrolled {  
  opacity: 1;  
}  
  
.scroll-animation[data-animation="fadeIn"].scrolled {  
  animation: fadeIn 1s ease forwards;  
}  
  
.scroll-animation[data-animation="slideIn"].scrolled  
{  
  animation: slideIn 1s ease forwards;  
}  
  
@keyframes slideIn {  
  from {  
    transform: translateX(-50px);  
    opacity: 0;  
  }  
  to {  
    transform: translateX(0);  
    opacity: 1;  
  }  
}
```

Dobre praktyki: - Używaj `prefers-reduced-motion` media query dla dostępności - Nie przesadzaj z animacjami, które mogą

rozpraszać i irytować - Optymalizuj animacje pod kątem wydajności (używaj properties które nie wywołują reflow) - Testuj na różnych urządzeniach i przeglądarkach

Dlaczego są ważne dla początkujących: - Dodają profesjonalizmu i nowoczesności do projektów - Przyciągają uwagę do ważnych elementów i akcji - Poprawiają wrażenia użytkownika i zrozumienie interfejsu

CZCIONKI

Typografia jest kluczowym elementem designu strony, wpływającym na czytelność, ton komunikacji i ogólną estetykę.

Kluczowe cechy: - Dobór odpowiednich fontów dla różnych elementów (nagłówki, treść, menu) - Hierarchia typograficzna dla lepszej organizacji treści - Odpowiednie rozmiary i odstępy dla zapewnienia czytelności - Responsywność typografii na różnych urządzeniach

Rodzaje fontów: - **Serif** - z szeryfami, dobre dla dłuższych tekstów w druku (Times New Roman, Georgia) - **Sans-serif** - bez szeryfów, czytelne na ekranach (Arial, Roboto, Open Sans) - **Monospace** - stała szerokość znaków, używane do kodu (Courier, Consolas) - **Display** - ozdobne, używane dla nagłówków i specjalnych elementów - **Script** - przypominające pismo odręczne, używane z umiarem dla akcentów

Implementacja fontów webowych:

```
<!-- W sekcji head dokumentu HTML -->
<head>
  <!-- Metoda 1: Linkowanie do Google Fonts -->
  <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;700&family=Playfair+Display:wght@700&display=swap" rel="stylesheet">
```

```
<!-- Metoda 2: Własne fonty poprzez @font-face w CSS
```

```

S -->
<style>
  @font-face {
    font-family: 'MojFont';
    src: url('sciezka/do/mojfont.woff2') format('woff2'),
        url('sciezka/do/mojfont.woff') format('woff');
    font-weight: normal;
    font-style: normal;
    font-display: swap; /* Strategia wyświetlania podczas ładowania */
  }
</style>
</head>

```

CSS dla typografii:

```

/* System fontów */
body {
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
  font-size: 16px;
  line-height: 1.6;
  color: #333;
}

/* Hierarchia typograficzna */
h1 {
  font-family: 'Playfair Display', serif;
  font-size: 2.5rem;
  font-weight: 700;
  margin-bottom: 1em;
  line-height: 1.2;
}

h2 {
  font-family: 'Playfair Display', serif;
  font-size: 2rem;

```



```
font-weight: 700;
margin: 1.5em 0 0.5em;
}

h3 {
font-size: 1.5rem;
font-weight: 600;
margin: 1em 0 0.5em;
}

p {
margin-bottom: 1em;
}

/* Responsywna typografia */
@media (max-width: 768px) {
body {
font-size: 14px;
}

h1 {
font-size: 2rem;
}

h2 {
font-size: 1.7rem;
}

h3 {
font-size: 1.3rem;
}
}

/* Dodatkowe style typograficzne */
.lead {
font-size: 1.2em;
font-weight: 300;
line-height: 1.7;
}
```

```
.caption {  
  font-size: 0.8em;  
  color: #666;  
}  
  
.quote {  
  font-style: italic;  
  border-left: 3px solid #333;  
  padding-left: 1em;  
  margin-left: 0;  
}
```

Jednostki dla czcionek: - **px** - piksele, jednostka absolutna - **em** - względna do rozmiaru czcionki elementu nadrzędnego - **rem** - względna do rozmiaru czcionki elementu root (html) - **%** - procentowa względem elementu nadrzędnego - **vw** - procent szerokości viewportu (1vw = 1% szerokości okna)

Właściwości CSS dla typografii: - **font-family** - rodzina czcionek - **font-size** - rozmiar czcionki - **font-weight** - grubość (100-900 lub normal, bold) - **line-height** - wysokość linii - **letter-spacing** - odstępy między literami - **text-align** - wyrównanie tekstu - **text-transform** - transformacja tekstu (uppercase, lowercase, capitalize) - **text-decoration** - dekoracja tekstu (underline, line-through)

Dobre praktyki: - Ograniczaj liczbę używanych fontów (zazwyczaj 2-3 na stronę) - Zapewnij odpowiedni kontrast tekstu i tła - Używaj jednostek względnych (rem, em) dla lepszej responsywności - Stosuj zmienne CSS do zarządzania typografią - Pamiętaj o fallback fontach w font-family

Dlaczego są ważne dla początkujących: - Typografia stanowi ponad 90% designu strony internetowej - Dobra typografia znacząco wpływa na czytelność i UX - Spójna typografia buduje profesjonalny wizerunek

4. Elementy interaktywne

FORMULARZE

Formularze są kluczowymi elementami umożliwiającymi zbieranie danych od użytkowników, od prostych formularzy kontaktowych po rozbudowane systemy rejestracji.

Kluczowe cechy: - Różnorodne pola do zbierania różnych typów danych - Walidacja danych po stronie klienta i serwera - Dostępność dla wszystkich użytkowników - Odpowiednie etykiety i informacje pomocnicze

Elementy formularzy: - Pola tekstowe i textarea - Checkboxy i radio buttony - Select (listy rozwijane) - Upload plików - Przyciski (submit, reset) - Ukryte pola

Przykładowy kod HTML:

```
<form action="/submit-form" method="post" class="contact-form">
  <div class="form-group">
    <label for="name">Imię i nazwisko *</label>
    <input type="text" id="name" name="name" required
  >
    <small class="form-hint">Wprowadź swoje pełne imię i nazwisko</small>
  </div>

  <div class="form-group">
    <label for="email">Email *</label>
    <input type="email" id="email" name="email" required
  >
  </div>

  <div class="form-group">
    <label for="phone">Telefon</label>
    <input type="tel" id="phone" name="phone" pattern=
    ="[0-9]{9}">
```

```

    <small class="form-hint">Format: 123456789</small
  >
</div>

<div class="form-group">
  <label for="subject">Temat</label>
  <select id="subject" name="subject">
    <option value="">Wybierz temat...</option>
    <option value="question">Pytanie</option>
    <option value="feedback">Opinia</option>
    <option value="other">Inne</option>
  </select>
</div>

<div class="form-group">
  <label for="message">Wiadomość *</label>
  <textarea id="message" name="message" rows="5" re
quired></textarea>
</div>

<div class="form-group checkbox">
  <input type="checkbox" id="terms" name="terms" re
quired>
  <label for="terms">Akceptuję <a href="/terms">reg
ulamin</a> i <a href="/privacy">politykę prywatności<
/a> *</label>
</div>

<div class="form-group newsletter">
  <input type="checkbox" id="newsletter" name="news
letter">
  <label for="newsletter">Chcę otrzymywać newslette
r</label>
</div>

<div class="form-buttons">
  <button type="reset" class="btn-secondary">Wyczyś
ć</button>
  <button type="submit" class="btn-primary">Wyślij<

```

```
/button>
</div>

<p class="required-note">* Pola wymagane</p>
</form>
```

CSS dla formularzy:

```
.contact-form {
  max-width: 600px;
  margin: 0 auto;
}

.form-group {
  margin-bottom: 20px;
}

label {
  display: block;
  margin-bottom: 5px;
  font-weight: 500;
}

input[type="text"],
input[type="email"],
input[type="tel"],
select,
textarea {
  width: 100%;
  padding: 10px;
  border: 1px solid #ddd;
  border-radius: 4px;
  font-size: 16px;
}

textarea {
  resize: vertical;
}

.form-hint {
```

```
display: block;
margin-top: 5px;
color: #666;
font-size: 0.9em;
}

.checkbox {
display: flex;
align-items: flex-start;
}

.checkbox input {
margin-top: 5px;
margin-right: 10px;
}

.form-buttons {
display: flex;
justify-content: space-between;
gap: 10px;
}

button {
padding: 10px 20px;
border: none;
border-radius: 4px;
font-size: 16px;
cursor: pointer;
}

.btn-primary {
background-color: #4CAF50;
color: white;
}

.btn-secondary {
background-color: #f1f1f1;
color: #333;
}
```

```

/* Walidacja */
input:invalid,
textarea:invalid {
  border-color: #ff6b6b;
}

input:focus,
textarea:focus,
select:focus {
  outline: none;
  border-color: #4CAF50;
  box-shadow: 0 0 3px rgba(76, 175, 80, 0.5);
}

.required-note {
  margin-top: 20px;
  font-size: 0.9em;
  color: #666;
}

```

JavaScript dla walidacji formularza:

```

document.addEventListener('DOMContentLoaded', function() {
  const form = document.querySelector('.contact-form');

  form.addEventListener('submit', function(event) {
    let valid = true;

    // Podstawowa walidacja
    const requiredFields = form.querySelectorAll('[required]');
    requiredFields.forEach(field => {
      if (!field.value.trim()) {
        valid = false;
        showError(field, 'To pole jest wymagane');
      } else {
        clearError(field);
      }
    });
  });

```

```

    }
  });

  // Walidacja email
  const emailField = form.querySelector('#email');
  if (emailField.value && !isValidEmail(emailField.value)) {
    valid = false;
    showError(emailField, 'Wprowadź poprawny adres email');
  }

  // Walidacja telefonu
  const phoneField = form.querySelector('#phone');
  if (phoneField.value && !isValidPhone(phoneField.value)) {
    valid = false;
    showError(phoneField, 'Wprowadź poprawny numer telefonu (9 cyfr)');
  }

  if (!valid) {
    event.preventDefault();
  }
});

function showError(field, message) {
  // Usunięcie istniejącego komunikatu
  clearError(field);

  // Dodanie komunikatu błędu
  const error = document.createElement('div');
  error.className = 'error-message';
  error.textContent = message;
  field.parentNode.appendChild(error);

  // Zaznaczenie pola z błędem
  field.classList.add('error');
}

```



```

function clearError(field) {
  const errorElement = field.parentNode.querySelector(
    '.error-message');
  if (errorElement) {
    errorElement.remove();
  }
  field.classList.remove('error');
}

function isValidEmail(email) {
  return /^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(email);
}

function isValidPhone(phone) {
  return /^[0-9]{9}$/.test(phone);
}
});

```

Dostępność formularzy: - Używaj elementu `<label>` dla wszystkich pól formularza - Dodawaj atrybut `for` dla powiązania etykiety z polem - Używaj atrybutu `aria-describedby` dla powiązania pola z dodatkowym opisem - Zapewnij czytelne komunikaty o błędach - Grupuj powiązane pola za pomocą `<fieldset>` i `<legend>`

Dlaczego są ważne dla początkujących: - Formularze są głównym sposobem interakcji z użytkownikami - Dobry formularz zwiększa konwersję (wypełnienia i wystania) - Źle zaprojektowane formularze mogą frustrować użytkowników i prowadzić do porzucenia strony

PRZYCISKI

Przyciski są podstawowymi elementami interakcji, które pozwalają użytkownikom na wykonywanie akcji na stronie.

Kluczowe cechy: - Czytelne i jasne komunikaty dotyczące akcji - Wyraźne stany (domyślny, hover, active, focus, disabled) -

Hierarchia ważności (primary, secondary, tertiary) - Spójny design w całej stronie

Rodzaje przycisków: - Standardowe przyciski akcji - Przyciski formularza (submit, reset) - Przyciski ikonowe (często w interfejsach mobilnych) - Przyciski przełączające (toggle) - Przyciski Call-to-action (CTA)

Przykładowy kod HTML:

```
<!-- Standardowe przyciski -->
<button class="btn">Przycisk domyślny</button>
<button class="btn btn-primary">Przycisk główny</button>
<button class="btn btn-secondary">Przycisk drugorzędny</button>
<button class="btn btn-danger">Przycisk usuwania</button>
<button class="btn" disabled>Przycisk wyłączony</button>
```

```
<!-- Przyciski formularza -->
<button type="submit" class="btn btn-primary">Wyślij formularz</button>
<button type="reset" class="btn btn-secondary">Resetuj</button>
```

```
<!-- Przyciski linkowe -->
<a href="#" class="btn btn-primary">Link jako przycisk</a>
```

```
<!-- Przyciski z ikonami -->
<button class="btn btn-icon">
  <svg class="icon">...</svg>
  Z ikoną
</button>
```

```
<!-- Tylko ikona -->
<button class="btn btn-icon-only" aria-label="Dodaj d
```

```
o ulubionych">
  <svg class="icon">...</svg>
</button>

<!-- Przycisk CTA -->
<button class="btn btn-cta">Rozpocznij teraz!</button
>
```

CSS dla przycisków:

```
/* Podstawowe style przycisków */
.btn {
  display: inline-block;
  padding: 10px 20px;
  font-size: 16px;
  font-weight: 500;
  text-align: center;
  text-decoration: none;
  border: 1px solid #ddd;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s, color 0.3s, border-color 0.3s, transform 0.1s;
  background-color: #f8f8f8;
  color: #333;
}

/* Stany przycisków */
.btn:hover {
  background-color: #eee;
}

.btn:active {
  transform: translateY(1px);
}

.btn:focus {
  outline: none;
  box-shadow: 0 0 0 3px rgba(66, 153, 225, 0.5);
}
```

```
/* Warianty przycisków */
.btn-primary {
  background-color: #4CAF50;
  border-color: #4CAF50;
  color: white;
}

.btn-primary:hover {
  background-color: #43a047;
  border-color: #43a047;
}

.btn-secondary {
  background-color: #f1f1f1;
  border-color: #d9d9d9;
  color: #333;
}

.btn-secondary:hover {
  background-color: #e7e7e7;
}

.btn-danger {
  background-color: #f44336;
  border-color: #f44336;
  color: white;
}

.btn-danger:hover {
  background-color: #e53935;
  border-color: #e53935;
}

/* Przycisk wyłączony */
.btn:disabled,
.btn.disabled {
  opacity: 0.5;
  cursor: not-allowed;
}
```

```
    pointer-events: none;
}

/* Przyciski z ikonami */
.btn-icon {
    display: inline-flex;
    align-items: center;
    gap: 8px;
}

.btn-icon-only {
    padding: 10px;
    border-radius: 50%;
    display: inline-flex;
    align-items: center;
    justify-content: center;
}

.icon {
    width: 16px;
    height: 16px;
}

/* Przycisk CTA */
.btn-cta {
    background-color: #ff9800;
    border-color: #ff9800;
    color: white;
    font-weight: 700;
    padding: 12px 24px;
    font-size: 18px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

.btn-cta:hover {
    background-color: #fb8c00;
    border-color: #fb8c00;
    transform: translateY(-2px);
    box-shadow: 0 6px 8px rgba(0, 0, 0, 0.15);
}
```

```
}

/* Rozmiary przycisków */
.btn-small {
    padding: 6px 12px;
    font-size: 14px;
}

.btn-large {
    padding: 12px 24px;
    font-size: 18px;
}

/* Przyciski blokowe (na całą szerokość) */
.btn-block {
    display: block;
    width: 100%;
}

/* Grupy przycisków */
.btn-group {
    display: inline-flex;
}

.btn-group .btn {
    border-radius: 0;
    margin-right: -1px;
}

.btn-group .btn:first-child {
    border-top-left-radius: 4px;
    border-bottom-left-radius: 4px;
}

.btn-group .btn:last-child {
    border-top-right-radius: 4px;
    border-bottom-right-radius: 4px;
    margin-right: 0;
}
```

Dobre praktyki: - Używaj opisowych etykiet na przyciskach - Zawsze wskazuj, co stanie się po kliknięciu - Używaj atrybutu **disabled** dla niedostępnych akcji - Zapewnij wyraźny kontrast dla lepszej dostępności - Zachowuj spójność w całej stronie - Przycisk "Submit" formularza powinien mieć wyróżniający się design

Dlaczego są ważne dla początkujących: - Przyciski są kluczowym elementem interakcji użytkownika ze stroną - Dobrze zaprojektowane przyciski zwiększają konwersję - Hierarchia przycisków pomaga użytkownikom w podejmowaniu decyzji

LINKI

Linki są fundamentalnym elementem internetu, umożliwiającym nawigację i łączenie treści zarówno w obrębie jednej strony, jak i między stronami.

Kluczowe cechy: - Wyraźne odróżnienie od zwykłego tekstu - Intuicyjne stany (domyślny, hover, active, visited, focus) - Spójny wygląd w całej witrynie - Jasne wskazanie celu lub akcji

Rodzaje linków: - Wewnętrzne (do innych sekcji tej samej strony) - Zewnętrzne (do innych stron) - Do pobrania plików - Kotwice (anchors) do konkretnych sekcji strony - Linki typu mailto i tel

Przykładowy kod HTML:

```
<!-- Podstawowy link -->
<a href="https://example.com">Standardowy link</a>

<!-- Link wewnętrzny (do sekcji na tej samej stronie) -->
<a href="#section-1">Przejdź do sekcji 1</a>

<!-- Link z atrybutem title (podpowiedź) -->
<a href="https://example.com" title="Przejdź do Example.com">Link z podpowiedzią</a>
```

```

<!-- Link do pobrania pliku -->
<a href="dokument.pdf" download>Pobierz PDF</a>

<!-- Link otwierający się w nowym oknie/karcie -->
<a href="https://example.com" target="_blank" rel="noopener noreferrer">Otwórz w nowej karcie</a>

<!-- Link email -->
<a href="mailto:kontakt@przyklad.pl">Wyślij email</a>

<!-- Link telefoniczny -->
<a href="tel:+48123456789">+48 123 456 789</a>

<!-- Link z ikoną -->
<a href="https://example.com" class="icon-link">
  Odwiedź stronę
  <svg class="icon">...</svg>
</a>

```

CSS dla linków:

```

/* Podstawowe style linków */
a {
  color: #0066cc;
  text-decoration: underline;
  transition: color 0.2s;
}

/* Stany linków */
a:hover {
  color: #004499;
}

a:active {
  color: #002255;
}

a:visited {
  color: #551A8B;
}

```



```
a:focus {
  outline: 2px solid #0066cc;
  outline-offset: 2px;
}

/* Usuwanie podkreślenia tylko przy hover */
.clean-link {
  text-decoration: none;
}

.clean-link:hover {
  text-decoration: underline;
}

/* Linki z ikonami */
.icon-link {
  display: inline-flex;
  align-items: center;
  gap: 5px;
}

.icon-link .icon {
  width: 16px;
  height: 16px;
  transition: transform 0.2s;
}

.icon-link:hover .icon {
  transform: translateX(3px);
}

/* Link jako przycisk */
.button-link {
  display: inline-block;
  padding: 10px 20px;
  background-color: #0066cc;
  color: white;
  text-decoration: none;
}
```

```

border-radius: 4px;
font-weight: 500;
transition: background-color 0.3s;
}

.button-link:hover {
background-color: #004499;
color: white;
}

/* Linki w nawigacji */
.nav-link {
text-decoration: none;
padding: 10px 15px;
color: #333;
font-weight: 500;
border-bottom: 2px solid transparent;
transition: border-color 0.3s, color 0.3s;
}

.nav-link:hover,
.nav-link.active {
border-bottom-color: #0066cc;
color: #0066cc;
}

/* Linki do zewnętrznych stron */
.external-link::after {
content: '↗';
display: inline-block;
margin-left: 3px;
}

/* Linki do pobrania */
.download-link::after {
content: '↓';
display: inline-block;
margin-left: 3px;
}

```

Zaawansowane linki z JavaScript:

```
// Oznaczanie linków zewnętrznych
document.addEventListener('DOMContentLoaded', function() {
    const links = document.getElementsByTagName('a');

    for (let i = 0; i < links.length; i++) {
        const link = links[i];
        const href = link.getAttribute('href');

        // Sprawdzanie czy link jest zewnętrzny
        if (href && href.indexOf('http') === 0 && !href.includes(window.location.hostname)) {
            // Dodawanie klasy i atrybutów bezpieczeństwa
            link.classList.add('external-link');
            link.setAttribute('target', '_blank');
            link.setAttribute('rel', 'noopener noreferrer');
        }

        // Opcjonalnie: dodawanie informacji o otwieraniu w nowym oknie
        link.setAttribute('title', link.title + ' (otwiera się w nowej karcie)');
    }

    // Łagodne scrollowanie dla linków wewnętrznych
    document.querySelectorAll('a[href^="#"]').forEach(anchor => {
        anchor.addEventListener('click', function(e) {
            e.preventDefault();

            const targetId = this.getAttribute('href');
            if (targetId === '#') return;

            const targetElement = document.querySelector(targetId);
            if (targetElement) {

```

```
targetElement.scrollToView({
  behavior: 'smooth',
  block: 'start'
});

// Opcjonalnie: aktualizacja URL
history.pushState(null, null, targetId);
}
});
});
});
```

Dobre praktyki: - Używaj opisowych tekstów linków (zamiast “kliknij tutaj”) - Stosuj atrybuty **title** dla dodatkowych informacji - Dla linków zewnętrznych używaj **target="_blank"** i **rel="noopener noreferrer"** - Nie stylizuj zwykłego tekstu, aby wyglądał jak link - Zapewnij odpowiedni kontrast kolorów - Używaj atrybutu **download** dla plików do pobrania

Dlaczego są ważne dla początkujących: - Linki są podstawowym elementem nawigacji w internecie - Prawidłowe wykorzystanie linków wpływa na UX i SEO - Spójne stylowanie linków buduje zaufanie i profesjonalizm strony

MENU ROZWIJANE

Menu rozwijane pozwalają na hierarchiczną organizację treści i nawigację, oszczędzając przestrzeń i porządkując interfejs użytkownika.

Kluczowe cechy: - Ukrywanie i pokazywanie dodatkowych opcji na żądanie - Intuicyjna interakcja (hover, click) - Jasna hierarchia i organizacja elementów - Dostępność z klawiatury

Rodzaje menu rozwijanych: - Menu nawigacyjne z podmenu - Select (listy rozwijane formularzy) - Dropdown dla funkcji (filtry, sortowanie) - Customowe komponenty dropdown

Przykładowy kod HTML dla menu nawigacyjnego:

```
<nav class="main-nav">
  <ul class="menu">
    <li><a href="/">Strona główna</a></li>
    <li class="dropdown">
      <a href="/produkty" class="dropdown-toggle">Pro
dukty <span class="arrow">▼</span></a>
      <ul class="submenu">
        <li><a href="/produkty/kategoria1">Kategoria
1</a></li>
        <li><a href="/produkty/kategoria2">Kategoria
2</a></li>
        <li><a href="/produkty/kategoria3">Kategoria
3</a></li>
      </ul>
    </li>
    <li class="dropdown">
      <a href="/uslugi" class="dropdown-toggle">Usług
i <span class="arrow">▼</span></a>
      <ul class="submenu">
        <li><a href="/uslugi/usluga1">Usługa 1</a></l
i>
        <li><a href="/uslugi/usluga2">Usługa 2</a></l
i>
        <li><a href="/uslugi/usluga3">Usługa 3</a></l
i>
      </ul>
    </li>
    <li><a href="/o-nas">O nas</a></li>
    <li><a href="/kontakt">Kontakt</a></li>
  </ul>
</nav>
```

CSS dla menu rozwijanego:

```
/* Podstawowe style menu */
.main-nav {
  background-color: #333;
}
```

```
.menu {
  display: flex;
  list-style: none;
  margin: 0;
  padding: 0;
}

.menu li {
  position: relative;
}

.menu a {
  display: block;
  padding: 15px 20px;
  color: white;
  text-decoration: none;
  transition: background-color 0.3s;
}

.menu a:hover {
  background-color: #444;
}

/* Strzałka w menu rozwijanym */
.arrow {
  display: inline-block;
  margin-left: 5px;
  font-size: 10px;
  transition: transform 0.3s;
}

/* Style dla podmenu - kontynuacja */
.dropdown:hover .submenu,
.dropdown:focus-within .submenu {
  opacity: 1;
  visibility: visible;
  transform: translateY(0);
}
```

```

    transition: opacity 0.3s, transform 0.3s;
}

.dropdown:hover .arrow,
.dropdown:focus-within .arrow {
    transform: rotate(180deg);
}

.submenu a {
    padding: 10px 20px;
}

.submenu a:hover {
    background-color: #555;
}

/* Style responsywne dla menu */
@media (max-width: 768px) {
    .menu {
        flex-direction: column;
    }

    .submenu {
        position: static;
        display: none;
        background-color: #333;
        padding-left: 20px;
        transform: none;
        opacity: 1;
        visibility: visible;
    }

    .dropdown.active .submenu {
        display: block;
    }
}

```

JavaScript dla responsywnego menu rozwijanego:

```

document.addEventListener('DOMContentLoaded', function() {
    // Obsługa menu mobilnego
    const dropdowns = document.querySelectorAll('.dropdown');

    if (window.innerWidth <= 768) {
        dropdowns.forEach(dropdown => {
            const dropdownToggle = dropdown.querySelector('.dropdown-toggle');

            dropdownToggle.addEventListener('click', function(e) {
                // Zapobieganie przejściu do linku
                e.preventDefault();

                // Przetączanie klasy active
                dropdown.classList.toggle('active');

                // Zamykanie innych otwartych menu
                dropdowns.forEach(otherDropdown => {
                    if (otherDropdown !== dropdown) {
                        otherDropdown.classList.remove('active');
                    }
                });
            });
        });

        // Obsługa dostępności (accessibility)
        dropdowns.forEach(dropdown => {
            const dropdownToggle = dropdown.querySelector('.dropdown-toggle');
            const submenu = dropdown.querySelector('.submenu');

            // Dodawanie atrybutów ARIA
            dropdownToggle.setAttribute('aria-haspopup', 'true');
        });
    }
});

```



```

dropdownToggle.setAttribute('aria-expanded', 'false');
submenu.setAttribute('aria-label', 'Podmenu');

// Obsługa klawiszy
dropdownToggle.addEventListener('keydown', function(e) {
    if (e.key === 'Enter' || e.key === ' ') {
        e.preventDefault();
        dropdown.classList.toggle('active');
        dropdownToggle.setAttribute('aria-expanded',
dropdown.classList.contains('active'));
    }
});
});
});
});
});

```

Przykład customowego dropdown:

```

<div class="custom-dropdown">
  <button class="dropdown-button">Wybierz opcję <span
class="arrow">▼</span></button>
  <div class="dropdown-content">
    <a href="#" data-value="option1">Opcja 1</a>
    <a href="#" data-value="option2">Opcja 2</a>
    <a href="#" data-value="option3">Opcja 3</a>
    <a href="#" data-value="option4">Opcja 4</a>
  </div>
  <input type="hidden" name="selected-option" id="selected-option">
</div>

.custom-dropdown {
  position: relative;
  display: inline-block;
  width: 200px;
}

.dropdown-button {
  width: 100%;

```

```
padding: 10px 15px;
background-color: white;
border: 1px solid #ddd;
border-radius: 4px;
text-align: left;
cursor: pointer;
display: flex;
justify-content: space-between;
align-items: center;
}

.dropdown-content {
  display: none;
  position: absolute;
  top: 100%;
  left: 0;
  width: 100%;
  background-color: white;
  border: 1px solid #ddd;
  border-top: none;
  border-radius: 0 0 4px 4px;
  z-index: 100;
  box-shadow: 0 4px 8px rgba(0,0,0,0.1);
}

.dropdown-content a {
  display: block;
  padding: 10px 15px;
  text-decoration: none;
  color: #333;
}

.dropdown-content a:hover {
  background-color: #f5f5f5;
}

.custom-dropdown.active .dropdown-content {
  display: block;
}
```

```

.custom-dropdown.active .arrow {
  transform: rotate(180deg);
}

document.addEventListener('DOMContentLoaded', function() {
  const customDropdowns = document.querySelectorAll(
    '.custom-dropdown');

  customDropdowns.forEach(dropdown => {
    const button = dropdown.querySelector('.dropdown-
button');
    const content = dropdown.querySelector('.dropdown
-content');
    const options = dropdown.querySelectorAll('.dropd
own-content a');
    const input = dropdown.querySelector('input[type=
"hidden"]');

    // Otwieranie/zamykanie dropdown
    button.addEventListener('click', function() {
      dropdown.classList.toggle('active');
    });

    // Zamykanie dropdown po kliknięciu poza nim
    document.addEventListener('click', function(e) {
      if (!dropdown.contains(e.target)) {
        dropdown.classList.remove('active');
      }
    });

    // Wybór opcji
    options.forEach(option => {
      option.addEventListener('click', function(e) {
        e.preventDefault();

        const value = this.getAttribute('data-value')

```

```

;

```

```

const text = this.textContent;

button.textContent = text;
button.appendChild(document.createElement('span')).className = 'arrow';
button.querySelector('.arrow').textContent = '▼';

input.value = value;

dropdown.classList.remove('active');

// Zdarzenie zmiany dla formularzy
const event = new Event('change', { bubbles:
true });
input.dispatchEvent(event);
});
});
});
});
});

```

Dobre praktyki: - Zapewnij wyraźne wskaźniki dla menu zawierających podmenu - Dbaj o dostępność - menu powinno być obsługiwane z klawiatury - Zapewnij responsywność - mobilne menu często wymaga innego podejścia - Przechowuj stan rozwinięcia/zwinięcia na urządzeniach dotykowych - Zamykaj menu po kliknięciu poza nim - Używaj animacji dla płynnych przejść

Dlaczego są ważne dla początkujących: - Menu rozwijane pozwalają na efektywne wykorzystanie przestrzeni - Umożliwiają organizację dużej liczby linków w przejrzysty sposób - Stanowią wyzwanie techniczne łączące HTML, CSS i JavaScript

GALERIE

Galerie umożliwiają prezentację wielu obrazów, filmów lub prac w uporządkowany i interaktywny sposób.

Kluczowe cechy: - Organizacja wielu elementów wizualnych - Interaktywne przeglądanie i nawigacja - Różne układy i style prezentacji - Responsywność na różnych urządzeniach

Rodzaje galerii: - Siatka (grid) - Karuzela (slider) - Lightbox (powiększenie po kliknięciu) - Masonry (różne wysokości elementów) - Portfolio (z filtrowaniem)

Przykładowy kod HTML dla galerii siatki:

```
<div class="gallery">
  <figure class="gallery-item">
    <a href="images/large/image1.jpg" class="lightbox
">
      
    </a>
    <figcaption>Tytuł obrazu 1</figcaption>
  </figure>

  <figure class="gallery-item">
    <a href="images/large/image2.jpg" class="lightbox
">
      
    </a>
    <figcaption>Tytuł obrazu 2</figcaption>
  </figure>

  <figure class="gallery-item">
    <a href="images/large/image3.jpg" class="lightbox
">
      
    </a>
    <figcaption>Tytuł obrazu 3</figcaption>
  </figure>

  <figure class="gallery-item">
```

```

    <a href="images/large/image4.jpg" class="lightbox
">
    
    </a>
    <figcaption>Tytuł obrazu 4</figcaption>
</figure>

<!-- Więcej elementów galerii... -->
</div>

```

CSS dla galerii siatki:

```

.gallery {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(250
px, 1fr));
  grid-gap: 20px;
  margin: 30px 0;
}

.gallery-item {
  margin: 0;
  overflow: hidden;
  position: relative;
  border-radius: 4px;
  box-shadow: 0 2px 5px rgba(0,0,0,0.1);
  transition: transform 0.3s;
}

.gallery-item:hover {
  transform: translateY(-5px);
  box-shadow: 0 5px 15px rgba(0,0,0,0.2);
}

.gallery-item img {
  width: 100%;
  height: 200px;
  object-fit: cover;
  display: block;
}

```

```

    transition: transform 0.3s;
  }

  .gallery-item:hover img {
    transform: scale(1.05);
  }

  figcaption {
    padding: 10px;
    background-color: white;
    text-align: center;
    font-weight: 500;
  }

  /* Responsywność */
  @media (max-width: 768px) {
    .gallery {
      grid-template-columns: repeat(auto-fill, minmax(1
50px, 1fr));
      grid-gap: 10px;
    }

    .gallery-item img {
      height: 150px;
    }
  }
}

```

JavaScript dla prostego lightbox:

```

document.addEventListener('DOMContentLoaded', function() {
  // Tworzenie elementów lightbox
  const lightboxOverlay = document.createElement('div
');
  lightboxOverlay.className = 'lightbox-overlay';

  const lightboxContainer = document.createElement('d
iv');
  lightboxContainer.className = 'lightbox-container';

```

```
const lightboxImage = document.createElement('img')
;
lightboxImage.className = 'lightbox-image';

const lightboxCaption = document.createElement('div');
lightboxCaption.className = 'lightbox-caption';

const lightboxClose = document.createElement('button');
lightboxClose.className = 'lightbox-close';
lightboxClose.innerHTML = '&times;';

const lightboxNav = document.createElement('div');
lightboxNav.className = 'lightbox-nav';

const lightboxPrev = document.createElement('button');
lightboxPrev.className = 'lightbox-prev';
lightboxPrev.innerHTML = '&#10094;';

const lightboxNext = document.createElement('button');
lightboxNext.className = 'lightbox-next';
lightboxNext.innerHTML = '&#10095;';

// Układanie elementów
lightboxContainer.appendChild(lightboxImage);
lightboxContainer.appendChild(lightboxCaption);
lightboxContainer.appendChild(lightboxClose);

lightboxNav.appendChild(lightboxPrev);
lightboxNav.appendChild(lightboxNext);
lightboxContainer.appendChild(lightboxNav);

lightboxOverlay.appendChild(lightboxContainer);
document.body.appendChild(lightboxOverlay);

// Pobieranie wszystkich linków Lightbox
```



```

const lightboxLinks = document.querySelectorAll('a.
lightbox');
let currentIndex = 0;
const galleryItems = [];

// Zbieranie danych o elementach galerii
lightboxLinks.forEach((link, index) => {
  const item = {
    src: link.getAttribute('href'),
    caption: link.querySelector('img').getAttribute
('alt')
  };

  galleryItems.push(item);

  // Obsługa kliknięcia w link
  link.addEventListener('click', function(e) {
    e.preventDefault();
    openLightbox(index);
  });
});

// Funkcja otwierająca lightbox
function openLightbox(index) {
  currentIndex = index;
  updateLightboxContent();
  lightboxOverlay.style.display = 'flex';

  // Blokowanie scrollowania strony
  document.body.style.overflow = 'hidden';
}

// Funkcja zamykająca lightbox
function closeLightbox() {
  lightboxOverlay.style.display = 'none';

  // Przywracanie scrollowania
  document.body.style.overflow = 'auto';
}

```

```

// Funkcja aktualizująca zawartość lightbox
function updateLightboxContent() {
    const item = galleryItems[currentIndex];

    lightboxImage.src = item.src;
    lightboxCaption.textContent = item.caption;

    // Aktualizacja stanu przycisków nawigacji
    if (currentIndex === 0) {
        lightboxPrev.classList.add('disabled');
    } else {
        lightboxPrev.classList.remove('disabled');
    }

    if (currentIndex === galleryItems.length - 1) {
        lightboxNext.classList.add('disabled');
    } else {
        lightboxNext.classList.remove('disabled');
    }
}

// Obsługa nawigacji
lightboxPrev.addEventListener('click', function() {
    if (currentIndex > 0) {
        currentIndex--;
        updateLightboxContent();
    }
});

lightboxNext.addEventListener('click', function() {
    if (currentIndex < galleryItems.length - 1) {
        currentIndex++;
        updateLightboxContent();
    }
});

// Zamykanie lightbox
lightboxClose.addEventListener('click', closeLightb

```

```

ox);
lightboxOverlay.addEventListener('click', function(
e) {
    if (e.target === lightboxOverlay) {
        closeLightbox();
    }
});

// Obsługa klawiszy
document.addEventListener('keydown', function(e) {
    if (lightboxOverlay.style.display === 'flex') {
        if (e.key === 'Escape') {
            closeLightbox();
        } else if (e.key === 'ArrowLeft' && currentInde
x > 0) {
            currentIndex--;
            updateLightboxContent();
        } else if (e.key === 'ArrowRight' && currentInd
ex < galleryItems.length - 1) {
            currentIndex++;
            updateLightboxContent();
        }
    }
});
});
});

```

CSS dla lightbox:

```

.lightbox-overlay {
    display: none;
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0,0,0,0.9);
    z-index: 1000;
    justify-content: center;
    align-items: center;

```

```

}

```

```
.lightbox-container {  
  position: relative;  
  max-width: 80%;  
  max-height: 80%;  
}  
  
.lightbox-image {  
  max-width: 100%;  
  max-height: 80vh;  
  display: block;  
  margin: 0 auto;  
}  
  
.lightbox-caption {  
  color: white;  
  text-align: center;  
  padding: 10px 0;  
  font-size: 16px;  
}  
  
.lightbox-close {  
  position: absolute;  
  top: -40px;  
  right: 0;  
  font-size: 30px;  
  color: white;  
  background: transparent;  
  border: none;  
  cursor: pointer;  
}  
  
.lightbox-nav {  
  position: absolute;  
  top: 50%;  
  width: 100%;  
  transform: translateY(-50%);  
  display: flex;  
  justify-content: space-between;
```

```

}

.lightbox-prev,
.lightbox-next {
  background: rgba(0,0,0,0.5);
  color: white;
  font-size: 24px;
  border: none;
  padding: 15px;
  cursor: pointer;
  transition: background-color 0.3s;
}

.lightbox-prev {
  margin-left: -50px;
}

.lightbox-next {
  margin-right: -50px;
}

.lightbox-prev:hover,
.lightbox-next:hover {
  background: rgba(0,0,0,0.8);
}

.lightbox-prev.disabled,
.lightbox-next.disabled {
  opacity: 0.3;
  cursor: not-allowed;
}

/* Responsywność Lightbox */
@media (max-width: 768px) {
  .lightbox-container {
    max-width: 95%;
  }

  .lightbox-prev,

```

```

.lightbox-next {
  position: absolute;
  padding: 10px;
}

.lightbox-prev {
  left: 0;
  margin-left: 0;
}

.lightbox-next {
  right: 0;
  margin-right: 0;
}
}

```

Przykład dla karuzeli/slidera:

```

<div class="carousel">
  <div class="carousel-container">
    <div class="carousel-slide">
      
    </div>
    <div class="carousel-slide">
      
    </div>
    <div class="carousel-slide">
      
    </div>
    <div class="carousel-slide">
      
    </div>
  </div>

  <button class="carousel-prev">&#10094;</button>
  <button class="carousel-next">&#10095;</button>

  <div class="carousel-dots">
    <span class="dot active" data-slide="0"></span>
    <span class="dot" data-slide="1"></span>
  </div>
</div>

```

```
        <span class="dot" data-slide="2"></span>
        <span class="dot" data-slide="3"></span>
    </div>
</div>
```

```
.carousel {
    position: relative;
    max-width: 800px;
    margin: 0 auto;
    overflow: hidden;
}
```

```
.carousel-container {
    display: flex;
    transition: transform 0.5s ease;
}
```

```
.carousel-slide {
    min-width: 100%;
    height: 400px;
}
```

```
.carousel-slide img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}
```

```
.carousel-prev,
.carousel-next {
    position: absolute;
    top: 50%;
    transform: translateY(-50%);
    background: rgba(0,0,0,0.5);
    color: white;
    border: none;
    padding: 15px;
    cursor: pointer;
    z-index: 10;
}
```

```
}

.carousel-prev {
  left: 10px;
}

.carousel-next {
  right: 10px;
}

.carousel-dots {
  position: absolute;
  bottom: 20px;
  left: 0;
  right: 0;
  display: flex;
  justify-content: center;
  gap: 10px;
}

.dot {
  width: 10px;
  height: 10px;
  background-color: rgba(255,255,255,0.5);
  border-radius: 50%;
  cursor: pointer;
  transition: background-color 0.3s;
}

.dot.active {
  background-color: white;
}

document.addEventListener('DOMContentLoaded', function() {
  const carousel = document.querySelector('.carousel');
  const container = carousel.querySelector('.carousel-
    container');
```



```

const slides = carousel.querySelectorAll('.carousel
-slide');
const prevBtn = carousel.querySelector('.carousel-p
rev');
const nextBtn = carousel.querySelector('.carousel-n
ext');
const dots = carousel.querySelectorAll('.dot');

let currentIndex = 0;
const slideCount = slides.length;

// Funkcja zmieniająca slajd
function goToSlide(index) {
  if (index < 0) {
    index = slideCount - 1;
  } else if (index >= slideCount) {
    index = 0;
  }

  currentIndex = index;
  const translateX = -currentIndex * 100 + '%';
  container.style.transform = `translateX(${transla
teX})`;

  // Aktualizacja kropek
  dots.forEach((dot, idx) => {
    dot.classList.toggle('active', idx === currentI
ndex);
  });
}

// Obsługa przycisków
prevBtn.addEventListener('click', () => {
  goToSlide(currentIndex - 1);
});

nextBtn.addEventListener('click', () => {
  goToSlide(currentIndex + 1);
});

```

```

// Obsługa kropek
dots.forEach((dot, index) => {
  dot.addEventListener('click', () => {
    goToSlide(index);
  });
});

// Automatyczne przewijanie (opcjonalne)
let interval;

function startAutoSlide() {
  interval = setInterval(() => {
    goToSlide(currentIndex + 1);
  }, 5000); // Zmiana co 5 sekund
}

function stopAutoSlide() {
  clearInterval(interval);
}

// Zatrzymanie automatycznego przewijania przy interakcji
carousel.addEventListener('mouseenter', stopAutoSlide);
carousel.addEventListener('mouseleave', startAutoSlide);

// Uruchomienie automatycznego przewijania
startAutoSlide();

// Obsługa swipe na urządzeniach mobilnych
let touchStartX = 0;
let touchEndX = 0;

carousel.addEventListener('touchstart', (e) => {
  touchStartX = e.changedTouches[0].screenX;
});

```

```

carousel.addEventListener('touchend', (e) => {
  touchEndX = e.changedTouches[0].screenX;
  handleSwipe();
});

function handleSwipe() {
  if (touchEndX < touchStartX) {
    // Swipe w lewo - następny slajd
    goToSlide(currentIndex + 1);
  } else if (touchEndX > touchStartX) {
    // Swipe w prawo - poprzedni slajd
    goToSlide(currentIndex - 1);
  }
}
});

```

Dobre praktyki: - Używaj miniatur dla lepszej wydajności ładowania - Zapewnij responsywność galerii na różnych urządzeniach - Wdrażaj lazy loading dla poprawy wydajności - Dodawaj opcje filtrowania i sortowania dla dużych kolekcji - Zapewnij sterowanie z klawiatury i dotyku

Dlaczego są ważne dla początkujących: - Pozwalają na efektywną prezentację treści wizualnych - Łączą różne technologie webowe (HTML, CSS, JavaScript) - Są często wymagany elementem portfolio, stron fotograficznych i e-commerce

5. Elementy techniczne (niewidoczne dla użytkownika)

META TAGI

Meta tagi są niewidocznymi elementami HTML umieszczanymi w sekcji `<head>` dokumentu, które dostarczają informacji o stronie dla przeglądarek, wyszukiwarek i innych serwisów.

Kluczowe cechy: - Dostarczają metadanych o stronie - Wpływają na SEO (Search Engine Optimization) - Kontrolują wyświetlanie

podglądu strony w mediach społecznościowych - Definiują sposób, w jaki strona jest renderowana przez przeglądarki

Najważniejsze meta tagi:

```
<!DOCTYPE html>
<html lang="pl">
<head>
  <!-- Podstawowe meta tagi -->
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge
">

  <!-- Tytuł strony (wyświetlany na karcie przeglądar
ki) -->
  <title>Tytuł mojej strony | Moja firma</title>

  <!-- Meta tagi SEO -->
  <meta name="description" content="Krótki, zwięzły o
pis strony (do 160 znaków) zawierający kluczowe słowa
.">
  <meta name="keywords" content="słowo kluczowe 1, sł
owo kluczowe 2, fraza kluczowa">
  <meta name="author" content="Nazwa firmy lub autora
">
  <meta name="robots" content="index, follow">

  <!-- Open Graph dla mediów społecznościowych (Faceb
ook, LinkedIn) -->
  <meta property="og:type" content="website">
  <meta property="og:url" content="https://www.przysl
ad.pl/">
  <meta property="og:title" content="Tytuł dla mediów
społecznościowych">
  <meta property="og:description" content="Opis dla m
ediów społecznościowych (może różnić się od meta desc
ription).">
```

```

<meta property="og:image" content="https://www.przyklad.pl/obrazek-social.jpg">

<!-- Twitter Card -->
<meta name="twitter:card" content="summary_large_image">
<meta name="twitter:site" content="@nazwa uzytkownika">
<meta name="twitter:title" content="Tytuł dla Twittera">
<meta name="twitter:description" content="Opis dla Twittera (może różnić się od meta description).">
<meta name="twitter:image" content="https://www.przyklad.pl/obrazek-twitter.jpg">

<!-- Dodatkowe meta tagi -->
<meta name="theme-color" content="#4285f4"> <!-- Kolor motywu dla Chrome na Androidzie -->
<meta name="apple-mobile-web-app-capable" content="yes"> <!-- App mode dla iOS -->
<meta name="format-detection" content="telephone=no"> <!-- Zapobieganie formatowaniu numerów telefonu -->

<!-- Kanoniczny URL (zapobieganie duplicate content) -->
<link rel="canonical" href="https://www.przyklad.pl/strona">

<!-- Favicon -->
<link rel="icon" href="/favicon.ico">
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon.png">
<link rel="icon" type="image/png" sizes="32x32" href="/favicon-32x32.png">
<link rel="icon" type="image/png" sizes="16x16" href="/favicon-16x16.png">
<link rel="manifest" href="/site.webmanifest">
</head>

```

Wyjaśnienie najważniejszych meta tagów:

1. `<meta charset="UTF-8">`
 - Definiuje kodowanie znaków używane na stronie (UTF-8 obsługuje wszystkie języki i znaki)
2. `<meta name="viewport">`
 - Kontroluje, jak strona jest wyświetlana na urządzeniach mobilnych
 - `width=device-width` dostosowuje szerokość strony do szerokości urządzenia
 - `initial-scale=1.0` ustawia początkowy poziom przybliżenia
3. `<meta name="description">`
 - Krótki opis strony (150-160 znaków)
 - Wyświetlany w wynikach wyszukiwania pod tytułem strony
 - Powinien zawierać słowa kluczowe i zachęcać do kliknięcia
4. `<meta name="keywords">`
 - Lista słów kluczowych związanych ze stroną
 - Obecnie ma mniejsze znaczenie dla SEO, ale wciąż używany
5. `<meta name="robots">`
 - Instrukcje dla robotów wyszukiwarek
 - `index` - zezwól na indeksowanie strony
 - `follow` - zezwól na śledzenie linków
 - `noindex` - nie indeksuj strony
 - `nofollow` - nie śledź linków
6. **Open Graph Meta Tagi:**
 - Kontrolują, jak strona jest prezentowana podczas udostępniania w mediach społecznościowych
 - `og:title` - tytuł udostępnianego linka

- **og:description** - opis udostępnianego linka
- **og:image** - obraz wyświetlany przy udostępnianiu (zalecany rozmiar: 1200x630px)
- **og:url** - kanoniczny URL udostępnianej strony

7. Twitter Card Meta Tagi:

- Podobne do Open Graph, ale specyficzne dla Twittera
- **twitter:card** - typ karty (summary, summary_large_image, app, player)
- **twitter:site** - oficjalny identyfikator Twitter dla strony

8. <link rel="canonical">

- Wskazuje preferowaną wersję strony, gdy istnieje wiele podobnych wersji
- Pomaga zapobiegać problemom z duplikacją treści
- Pomaga przeglądarkom zrozumieć, który URL jest "oficjalny"

8. Favicon

- Małe ikony wyświetlane w zakładkach przeglądarki i zakładkach
- Różne wersje dla różnych kontekstów (browser, mobile, etc.)
- Zazwyczaj zestawy ikon w różnych rozmiarach

Dobra praktyki dla meta tagów: - Twórz unikalne, opisowe tytuły (idealnie 50-60 znaków) - Pisz atrakcyjne i zwięzłe meta opisy (150-160 znaków) - Używaj właściwych obrazów dla Open Graph i Twitter Cards - Stosuj poprawne kodowanie znaków (UTF-8) - Aktualizuj meta tagi dla każdej podstrony - Używaj narzędzi walidacyjnych, takich jak Facebook Sharing Debugger czy Twitter Card Validator - Zawsze dodawaj tag viewport dla responsywności

Dlaczego są ważne dla początkujących: - Znacząco wpływają na SEO i widoczność w wyszukiwarkach - Determinują sposób, w jaki strona jest prezentowana w mediach społecznościowych - Są istotnym elementem technicznym, mimo że nie są widoczne dla użytkowników - Prawidłowe meta tagi mogą zwiększyć CTR (współczynnik klikalności) w wynikach wyszukiwania

SKRYPTY

Skrypty są blokami kodu, najczęściej JavaScript, które dodają interaktywność i dynamikę do stron internetowych, wykonując się w tle.

Kluczowe cechy: - Dodają interaktywność i funkcjonalność do strony - Mogą być osadzone bezpośrednio w HTML lub w zewnętrznych plikach - Mogą ładować się asynchronicznie lub synchronicznie - Wpływają na wydajność i czas ładowania strony

Sposoby dodawania skryptów:

```
<!-- 1. Bezpośrednio w HTML (inline) -->
<script>
  document.addEventListener('DOMContentLoaded', funct
ion() {
    console.log('Strona załadowana!');
    // Kod JavaScript
  });
</script>
```

```
<!-- 2. Z zewnętrznego pliku - standardowo -->
<script src="script.js"></script>
```

```
<!-- 3. Z atrybutem async - ładowanie asynchroniczne -->
<script src="analytics.js" async></script>
```

```
<!-- 4. Z atrybutem defer - opóźnione wykonanie -->
<script src="secondary.js" defer></script>
```



```
<!-- 5. Z określonym typem (dla starszych przeglądarek) -->
<script type="text/javascript" src="compatibility.js"></script>

<!-- 6. Skrypty modułowe (ES modules) -->
<script type="module" src="module.js"></script>

<!-- 7. Z atrybutami integrity i crossorigin dla bezpieczeństwa -->
<script
  src="https://cdn.example.com/library.js"
  integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/
  uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"
  crossorigin="anonymous">
</script>
```

Wyjaśnienie atrybutów:

1. `async`

- Skrypt ładuje się asynchronicznie (nie blokuje parsowania HTML)
- Wykonuje się natychmiast po załadowaniu, może przerwać parsowanie HTML
- Dobry dla skryptów niezależnych od DOM i innych skryptów (np. analytics)

2. `defer`

- Skrypt ładuje się asynchronicznie (nie blokuje parsowania HTML)
- Wykonuje się dopiero po zakończeniu parsowania HTML, ale przed zdarzeniem DOMContentLoaded
- Zachowuje kolejność wykonania wielu skryptów defer
- Zalecany dla większości skryptów, które działają na DOM

3. `type="module"`

- Traktuje skrypt jako moduł ES6
- Automatycznie stosuje `defer` (nie blokuje parsowania)
- Pozwala na używanie `import` i `export`
- Działa w “strict mode”

4. `integrity`

- Zapewnia integralność skryptu (zabezpieczenie przed modyfikacją)
- Zawiera skrót kryptograficzny zawartości pliku
- Przeglądarka sprawdza, czy pobrany plik zgadza się ze skrótem

5. `crossorigin`

- Określa, jak przeglądarka ma obsługiwać żądania CORS (Cross-Origin Resource Sharing)
- Wartości: “anonymous” lub “use-credentials”

Popularne biblioteki i ich dodawanie:

```
<!-- jQuery -->
<script src="https://code.jquery.com/jquery-3.7.0.min.js"></script>
```

```
<!-- Bootstrap (CSS i JS) -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
```

```
<!-- Font Awesome (ikony) -->
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/@fortawesome/fontawesome-free@6.4.0/css/all.min.css">
```

```
<!-- Google Analytics -->
<script async src="https://www.googletagmanager.com/gtag/js?id=G-XXXXXXXXXX"></script>
```

```

<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());
  gtag('config', 'G-XXXXXXXXXX');
</script>

```

Dobre praktyki: - Umieszczaj skrypty przed zamknięciem znacznika `</body>` lub używaj `defer` - Łącz i minifikuj skrypty w środowisku produkcyjnym - Używaj asynchronicznego ładowania dla skryptów niekrytycznych - Stosuj lazy loading dla skryptów ładowanych na żądanie - Redukuj liczbę zewnętrznych zależności - Testuj wydajność ładowania skryptów - Używaj nowoczesnych formatów (moduły ES6) gdy to możliwe - Dodawaj komentarze wyjaśniające przeznaczenie skryptów

Przykład organizacji skryptów:

```

<!DOCTYPE html>
<html lang="pl">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Moja strona</title>

  <!-- Krytyczne skrypty potrzebne od razu -->
  <script src="critical.js"></script>

  <!-- Preload ważnych skryptów -->
  <link rel="preload" href="important.js" as="script"
>
</head>
<body>
  <!-- Zawartość strony -->

  <!-- Główne skrypty aplikacji - defer dla lepszej w
ydajności -->
  <script src="main.js" defer></script>

```

```

<!-- Biblioteki zewnętrzne -->
<script src="vendor/library.js" defer></script>

<!-- Niekrytyczne skrypty analityczne -->
<script src="analytics.js" async></script>

<!-- Skrypty ładowane warunkowo -->
<script>
  if (window.matchMedia('(min-width: 768px)').match
es) {
    // ładuj skrypty tylko dla desktopa
    const desktopScript = document.createElement('s
cript');
    desktopScript.src = 'desktop-features.js';
    document.body.appendChild(desktopScript);
  }
</script>
</body>
</html>

```

Skrypt dla lazy loading komponentów:

```

// Przykład Lazy Loading dla ciężkich komponentów
document.addEventListener('DOMContentLoaded', functio
n() {
  // Funkcja sprawdzająca czy element jest w widoku
  function isElementInViewport(el) {
    const rect = el.getBoundingClientRect();
    return (
      rect.top >= 0 &&
      rect.left >= 0 &&
      rect.bottom <= (window.innerHeight || document.
documentElement.clientHeight) &&
      rect.right <= (window.innerWidth || document.do
cumentElement.clientWidth)
    );
  }

  // Elementy do lazy loadingu

```

```

const lazyComponents = document.querySelectorAll('[
data-lazy]');

// Funkcja ładująca komponenty
function loadLazyComponents() {
    lazyComponents.forEach(component => {
        if (isElementInViewport(component) && !componen
t.dataset.loaded) {
            const componentType = component.dataset.lazy;

            switch(componentType) {
                case 'map':
                    loadMap(component);
                    break;
                case 'video':
                    loadVideo(component);
                    break;
                case 'comments':
                    loadComments(component);
                    break;
                // Inne typy komponentów...
            }

            component.dataset.loaded = 'true';
        }
    });
}

// Przykładowa funkcja ładująca mapę
function loadMap(container) {
    const mapScript = document.createElement('script'
);
    mapScript.src = 'https://maps.googleapis.com/maps
/api/js?key=YOUR_API_KEY&callback=initMap';
    document.body.appendChild(mapScript);

    window.initMap = function() {
        const map = new google.maps.Map(container, {
            center: { lat: parseFloat(container.dataset.l

```

```

at), lng: parseFloat(container.dataset.lng) },
    zoom: 12
  });

  // Dodatkowa konfiguracja mapy...
};

// Inne funkcje ładujące komponenty...

// Nastuchiwanie na scroll i resize
window.addEventListener('scroll', loadLazyComponent
s);
window.addEventListener('resize', loadLazyComponent
s);

// Inicjalne sprawdzenie
loadLazyComponents();
});

```

Dlaczego są ważne dla początkujących: - Umożliwiają tworzenie interaktywnych stron i aplikacji webowych - Wpływają znacząco na wydajność i czas ładowania strony - Prawidłowe zarządzanie skryptami jest kluczowe dla optymalizacji - Stanowią podstawę nowoczesnego tworzenia stron internetowych

PLIKI COOKIE

Pliki cookie to małe pliki tekstowe przechowujące dane, które przeglądarka zapisuje na urządzeniu użytkownika, umożliwiając stronom “zapamiętywanie” informacji między wizytami.

Kluczowe cechy: - Przechowują dane sesji, preferencje użytkownika, dane analityczne - Mają określony czas życia (tymczasowe lub trwałe) - Podlegają regulacjom prawnym (np. RODO/GDPR, ePrivacy) - Mogą być tworzone i zarządzane przez JavaScript lub serwer

Rodzaje plików cookie: - **Sesyjne** - usuwane po zamknięciu przeglądarki - **Trwałe** - przechowywane przez określony czas (dni, miesiące) - **Własne** - pochodzące z odwiedzanej domeny - **Zewnętrzne (third-party)** - pochodzące z innych domen (np. analytics, reklamy) - **Niezbędne** - konieczne do działania strony - **Preferencyjne** - zapamiętujące ustawienia użytkownika - **Statystyczne** - zbierające dane o użytkowaniu strony - **Marketingowe** - śledzące użytkownika w celach reklamowych

JavaScript do zarządzania plikami cookie:

```
// Ustawienie pliku cookie
function setCookie(name, value, days) {
    let expires = '';
    if (days) {
        const date = new Date();
        date.setTime(date.getTime() + (days * 24 * 60 * 60 * 1000));
        expires = '; expires=' + date.toUTCString();
    }
    document.cookie = name + '=' + (value || '') + expires + '; path=/; SameSite=Strict';
}
```

```
// Pobieranie wartości pliku cookie
function getCookie(name) {
    const nameEQ = name + '=';
    const ca = document.cookie.split(';');
    for (let i = 0; i < ca.length; i++) {
        let c = ca[i];
        while (c.charAt(0) === ' ') c = c.substring(1, c.length);
        if (c.indexOf(nameEQ) === 0) return c.substring(nameEQ.length, c.length);
    }
    return null;
}
```

```
// Usuwanie pliku cookie
```

```
function eraseCookie(name) {
    document.cookie = name + '=; Max-Age=-99999999; path=/'
}
```

// Przykłady użycia

```
setCookie('preferences', JSON.stringify({ theme: 'dark',
fontSize: 'large' })), 30); // 30 dni
const preferences = JSON.parse(getCookie('preferences'));
eraseCookie('temporary_data');
```

Banner informujący o plikach cookie (zgodny z RODO):

```
<div id="cookie-banner" class="cookie-banner">
  <div class="cookie-content">
    <h3>Używamy plików cookie</h3>
    <p>
      Nasza strona używa plików cookie w celu zapewnienia
      poprawnego działania, personalizacji treści,
      analizy ruchu i celów marketingowych. Możesz za
      rządzić swoimi preferencjami dotyczącymi plików cookie.
    </p>
    <div class="cookie-buttons">
      <button id="cookie-accept-all" class="btn btn-primary">Akceptuj wszystkie</button>
      <button id="cookie-settings" class="btn btn-secondary">Ustawienia</button>
      <button id="cookie-reject-all" class="btn btn-outline">Odrzuć wszystkie oprócz niezbędnych</button>
    </div>
  </div>
</div>

<div id="cookie-settings-panel" class="cookie-setting
s-panel" style="display: none;">
  <div class="settings-content">
    <h3>Ustawienia plików cookie</h3>
    <div class="cookie-options">
```



```

<div class="cookie-option">
  <input type="checkbox" id="essential-cookies"
checked disabled>
  <label for="essential-cookies">Niezbędne (wym
agane)</label>
  <p>Są konieczne do działania podstawowych fun
kcji strony.</p>
</div>

```

```

<div class="cookie-option">
  <input type="checkbox" id="preferences-cookie
s" checked>
  <label for="preferences-cookies">Preferencyjn
e</label>
  <p>Zapamiętują Twoje preferencje, takie jak w
ybór języka czy ustawienia wyglądu.</p>
</div>

```

```

<div class="cookie-option">
  <input type="checkbox" id="analytics-cookies"
checked>
  <label for="analytics-cookies">Analityczne</l
abel>
  <p>Pomagają nam zrozumieć, jak użytkownicy ko
rzystają z naszej strony.</p>
</div>

```

```

<div class="cookie-option">
  <input type="checkbox" id="marketing-cookies"
>
  <label for="marketing-cookies">Marketingowe</
label>
  <p>Umożliwiają wyświetlanie spersonalizowanyc
h reklam.</p>
</div>
</div>

```

```

<div class="settings-buttons">
  <button id="save-preferences" class="btn btn-pr

```

```

imary">Zapisz preferencje</button>
  <button id="close-settings" class="btn btn-seco
ndary">Zamknij</button>
</div>
</div>
</div>
</div>

.cookie-banner {
  position: fixed;
  bottom: 0;
  left: 0;
  right: 0;
  background-color: white;
  box-shadow: 0 -2px 10px rgba(0,0,0,0.1);
  z-index: 1000;
  padding: 20px;
}

.cookie-content {
  max-width: 1200px;
  margin: 0 auto;
}

.cookie-buttons {
  margin-top: 15px;
  display: flex;
  gap: 10px;
  flex-wrap: wrap;
}

.cookie-settings-panel {
  position: fixed;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: rgba(0,0,0,0.5);
  z-index: 1001;
  display: flex;

```

```
    align-items: center;
    justify-content: center;
}

.settings-content {
    background-color: white;
    border-radius: 8px;
    max-width: 600px;
    width: 100%;
    padding: 30px;
    max-height: 80vh;
    overflow-y: auto;
}

.cookie-options {
    margin: 20px 0;
}

.cookie-option {
    margin-bottom: 15px;
    padding-bottom: 15px;
    border-bottom: 1px solid #eee;
}

.cookie-option p {
    margin-top: 5px;
    color: #666;
    font-size: 0.9em;
}

.settings-buttons {
    display: flex;
    justify-content: space-between;
    margin-top: 20px;
}

.btn {
    padding: 10px 20px;
    border-radius: 4px;
}
```

```
    cursor: pointer;
    font-weight: 500;
    border: none;
}

.btn-primary {
    background-color: #4CAF50;
    color: white;
}

.btn-secondary {
    background-color: #f1f1f1;
    color: #333;
}

.btn-outline {
    background-color: transparent;
    border: 1px solid #ddd;
    color: #333;
}

document.addEventListener('DOMContentLoaded', function() {
    const cookieBanner = document.getElementById('cookie-banner');
    const cookieSettingsPanel = document.getElementById('cookie-settings-panel');
    const acceptAllButton = document.getElementById('cookie-accept-all');
    const settingsButton = document.getElementById('cookie-settings');
    const rejectAllButton = document.getElementById('cookie-reject-all');
    const savePreferencesButton = document.getElementById('save-preferences');
    const closeSettingsButton = document.getElementById('close-settings');

    // Checkboxy
```

```

const preferencesCheckbox = document.getElementById(
  'preferences-cookies');
const analyticsCheckbox = document.getElementById(
  'analytics-cookies');
const marketingCheckbox = document.getElementById(
  'marketing-cookies');

// Sprawdzenie, czy użytkownik już dokonał wyboru
const cookieConsent = getCookie('cookie_consent');

if (!cookieConsent) {
  cookieBanner.style.display = 'block';
} else {
  // Zastosowanie zapisanych preferencji
  applyStoredPreferences(JSON.parse(cookieConsent))
;
}

// Obsługa przycisków
acceptAllButton.addEventListener('click', function(
) {
  const preferences = {
    essential: true,
    preferences: true,
    analytics: true,
    marketing: true
  };

  savePreferences(preferences);
  cookieBanner.style.display = 'none';
});

settingsButton.addEventListener('click', function()
{
  cookieSettingsPanel.style.display = 'flex';
});

rejectAllButton.addEventListener('click', function(
) {

```

```

const preferences = {
  essential: true,
  preferences: false,
  analytics: false,
  marketing: false
};

// Aktualizacja checkboxów w panelu ustawień
preferencesCheckbox.checked = false;
analyticsCheckbox.checked = false;
marketingCheckbox.checked = false;

savePreferences(preferences);
cookieBanner.style.display = 'none';
});

savePreferencesButton.addEventListener('click', function() {
  const preferences = {
    essential: true, // Zawsze włączone
    preferences: preferencesCheckbox.checked,
    analytics: analyticsCheckbox.checked,
    marketing: marketingCheckbox.checked
  };

  savePreferences(preferences);
  cookieSettingsPanel.style.display = 'none';
  cookieBanner.style.display = 'none';
});

closeSettingsButton.addEventListener('click', function() {
  cookieSettingsPanel.style.display = 'none';
});

// Funkcja zapisująca preferencje
function savePreferences(preferences) {
  setCookie('cookie_consent', JSON.stringify(preferences), 365); // 1 rok

```

```

    applyPreferences(preferences);
}

// Zastosowanie preferencji do skryptów strony
function applyPreferences(preferences) {
    // Przykład: aktywowanie skryptów analitycznych,
    // jeśli dozwolone
    if (preferences.analytics) {
        enableAnalytics();
    }

    // Przykład: aktywowanie skryptów marketingowych,
    // jeśli dozwolone
    if (preferences.marketing) {
        enableMarketing();
    }
}

// Zastosowanie zapisanych preferencji
function applyStoredPreferences(preferences) {
    preferencesCheckbox.checked = preferences.preferences;
    analyticsCheckbox.checked = preferences.analytics;
;
    marketingCheckbox.checked = preferences.marketing;
;

    applyPreferences(preferences);
}

// Funkcja aktywująca skrypty analityczne
function enableAnalytics() {
    // Przykład: Google Analytics
    const analyticsScript = document.createElement('script');
    analyticsScript.src = 'https://www.googletagmanager.com/gtag/js?id=G-XXXXXXXXXX';
    analyticsScript.async = true;
    document.head.appendChild(analyticsScript);
}

```

```

window.dataLayer = window.dataLayer || [];
function gtag(){dataLayer.push(arguments);}
gtag('js', new Date());
gtag('config', 'G-XXXXXXXXXX');
}

// Funkcja aktywująca skrypty marketingowe
function enableMarketing() {
  // Przykład: Facebook Pixel
  !function(f,b,e,v,n,t,s)
  {if(f.fbq)return;n=f.fbq=function(){n.callMethod?
  n.callMethod.apply(n,arguments):n.queue.push(aru
ments)}};
  if(!f._fbq)f._fbq=n;n.push=n;n.loaded=!0;n.version=
n='2.0';
  n.queue=[];t=b.createElement(e);t.async=!0;
  t.src=v;s=b.getElementsByTagName(e)[0];
  s.parentNode.insertBefore(t,s)}(window, document,
'script',
'https://connect.facebook.net/en_US/fbevents.js')
;
  fbq('init', 'XXXXXXXXXXXXX');
  fbq('track', 'PageView');
}
});

```

Dobre praktyki dotyczące plików cookie: - Informuj użytkowników o używaniu plików cookie (wymóg prawny w UE) - Uzyskaj wyraźną zgodę przed ustawianiem nieistotnych plików cookie - Umożliw użytkownikom zarządzanie preferencjami cookie - Przechowuj minimalną ilość danych potrzebnych do działania - Ustaw odpowiednią datę wygaśnięcia (nie przechowuj dłużej niż potrzeba) - Zabezpiecz pliki cookie przed nieupoważnionym dostępem - Skonsultuj się z prawnikiem w zakresie zgodności z regulacjami (RODO/GDPR)

Dlaczego są ważne dla początkujących: - Pliki cookie są fundamentem personalizacji i śledzenia aktywności użytkowników - Prawidłowe zarządzanie plikami cookie jest kluczowe dla zgodności z przepisami - Zrozumienie działania plików cookie pomaga w implementacji funkcji pamiętających stan - Są niezbędne w e-commerce, systemach logowania i wielu innych funkcjonalnościach

API (APPLICATION PROGRAMMING INTERFACE)

API to interfejsy programistyczne umożliwiające komunikację między różnymi systemami i aplikacjami, pozwalające na integrację zewnętrznych usług ze stroną internetową.

Kluczowe cechy: - Umożliwiają dostęp do funkcji i danych zewnętrznych serwisów - Działają według określonych protokołów i formatów danych - Mogą być publiczne (otwarte dla wszystkich) lub prywatne (wymagające autoryzacji) - Pozwalają na rozszerzenie funkcjonalności strony bez tworzenia wszystkiego od podstaw

Popularne typy API w kontekście stron internetowych: - REST

API - najpopularniejszy typ API wykorzystujący standardowe metody HTTP - **GraphQL** - elastyczne API pozwalające klientowi precyzyjnie określić żądane dane - **WebSockets** - umożliwiają dwukierunkową komunikację w czasie rzeczywistym - **SOAP** - protokół dostępu do usług webowych, bardziej złożony i formalny - **APIs przeglądarek** - np. Geolocation API, Web Storage API, Fetch API

Przykład użycia REST API (Fetch):

```
// Pobieranie danych z API
function fetchData() {
  // Pokazanie loadera
  document.getElementById('loader').style.display = 'block';
```

```

fetch('https://api.example.com/data')
  .then(response => {
    // Sprawdzenie, czy odpowiedź jest OK
    if (!response.ok) {
      throw new Error('Network response was not ok:
' + response.status);
    }
    return response.json();
  })
  .then(data => {
    // Przetwarzanie danych
    displayData(data);
  })
  .catch(error => {
    // Obsługa błędów
    console.error('Error fetching data:', error);
    document.getElementById('error-message').textCo
ntent = 'Nie udało się pobrać danych. Spróbuj ponowni
e później.';
    document.getElementById('error-message').style.
display = 'block';
  })
  .finally(() => {
    // Ukrycie loadera niezależnie od wyniku
    document.getElementById('loader').style.display
= 'none';
  });
}

// Wyświetlanie danych na stronie
function displayData(data) {
  const container = document.getElementById('data-con
tainer');
  container.innerHTML = '';

  if (data.length === 0) {
    container.innerHTML = '<p>Brak danych do wyświetl
enia.</p>';
    return;
  }

```

```

    }

    data.forEach(item => {
      const element = document.createElement('div');
      element.className = 'data-item';
      element.innerHTML = `
        <h3>${item.title}</h3>
        <p>${item.description}</p>
        <span class="date">${new Date(item.timestamp).toLocaleDateString()}</span>
      `;
      container.appendChild(element);
    });
  }

  // Wywołanie funkcji po załadowaniu strony
  document.addEventListener('DOMContentLoaded', fetchData);

  // Dodanie odświeżania danych
  document.getElementById('refresh-button').addEventListener('click', fetchData);

```

Przykład wysyłania danych z formularza do API:

```

document.getElementById('submit-form').addEventListener('submit', function(event) {
  event.preventDefault();

  // Pobieranie danych z formularza
  const formData = new FormData(this);
  const formObject = {};

  formData.forEach((value, key) => {
    formObject[key] = value;
  });

  // Walidacja (przykład)
  if (!formObject.name || !formObject.email) {
    document.getElementById('form-error').textContent

```

```

= 'Proszę wypełnić wszystkie wymagane pola.';
    return;
}

// Pokazanie Loadera
document.getElementById('form-loader').style.display = 'block';
document.getElementById('form-error').textContent = '';

// Wysłanie danych do API
fetch('https://api.example.com/submit', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify(formObject)
})
.then(response => {
    if (!response.ok) {
        return response.json().then(errorData => {
            throw new Error(errorData.message || 'Wystąpił błąd podczas wysyłania formularza.');
        });
    }
    return response.json();
})
.then(data => {
    // Sukces
    document.getElementById('form-success').textContent = 'Formularz został pomyślnie wysłany!';
    document.getElementById('submit-form').reset();
})
.catch(error => {
    // Błąd
    document.getElementById('form-error').textContent = error.message;
})
.finally(() => {

```

```
// Ukrycie loadera
document.getElementById('form-loader').style.display = 'none';
});
});
```

Przykład użycia API z autoryzacją:

```
// Funkcja logowania
function login(username, password) {
  return fetch('https://api.example.com/auth/login',
    {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({
        username: username,
        password: password
      })
    })
  .then(response => {
    if (!response.ok) {
      throw new Error('Nieprawidłowe dane logowania')
    }
    return response.json();
  })
  .then(data => {
    // Zapisanie tokenu autoryzacyjnego
    localStorage.setItem('authToken', data.token);
    localStorage.setItem('tokenExpiry', data.expiresAt);
    return data;
  });
}

// Funkcja sprawdzająca, czy token jest ważny
function isTokenValid() {
  const expiryTimestamp = localStorage.getItem('token
```

```

Expiry');
    if (!expiryTimestamp) return false;

    return new Date().getTime() < parseInt(expiryTimestamp, 10);
}

// Funkcja do wykonywania autoryzowanych żądań
function authenticatedRequest(url, method = 'GET', data = null) {
    // Sprawdzenie, czy token istnieje i jest ważny
    if (!isTokenValid()) {
        // Przekierowanie do strony logowania lub odświeżenie tokenu
        window.location.href = '/login.html';
        return Promise.reject('Sesja wygasła. Zaloguj się ponownie.');
```

```

    }

    const authToken = localStorage.getItem('authToken')
    ;
```

```

    const options = {
        method: method,
        headers: {
            'Authorization': `Bearer ${authToken}`,
            'Content-Type': 'application/json'
        }
    };
};
```

```

    if (data && (method === 'POST' || method === 'PUT' || method === 'PATCH')) {
        options.body = JSON.stringify(data);
    }
}
```

```

    return fetch(url, options)
        .then(response => {
            // Token mógł wygasnąć mimo sprawdzenia (np. został anulowany na serwerze)
```

```

    if (response.status === 401) {
      localStorage.removeItem('authToken');
      localStorage.removeItem('tokenExpiry');
      window.location.href = '/login.html';
      throw new Error('Sesja wygasła. Zaloguj się p
onownie.');
```

}

```

    if (!response.ok) {
      throw new Error('Błąd komunikacji z API: ' +
response.status);
    }

    return response.json();
  });
}
```

// Przykład użycia

```

document.getElementById('login-form').addEventListener(
  'submit', function(event) {
    event.preventDefault();

    const username = document.getElementById('username'
).value;
    const password = document.getElementById('password'
).value;

    login(username, password)
      .then(data => {
        console.log('Zalogowano pomyślnie', data);
        window.location.href = '/dashboard.html';
      })
      .catch(error => {
        document.getElementById('login-error').textCont
ent = error.message;
      });
  });
```

// Przykład pobrania danych użytkownika (na zabezpiec

```

zonej stronie)
function loadUserData() {
  authenticatedRequest('https://api.example.com/user/
profile')
    .then(userData => {
      displayUserProfile(userData);
    })
    .catch(error => {
      console.error('Błąd pobierania danych użytkowni
ka:', error);
    });
}

function displayUserProfile(userData) {
  document.getElementById('user-name').textContent =
userData.name;
  document.getElementById('user-email').textContent =
userData.email;
  // Więcej pól...
}

```

Przykład integracji z API Google Maps:

```

<!DOCTYPE html>
<html lang="pl">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Integracja z Google Maps</title>
  <style>
    #map {
      height: 400px;
      width: 100%;
    }
    .map-container {
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
    }
  </style>

```



```

    .controls {
      margin-bottom: 20px;
    }
  </style>
</head>
<body>
  <div class="map-container">
    <h1>Nasza lokalizacja</h1>

    <div class="controls">
      <input id="search-input" type="text" placeholder="Wyszukaj miejsce">
      <button id="search-button">Szukaj</button>
      <button id="show-route">Pokaż trasę do nas</button>
    </div>

    <div id="map"></div>

    <div id="directions-panel"></div>
  </div>

  <script>
    let map;
    let marker;
    let directionsService;
    let directionsRenderer;

    // Inicjalizacja mapy
    function initMap() {
      // Współrzędne naszej lokalizacji
      const companyLocation = { lat: 52.232222, lng:
21.008333 }; // Warszawa

      // Tworzenie mapy
      map = new google.maps.Map(document.getElementById('map'), {
        center: companyLocation,
        zoom: 15

```

```

});

// Marker Lokalizacji firmy
marker = new google.maps.Marker({
  position: companyLocation,
  map: map,
  title: 'Nasza firma'
});

// Okno informacyjne
const infoWindow = new google.maps.InfoWindow({
  content: '<h3>Nasza firma</h3><p>ul. Przykład  
owa 123<br>00-001 Warszawa<br>tel. +48 123 456 789</p>'
});

marker.addListener('click', () => {
  infoWindow.open(map, marker);
});

// Inicjalizacja serwisów do wyszukiwania i naw
igacji
const searchInput = document.getElementById('se
arch-input');
const searchBox = new google.maps.places.Search
Box(searchInput);

directionsService = new google.maps.DirectionsS
ervice();
directionsRenderer = new google.maps.Directions
Renderer();
directionsRenderer.setMap(map);
directionsRenderer.setPanel(document.getElement
ById('directions-panel'));

// Nastuchiwanie zmian w wyszukiwarce
map.addListener('bounds_changed', () => {
  searchBox.setBounds(map.getBounds());
});

```

```

// Obsługa wyszukiwania
document.getElementById('search-button').addEventListener('click', () => {
    searchLocation();
});

searchInput.addEventListener('keypress', (e) =>
{
    if (e.key === 'Enter') {
        searchLocation();
    }
});

// Obsługa wyznaczania trasy
document.getElementById('show-route').addEventListener('click', () => {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(
            (position) => {
                const userLocation = {
                    lat: position.coords.latitude,
                    lng: position.coords.longitude
                };
                calculateRoute(userLocation, companyLocation);
            },
            () => {
                alert('Nie udało się pobrać Twojej lokalizacji. Wprowadź adres w wyszukiwarce i spróbuj ponownie.');
```

```

// Funkcja wyszukiwania
function searchLocation() {
    const searchInput = document.getElementById('search-input');
    const places = new google.maps.places.PlacesService(map);

    places.findPlaceFromQuery({
        query: searchInput.value,
        fields: ['name', 'geometry']
    }, (results, status) => {
        if (status === google.maps.places.PlacesServiceStatus.OK && results) {
            const location = results[0].geometry.location;
            map.setCenter(location);

            // Dodanie markera dla znalezionej lokalizacji
            new google.maps.Marker({
                map: map,
                position: location,
                title: results[0].name
            });
        } else {
            alert('Nie znaleziono takiego miejsca. Spróbuj innego zapytania.');
```

```

// Funkcja wyznaczania trasy

```

```

function calculateRoute(origin, destination) {
    directionsService.route(
        {
            origin: origin,
            destination: destination,
            travelMode: google.maps.TravelMode.DRIVING
        }
    );
}

```

```

    },
    (result, status) => {
      if (status === 'OK') {
        directionsRenderer.setDirections(result);
      } else {
        alert('Nie udało się wyznaczyć trasy: ' +
status);
      }
    }
  );
}
</script>

<!-- Skrypt Google Maps -->
<script src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&libraries=places&callback=initMap"
  async defer></script>
</body>
</html>

```

Przykład integracji z API pogodowym:

```

// Funkcja pobierająca dane pogodowe
function getWeather(city) {
  const apiKey = 'YOUR_API_KEY';
  const url = `https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${apiKey}&units=metric&lang=pl`;

  return fetch(url)
    .then(response => {
      if (!response.ok) {
        throw new Error('Nie znaleziono danych pogodowych dla podanego miasta');
      }
      return response.json();
    });
}

```

```

// Funkcja wyświetlająca dane pogodowe

```

```

function displayWeather(data) {
  const weatherContainer = document.getElementById('weather-widget');

  const html = `
    <div class="weather-card">
      <h3>${data.name}, ${data.sys.country}</h3>
      <div class="weather-main">
        
        <div class="temperature">${Math.round(data.main.temp)}°C</div>
      </div>
      <div class="weather-details">
        <p>${data.weather[0].description}</p>
        <p>Odczuwalna: ${Math.round(data.main.feels_like)}°C</p>
        <p>Wilgotność: ${data.main.humidity}%</p>
        <p>Wiatr: ${data.wind.speed} m/s</p>
      </div>
    </div>
  `;

  weatherContainer.innerHTML = html;
}

// Obsługa formularza wyszukiwania
document.getElementById('weather-form').addEventListener('submit', function(event) {
  event.preventDefault();

  const city = document.getElementById('city-input').value.trim();

  if (!city) {
    alert('Wprowadź nazwę miasta');
    return;
  }
}

```

```

// Pokazanie loadera
document.getElementById('weather-widget').innerHTML
= '<div class="loader"></div>';

getWeather(city)
  .then(data => {
    displayWeather(data);
  })
  .catch(error => {
    document.getElementById('weather-widget').inner
HTML = `<div class="error">${error.message}</div>`;
  });
});

// Automatyczne pobieranie pogody dla domyślnego mias
ta
document.addEventListener('DOMContentLoaded', functio
n() {
  getWeather('Warszawa')
    .then(data => {
      displayWeather(data);
    })
    .catch(error => {
      document.getElementById('weather-widget').inner
HTML = `<div class="error">${error.message}</div>`;
    });
});

```

Dobre praktyki przy korzystaniu z API: - Zabezpieczaj klucze API (nie umieszczaj ich bezpośrednio w kodzie frontend) - Używaj HTTPS do komunikacji z API - Implementuj obsługę błędów i przypadków granicznych - Buforuj odpowiedzi API w celu ograniczenia liczby żądań - Pamiętaj o limitach żądań dla danego API - Używaj wskaźników ładowania (loaderów) podczas oczekiwania na odpowiedź - Testuj API z różnymi parametrami i w różnych warunkach - Dokumentuj wykorzystane API i ich funkcjonalności

Dlaczego są ważne dla początkujących: - Pozwalają na rozszerzenie funkcjonalności strony bez konieczności budowania wszystkiego od podstaw - Umożliwiają integrację z popularnymi usługami i serwisami - Znajomość pracy z API jest kluczową umiejętnością w nowoczesnym tworzeniu stron - Stanowią fundament dla aplikacji webowych i mobilnych

6. Responsywność

MEDIA QUERIES

Media queries to technika CSS pozwalająca na dostosowanie stylów strony do różnych urządzeń, rozmiarów ekranu i innych cech środowiska przeglądania.

Kluczowe cechy: - Pozwalają na określenie różnych stylów dla różnych warunków - Są podstawą responsywnego projektowania (RWD) - Umożliwiają tworzenie różnych układów dla różnych urządzeń - Pozwalają na optymalizację doświadczenia użytkownika na różnych ekranach

Podstawowa składnia:

```
@media [warunek] {  
    /* Style stosowane, gdy warunek jest spełniony */  
}
```

Przykłady podstawowych media queries:

```
/* Style dla ekranów o szerokości do 768px (urządzeni  
a mobilne) */  
@media (max-width: 768px) {  
    .container {  
        width: 100%;  
        padding: 0 15px;  
    }  
  
    .nav-menu {  
        display: none;  
    }  
}
```



```
}

.mobile-menu-toggle {
    display: block;
}

}

/* Style dla ekranów od 769px do 1024px (tablety) */
@media (min-width: 769px) and (max-width: 1024px) {
    .container {
        width: 90%;
        max-width: 900px;
    }

    .sidebar {
        width: 30%;
    }

    .main-content {
        width: 70%;
    }
}

/* Style dla ekranów powyżej 1024px (desktop) */
@media (min-width: 1025px) {
    .container {
        width: 80%;
        max-width: 1200px;
    }

    .sidebar {
        width: 25%;
    }

    .main-content {
        width: 75%;
    }
}
```

```

/* Style dla orientacji poziomej */
@media (orientation: landscape) {
  .hero-section {
    height: 70vh;
  }
}

/* Style dla drukarki */
@media print {
  .no-print {
    display: none;
  }

  body {
    font-size: 12pt;
    color: black;
    background: white;
  }

  a[href]:after {
    content: " (" attr(href) ")";
  }
}

```

Przykładowy pełny zestaw media queries dla responsywnej strony:

```

/* Base styles (mobile first) */
body {
  font-size: 16px;
  line-height: 1.6;
}

.container {
  width: 100%;
  padding: 0 15px;
  margin: 0 auto;
}

```

```
/* Header styles */
.header {
  padding: 10px 0;
}

.logo {
  font-size: 1.5rem;
}

.nav-menu {
  display: none;
}

.mobile-toggle {
  display: block;
}

/* Main Layout */
.main-content {
  width: 100%;
}

.sidebar {
  width: 100%;
  margin-top: 20px;
}

/* Grid items */
.grid {
  display: grid;
  grid-template-columns: 1fr;
  gap: 20px;
}

/* Typography */
h1 {
  font-size: 1.8rem;
}
```

```
h2 {
  font-size: 1.5rem;
}

/* Small tablets (portrait) */
@media (min-width: 576px) {
  .grid {
    grid-template-columns: repeat(2, 1fr);
  }

  h1 {
    font-size: 2rem;
  }
}

/* Tablets (portrait) */
@media (min-width: 768px) {
  .container {
    width: 90%;
    max-width: 720px;
  }

  .header {
    padding: 20px 0;
  }

  .nav-menu {
    display: flex;
  }

  .mobile-toggle {
    display: none;
  }

  .grid {
    grid-template-columns: repeat(2, 1fr);
  }

  .featured-grid {
```

```
    grid-template-columns: repeat(3, 1fr);
  }

  h1 {
    font-size: 2.2rem;
  }

  h2 {
    font-size: 1.8rem;
  }
}

/* Tablets (landscape) and small desktops */
@media (min-width: 992px) {
  .container {
    max-width: 960px;
  }

  .main-content {
    width: 70%;
    float: left;
  }

  .sidebar {
    width: 25%;
    float: right;
    margin-top: 0;
  }

  .grid {
    grid-template-columns: repeat(3, 1fr);
  }

  .featured-grid {
    grid-template-columns: repeat(4, 1fr);
  }

  h1 {
    font-size: 2.5rem;
  }
}
```

```
}  
}  
  
/* Large desktops */  
@media (min-width: 1200px) {  
  .container {  
    max-width: 1140px;  
  }  
  
  .grid {  
    grid-template-columns: repeat(4, 1fr);  
  }  
  
  .featured-grid {  
    grid-template-columns: repeat(5, 1fr);  
  }  
  
  h1 {  
    font-size: 3rem;  
  }  
  
  body {  
    font-size: 18px;  
  }  
}  
  
/* Extra large desktops */  
@media (min-width: 1400px) {  
  .container {  
    max-width: 1320px;  
  }  
}  
  
/* Special case for extra small screens */  
@media (max-width: 320px) {  
  body {  
    font-size: 14px;  
  }  
}
```

```
h1 {
  font-size: 1.5rem;
}

h2 {
  font-size: 1.3rem;
}

/* High-resolution displays (Retina) */
@media (-webkit-min-device-pixel-ratio: 2), (min-resolution: 192dpi) {
  .logo-image {
    background-image: url('logo@2x.png');
  }
}

/* Print styles */
@media print {
  .header-nav,
  .footer,
  .sidebar,
  .comments,
  .share-buttons,
  .ads {
    display: none;
  }

  .container {
    width: 100%;
    max-width: 100%;
  }

  body {
    font-size: 12pt;
    line-height: 1.5;
    color: #000;
    background: #fff;
  }
}
```

```

a[href]:after {
  content: " (" attr(href) ")";
  font-size: 90%;
}

img {
  max-width: 100% !important;
  page-break-inside: avoid;
}

h1, h2, h3, h4, h5, h6 {
  page-break-after: avoid;
}

p, h2, h3 {
  orphans: 3;
  widows: 3;
}

```

Dodatkowe rodzaje zapytań medialnych:

```

/* Preferencje użytkownika - ciemny motyw */
@media (prefers-color-scheme: dark) {
  body {
    background-color: #121212;
    color: #e0e0e0;
  }

  a {
    color: #90caf9;
  }
}

/* Preferencje użytkownika - redukcja ruchu */
@media (prefers-reduced-motion: reduce) {
  * {
    animation: none !important;
    transition: none !important;
  }
}

```



```

    }
  }

  /* Komunikaty o funkcjach */
  @supports (display: grid) {
    .container {
      display: grid;
      grid-template-columns: repeat(auto-fill, minmax(2
50px, 1fr));
    }
  }

  @supports not (display: grid) {
    .container {
      display: flex;
      flex-wrap: wrap;
    }

    .item {
      width: calc(33.333% - 20px);
      margin: 10px;
    }
  }

  /* Obsługa hover */
  @media (hover: hover) {
    .card:hover {
      transform: translateY(-5px);
      box-shadow: 0 10px 20px rgba(0,0,0,0.1);
    }
  }

  @media (hover: none) {
    .card:active {
      background-color: #f5f5f5;
    }
  }

```

Dobre praktyki dla media queries: - Używaj podejścia “mobile first” (domyślne style dla urządzeń mobilnych) - Stosuj standardowe punkty przerwania (breakpoints) dla spójności - Unikaj nadmiernej liczby breakpointsów - Testuj na prawdziwych urządzeniach, a nie tylko na zmieniającym rozmiar oknie przeglądarki - Używaj jednostek relatywnych (em, rem) zamiast px - Utrzymuj media queries zorganizowane i spójne w całym projekcie - Rozważaj nie tylko szerokość ekranu, ale również inne czynniki (orientację, gęstość pikseli)

Dlaczego są ważne dla początkujących: - Media queries są podstawowym narzędziem do tworzenia responsywnych stron - Umożliwiają dotarcie do szerszej publiczności na różnych urządzeniach - Są standardem w nowoczesnym projektowaniu stron internetowych - Google faworyzuje responsywne strony w wynikach wyszukiwania

ELASTYCZNE SIATKI (GRID)

Elastyczne siatki to systemy układu, które automatycznie dostosowują się do różnych rozmiarów ekranu, umożliwiając tworzenie responsywnych projektów stron internetowych.

Kluczowe cechy: - Automatycznie dostosowują się do dostępnej przestrzeni - Pozwalają na precyzyjne rozmieszczenie elementów w dwóch wymiarach - Mogą być implementowane za pomocą CSS Grid lub Flexbox - Umożliwiają tworzenie złożonych układów, które reagują na zmiany rozmiaru ekranu

CSS Grid - podstawy:

```
/* Kontener siatki */
.grid-container {
  display: grid;
  grid-template-columns: repeat(3, 1fr); /* Trzy kolumny o równej szerokości */
  grid-template-rows: auto auto auto; /* Trzy wiersze
```

```

o automatycznej wysokości */
gap: 20px; /* Odstęp między elementami */
}

/* Elementy zajmujące więcej komórek */
.header {
  grid-column: 1 / -1; /* Od pierwszej do ostatniej kolumny */
}

.sidebar {
  grid-row: 2 / 4; /* Od drugiego do czwartego wiersza */
}

.featured {
  grid-column: 2 / 4; /* Od drugiej do czwartej kolumny */
  grid-row: 2 / 3; /* Tylko drugi wiersz */
}

/* Responsywna siatka */
@media (max-width: 768px) {
  .grid-container {
    grid-template-columns: 1fr; /* Jedna kolumna na urządzeniach mobilnych */
  }

  .sidebar, .featured {
    grid-column: 1; /* Resetowanie pozycji kolumn */
    grid-row: auto; /* Automatyczna pozycja wierszy */
  }
}

```

Przykład zaawansowanej siatki responsywnej:

```

.grid-container {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(250

```

```

px, 1fr));
    gap: 20px;
    padding: 20px;
}

.grid-item {
    background-color: #f8f8f8;
    border-radius: 5px;
    padding: 20px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}

/* Style dla elementów specjalnych */
.grid-item.featured {
    grid-column: span 2;
    grid-row: span 2;
    background-color: #fff8e1;
}

/* Układ dla różnych rozmiarów ekranu */
@media (max-width: 768px) {
    .grid-container {
        grid-template-columns: repeat(auto-fill, minmax(2
00px, 1fr));
    }

    .grid-item.featured {
        grid-column: span 1;
        grid-row: span 1;
    }
}

@media (max-width: 480px) {
    .grid-container {
        grid-template-columns: 1fr;
        gap: 15px;
    }
}

```

Przykładowy układ strony z CSS Grid:

```
body {  
  margin: 0;  
  font-family: Arial, sans-serif;  
}  
  
.page-container {  
  display: grid;  
  min-height: 100vh;  
  grid-template-areas:  
    "header header header"  
    "nav main sidebar"  
    "footer footer footer";  
  grid-template-columns: 200px 1fr 250px;  
  grid-template-rows: auto 1fr auto;  
}  
  
.header {  
  grid-area: header;  
  background-color: #333;  
  color: white;  
  padding: 1rem;  
}  
  
.nav {  
  grid-area: nav;  
  background-color: #f5f5f5;  
  padding: 1rem;  
}  
  
.main {  
  grid-area: main;  
  padding: 1rem;  
}  
  
.sidebar {  
  grid-area: sidebar;  
  background-color: #f9f9f9;
```

```
padding: 1rem;
}

.footer {
  grid-area: footer;
  background-color: #333;
  color: white;
  padding: 1rem;
  text-align: center;
}

/* Responsywność */
@media (max-width: 992px) {
  .page-container {
    grid-template-areas:
      "header header"
      "nav main"
      "sidebar sidebar"
      "footer footer";
    grid-template-columns: 150px 1fr;
  }
}

@media (max-width: 768px) {
  .page-container {
    grid-template-areas:
      "header"
      "nav"
      "main"
      "sidebar"
      "footer";
    grid-template-columns: 1fr;
  }

  .nav {
    display: flex;
    overflow-x: auto;
  }
}
```

```

.nav ul {
    display: flex;
    list-style: none;
    padding: 0;
    margin: 0;
}

.nav li {
    margin-right: 15px;
}
}

```

Flexbox - podstawy:

```

/* Kontener flex */
.flex-container {
    display: flex;
    flex-wrap: wrap; /* Umożliwia zawijanie elementów */
    /
    justify-content: space-between; /* Rozłożenie elementów */
    align-items: stretch; /* Wyrównanie w pionie */
    gap: 20px; /* Odstęp między elementami */
}

/* Elementy flex */
.flex-item {
    flex: 1 1 300px; /* grow | shrink | basis */
    margin-bottom: 20px;
}

/* Elementy specjalne */
.flex-item.featured {
    flex: 2 1 600px; /* Podwójna szerokość */
}

/* Responsywność */
@media (max-width: 768px) {
    .flex-container {
        flex-direction: column; /* Układ kolumnowy na urz

```

```
ądzeniach mobilnych */  
}
```

```
.flex-item, .flex-item.featured {  
    flex: 1 1 auto; /* Resetowanie specjalnych rozmiarów */  
}  
}
```

Porównanie Grid i Flexbox:

CSS Grid: - Doskonały do układów dwuwymiarowych (wiersze i kolumny) - Pozwala na precyzyjne rozmieszczenie elementów - Idealny do złożonych układów stron - Możliwość nakładania elementów

Flexbox: - Doskonały do układów jednowymiarowych (wiersz lub kolumna) - Świetny do wyrównywania elementów - Idealny do komponentów UI jak menu, paski narzędzi, karty - Łatwiejszy w użyciu dla prostych układów

Przykład łączenia Grid i Flexbox:

```
/* Główny układ strony z CSS Grid */  
.page-layout {  
    display: grid;  
    grid-template-columns: 1fr 3fr 1fr;  
    grid-template-areas:  
        "header header header"  
        "sidebar content aside"  
        "footer footer footer";  
    gap: 20px;  
}  
  
.header { grid-area: header; }  
.sidebar { grid-area: sidebar; }  
.content { grid-area: content; }  
.aside { grid-area: aside; }  
.footer { grid-area: footer; }
```



```
/* Karty produktów z Flexbox */
.product-grid {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
}

.product-card {
  flex: 1 1 300px;
  display: flex;
  flex-direction: column;
  border: 1px solid #eee;
  border-radius: 5px;
}

.product-image {
  width: 100%;
  height: 200px;
  object-fit: cover;
}

.product-info {
  flex: 1;
  padding: 15px;
  display: flex;
  flex-direction: column;
}

.product-title {
  margin-top: 0;
}

.product-description {
  flex: 1;
}

.product-price {
  font-size: 1.2em;
}
```

```

font-weight: bold;
margin-top: auto;
}

.product-actions {
display: flex;
justify-content: space-between;
padding: 15px;
border-top: 1px solid #eee;
}

/* Responsywność */
@media (max-width: 992px) {
.page-layout {
grid-template-columns: 1fr 2fr;
grid-template-areas:
    "header header"
    "sidebar content"
    "aside aside"
    "footer footer";
}
}

@media (max-width: 768px) {
.page-layout {
grid-template-columns: 1fr;
grid-template-areas:
    "header"
    "sidebar"
    "content"
    "aside"
    "footer";
}
}

```

HTML struktura dla powyższego CSS:

```

<div class="page-layout">
  <header class="header">
    <h1>Nazwa sklepu</h1>

```

```

    <nav><!-- Menu --></nav>
</header>

<aside class="sidebar">
    <h2>Kategorie</h2>
    <ul>
        <li><a href="#">Kategoria 1</a></li>
        <li><a href="#">Kategoria 2</a></li>
        <li><a href="#">Kategoria 3</a></li>
    </ul>
    <h2>Filtry</h2>
    <!-- Filtry -->
</aside>

<main class="content">
    <h2>Produkty</h2>

    <div class="product-grid">
        <div class="product-card">
            
            <div class="product-info">
                <h3 class="product-title">Produkt 1</h3>
                <p class="product-description">Lorem ipsum
dolor sit amet, consectetur adipiscing elit.</p>
                <div class="product-price">99,99 zł</div>
            </div>
            <div class="product-actions">
                <button class="btn-wishlist">♥</button>
                <button class="btn-add-cart">Dodaj do koszy
ka</button>
            </div>
        </div>

        <!-- Więcej produktów -->
    </div>
</main>

<aside class="aside">

```

```

<div class="cart-summary">
  <h2>Koszyk</h2>
  <!-- Zawartość koszyka -->
</div>
<div class="bestsellers">
  <h2>Bestsellery</h2>
  <!-- Lista bestsellerów -->
</div>
</aside>

<footer class="footer">
  <div class="footer-columns">
    <!-- Zawartość stopki -->
  </div>
  <div class="copyright">© 2025 Nazwa sklepu. Wszel
kie prawa zastrzeżone.</div>
</footer>
</div>

```

Dobre praktyki dla elastycznych siatek: - Używaj jednostek relatywnych (fr, %, em) zamiast jednostek absolutnych (px) - Stosuj `minmax()` w Grid, aby zapewnić minimalną i maksymalną szerokość - Używaj `auto-fill` lub `auto-fit` dla automatycznego układania elementów - Korzystaj z `@media` queries do dostosowania układu do różnych rozmiarów ekranów - Wybieraj odpowiednią technikę do odpowiedniego zadania (Grid dla układu strony, Flexbox dla komponentów) - Testuj na różnych urządzeniach i rozmiarach ekranu - Pamiętaj o dostępności i kolejności elementów w DOM

Dlaczego są ważne dla początkujących: - Elastyczne siatki są fundamentem nowoczesnego, responsywnego designu - Pozwalają na tworzenie złożonych układów bez skomplikowanego kodu - Zastępują przestarzałe techniki jak tabele i float - Są wspierane przez wszystkie nowoczesne przeglądarki - Znacznie przyspieszają proces tworzenia responsywnych layoutów

OBRAZY RESPONSYWNE

Obrazy responsywne to techniki pozwalające na dostosowanie obrazów do różnych rozmiarów ekranu, gęstości pikseli i możliwości urządzeń, optymalizując wydajność i doświadczenie użytkownika.

Kluczowe cechy: - Dostosowują się do różnych rozmiarów ekranu - Oszczędzają transfer danych na urządzeniach mobilnych - Zapewniają wysoką jakość obrazów na wyświetlaczach o wysokiej gęstości pikseli (Retina) - Mogą oferować różne formaty obrazów w zależności od wsparcia przeglądarki

Podstawowe techniki responsywnych obrazów:

1. Podstawowe skalowanie CSS:

```
img {  
  max-width: 100%;  
  height: auto;  
}
```

2. Element `<picture>` i źródła:

```
<picture>  
  <!-- Różne rozmiary dla różnych breakpointów -->  
  <source media="(max-width: 480px)" srcset="image-small.jpg">  
  <source media="(max-width: 768px)" srcset="image-medium.jpg">  
  <source media="(max-width: 1200px)" srcset="image-large.jpg">  
  
  <!-- Obraz domyślny -->  
    
</picture>
```

3. Atrybut `srcset` z deskryptorami szerokości:

```


```

4. Atrybut `srcset` z deskryptorami gęstości pikseli:

```



```

5. Element `<picture>` z różnymi formatami obrazu:

```

<picture>
  <!-- WebP dla przeglądarek wspierających ten format
  -->
  <source type="image/webp" srcset="image.webp">

  <!-- AVIF dla przeglądarek wspierających ten format
  -->
  <source type="image/avif" srcset="image.avif">

  <!-- Fallback do JPEG dla pozostałych przeglądarek
  -->
  
</picture>

```

Przykład kompletnego responsywnego obrazu z różnymi formatami i rozmiarami:

```
<picture>
  <!-- WebP format, różne rozmiary -->
  <source
    type="image/webp"
    media="(max-width: 480px)"
    srcset="
      images/small.webp 1x,
      images/small@2x.webp 2x
    "
  >
  <source
    type="image/webp"
    media="(max-width: 768px)"
    srcset="
      images/medium.webp 1x,
      images/medium@2x.webp 2x
    "
  >
  <source
    type="image/webp"
    srcset="
      images/large.webp 1x,
      images/large@2x.webp 2x
    "
  >

  <!-- JPEG format, różne rozmiary (fallback) -->
  <source
    media="(max-width: 480px)"
    srcset="
      images/small.jpg 1x,
      images/small@2x.jpg 2x
    "
  >
  <source
    media="(max-width: 768px)"
```

```

srcset="
    images/medium.jpg 1x,
    images/medium@2x.jpg 2x
"
>

<!-- Domyślny obraz -->

</picture>

```

Techniki CSS dla responsywnych obrazów tła:

```

/* Podstawowe tło responsywne */
.hero-banner {
    background-image: url('banner-large.jpg');
    background-size: cover;
    background-position: center;
    height: 500px;
}

/* Różne obrazy tła dla różnych rozmiarów ekranu */
@media (max-width: 768px) {
    .hero-banner {
        background-image: url('banner-medium.jpg');
        height: 300px;
    }
}

@media (max-width: 480px) {
    .hero-banner {
        background-image: url('banner-small.jpg');
        height: 200px;
    }
}

```



```

}

/* Obrazy tła o wysokiej gęstości pikseli (Retina) */
@media (-webkit-min-device-pixel-ratio: 2), (min-resolution: 192dpi) {
  .hero-banner {
    background-image: url('banner-large@2x.jpg');
  }
}

/* Responsywne proporcje obrazu (aspect ratio) */
.responsive-container {
  position: relative;
  width: 100%;
  height: 0;
  padding-bottom: 56.25%; /* Proporcje 16:9 */
  overflow: hidden;
}

.responsive-container img,
.responsive-container iframe,
.responsive-container video {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  object-fit: cover; /* lub contain, w zależności od
potrzeb */
}

```

Lazy loading obrazów:

```

<!-- Natywny Lazy Loading (wspierany przez nowoczesne
przeglądarki) -->


```

```

<!-- Lazy Loading z atrybutem data-src (wymaga JavaScript) -->

```

```

```

```
// Implementacja lazy loading z Intersection Observer
document.addEventListener('DOMContentLoaded', function() {
  const lazyImages = document.querySelectorAll('img.lazy');
```

```
  if ('IntersectionObserver' in window) {
    const imageObserver = new IntersectionObserver(function(entries, observer) {
      entries.forEach(function(entry) {
        if (entry.isIntersecting) {
          const img = entry.target;
          img.src = img.dataset.src;
          img.classList.remove('lazy');
          imageObserver.unobserve(img);
        }
      });
    });
  }
```

```
  lazyImages.forEach(function(img) {
    imageObserver.observe(img);
  });
} else {
  // Fallback dla starszych przeglądarek
  let active = false;
```

```
  const lazyLoad = function() {
    if (active === false) {
      active = true;

      setTimeout(function() {
        lazyImages.forEach(function(img) {
          if ((img.getBoundingClientRect().top <= window.innerHeight &&
            img.getBoundingClientRect().bottom >= 0) &&
            getComputedStyle(img).display !== 'none') {

```

```

img.src = img.dataset.src;
img.classList.remove('lazy');

lazyImages = lazyImages.filter(function
(image) {
    return image !== img;
});

if (lazyImages.length === 0) {
    document.removeEventListener('scroll'
, lazyLoad);
    window.removeEventListener('resize',
lazyLoad);
    window.removeEventListener('orientati
onchange', lazyLoad);
}
}
});

    active = false;
}, 200);
}
};

document.addEventListener('scroll', lazyLoad);
window.addEventListener('resize', lazyLoad);
window.addEventListener('orientationchange', lazy
Load);
}
});

```

Art direction z CSS (alternatywne podejście):

```

.responsive-image {
    background-image: url('image-desktop.jpg');
    background-size: cover;
    background-position: center;
    height: 400px;
}

```

```

@media (max-width: 768px) {
  .responsive-image {
    background-image: url('image-tablet.jpg');
    height: 300px;
    background-position: top center;
  }
}

@media (max-width: 480px) {
  .responsive-image {
    background-image: url('image-mobile.jpg');
    height: 200px;
    background-position: center 30%;
  }
}

```

Dobre praktyki dla responsywnych obrazów: - Optymalizuj obrazy pod kątem rozmiaru pliku (kompresja, odpowiednie wymiary) - Używaj nowoczesnych formatów obrazów (WebP, AVIF) z fallbackiem do starszych formatów - Dostarczaj obrazy w różnych rozmiarach dla różnych urządzeń - Uwzględniaj wyświetlacze o wysokiej gęstości pikseli (Retina) - Implementuj lazy loading dla poprawy wydajności - Zawsze dodawaj alternatywny tekst (atrybut alt) dla dostępności - Wybieraj odpowiednią strategię w zależności od kontekstu (art direction vs. responsywność) - Określaj wymiary obrazu (width i height), aby uniknąć przeskoków podczas ładowania - Testuj na różnych urządzeniach i przy różnych prędkościach internetu

Dlaczego są ważne dla początkujących: - Obrazy często stanowią największą część transferu danych na stronie - Responsywne obrazy znacząco poprawiają wydajność strony, zwłaszcza na urządzeniach mobilnych - Wpływają na SEO (szybkość ładowania strony jest czynnikiem rankingowym) - Zapewniają optymalne doświadczenie użytkownika na wszystkich urządzeniach - Są standardem w nowoczesnym projektowaniu stron

7. Warstwa backendowa (zaplecze)

SERWER

Serwer to komputer lub system przechowujący pliki strony internetowej i udostępniający je użytkownikom za pośrednictwem internetu.

Kluczowe cechy: - Przechowuje pliki strony (HTML, CSS, JavaScript, obrazy, itp.) - Przetwarza żądania od przeglądarek użytkowników - Może wykonywać kod po stronie serwera (PHP, Python, Node.js, itp.) - Zarządza bazami danych, przechowywaniem plików i logiką aplikacji - Zapewnia bezpieczeństwo danych i kontrolę dostępu

Rodzaje serwerów: - **Serwery współdzielone (shared hosting)** - wiele stron korzysta z zasobów jednego serwera - **Serwery VPS (Virtual Private Server)** - wirtualny serwer dedykowany dla jednego klienta - **Serwery dedykowane** - fizyczny serwer zarezerwowany dla jednego klienta - **Serwery w chmurze** - elastyczne zasoby udostępniane przez dostawcę chmury (AWS, Google Cloud, Azure) - **Serwery statyczne** - specjalizują się w obsłudze statycznych stron (GitHub Pages, Netlify)

Podstawowe komponenty serwerowe: - **Serwer HTTP** (Apache, Nginx, IIS) - obsługuje żądania HTTP od przeglądarek - **Środowisko uruchomieniowe** (PHP, Python, Node.js, Ruby) - wykonuje kod backendowy - **System zarządzania bazami danych** (MySQL, PostgreSQL, MongoDB) - przechowuje i zarządza danymi - **System operacyjny** (Linux, Windows Server) - zarządza zasobami sprzętowymi - **Firewall i zabezpieczenia** - chronią przed nieautoryzowanym dostępem

Cykl żądanie-odpowiedź: 1. Użytkownik wprowadza adres URL lub klika link 2. Przeglądarka wysyła żądanie HTTP do serwera 3. Serwer HTTP odbiera żądanie 4. Jeśli potrzeba, żądanie jest przekazywane

do interpretera języka (PHP, Python, itd.) 5. Kod backendowy przetwarza żądanie, często korzystając z bazy danych 6. Serwer generuje odpowiedź HTTP (zazwyczaj strona HTML) 7. Odpowiedź jest wysyłana z powrotem do przeglądarki użytkownika 8. Przeglądarka renderuje otrzymaną stronę

Podstawowa konfiguracja serwera Apache (plik .htaccess):

```
# Określenie strony domyślnej
DirectoryIndex index.html index.php

# Włączenie przekierowań URL
RewriteEngine On

# Przekierowanie z HTTP na HTTPS
RewriteCond %{HTTPS} off
RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI}
[L,R=301]

# Przekierowanie z www na non-www
RewriteCond %{HTTP_HOST} ^www\.(.+)$ [NC]
RewriteRule ^(.*)$ https://%1/$1 [R=301,L]

# Usuwanie końcowego ukośnika
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)/$ /$1 [R=301,L]

# Obsługa adresów przyjaznych (bez rozszerzenia .php)
RewriteCond %{REQUEST_FILENAME}.php -f
RewriteRule ^(.*)$ $1.php [L]

# Obsługa błędów
ErrorDocument 404 /404.html
ErrorDocument 500 /500.html

# Kompresja GZIP
<IfModule mod_deflate.c>
    AddOutputFilterByType DEFLATE text/html text/plain
    text/xml text/css application/javascript application/
```

```

json
</IfModule>

# Cache-Control
<IfModule mod_expires.c>
    ExpiresActive On
    ExpiresByType image/jpg "access plus 1 year"
    ExpiresByType image/jpeg "access plus 1 year"
    ExpiresByType image/gif "access plus 1 year"
    ExpiresByType image/png "access plus 1 year"
    ExpiresByType image/webp "access plus 1 year"
    ExpiresByType image/svg+xml "access plus 1 year"
    ExpiresByType text/css "access plus 1 month"
    ExpiresByType application/javascript "access plus 1
month"
    ExpiresByType text/html "access plus 1 day"
</IfModule>

# Blokowanie dostępu do plików wrażliwych
<FilesMatch "^\.(?!well-known)|\.(?:git|htaccess|env|
config\.php|ini)$">
    Order Allow,Deny
    Deny from all
</FilesMatch>

```

Podstawowa konfiguracja Nginx:

```

server {
    listen 80;
    server_name example.com www.example.com;

    # Przekierowanie HTTP na HTTPS
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl;
    server_name example.com www.example.com;

    # Konfiguracja SSL

```

```
ssl_certificate /etc/letsencrypt/live/example.com
/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/example
.com/privkey.pem;
```

```
# Przekierowanie z www na non-www
if ($host = www.example.com) {
    return 301 https://example.com$request_uri;
}
```

```
# Główny katalog strony
root /var/www/example.com/public;
index index.html index.php;
```

```
# Obsługa błędów
error_page 404 /404.html;
error_page 500 502 503 504 /50x.html;
```

```
# Obsługa plików statycznych
location ~* \.(jpg|jpeg|png|gif|ico|css|js)$ {
    expires 1y;
    add_header Cache-Control "public, max-age=315
36000";
}
```

```
# Obsługa PHP
location ~ /\.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/run/php/php7.4-fpm.sock;
}
```

```
# Blokowanie dostępu do plików wrażliwych
location ~ /\.(!well-known) {
    deny all;
}
```

```
# Przyjazne adresy URL
location / {
    try_files $uri $uri/ /index.php?$query_string
```


Popularne usługi hostingowe: - **Współdzielone:** Hostinger, Bluehost, HostGator, OVH - **VPS/Dedykowane:** DigitalOcean, Vultr, Linode, OVH - **Chmurowe:** AWS (Amazon Web Services), Google Cloud Platform, Microsoft Azure - **Statyczne/JAMstack:** Netlify, Vercel, GitHub Pages, Cloudflare Pages

Dobre praktyki dla serwerów: - Regularnie aktualizuj oprogramowanie serwerowe - Zabezpieczaj serwer (firewall, HTTPS, regularne audyty) - Monitoruj wydajność i dostępność - Twórz kopie zapasowe danych - Optymalizuj pod kątem wydajności (cache, kompresja, CDN) - Planuj skalowanie w miarę wzrostu ruchu - Dokumentuj konfiguracje i procedury - Stosuj automatyzację (CI/CD, skrypty)

Dlaczego są ważne dla początkujących: - Nawet prosta strona wymaga serwera do jej publikacji - Zrozumienie podstaw serwerów pomaga w diagnostyce problemów - Znajomość opcji hostingowych pomaga wybrać odpowiednią usługę - Podstawy konfiguracji serwerów są kluczowe dla wydajności i bezpieczeństwa strony

BAZA DANYCH

Baza danych to zorganizowany zbiór danych, który można przechowywać, zarządzać i odpytywać, umożliwiając dynamiczną zawartość i funkcjonalności stron internetowych.

Kluczowe cechy: - Umożliwia trwałe przechowywanie danych - Zapewnia szybki dostęp do informacji - Pozwala na relacje między danymi - Umożliwia zarządzanie uprawnieniami i bezpieczeństwem danych - Wspiera transakcje i integralność danych

Rodzaje baz danych: - Relacyjne (SQL) - MySQL, PostgreSQL, SQLite, MariaDB - Dane przechowywane w tabelach z relacjami - Sztywna struktura danych (schemat) - Język zapytań SQL - ACID compliance (Atomicity, Consistency, Isolation, Durability)

- **Nierelacyjne (NoSQL)** - MongoDB, CouchDB, Redis, Firebase
 - Elastyczna struktura danych
 - Różne modele danych (dokumentowe, klucz-wartość, grafowe)
 - Wysoka skalowalność
 - Zazwyczaj bez standardowego języka zapytań

Podstawowe operacje na bazie danych (SQL):

1. Tworzenie tabeli:

```
CREATE TABLE users (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  username VARCHAR(50) NOT NULL UNIQUE,  
  email VARCHAR(100) NOT NULL UNIQUE,  
  password VARCHAR(255) NOT NULL,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U  
PDATE CURRENT_TIMESTAMP  
);
```

```
CREATE TABLE posts (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  user_id INT NOT NULL,  
  title VARCHAR(255) NOT NULL,  
  content TEXT,  
  status ENUM('draft', 'published', 'archived') DEFAU  
LT 'draft',  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U  
PDATE CURRENT_TIMESTAMP,  
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELET
```

E CASCADE

```
);
```

2. Wstawianie danych:

```
INSERT INTO users (username, email, password)
VALUES ('john_doe', 'john@example.com', 'hashed_password_here');
```

```
INSERT INTO posts (user_id, title, content, status)
VALUES (1, 'Mój pierwszy post', 'Treść mojego pierwszego postu...', 'published');
```

3. Pobieranie danych:

```
-- Pobieranie wszystkich użytkowników
SELECT * FROM users;
```

```
-- Pobieranie konkretnego użytkownika
SELECT * FROM users WHERE id = 1;
```

```
-- Pobieranie postów z informacjami o autorze
SELECT posts.*, users.username
FROM posts
JOIN users ON posts.user_id = users.id
WHERE posts.status = 'published'
ORDER BY posts.created_at DESC
LIMIT 10;
```

4. Aktualizacja danych:

```
UPDATE users
SET email = 'newemail@example.com'
WHERE id = 1;
```

```
UPDATE posts
SET status = 'archived'
WHERE created_at < DATE_SUB(NOW(), INTERVAL 1 YEAR);
```

5. Usuwanie danych:

```
DELETE FROM posts WHERE id = 5;
```

```
DELETE FROM users WHERE id = 2;
```

Przykład połączenia z bazą danych MySQL w PHP:

```
<?php
// Dane połączenia
$host = 'localhost';
$dbname = 'mywebsite';
$username = 'dbuser';
$password = 'dbpassword';

try {
    // Nawiązanie połączenia z bazą danych
    $pdo = new PDO("mysql:host=$host;dbname=$dbname;character
    set=utf8mb4", $username, $password);

    // Ustawienie trybu błędów na wyjątki
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE
    E_EXCEPTION);

    // Przygotowanie zapytania
    $stmt = $pdo->prepare("SELECT * FROM posts WHERE
    status = :status ORDER BY created_at DESC LIMIT :limit");

    // Przypisanie parametrów
    $stmt->bindValue(':status', 'published', PDO::PARAM_
    AM_STR);
    $stmt->bindValue(':limit', 10, PDO::PARAM_INT);

    // Wykonanie zapytania
    $stmt->execute();

    // Pobranie wyników
    $posts = $stmt->fetchAll(PDO::FETCH_ASSOC);

    // Wyświetlenie wyników
    foreach ($posts as $post) {
```

```

        echo "<h2>" . htmlspecialchars($post['title']
    ) . "</h2>";
        echo "<p>" . htmlspecialchars($post['content'
    ]) . "</p>";
        echo "<small>Data publikacji: " . $post['crea
    ted_at'] . "</small>";
    }

} catch (PDOException $e) {
    // Obsługa błędu
    die("Błąd połączenia z bazą danych: " . $e->getMe
    ssage());
}
?>

```

Przykład połączenia z bazą danych MongoDB w Node.js:

```

const { MongoClient } = require('mongodb');

// Adres URI bazy danych
const uri = "mongodb+srv://username:password@cluster0
.mongodb.net/mywebsite?retryWrites=true&w=majority";
const client = new MongoClient(uri);

async function getPosts() {
    try {
        // Połączenie z bazą danych
        await client.connect();

        // Wybór bazy danych i kolekcji
        const database = client.db("mywebsite");
        const posts = database.collection("posts");

        // Zapytanie o posty
        const query = { status: "published" };
        const options = {
            sort: { created_at: -1 },
            limit: 10
        };
    }

```

```

// Wykonanie zapytania
const cursor = posts.find(query, options);

// Sprawdzenie, czy są wyniki
if ((await cursor.count()) === 0) {
  console.log("Nie znaleziono postów");
}

// Iteracja przez wyniki
const results = [];
await cursor.forEach(post => {
  results.push(post);
});

return results;

} finally {
  // Zamknięcie połączenia po zakończeniu
  await client.close();
}
}

// Wywołanie funkcji i obsługa wyniku
getPosts()
  .then(posts => {
    posts.forEach(post => {
      console.log(`${post.title} - ${post.created_at}`);
    });
  })
  .catch(err => {
    console.error("Błąd pobierania postów:", err);
  });

```

ORM (Object-Relational Mapping): ORM to technika programowania pozwalająca na pracę z bazą danych przy użyciu obiektów zamiast bezpośrednich zapytań SQL.

Przykład z Sequelize (ORM dla Node.js):

```
const { Sequelize, DataTypes } = require('sequelize')
;

// Inicjalizacja połączenia
const sequelize = new Sequelize('mywebsite', 'dbuser'
, 'dbpassword', {
  host: 'localhost',
  dialect: 'mysql'
});

// Definicja modelu User
const User = sequelize.define('User', {
  username: {
    type: DataTypes.STRING(50),
    allowNull: false,
    unique: true
  },
  email: {
    type: DataTypes.STRING(100),
    allowNull: false,
    unique: true,
    validate: {
      isEmail: true
    }
  },
  password: {
    type: DataTypes.STRING(255),
    allowNull: false
  }
}, {
  timestamps: true,
  createdAt: 'created_at',
  updatedAt: 'updated_at'
});

// Definicja modelu Post
const Post = sequelize.define('Post', {
  title: {
    type: DataTypes.STRING(255),
```

```

    allowNull: false
  },
  content: {
    type: DataTypes.TEXT
  },
  status: {
    type: DataTypes.ENUM('draft', 'published', 'archi
ved'),
    defaultValue: 'draft'
  }
}, {
  timestamps: true,
  createdAt: 'created_at',
  updatedAt: 'updated_at'
});

// Definicja relacji
User.hasMany(Post);
Post.belongsTo(User);

// Synchronizacja modeli z bazą danych
sequelize.sync()
  .then(() => {
    console.log('Models synced with database');
  })
  .catch(err => {
    console.error('Error syncing models:', err);
  });

// Przykład użycia modeli
async function createUserWithPost() {
  try {
    // Tworzenie użytkownika
    const user = await User.create({
      username: 'jane_doe',
      email: 'jane@example.com',
      password: 'hashed_password_here'
    });
  }

```



```

// Tworzenie posta powiązanego z użytkownikiem
const post = await Post.create({
  title: 'Mój pierwszy post z ORM',
  content: 'Treść posta utworzonego za pomocą ORM
...',
  status: 'published',
  UserId: user.id
});

console.log('User and post created successfully')
;
} catch (error) {
  console.error('Error creating user and post:', error);
}
}

// Przykład pobierania danych
async function getPublishedPosts() {
  try {
    const posts = await Post.findAll({
      where: {
        status: 'published'
      },
      include: {
        model: User,
        attributes: ['username', 'email']
      },
      order: [
        ['created_at', 'DESC']
      ],
      limit: 10
    });

    return posts;
  } catch (error) {
    console.error('Error fetching posts:', error);
    return [];
  }
}

```



Modelowanie danych:

1. **Normalizacja** - proces organizowania danych w bazie w celu minimalizacji redundancji i zwiększenia integralności danych.
2. **Klucze:**
 - **Klucz podstawowy (Primary Key)** - unikalny identyfikator rekordu w tabeli
 - **Klucz obcy (Foreign Key)** - odwołanie do klucza podstawowego w innej tabeli
3. **Relacje:**
 - **Jeden do jednego (1:1)** - jeden rekord w tabeli A odnosi się do dokładnie jednego rekordu w tabeli B
 - **Jeden do wielu (1:N)** - jeden rekord w tabeli A odnosi się do wielu rekordów w tabeli B
 - **Wiele do wielu (N:N)** - wiele rekordów w tabeli A odnosi się do wielu rekordów w tabeli B (wymaga tabeli łączącej)

Przykład modelu danych dla bloga:

```
-- Użytkownicy
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(50) NOT NULL UNIQUE,
  email VARCHAR(100) NOT NULL UNIQUE,
  password VARCHAR(255) NOT NULL,
  bio TEXT,
  avatar_url VARCHAR(255),
  is_admin BOOLEAN DEFAULT FALSE,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U
PDATE CURRENT_TIMESTAMP
);
```

-- Kategorie

```
CREATE TABLE categories (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  name VARCHAR(50) NOT NULL UNIQUE,  
  slug VARCHAR(50) NOT NULL UNIQUE,  
  description TEXT,  
  parent_id INT,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U  
PDATE CURRENT_TIMESTAMP,  
  FOREIGN KEY (parent_id) REFERENCES categories(id) O  
N DELETE SET NULL  
);
```

-- Posty

```
CREATE TABLE posts (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  user_id INT NOT NULL,  
  title VARCHAR(255) NOT NULL,  
  slug VARCHAR(255) NOT NULL UNIQUE,  
  content TEXT,  
  excerpt TEXT,  
  featured_image VARCHAR(255),  
  status ENUM('draft', 'published', 'archived') DEFAU  
LT 'draft',  
  comment_status ENUM('open', 'closed') DEFAULT 'open  
' ,  
  view_count INT DEFAULT 0,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U  
PDATE CURRENT_TIMESTAMP,  
  published_at TIMESTAMP NULL,  
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELET  
E CASCADE  
);
```

-- Relacja wiele do wielu między postami i kategoriami

```
CREATE TABLE post_categories (  
    post_id INT NOT NULL,  
    category_id INT NOT NULL,  
    PRIMARY KEY (post_id, category_id),  
    FOREIGN KEY (post_id) REFERENCES posts(id) ON DELETE  
E CASCADE,  
    FOREIGN KEY (category_id) REFERENCES categories(id)  
ON DELETE CASCADE  
);
```

-- Tagi

```
CREATE TABLE tags (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    name VARCHAR(50) NOT NULL UNIQUE,  
    slug VARCHAR(50) NOT NULL UNIQUE,  
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

-- Relacja wiele do wielu między postami i tagami

```
CREATE TABLE post_tags (  
    post_id INT NOT NULL,  
    tag_id INT NOT NULL,  
    PRIMARY KEY (post_id, tag_id),  
    FOREIGN KEY (post_id) REFERENCES posts(id) ON DELETE  
E CASCADE,  
    FOREIGN KEY (tag_id) REFERENCES tags(id) ON DELETE  
CASCADE  
);
```

-- Komentarze

```
CREATE TABLE comments (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    post_id INT NOT NULL,  
    parent_id INT,  
    author_name VARCHAR(100),  
    author_email VARCHAR(100),  
    author_url VARCHAR(100),  
    author_ip VARCHAR(100),  
    content TEXT NOT NULL,
```

```
user_id INT,  
status ENUM('pending', 'approved', 'spam') DEFAULT  
'pending',  
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
FOREIGN KEY (post_id) REFERENCES posts(id) ON DELET  
E CASCADE,  
FOREIGN KEY (parent_id) REFERENCES comments(id) ON  
DELETE CASCADE,  
FOREIGN KEY (user_id) REFERENCES users(id) ON DELET  
E SET NULL  
);
```

Dobre praktyki dla baz danych: - Projektuj schemat danych z myślą o skalowalności - Używaj odpowiednich typów danych dla kolumn - Twórz indeksy dla często wyszukiwanych kolumn - Stosuj transakcje dla operacji złożonych - Zabezpieczaj przed SQL Injection (parametryzowane zapytania) - Regularnie wykonuj kopie zapasowe - Implementuj walidację danych po stronie serwera - Nie przechowuj wrażliwych danych w formie niezaszyfrowanej - Monitoruj wydajność zapytań - Optymalizuj zapytania dla dużych zbiorów danych

Dlaczego są ważne dla początkujących: - Bazy danych są podstawą dynamicznych stron internetowych - Umożliwiają przechowywanie i zarządzanie treścią, użytkownikami i innymi danymi - Znajomość podstaw baz danych jest kluczowa dla tworzenia funkcjonalnych aplikacji webowych - Prawidłowe projektowanie baz danych wpływa na wydajność i skalowalność aplikacji

JĘZYKI BACKENDOWE

Języki backendowe to języki programowania działające po stronie serwera, które przetwarzają żądania, komunikują się z bazami danych i generują dynamiczne treści dla stron internetowych.

Kluczowe cechy: - Wykonują się na serwerze, a nie w przeglądarce użytkownika - Generują dynamiczną zawartość HTML w odpowiedzi na żądania - Komunikują się z bazami danych i innymi serwisami - Implementują logikę biznesową i przetwarzanie danych - Zarządzają sesjami, uwierzytelnianiem i autoryzacją

Popularne języki backendowe:

1. PHP

- Najbardziej rozpowszechniony język do tworzenia stron WWW
- Łatwy w nauce i wdrożeniu
- Wspiera większość popularnych CMS-ów (WordPress, Drupal, Joomla)
- Frameworki: Laravel, Symfony, CodeIgniter

2. Node.js (JavaScript)

- Pozwala na używanie JavaScript zarówno na frontendzie, jak i backendzie
- Asynchroniczny, zorientowany na wydarzenia
- Świetny do aplikacji w czasie rzeczywistym
- Frameworki: Express.js, Nest.js, Koa.js

3. Python

- Czytelna składnia, łatwy do nauczenia
- Wszechstronny, używany również w nauce danych, AI
- Frameworki: Django, Flask, FastAPI

4. Ruby

- Zorientowany obiektowo, elegancka składnia
- Nacisk na konwencje ponad konfigurację
- Framework: Ruby on Rails

5. Java

- Silnie typowany, obiektowy język
- Wysoka wydajność, stabilność i skalowalność
- Frameworki: Spring, Jakarta EE

6. C# (.NET)

- Rozwijany przez Microsoft
- Dobrze zintegrowany z ekosystemem Microsoft
- Framework: ASP.NET Core

Przykłady kodu w różnych językach backendowych:

PHP (prosta strona z listą postów):

```
<?php
// Połączenie z bazą danych
$db = new PDO('mysql:host=localhost;dbname=blog;charset=utf8mb4', 'username', 'password');

// Pobranie postów
$stmt = $db->query('SELECT * FROM posts WHERE status = "published" ORDER BY published_at DESC LIMIT 10');
$posts = $stmt->fetchAll(PDO::FETCH_ASSOC);
?>

<!DOCTYPE html>
<html>
<head>
    <title>Mój Blog</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <h1>Mój Blog</h1>
        <nav>
            <ul>
                <li><a href="index.php">Strona główna
            </a></li>
                <li><a href="about.php">O mnie</a></li>
                <li><a href="contact.php">Kontakt</a>
            </li>
            </ul>
```

```

        </nav>
    </header>

    <main>
        <section class="posts">
            <?php if (empty($posts)): ?>
                <p>Brak postów do wyświetlenia.</p>
            <?php else: ?>
                <?php foreach ($posts as $post): ?>
                    <article class="post">
                        <h2><?= htmlspecialchars($pos
t['title']) ?></h2>
                        <div class="meta">
                            Opublikowany: <?= date('d
.m.Y', strtotime($post['published_at'])) ?>
                        </div>
                        <div class="excerpt">
                            <?= htmlspecialchars($pos
t['excerpt']) ?>
                        </div>
                        <a href="post.php?id=<?= $pos
t['id'] ?>" class="read-more">Czytaj więcej</a>
                    </article>
                <?php endforeach; ?>
            <?php endif; ?>
        </section>

        <aside>
            <!-- Sidebar z kategoriami, tagami, itp.
-->
        </aside>
    </main>

    <footer>
        <p>&copy; <?= date('Y') ?> Mój Blog. Wszelkie
prawa zastrzeżone.</p>
    </footer>
</body>
</html>

```


Node.js z Express (prosta aplikacja API):

```
const express = require('express');
const mysql = require('mysql2/promise');
const bodyParser = require('body-parser');

const app = express();
const port = 3000;

// Middleware
app.use(bodyParser.json());

// Konfiguracja bazy danych
const dbConfig = {
  host: 'localhost',
  user: 'username',
  password: 'password',
  database: 'blog'
};

// Tworzenie puli połączeń
const pool = mysql.createPool(dbConfig);

// Trasa do pobierania postów
app.get('/api/posts', async (req, res) => {
  try {
    const [rows] = await pool.query(
      'SELECT id, title, excerpt, published_at FROM posts WHERE status = ? ORDER BY published_at DESC LIMIT 10',
      ['published']
    );

    res.json({
      success: true,
      data: rows
    });
  } catch (error) {
    res.status(500).json({
```

```

        success: false,
        message: 'Wystąpił błąd podczas pobierania post
ów',
        error: error.message
    });
}
});

// Trasa do pobierania pojedynczego posta
app.get('/api/posts/:id', async (req, res) => {
    try {
        const [rows] = await pool.query(
            'SELECT * FROM posts WHERE id = ? AND status =
?',
            [req.params.id, 'published']
        );

        if (rows.length === 0) {
            return res.status(404).json({
                success: false,
                message: 'Post nie został znaleziony'
            });
        }

        res.json({
            success: true,
            data: rows[0]
        });
    } catch (error) {
        res.status(500).json({
            success: false,
            message: 'Wystąpił błąd podczas pobierania post
a',
            error: error.message
        });
    }
});

```

// Trasa do tworzenia nowego posta (z autoryzacją)

```

app.post('/api/posts', async (req, res) => {
  // Tutaj byłaby weryfikacja autoryzacji

  const { title, content, excerpt, status } = req.body;

  if (!title || !content) {
    return res.status(400).json({
      success: false,
      message: 'Tytuł i treść są wymagane'
    });
  }

  try {
    const [result] = await pool.query(
      'INSERT INTO posts (user_id, title, content, excerpt, status, published_at) VALUES (?, ?, ?, ?, ?, ?)',
      [1, title, content, excerpt, status || 'draft',
      status === 'published' ? new Date() : null]
    );

    res.status(201).json({
      success: true,
      message: 'Post został utworzony',
      data: {
        id: result.insertId
      }
    });
  } catch (error) {
    res.status(500).json({
      success: false,
      message: 'Wystąpił błąd podczas tworzenia posta',
      error: error.message
    });
  }
});

```

```
// Uruchomienie serwera
app.listen(port, () => {
  console.log(`Serwer działa na porcie ${port}`);
});
```

Python z Flask:

```
from flask import Flask, render_template, request, jsonify
import mysql.connector
from mysql.connector import Error
import datetime
```

```
app = Flask(__name__)
```

```
# Konfiguracja bazy danych
db_config = {
  'host': 'localhost',
  'user': 'username',
  'password': 'password',
  'database': 'blog'
}
```

```
# Funkcja do połączenia z bazą
def get_db_connection():
  try:
    conn = mysql.connector.connect(**db_config)
    return conn
  except Error as e:
    print(f"Błąd połączenia z bazą danych: {e}")
    return None
```

```
# Strona główna z listą postów
@app.route('/')
def home():
  conn = get_db_connection()
  if conn:
    cursor = conn.cursor(dictionary=True)
    cursor.execute("""
      SELECT id, title, excerpt, published_at
```

```

        FROM posts
        WHERE status = 'published'
        ORDER BY published_at DESC
        LIMIT 10
    """)
    posts = cursor.fetchall()
    cursor.close()
    conn.close()
    return render_template('home.html', posts=posts)
ts)
else:
    return render_template('error.html', message=
"Błąd połączenia z bazą danych")

# Pojedynczy post
@app.route('/post/<int:post_id>')
def post(post_id):
    conn = get_db_connection()
    if conn:
        cursor = conn.cursor(dictionary=True)
        cursor.execute("""
            SELECT p.*, u.username
            FROM posts p
            JOIN users u ON p.user_id = u.id
            WHERE p.id = %s AND p.status = 'published
,

        """, (post_id,))
        post = cursor.fetchone()

    # Zwiększenie licznika wyświetleń
    if post:
        cursor.execute("UPDATE posts SET view_cou
nt = view_count + 1 WHERE id = %s", (post_id,))
        conn.commit()

        cursor.close()
        conn.close()

    if post:

```

```

        return render_template('post.html', post=
post)
    else:
        return render_template('error.html', mess
age="Post nie został znaleziony"), 404
    else:
        return render_template('error.html', message=
"Błąd połączenia z bazą danych")

# API do pobierania postów
@app.route('/api/posts')
def api_posts():
    conn = get_db_connection()
    if conn:
        cursor = conn.cursor(dictionary=True)
        cursor.execute("""
            SELECT id, title, excerpt, published_at
            FROM posts
            WHERE status = 'published'
            ORDER BY published_at DESC
            LIMIT 10
        """)
        posts = cursor.fetchall()

        # Konwersja dat do formatu JSON
        for post in posts:
            if post['published_at']:
                post['published_at'] = post['publishe
d_at'].isoformat()

        cursor.close()
        conn.close()
        return jsonify({'success': True, 'data': post
s})
    else:
        return jsonify({'success': False, 'message':
'Błąd połączenia z bazą danych'}), 500

```

```
if __name__ == '__main__':  
    app.run(debug=True)
```

Dobre praktyki dla kodowania backendowego: - Stosuj zasady bezpieczeństwa (walidacja danych wejściowych, parametryzowane zapytania) - Organizuj kod według wzorców projektowych (MVC, Repository, Service) - Stosuj zasadę DRY (Don't Repeat Yourself) - Implementuj obsługę błędów i logowanie - Używaj kontroli wersji (Git) - Pisz testy jednostkowe i integracyjne - Optymalizuj zapytania do bazy danych - Stosuj asynchroniczne operacje dla zadań intensywnych obliczeniowo - Wdrażaj caching w celu poprawy wydajności - Regularnie aktualizuj zależności i biblioteki

Dlaczego są ważne dla początkujących: - Języki backendowe umożliwiają tworzenie dynamicznych, interaktywnych stron - Pozwalają na implementację funkcji wymagających przetwarzania danych - Są niezbędne do integracji z bazami danych i zewnętrznymi API - Zapewniają bezpieczeństwo i kontrolę dostępu do danych

CMS (CONTENT MANAGEMENT SYSTEM)

CMS to system zarządzania treścią, który umożliwia tworzenie, edytowanie i zarządzanie treścią cyfrową poprzez interfejs użytkownika, bez konieczności bezpośredniej pracy z kodem.

Kluczowe cechy: - Interfejs administracyjny do zarządzania treścią - Oddzielenie treści od prezentacji - Zarządzanie użytkownikami i uprawnieniami - System szablonów i motywów - Rozszerzalność przez wtyczki i dodatki

Popularne systemy CMS:

1. WordPress

- Najbardziej popularny CMS na świecie (ok. 40% wszystkich stron)

- Początkowo system blogowy, obecnie wszechstronna platforma
- Duża społeczność, tysiące wtyczek i motywów
- Łatwy w instalacji i konfiguracji
- Dobry wybór dla blogów, stron firmowych, portfolio, e-commerce (WooCommerce)

2. **Drupal**

- Bardziej techniczny i złożony
- Wysoka skalowalność i elastyczność
- Zaawansowane możliwości zarządzania treścią
- Rozbudowany system ról i uprawnień
- Dobry wybór dla złożonych, dużych portali, stron rządowych, edukacyjnych

3. **Joomla**

- Pośredni poziom złożoności między WordPress a Drupal
- Dobra równowaga między elastycznością a łatwością obsługi
- Zaawansowane zarządzanie wielojęzycznością
- Dobre rozwiązanie dla stron społecznościowych, intranetów

4. **Wix, Squarespace, Webflow**

- CMS-y typu “drag and drop”
- Nie wymagają znajomości kodowania
- Hostowane rozwiązania (SaaS)
- Ograniczona elastyczność, ale szybkie wdrożenie
- Dobre dla małych stron, portfolio, prostych sklepów

5. **Headless CMS (Strapi, Contentful, Sanity)**

- Systemy zapewniające tylko backend dla treści
- API-first (dostarczają treść przez API)
- Oddzielenie backendu od frontendu

- Elastyczność w wyborze technologii frontendowych
- Dobre dla projektów wieloplatformowych (web, mobile, IoT)

Elementy CMS:

1. Panel administracyjny

- Logowanie i zarządzanie kontem
- Dashboard z przeglądem systemu
- Edytory treści (wizualne, tekstowe)
- Zarządzanie mediami (obrazy, pliki)
- Konfiguracja systemu i ustawienia

2. System zarządzania treścią

- Tworzenie, edycja, publikowanie treści
- Kategoryzacja i tagowanie
- Planowanie publikacji
- Wersjonowanie treści
- Komentarze i moderacja

3. System szablonów

- Motywy/szablony definiujące wygląd
- Układy stron (layouts)
- Widżety i bloki treści
- Responsywny design

4. System rozszerzeń

- Wtyczki/moduły/dodatki
- Własne typy treści
- Integracje z zewnętrznymi serwisami
- Funkcje specjalne (formularze, galerie, etc.)

5. System użytkowników

- Rejestracja i logowanie
- Role i uprawnienia
- Profile użytkowników

- Kontrola dostępu do treści

WordPress - instalacja i konfiguracja:

1. Wymagania systemowe:

- PHP 7.4 lub nowszy
- MySQL 5.7 lub nowszy / MariaDB 10.2 lub nowszy
- Serwer HTTP (Apache, Nginx)

2. Instalacja:

1. Pobierz WordPress z wordpress.org
2. Utwórz bazę danych MySQL
3. Rozpakuj pliki WordPress do katalogu głównego serwera
4. Otwórz stronę w przeglądarce i wykonaj instalację:
 - a. Podaj dane połączenia z bazą danych
 - b. Ustaw nazwę witryny, login i hasło administratora

3. Podstawowa konfiguracja:

1. Ustawienia ogólne: nazwa witryny, adres URL, opis
2. Ustawienia wyświetlania: liczba postów na stronie, format daty
3. Ustawienia permalinków: struktura przyjaznych adresów URL
4. Instalacja i aktywacja motywu
5. Instalacja niezbędnych wtyczek

Struktura motywu WordPress:

```
theme-name/  
├── style.css           # Główny plik stylu, zawiera  
a metadane motywu  
├── index.php          # Główny szablon  
├── header.php         # Nagłówek strony  
└── footer.php         # Stopka strony
```

u	sidebar.php	# Pasek boczny
	functions.php	# Funkcje i hooki motywu
	single.php	# Szablon pojedynczego wpis
ie, tagi)	page.php	# Szablon strony
	archive.php	# Szablon archiwum (kategor
	404.php	# Szablon błędu 404
	search.php	# Szablon wyników wyszukiwa
nia	comments.php	# Szablon komentarzy
1200x900px)	screenshot.png	# Obrazek podglądu motywu (
	assets/	
	css/	# Dodatkowe pliki CSS
	js/	# Pliki JavaScript
	images/	# Obrazy używane w motywie
	template-parts/	# Fragmenty szablonów do po
nownego użycia	content.php	# Szablon treści wpisu
	content-page.php	# Szablon treści strony
	content-none.php	# Szablon "brak treści"

Przykład szablonu WordPress (single.php):

```
<?php get_header(); ?>

<main id="primary" class="site-main">

    <?php
    while ( have_posts() ) :
        the_post();

        get_template_part( 'template-parts/content',
get_post_type() );

        // Nawigacja między wpisami
        the_post_navigation(
            array(
                'prev_text' => '<span class="nav-subt
```

```

itle">' . esc_html__( 'Poprzedni:', 'mytheme' ) . '</
span> <span class="nav-title">%title</span>',
        'next_text' => '<span class="nav-subt
itle">' . esc_html__( 'Następny:', 'mytheme' ) . '</s
pan> <span class="nav-title">%title</span>',
    );

    // Komentarze
    if ( comments_open() || get_comments_number()
) :
        comments_template();
    endif;

endwhile;
?>

</main>

<?php get_sidebar(); ?>
<?php get_footer(); ?>

```

Główne hooki WordPress:

```

// Dodanie stylu i skryptu
function mytheme_scripts() {
    wp_enqueue_style( 'mytheme-style', get_stylesheet
_uri(), array(), '1.0.0' );
    wp_enqueue_script( 'mytheme-navigation', get_temp
late_directory_uri() . '/assets/js/navigation.js', ar
ray(), '1.0.0', true );
}
add_action( 'wp_enqueue_scripts', 'mytheme_scripts' )
;

// Rejestracja menu
function mytheme_menus() {
    register_nav_menus(
        array(
            'primary-menu' => esc_html__( 'Menu główn

```

```

e', 'mytheme' ),
        'footer-menu' => esc_html__( 'Menu w sto
pce', 'mytheme' ),
    )
);
}
add_action( 'init', 'mytheme_menus' );

// Rejestracja panelu widgetów (sidebary)
function mytheme_widgets_init() {
    register_sidebar(
        array(
            'name' => esc_html__( 'Sidebar',
'mytheme' ),
            'id' => 'sidebar-1',
            'description' => esc_html__( 'Dodaj wid
gety tutaj.', 'mytheme' ),
            'before_widget' => '<section id="%1$s" cl
ass="widget %2$s">',
            'after_widget' => '</section>',
            'before_title' => '<h2 class="widget-tit
le">',
            'after_title' => '</h2>',
        )
    );
}
add_action( 'widgets_init', 'mytheme_widgets_init' );

// Dodanie własnych typów treści
function mytheme_custom_post_types() {
    register_post_type(
        'portfolio',
        array(
            'labels' => array(
                'name' => esc_html__( 'Portf
olio', 'mytheme' ),
                'singular_name' => esc_html__( 'Proje
kt', 'mytheme' ),
            ),
        ),
    );
}

```

```

        'public'           => true,
        'has_archive'      => true,
        'menu_icon'        => 'dashicons-portfolio',
        'supports'         => array( 'title', 'editor'
, 'thumbnail', 'excerpt' ),
        'rewrite'          => array( 'slug' => 'portfo
lio' ),
    )
    );
}
add_action( 'init', 'mytheme_custom_post_types' );

// Hook po załadowaniu motywu
function mytheme_setup() {
    // Dodanie wsparcia dla tłumaczeń
    load_theme_textdomain( 'mytheme', get_template_di
rectory() . '/languages' );

    // Dodanie wsparcia dla miniatur wpisów
    add_theme_support( 'post-thumbnails' );

    // Dodanie wsparcia dla RSS
    add_theme_support( 'automatic-feed-links' );

    // Dodanie wsparcia dla tytułu strony
    add_theme_support( 'title-tag' );

    // Dodanie wsparcia dla HTML5
    add_theme_support(
        'html5',
        array(
            'search-form',
            'comment-form',
            'comment-list',
            'gallery',
            'caption',
            'style',
            'script',
        )
    );
}

```

```

    );
}
add_action( 'after_setup_theme', 'mytheme_setup' );

// Modyfikacja zapytania głównego
function mytheme_modify_query( $query ) {
    if ( is_home() && $query->is_main_query() ) {
        $query->set( 'posts_per_page', 6 );
        $query->set( 'category__not_in', array( 5 ) )
    ; // Wyklucz kategorię o ID 5
    }
}
add_action( 'pre_get_posts', 'mytheme_modify_query' )
;

```

Tworzenie wtyczki WordPress:

<?php

```

/**
 * Plugin Name: Moja Przykładowa Wtyczka
 * Plugin URI: https://przyklad.pl/moja-wtyczka
 * Description: Przykładowa wtyczka demonstrująca pod
stawową funkcjonalność.
 * Version: 1.0.0
 * Author: Twoje Imię
 * Author URI: https://przyklad.pl
 * Text Domain: moja-wtyczka
 * Domain Path: /languages
 */

// Zabezpieczenie przed bezpośrednim dostępem
if ( ! defined( 'ABSPATH' ) ) {
    exit; // Exit if accessed directly
}

// Definicje stałych
define( 'MPW_VERSION', '1.0.0' );
define( 'MPW_PLUGIN_DIR', plugin_dir_path( __FILE__ )
);
define( 'MPW_PLUGIN_URL', plugin_dir_url( __FILE__ )

```

```

);

// Funkcja aktywacji wtyczki
function mpw_activate() {
    // Kod wykonywany przy aktywacji
    flush_rewrite_rules();
}
register_activation_hook( __FILE__, 'mpw_activate' );

// Funkcja deaktywacji wtyczki
function mpw_deactivate() {
    // Kod wykonywany przy deaktywacji
    flush_rewrite_rules();
}
register_deactivation_hook( __FILE__, 'mpw_deactivate' );

// Dodanie menu w panelu administracyjnym
function mpw_add_admin_menu() {
    add_menu_page(
        'Moja Wtyczka',           // Tytuł strony
        'Moja Wtyczka',           // Nazwa w menu
        'manage_options',         // Uprawnienia
        'moja-wtyczka',           // Slug
        'mpw_admin_page_content', // Funkcja wyświetl
        'dashicons-admin-generic', // Ikona
        20,                       // Pozycja w menu
    );
}
add_action( 'admin_menu', 'mpw_add_admin_menu' );

// Funkcja wyświetlająca zawartość strony administrac
yjnej
function mpw_admin_page_content() {
    // Sprawdzanie uprawnień
    if ( ! current_user_can( 'manage_options' ) ) {
        return;
    }
}

```



```

// Zapisywanie ustawień
if ( isset( $_POST['mpw_save_settings'] ) && check_admin_referer( 'mpw_settings_nonce' ) ) {
    $option_value = sanitize_text_field( $_POST['mpw_option'] );
    update_option( 'mpw_option', $option_value );

    echo '<div class="notice notice-success is-dismissible"><p>Ustawienia zostały zapisane.</p></div>';
}

// Pobranie aktualnych ustawień
$option_value = get_option( 'mpw_option', 'domyśl na wartość' );

// Wyświetlenie formularza
?>
<div class="wrap">
    <h1><?php echo esc_html( get_admin_page_title() ); ?></h1>
    <form method="post" action="">
        <?php wp_nonce_field( 'mpw_settings_nonce' ); ?>
        <table class="form-table">
            <tr>
                <th scope="row">
                    <label for="mpw_option">Przykładowa opcja</label>
                </th>
                <td>
                    <input type="text" id="mpw_option" name="mpw_option" value="<?php echo esc_attr( $option_value ); ?>" class="regular-text">
                    <p class="description">To jest przykładowa opcja konfiguracyjna.</p>
                </td>
            </tr>
        </table>
    </div>

```

```

        </table>

        <?php submit_button( 'Zapisz ustawienia',
'primary', 'mpw_save_settings' ); ?>
    </form>
</div>
<?php
}

// Dodanie shortcode
function mpw_example_shortcode( $atts ) {
    // Domyślne atrybuty
    $atts = shortcode_atts(
        array(
            'title' => 'Domyślny tytuł',
            'count' => 5,
        ),
        $atts,
        'mpw_example'
    );

    // Pobranie wpisów
    $posts = get_posts(
        array(
            'posts_per_page' => intval( $atts['count']
] ),
            'post_type'      => 'post',
        )
    );

    // Rozpoczęcie buforowania wyjścia
    ob_start();

    // Wygenerowanie HTML
    echo '<div class="mpw-shortcode-container">';
    echo '<h2>' . esc_html( $atts['title'] ) . '</h2>';
';
    echo '<ul>';

```

```

        foreach ( $posts as $post ) {
            setup_postdata( $post );
            echo '<li><a href="' . esc_url( get_permalink
( $post ) ) . '"' . esc_html( get_the_title( $post )
) . '</a></li>';
        }

        echo '</ul>';
        echo '</div>';

        wp_reset_postdata();

        // Zwrócenie buforowanego HTML
        return ob_get_clean();
    }
}
add_shortcode( 'mpw_example', 'mpw_example_shortcode'
);

// Dodanie własnych stylów i skryptów
function mpw_enqueue_scripts() {
    wp_enqueue_style( 'mpw-style', MPW_PLUGIN_URL . '
assets/css/style.css', array(), MPW_VERSION );
    wp_enqueue_script( 'mpw-script', MPW_PLUGIN_URL .
'assets/js/script.js', array( 'jquery' ), MPW_VERSION
N, true );

    // Przekazanie zmiennych do JavaScript
    wp_localize_script(
        'mpw-script',
        'mpwData',
        array(
            'ajax_url' => admin_url( 'admin-ajax.php'
),
            'nonce'     => wp_create_nonce( 'mpw-ajax-
nonce' ),
            'some_var' => get_option( 'mpw_option' ),
        )
    );
}

```

```

add_action( 'wp_enqueue_scripts', 'mpw_enqueue_scripts' );

// Obsługa AJAX
function mpw_ajax_handler() {
    // Sprawdzenie nonce
    check_ajax_referer( 'mpw-ajax-nonce', 'nonce' );

    // Przetwarzanie danych
    $data = isset( $_POST['data'] ) ? sanitize_text_field( $_POST['data'] ) : '';

    // Zwrócenie odpowiedzi
    wp_send_json_success(
        array(
            'message' => 'Dane otrzymane: ' . $data,
            'status'   => 'success',
        )
    );

    // Zawsze zakończ obsługę AJAX
    wp_die();
}
add_action( 'wp_ajax_mpw_ajax_action', 'mpw_ajax_handler' ); // Dla zalogowanych użytkowników
add_action( 'wp_ajax_nopriv_mpw_ajax_action', 'mpw_ajax_handler' ); // Dla niezalogowanych użytkowników

```

Dobre praktyki dla CMS: - Wybieraj CMS odpowiedni do potrzeb projektu - Regularnie aktualizuj CMS, motywy i wtyczki - Korzystaj z motywów i wtyczek z zaufanych źródeł - Twórz kopie zapasowe przed dokonywaniem zmian - Minimalizuj liczbę wtyczek/modułów (każda wpływa na wydajność) - Używaj rozwiązań typu “child theme” w WordPress - Stosuj zabezpieczenia odpowiednie dla CMS (np. captcha, limit logowań) - Optymalizuj media i szybkość ładowania - Testuj zmiany na środowisku testowym przed wdrożeniem na produkcję - Korzystaj z dokumentacji i społeczności CMS

Dlaczego są ważne dla początkujących: - CMS umożliwiają zarządzanie treścią bez znajomości kodowania - Przyspieszają proces tworzenia stron i aplikacji webowych - Oferują gotowe rozwiązania dla typowych funkcjonalności - Zapewniają aktualizacje bezpieczeństwa i nowe funkcje - Mają duże społeczności i dostępne zasoby edukacyjne

8. Funkcje i usługi

SYSTEM LOGOWANIA

System logowania umożliwia użytkownikom uwierzytelnianie się na stronie internetowej, zapewniając dostęp do funkcji i treści chronionych oraz personalizację doświadczenia.

Kluczowe cechy: - Rejestracja nowych użytkowników - Uwierzytelnianie (logowanie) - Zarządzanie sesją - Resetowanie i odzyskiwanie hasła - Role i uprawnienia użytkowników - Ochrona przed atakami

Komponenty systemu logowania:

1. Formularz rejestracji:

```
<form id="register-form" method="post" action="register.php">
  <div class="form-group">
    <label for="username">Nazwa użytkownika</label>
    <input type="text" id="username" name="username"
required minlength="4" maxlength="20" pattern="[a-zA-Z0-9_]+">
    <small>Dozwolone znaki: litery, cyfry i podkreślnik. Długość: 4-20 znaków.</small>
  </div>

  <div class="form-group">
    <label for="email">Adres email</label>
    <input type="email" id="email" name="email" required>
```

```

</div>

<div class="form-group">
  <label for="password">Hasło</label>
  <input type="password" id="password" name="password" required minlength="8">
  <small>Minimum 8 znaków, w tym: małe i wielkie litery, cyfry, znaki specjalne.</small>
</div>

<div class="form-group">
  <label for="confirm-password">Potwierdź hasło</label>
  <input type="password" id="confirm-password" name="confirm_password" required>
</div>

<div class="form-group checkbox">
  <input type="checkbox" id="terms" name="terms" required>
  <label for="terms">Akceptuję <a href="/terms">regulamin</a> i <a href="/privacy">politykę prywatności</a></label>
</div>

<div class="form-group">
  <div class="g-recaptcha" data-sitekey="YOUR_RECAPTCHA_SITE_KEY"></div>
</div>

<button type="submit" class="btn btn-primary">Zarejestruj się</button>
</form>

```

2. Formularz logowania:

```

<form id="login-form" method="post" action="login.php">
  <div class="form-group">
    <label for="login-username">Nazwa użytkownika lub

```

```

email</label>
    <input type="text" id="login-username" name="login_identifier" required>
</div>

    <div class="form-group">
        <label for="login-password">Hasło</label>
        <input type="password" id="login-password" name="login_password" required>
    </div>

    <div class="form-group checkbox">
        <input type="checkbox" id="remember-me" name="remember_me">
        <label for="remember-me">Zapamiętaj mnie</label>
    </div>

    <div class="form-actions">
        <button type="submit" class="btn btn-primary">Zaloguj się</button>
        <a href="forgot-password.php" class="forgot-password">Zapomniałem hasła</a>
    </div>
</form>

```

3. Obsługa rejestracji (PHP):

```

<?php
// Połączenie z bazą danych
require_once 'config.php';
require_once 'functions.php';

// Sprawdzenie, czy formularz został wysłany
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Pobranie i sanitizacja danych
    $username = filter_input(INPUT_POST, 'username',
        FILTER_SANITIZE_STRING);
    $email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
    $password = $_POST['password'];

```

```

$confirm_password = $_POST['confirm_password'];
$terms = isset($_POST['terms']) ? true : false;

$errors = [];

// Walidacja nazwy użytkownika
if (empty($username) || strlen($username) < 4 ||
    strlen($username) > 20 || !preg_match('/^[a-zA-Z0-9_]+$', $username)) {
    $errors[] = "Nazwa użytkownika musi mieć od 4
do 20 znaków i zawierać tylko litery, cyfry i podkre
ślniki.";
}

// Sprawdzenie, czy nazwa użytkownika jest już za
jęta
if (is_username_taken($username)) {
    $errors[] = "Ta nazwa użytkownika jest już za
jęta.";
}

// Walidacja email
if (empty($email) || !filter_var($email, FILTER_V
ALIDATE_EMAIL)) {
    $errors[] = "Podaj poprawny adres email.";
}

// Sprawdzenie, czy email jest już zajęty
if (is_email_taken($email)) {
    $errors[] = "Ten adres email jest już używany
.";
}

// Walidacja hasła
if (empty($password) || strlen($password) < 8) {
    $errors[] = "Hasło musi mieć co najmniej 8 zn
aków.";
} elseif (!preg_match('/[A-Z]/', $password) || !p
reg_match('/[a-z]/', $password) ||

```



```

        !preg_match('/[0-9]/', $password) || !preg_match('/[^A-Za-z0-9]/', $password)) {
            $errors[] = "Hasło musi zawierać małe i wielkie litery, cyfry oraz znaki specjalne.";
        }

        // Sprawdzenie zgodności haseł
        if ($password !== $confirm_password) {
            $errors[] = "Hasła nie są identyczne.";
        }

        // Sprawdzenie akceptacji regulaminu
        if (!$terms) {
            $errors[] = "Musisz zaakceptować regulamin i politykę prywatności.";
        }

        // Weryfikacja CAPTCHA
        $recaptcha_secret = 'YOUR_RECAPTCHA_SECRET_KEY';
        $recaptcha_response = $_POST['g-recaptcha-response'];

        $verify_response = file_get_contents('https://www.google.com/recaptcha/api/siteverify?secret='.$recaptcha_secret.'&response='.$recaptcha_response);
        $response_data = json_decode($verify_response);

        if (!$response_data->success) {
            $errors[] = "Weryfikacja CAPTCHA nie powiodła się. Spróbuj ponownie.";
        }

        // Jeśli nie ma błędów, zarejestruj użytkownika
        if (empty($errors)) {
            // Hashowanie hasła
            $hashed_password = password_hash($password, PASSWORD_DEFAULT);

            // Generowanie tokenu aktywacyjnego

```

```

$activation_token = bin2hex(random_bytes(32))
;

// Zapisanie użytkownika w bazie danych
$user_id = register_user($username, $email, $
hashed_password, $activation_token);

if ($user_id) {
    // Wysłanie emaila aktywacyjnego
    send_activation_email($email, $username,
$activation_token);

    // Przekierowanie do strony potwierdzenia
    header('Location: registration-success.ph
p');
    exit;
} else {
    $errors[] = "Wystąpił problem podczas rej
estracji. Spróbuj ponownie później.";
}
}
?>

```

4. Obsługa logowania (PHP):

```

<?php
// Połączenie z bazą danych
require_once 'config.php';
require_once 'functions.php';

// Rozpoczęcie sesji
session_start();

// Sprawdzenie, czy użytkownik jest już zalogowany
if (isset($_SESSION['user_id'])) {
    header('Location: dashboard.php');
    exit;
}

```

```

// Sprawdzenie, czy formularz został wysłany
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Pobranie danych
    $login_identifier = $_POST['login_identifier'];
    $password = $_POST['login_password'];
    $remember_me = isset($_POST['remember_me']) ? true
e : false;

    // Walidacja danych
    if (empty($login_identifier) || empty($password))
    {
        $error = "Wszystkie pola są wymagane.";
    } else {
        // Wyszukanie użytkownika
        $user = get_user_by_identifier($login_identif
ier);

        if ($user && $user['is_active'] && password_v
erify($password, $user['password'])) {
            // Ustawienie danych sesji
            $_SESSION['user_id'] = $user['id'];
            $_SESSION['username'] = $user['username']
;

            $_SESSION['email'] = $user['email'];
            $_SESSION['role'] = $user['role'];

            // Aktualizacja ostatniego logowania
            update_last_login($user['id']);

            // Obsługa "zapamiętaj mnie"
            if ($remember_me) {
                $token = bin2hex(random_bytes(32));
                $expire = time() + (30 * 24 * 60 * 60
); // 30 dni

                // Zapisanie tokena w bazie danych
                save_remember_token($user['id'], $tok
en, $expire);

```

```

        // Ustawienie ciasteczka
        setcookie('remember_token', $token, $
expire, '/', '', true, true);
        setcookie('user_id', $user['id'], $ex
pire, '/', '', true, true);
    }

    // Przekierowanie do panelu użytkownika
    header('Location: dashboard.php');
    exit;
} else {
    // Niepoprawne dane logowania
    $error = "Niepoprawna nazwa użytkownika/e
mail lub hasło.";

    // Logowanie nieudanej próby logowania
    log_failed_login_attempt($login_identifie
r, $_SERVER['REMOTE_ADDR']);
}
}

// Sprawdzenie, czy użytkownik ma zapisane ciasteczko
"zapamiętaj mnie"
if (!isset($_SESSION['user_id']) && isset($_COOKIE['r
emember_token']) && isset($_COOKIE['user_id'])) {
    $token = $_COOKIE['remember_token'];
    $user_id = $_COOKIE['user_id'];

    // Weryfikacja tokena
    $user = verify_remember_token($user_id, $token);

    if ($user) {
        // Ustawienie danych sesji
        $_SESSION['user_id'] = $user['id'];
        $_SESSION['username'] = $user['username'];
        $_SESSION['email'] = $user['email'];
        $_SESSION['role'] = $user['role'];
    }
}

```

```

        // Aktualizacja ostatniego logowania
        update_last_login($user['id']);

        // Przekierowanie do panelu użytkownika
        header('Location: dashboard.php');
        exit;
    }
}
?>

```

5. Funkcje pomocnicze (PHP):

```

<?php
// Funkcja rejestrująca użytkownika
function register_user($username, $email, $password,
$activation_token) {
    global $db;

    try {
        $query = "INSERT INTO users (username, email,
            password, activation_token, created_at)
                VALUES (:username, :email, :password,
            d, :activation_token, NOW())";

        $stmt = $db->prepare($query);
        $stmt->bindParam(':username', $username);
        $stmt->bindParam(':email', $email);
        $stmt->bindParam(':password', $password);
        $stmt->bindParam(':activation_token', $activation_token);
        $stmt->execute();

        return $db->lastInsertId();
    } catch (PDOException $e) {
        error_log("Database error: " . $e->getMessage());
        return false;
    }
}

```

// Funkcja sprawdzająca, czy nazwa użytkownika jest zajęta

```
function is_username_taken($username) {  
    global $db;  
  
    $query = "SELECT COUNT(*) FROM users WHERE username = :username";  
    $stmt = $db->prepare($query);  
    $stmt->bindParam(':username', $username);  
    $stmt->execute();  
  
    return $stmt->fetchColumn() > 0;  
}
```

// Funkcja sprawdzająca, czy email jest zajęty

```
function is_email_taken($email) {  
    global $db;  
  
    $query = "SELECT COUNT(*) FROM users WHERE email = :email";  
    $stmt = $db->prepare($query);  
    $stmt->bindParam(':email', $email);  
    $stmt->execute();  
  
    return $stmt->fetchColumn() > 0;  
}
```

// Funkcja wysyłająca email aktywacyjny

```
function send_activation_email($email, $username, $token) {  
    $subject = "Aktywacja konta";  
    $activation_link = "https://example.com/activate.php?token=" . $token;  
  
    $message = "Witaj {$username},\n\n";  
    $message .= "Dziękujemy za rejestrację. Aby aktywować konto, kliknij w poniższy link:\n";  
    $message .= $activation_link . "\n\n";  
    $message .= "Jeśli nie rejestrowałeś się na nasze
```

```

j stronie, zignoruj ten email.\n\n";
$message .= "Pozdrawiamy,\nZespół Example.com";

$headers = "From: noreply@example.com\r\n";
$headers .= "Reply-To: support@example.com\r\n";

return mail($email, $subject, $message, $headers)
;
}

```

// Funkcja wyszukiwająca użytkownika po nazwie lub emailu

```

function get_user_by_identifier($identifier) {
    global $db;

    $query = "SELECT * FROM users WHERE username = :identifier OR email = :identifier LIMIT 1";
    $stmt = $db->prepare($query);
    $stmt->bindParam(':identifier', $identifier);
    $stmt->execute();

    return $stmt->fetch(PDO::FETCH_ASSOC);
}

```

// Funkcja aktualizująca datę ostatniego logowania

```

function update_last_login($user_id) {
    global $db;

    $query = "UPDATE users SET last_login = NOW() WHERE id = :user_id";
    $stmt = $db->prepare($query);
    $stmt->bindParam(':user_id', $user_id);
    $stmt->execute();
}

```

// Funkcja zapisująca token "zapamiętaj mnie"

```

function save_remember_token($user_id, $token, $expire) {
    global $db;

```

```

        $query = "INSERT INTO auth_tokens (user_id, token
, expires) VALUES (:user_id, :token, :expires)";
        $stmt = $db->prepare($query);
        $stmt->bindParam(':user_id', $user_id);
        $stmt->bindParam(':token', $token);
        $stmt->bindParam(':expires', date('Y-m-d H:i:s',
$expire));
        $stmt->execute();
    }

```

// Funkcja weryfikująca token "zapamiętaj mnie"

```

function verify_remember_token($user_id, $token) {
    global $db;

```

```

        $query = "SELECT u.* FROM users u
                JOIN auth_tokens t ON u.id = t.user_id
                WHERE t.user_id = :user_id AND t.token
= :token AND t.expires > NOW()";

```

```

        $stmt = $db->prepare($query);
        $stmt->bindParam(':user_id', $user_id);
        $stmt->bindParam(':token', $token);
        $stmt->execute();

```

```

        return $stmt->fetch(PDO::FETCH_ASSOC);
    }

```

// Funkcja Logująca nieudane próby Logowania

```

function log_failed_login_attempt($identifier, $ip) {
    global $db;

```

```

        $query = "INSERT INTO login_attempts (identifier,
ip_address, attempted_at)
                VALUES (:identifier, :ip, NOW())";

```

```

        $stmt = $db->prepare($query);
        $stmt->bindParam(':identifier', $identifier);
        $stmt->bindParam(':ip', $ip);

```



```

$stmt->execute();

// Sprawdzenie liczby nieudanych prób
$query = "SELECT COUNT(*) FROM login_attempts
        WHERE identifier = :identifier AND ip_address = :ip
        AND attempted_at > DATE_SUB(NOW(), INTERVAL 1 HOUR)";

$stmt = $db->prepare($query);
$stmt->bindParam(':identifier', $identifier);
$stmt->bindParam(':ip', $ip);
$stmt->execute();

$count = $stmt->fetchColumn();

// Jeśli jest zbyt wiele prób, możemy zablokować dostęp
if ($count >= 5) {
    // Blokowanie IP lub identyfikatora na określony czas
}
}

```

6. Obsługa resetowania hasła:

```

<?php
// Połączenie z bazą danych
require_once 'config.php';
require_once 'functions.php';

// Funkcja resetowania hasła - etap 1: formularz żądania resetu
function request_password_reset() {
    global $db;

    if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['email'])) {
        $email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
    }
}

```

```

    )) {
        return "Podaj poprawny adres email.";
    }

    // Sprawdzenie, czy email istnieje w bazie
    $query = "SELECT id, username FROM users WHERE
E email = :email AND is_active = 1 LIMIT 1";
    $stmt = $db->prepare($query);
    $stmt->bindParam(':email', $email);
    $stmt->execute();

    $user = $stmt->fetch(PDO::FETCH_ASSOC);

    if ($user) {
        // Generowanie tokena resetu
        $token = bin2hex(random_bytes(32));
        $expires = date('Y-m-d H:i:s', time() + 3
600); // 1 godzina

        // Zapisanie tokena w bazie
        $query = "INSERT INTO password_resets (us
er_id, token, expires) VALUES (:user_id, :token, :exp
ires)";
        $stmt = $db->prepare($query);
        $stmt->bindParam(':user_id', $user['id'])
;
        $stmt->bindParam(':token', $token);
        $stmt->bindParam(':expires', $expires);
        $stmt->execute();

        // Wysłanie emaila z linkiem do resetu
        $reset_link = "https://example.com/reset-
password.php?token=" . $token;

        $subject = "Reset hasła";
        $message = "Witaj {$user['username']}, \n\
n";
    }

```

```

        $message .= "Otrzymałiśmy prośbę o reset
hasła. Kliknij w poniższy link, aby zresetować hasło:
\n";
        $message .= $reset_link . "\n\n";
        $message .= "Link wygaśnie za godzinę.\n\n";
        $message .= "Jeśli nie prosiłeś o reset h
asła, zignoruj ten email.\n\n";
        $message .= "Pozdrawiamy,\nZespół Example
.com";

        $headers = "From: noreply@example.com\r\n
";
        $headers .= "Reply-To: support@example.co
m\r\n";

        mail($email, $subject, $message, $headers
);

        return true;
    } else {
        // Opóźnienie, aby zapobiec atakom polega
jącym na sprawdzaniu, które emaile istnieją
        sleep(1);
        return true; // Zwracamy true nawet jeśli
email nie istnieje, aby nie ujawniać informacji
    }
}

return false;
}

```

// Funkcja resetowania hasła - etap 2: weryfikacja to
kena i zmiana hasła

```

function reset_password() {
    global $db;

    if (!isset($_GET['token']) || empty($_GET['token']
))) {

```

```

        return "Nieprawidłowy token resetowania hasła
        .";
    }

    $token = $_GET['token'];

    // Sprawdzenie czy token jest ważny
    $query = "SELECT r.user_id, u.username, u.email FROM
    password_resets r
                JOIN users u ON r.user_id = u.id
                WHERE r.token = :token AND r.expires >
    NOW() AND r.used = 0
                LIMIT 1";

    $stmt = $db->prepare($query);
    $stmt->bindParam(':token', $token);
    $stmt->execute();

    $reset = $stmt->fetch(PDO::FETCH_ASSOC);

    if (!$reset) {
        return "Token resetowania hasła jest nieprawid
        łowy lub wygasł.";
    }

    // Jeśli formularz został wysłany
    if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['password']) && isset($_POST['confirm_password'])) {
        $password = $_POST['password'];
        $confirm_password = $_POST['confirm_password'];

        // Walidacja hasła
        if (empty($password) || strlen($password) < 8) {
            return "Hasło musi mieć co najmniej 8 znaków.";
        } elseif (!preg_match('/[A-Z]/', $password) |

```

```

| !preg_match('/[a-z]/', $password) ||
| !preg_match('/[0-9]/', $password) |
| !preg_match('/^[A-Za-z0-9]/', $password)) {
    return "Hasło musi zawierać małe i wielkie litery, cyfry oraz znaki specjalne.";
}

// Sprawdzenie zgodności haseł
if ($password !== $confirm_password) {
    return "Hasła nie są identyczne.";
}

// Hashowanie nowego hasła
$hashed_password = password_hash($password, PASSWORD_DEFAULT);

// Aktualizacja hasła
$query = "UPDATE users SET password = :password WHERE id = :user_id";
$stmt = $db->prepare($query);
$stmt->bindParam(':password', $hashed_password);
$stmt->bindParam(':user_id', $reset['user_id']);
$stmt->execute();

// Oznaczenie tokena jako wykorzystanego
$query = "UPDATE password_resets SET used = 1 WHERE token = :token";
$stmt = $db->prepare($query);
$stmt->bindParam(':token', $token);
$stmt->execute();

// Wysłanie powiadomienia o zmianie hasła
$subject = "Twoje hasło zostało zmienione";
$message = "Witaj {$reset['username']},\n\n";
$message .= "Twoje hasło zostało pomyślnie zmienione.\n\n";
$message .= "Jeśli nie Ty dokonałeś tej zmiany";

```

```

y, skontaktuj się natychmiast z obsługą klienta.\n\n"
;
$message .= "Pozdrawiamy,\nZespół Example.com
";

$headers = "From: noreply@example.com\r\n";
$headers .= "Reply-To: support@example.com\r\n";

mail($reset['email'], $subject, $message, $headers);

return true;
}

return $reset;
}

```

7. JavaScript dla walidacji formularzy:

```

// Walidacja formularza rejestracji
document.addEventListener('DOMContentLoaded', function() {
    const registerForm = document.getElementById('register-form');

    if (registerForm) {
        const username = document.getElementById('username');
        const email = document.getElementById('email');
        const password = document.getElementById('password');
        const confirmPassword = document.getElementById('confirm-password');
        const terms = document.getElementById('terms');

        registerForm.addEventListener('submit', function(event) {

```

```

let isValid = true;

// Walidacja nazwy użytkownika
if (!username.value.match(/^[a-zA-Z0-9_]{4,20}$/)) {
    showError(username, 'Nazwa użytkownika musi mieć od 4 do 20 znaków i zawierać tylko litery, cyfry i podkreślniki.');
```

a musi mieć od 4 do 20 znaków i zawierać tylko litery, cyfry i podkreślniki.');

```

    isValid = false;
} else {
    clearError(username);
}

// Walidacja email
if (!validateEmail(email.value)) {
    showError(email, 'Podaj poprawny adres email.');
```

s email.');

```

    isValid = false;
} else {
    clearError(email);
}

// Walidacja hasła
if (!validatePassword(password.value)) {
    showError(password, 'Hasło musi mieć co najmniej 8 znaków, zawierać małe i wielkie litery, cyfry oraz znaki specjalne.');
```

co najmniej 8 znaków, zawierać małe i wielkie litery, cyfry oraz znaki specjalne.');

```

    isValid = false;
} else {
    clearError(password);
}

// Walidacja zgodności haseł
if (password.value !== confirmPassword.value) {
    showError(confirmPassword, 'Hasła nie są identyczne.');
```

są identyczne.');

```

    isValid = false;
} else {

```

```

        clearError(confirmPassword);
    }

    // Walidacja akceptacji regulaminu
    if (!terms.checked) {
        showError(terms, 'Musisz zaakceptować
regulamin i politykę prywatności.');
```

ym

```

        isValid = false;
    } else {
        clearError(terms);
    }

    if (!isValid) {
        event.preventDefault();
    }
});

// Sprawdzanie siły hasła w czasie rzeczywist
ym
    if (password) {
        password.addEventListener('input', functi
on() {
            const strength = calculatePasswordStr
ength(this.value);
            updatePasswordStrengthIndicator(stren
gth);
        }
    });
});

// Funkcje pomocnicze
function validateEmail(email) {
    const re = /^((^[^<>()\\[\]\\\\.,;:\s@""]+(\.^[^<>()\\[\]
]\\\\.,;:\s@""]+)*)|("[^"]*"|'['']*))@[^\s(){}|;,\.:\\[\]\\\\\
[0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}])|(([a-zA-Z\\-0-9]+\\.)+[a-zA-Z]
{2,}))$/;
    return re.test(String(email).toLowerCase());
}

```



```

function validatePassword(password) {
    // Minimum 8 znaków, mała i wielka litera, cyfra,
    // znak specjalny
    const re = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[
@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/;
    return re.test(password);
}

function showError(input, message) {
    const formGroup = input.closest('.form-group');
    formGroup.classList.add('has-error');

    // Usunięcie poprzedniego komunikatu błędu, jeśli
    // istnieje
    const existingError = formGroup.querySelector('.e
rror-message');
    if (existingError) {
        existingError.remove();
    }

    // Dodanie nowego komunikatu błędu
    const errorElement = document.createElement('div'
);
    errorElement.className = 'error-message';
    errorElement.textContent = message;
    formGroup.appendChild(errorElement);
}

function clearError(input) {
    const formGroup = input.closest('.form-group');
    formGroup.classList.remove('has-error');

    // Usunięcie komunikatu błędu, jeśli istnieje
    const existingError = formGroup.querySelector('.e
rror-message');
    if (existingError) {
        existingError.remove();
    }
}

```

```

}

function calculatePasswordStrength(password) {
    let strength = 0;

    // Długość hasła
    if (password.length >= 8) strength += 1;
    if (password.length >= 12) strength += 1;

    // Zawartość hasła
    if (password.match(/[a-z]+/)) strength += 1;
    if (password.match(/[A-Z]+/)) strength += 1;
    if (password.match(/[0-9]+/)) strength += 1;
    if (password.match(/^[A-Za-z0-9]+/)) strength +=
1;

    return Math.min(strength, 5);
}

function updatePasswordStrengthIndicator(strength) {
    const indicator = document.getElementById('password-strength');

    if (!indicator) return;

    // Usunięcie poprzednich klas
    indicator.className = 'password-strength';

    // Dodanie odpowiedniej klasy w zależności od siły hasła
    let strengthClass = '';
    let strengthText = '';

    switch(strength) {
        case 0:
        case 1:
            strengthClass = 'very-weak';
            strengthText = 'Bardzo słabe';
            break;

```

```

    case 2:
        strengthClass = 'weak';
        strengthText = 'Słabe';
        break;
    case 3:
        strengthClass = 'medium';
        strengthText = 'Średnie';
        break;
    case 4:
        strengthClass = 'strong';
        strengthText = 'Silne';
        break;
    case 5:
        strengthClass = 'very-strong';
        strengthText = 'Bardzo silne';
        break;
}

indicator.classList.add(strengthClass);
indicator.textContent = strengthText;
}

```

Zabezpieczenia systemu logowania:

1. Hashowanie haseł

- Używanie nowoczesnych algorytmów hashowania (bcrypt, Argon2)
- Unikanie przestarzałych metod (MD5, SHA-1)

2. Ochrona przed atakami

- Zabezpieczenie przed SQL Injection (parametryzowane zapytania)
- Ochrona przed CSRF (Cross-Site Request Forgery)
- Ochrona przed XSS (Cross-Site Scripting)
- Walidacja danych wejściowych po stronie serwera

3. Limity prób logowania

- Blokowanie konta po wielu nieudanych próbach

- Opóźnianie odpowiedzi po nieudanym logowaniu
- Captcha po kilku nieudanych próbach

4. Bezpieczne zarządzanie sesją

- Generowanie bezpiecznych identyfikatorów sesji
- Wygasanie sesji po czasie bezczynności
- Unieważnianie sesji po zmianie hasła
- Flaga HttpOnly dla ciasteczek sesji

5. Dwustopniowa weryfikacja (2FA)

- Kody SMS/email
- Aplikacje uwierzytelniające (Google Authenticator, Authy)
- Klucze bezpieczeństwa (YubiKey)

Przykład implementacji 2FA z kodem jednorazowym:

<?php

// Generowanie kodu jednorazowego

```
function generate_2fa_code($user_id) {  
    global $db;
```

// Generowanie 6-cyfrowego kodu

```
$code = sprintf('%06d', mt_rand(0, 999999));
```

// Ustawienie czasu wygaśnięcia (10 minut)

```
$expires = date('Y-m-d H:i:s', time() + 600);
```

// Zapisanie kodu w bazie danych

```
$query = "INSERT INTO auth_codes (user_id, code,  
expires) VALUES (:user_id, :code, :expires)";
```

```
$stmt = $db->prepare($query);
```

```
$stmt->bindParam(':user_id', $user_id);
```

```
$stmt->bindParam(':code', $code);
```

```
$stmt->bindParam(':expires', $expires);
```

```
$stmt->execute();
```

```
return $code;
```

```
}
```

// Wysyłanie kodu SMS

```
function send_2fa_sms($phone, $code) {  
    // Implementacja wysyłania SMS - wymaga integracji z usługą SMS  
    // Przykład z Twilio:  
    require_once 'vendor/autoload.php';  
  
    $sid = 'YOUR_TWILIO_SID';  
    $token = 'YOUR_TWILIO_TOKEN';  
    $twilioNumber = 'YOUR_TWILIO_PHONE_NUMBER';  
  
    $client = new Twilio\Rest\Client($sid, $token);  
  
    try {  
        $client->messages->create(  
            $phone,  
            [  
                'from' => $twilioNumber,  
                'body' => "Twój kod weryfikacyjny: $code"  
            ]  
        );  
        return true;  
    } catch (Exception $e) {  
        error_log("Błąd wysyłania SMS: " . $e->getMessage());  
        return false;  
    }  
}
```

// Weryfikacja kodu 2FA

```
function verify_2fa_code($user_id, $code) {  
    global $db;  
  
    $query = "SELECT id FROM auth_codes  
              WHERE user_id = :user_id AND code = :code  
              AND expires > NOW() AND used = 0  
              LIMIT 1";
```

```

$stmt = $db->prepare($query);
$stmt->bindParam(':user_id', $user_id);
$stmt->bindParam(':code', $code);
$stmt->execute();

$result = $stmt->fetch(PDO::FETCH_ASSOC);

if ($result) {
    // Oznaczenie kodu jako wykorzystanego
    $query = "UPDATE auth_codes SET used = 1 WHERE
E id = :id";
    $stmt = $db->prepare($query);
    $stmt->bindParam(':id', $result['id']);
    $stmt->execute();

    return true;
}

return false;
}

// Obsługa logowania z 2FA
function login_with_2fa() {
    global $db;

    session_start();

    // Sprawdzenie, czy użytkownik jest w trakcie procesu 2FA
    if (!isset($_SESSION['pending_user_id'])) {
        header('Location: login.php');
        exit;
    }

    $user_id = $_SESSION['pending_user_id'];

    // Pobranie danych użytkownika
    $query = "SELECT username, email, phone FROM user

```

```

s WHERE id = :user_id LIMIT 1";
$stmt = $db->prepare($query);
$stmt->bindParam(':user_id', $user_id);
$stmt->execute();

$user = $stmt->fetch(PDO::FETCH_ASSOC);

if (!$user) {
    // Nieprawidłowy użytkownik
    session_destroy();
    header('Location: login.php?error=invalid_session');
    exit;
}

// Jeśli formularz został wysłany
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['verification_code'])) {
    $code = preg_replace('/\D/', '', $_POST['verification_code']); // Usunięcie wszystkich znaków niebędących cyframi

    if (strlen($code) !== 6) {
        $error = "Kod weryfikacyjny musi składać się z 6 cyfr.";
    } else {
        // Weryfikacja kodu
        if (verify_2fa_code($user_id, $code)) {
            // Ukończenie logowania
            $_SESSION['user_id'] = $user_id;
            $_SESSION['username'] = $user['username'];

            unset($_SESSION['pending_user_id']);

            // Przekierowanie do panelu użytkownika
            header('Location: dashboard.php');
            exit;
        } else {

```

```

        $error = "Nieprawidłowy kod weryfikacyjny lub kod wygasł.";
    }
}
} else {
    // Generowanie nowego kodu i wysyłanie
    $code = generate_2fa_code($user_id);

    if ($user['phone']) {
        send_2fa_sms($user['phone'], $code);
    } else {
        // Wysyłanie kodu przez email jeśli nie ma numeru telefonu
        $subject = "Kod weryfikacyjny";
        $message = "Twój kod weryfikacyjny: $code\n\nKod jest ważny przez 10 minut.";
        mail($user['email'], $subject, $message);
    }
}

// Wyświetlenie formularza 2FA
// ...
}

```

Dobre praktyki dla systemów logowania: - Używaj HTTPS dla wszystkich stron związanych z uwierzytelnianiem - Stosuj silne hashowanie haseł (bcrypt, Argon2) - Implementuj limity prób logowania - Wymagaj silnych haseł od użytkowników - Implementuj bezpieczny proces odzyskiwania hasła - Używaj tokenów CSRF w formularzach - Zabezpiecz ciasteczka sesji (HttpOnly, Secure, SameSite) - Monitoruj i loguj podejrzane aktywności - Regularnie przeglądaj i aktualizuj zabezpieczenia - Rozważ implementację 2FA dla krytycznych funkcji

Dlaczego są ważne dla początkujących: - System logowania jest podstawowym elementem wielu stron internetowych - Prawidłowa implementacja systemu logowania jest kluczowa dla bezpieczeństwa - Wycieki danych użytkowników mogą mieć

poważne konsekwencje prawne i wizerunkowe - Dobrze zaimplementowany system logowania buduje zaufanie użytkowników

KOSZYK ZAKUPOWY

Koszyk zakupowy to kluczowy element każdego sklepu internetowego, umożliwiający klientom wybieranie, przechowywanie i zarządzanie produktami przed ich zakupem.

Kluczowe cechy: - Dodawanie i usuwanie produktów - Zmiana ilości produktów - Obliczanie sumy zakupów - Przechowywanie stanu koszyka między sesjami - Przejście do procesu zamówienia

Komponenty koszyka zakupowego:

1. Interfejs koszyka (HTML/CSS):

```
<div class="shopping-cart">
  <h2>Twój koszyk</h2>

  <!-- Mini koszyk (w nagłówku) -->
  <div class="mini-cart">
    <div class="cart-summary">
      <span class="cart-count">3</span> produkty
      <span class="cart-total">249,97 zł</span>
    </div>
    <button class="view-cart-btn">Zobacz koszyk</button>
  </div>

  <!-- Pełny koszyk (na stronie koszyka) -->
  <div class="cart-content">
    <table class="cart-table">
      <thead>
        <tr>
          <th class="product-thumbnail">Zdjęcie</th>
          <th class="product-name">Produkt</th>
          <th class="product-price">Cena</th>
          <th class="product-quantity">Ilość</th>
        </tr>
      </thead>
    </table>
  </div>
</div>
```

```

        <th class="product-subtotal">Suma</th>
        <th class="product-remove">Usuń</th>
    </tr>
</thead>

<tbody>
    <tr class="cart-item" data-product-id="123">
        <td class="product-thumbnail">
            
        </td>
        <td class="product-name">
            <a href="product.php?id=123">Nazwa produk
tu 1</a>
            <div class="product-variant">Kolor: Czarn
y, Rozmiar: M</div>
        </td>
        <td class="product-price">
            <span class="price">99,99 zł</span>
        </td>
        <td class="product-quantity">
            <div class="quantity-input">
                <button class="decrease-qty" data-produ
ct-id="123">-</button>
                <input type="number" min="1" max="10" v
alue="1" data-product-id="123">
                <button class="increase-qty" data-produ
ct-id="123">+</button>
            </div>
        </td>
        <td class="product-subtotal">
            <span class="price">99,99 zł</span>
        </td>
        <td class="product-remove">
            <button class="remove-item" data-product-
id="123">x</button>
        </td>
    </tr>

```

```

        <!-- Więcej produktów... -->
    </tbody>
</table>

<div class="cart-actions">
    <div class="coupon">
        <input type="text" name="coupon_code" placeholder="Kod kuponu">
        <button class="apply-coupon">Zastosuj kupon</button>
    </div>

    <button class="update-cart">Aktualizuj koszyk</button>
</div>

<div class="cart-collaterals">
    <div class="cart-totals">
        <h3>Podsumowanie koszyka</h3>

        <table>
            <tr class="cart-subtotal">
                <th>Suma częściowa</th>
                <td>249,97 zł</td>
            </tr>

            <tr class="shipping">
                <th>Wysyłka</th>
                <td>
                    <ul class="shipping-methods">
                        <li>
                            <input type="radio" name="shipping_method" id="shipping_method_1" value="flat_rate" checked>
                            <label for="shipping_method_1">Stawka ryczałtowa: 15,00 zł</label>
                        </li>
                        <li>
                            <input type="radio" name="shipping_

```

```

method" id="shipping_method_2" value="free">
    <label for="shipping_method_2">Darm
owa wysyłka</label>
    </li>
    </ul>
    </td>
    </tr>

    <tr class="order-total">
        <th>Suma</th>
        <td><strong>264,97 zł</strong></td>
    </tr>
</table>

<div class="checkout-button-wrapper">
    <a href="checkout.php" class="checkout-butt
on">Przejdź do kasy</a>
    </div>
</div>
</div>
</div>
</div>
</div>

```

2. CSS dla koszyka:

```

/* Mini koszyk */
.mini-cart {
    position: relative;
    padding: 10px;
    background-color: #f8f8f8;
    border-radius: 4px;
    margin-bottom: 20px;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.cart-summary {
    font-size: 14px;
}

```

```
.cart-count {
  font-weight: bold;
}

.cart-total {
  margin-left: 10px;
  font-weight: bold;
  color: #3c3c3c;
}

.view-cart-btn {
  padding: 8px 16px;
  background-color: #4CAF50;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

/* Tabela koszyka */
.cart-table {
  width: 100%;
  border-collapse: collapse;
  margin-bottom: 30px;
}

.cart-table th {
  text-align: left;
  padding: 12px;
  border-bottom: 1px solid #ddd;
  font-weight: 600;
}

.cart-table td {
  padding: 15px 12px;
  border-bottom: 1px solid #f1f1f1;
  vertical-align: middle;
}
```

```
.product-thumbnail img {
  width: 80px;
  height: 80px;
  object-fit: cover;
}

.product-name a {
  font-weight: 500;
  color: #333;
  text-decoration: none;
}

.product-name a:hover {
  color: #4CAF50;
}

.product-variant {
  font-size: 13px;
  color: #777;
  margin-top: 5px;
}

.product-price,
.product-subtotal {
  font-weight: 500;
}

/* Ilość produktu */
.quantity-input {
  display: flex;
  align-items: center;
  max-width: 120px;
}

.quantity-input button {
  width: 30px;
  height: 30px;
  background-color: #f5f5f5;
```

```
border: 1px solid #ddd;
font-size: 16px;
cursor: pointer;
display: flex;
align-items: center;
justify-content: center;
}

.quantity-input input {
width: 40px;
height: 30px;
text-align: center;
border: 1px solid #ddd;
border-left: none;
border-right: none;
padding: 0;
}

.product-remove button {
font-size: 24px;
color: #999;
background: none;
border: none;
cursor: pointer;
}

.product-remove button:hover {
color: #e74c3c;
}

/* Akcje koszyka */
.cart-actions {
display: flex;
justify-content: space-between;
margin-bottom: 30px;
}

.coupon {
display: flex;
```

```

}

.coupon input {
  width: 200px;
  padding: 8px 12px;
  border: 1px solid #ddd;
  border-radius: 4px 0 0 4px;
}

.apply-coupon {
  padding: 8px 16px;
  background-color: #333;
  color: white;
  border: none;
  border-radius: 0 4px 4px 0;
  cursor: pointer;
}

.update-cart {
  padding: 8px 16px;
  background-color: #f8f8f8;
  border: 1px solid #ddd;
  border-radius: 4px;
  cursor: pointer;
}

/* Podsumowanie koszyka */
.cart-collaterals {
  background-color: #f9f9f9;
  padding: 20px;
  border-radius: 4px;
}

.cart-totals h3 {
  margin-top: 0;
  padding-bottom: 10px;
  border-bottom: 1px solid #eee;
}

```



```
.cart-totals table {
  width: 100%;
  margin-bottom: 20px;
}

.cart-totals th,
.cart-totals td {
  padding: 12px 0;
  border-bottom: 1px solid #eee;
}

.cart-totals th {
  text-align: left;
  font-weight: 500;
}

.cart-totals td {
  text-align: right;
}

.shipping-methods {
  list-style: none;
  padding: 0;
  margin: 0;
}

.shipping-methods li {
  margin-bottom: 8px;
}

.order-total td {
  font-size: 18px;
  font-weight: bold;
  color: #333;
}

.checkout-button {
  display: block;
  width: 100%;
}
```

```
padding: 12px;
background-color: #4CAF50;
color: white;
text-align: center;
text-decoration: none;
font-weight: bold;
border-radius: 4px;
margin-top: 20px;
}

.checkout-button:hover {
  background-color: #45a049;
}

/* Responsywność koszyka */
@media (max-width: 768px) {
  .cart-table {
    display: block;
    overflow-x: auto;
  }

  .cart-actions {
    flex-direction: column;
    gap: 10px;
  }

  .coupon {
    width: 100%;
  }

  .coupon input {
    flex: 1;
  }

  .update-cart {
    width: 100%;
  }
}
```

```

@media (max-width: 576px) {
  .product-thumbnail {
    display: none;
  }

  .cart-table th,
  .cart-table td {
    padding: 8px;
  }

  .quantity-input {
    max-width: 90px;
  }

  .quantity-input input {
    width: 30px;
  }
}

```

3. JavaScript dla funkcjonalności koszyka:

```

document.addEventListener('DOMContentLoaded', function() {
  // Elementy DOM
  const cartItems = document.querySelectorAll('.cart-item');
  const quantityInputs = document.querySelectorAll('quantity-input input');
  const increaseButtons = document.querySelectorAll('increase-qty');
  const decreaseButtons = document.querySelectorAll('decrease-qty');
  const removeButtons = document.querySelectorAll('remove-item');
  const updateCartButton = document.querySelector('update-cart');
  const applyDiscountButton = document.querySelector('apply-coupon');
  const shippingMethods = document.querySelectorAll('input[name="shipping_method"]');

```

```

// Nastuchiwanie zmian ilości produktów
quantityInputs.forEach(input => {
  input.addEventListener('change', function() {
    const productId = this.getAttribute('data-product-id');
    const newQuantity = parseInt(this.value);

    if (isNaN(newQuantity) || newQuantity < 1) {
      this.value = 1;
      updateQuantity(productId, 1);
    } else {
      updateQuantity(productId, newQuantity);
    }
  });
});

// Przycisk zwiększania ilości
increaseButtons.forEach(button => {
  button.addEventListener('click', function() {
    const productId = this.getAttribute('data-product-id');
    const input = document.querySelector(`.quantity-input[data-product-id="${productId}"]`);
    let currentValue = parseInt(input.value);

    if (!isNaN(currentValue)) {
      const newValue = currentValue + 1;
      input.value = newValue;
      updateQuantity(productId, newValue);
    }
  });
});

// Przycisk zmniejszania ilości
decreaseButtons.forEach(button => {
  button.addEventListener('click', function() {
    const productId = this.getAttribute('data-product-id');

```

```

    const input = document.querySelector(`.quantity
-input input[data-product-id="${productId}"]`);
    let currentValue = parseInt(input.value);

    if (!isNaN(currentValue) && currentValue > 1) {
        const newValue = currentValue - 1;
        input.value = newValue;
        updateQuantity(productId, newValue);
    }
  });
});

// Przycisk usuwania produktu
removeButtons.forEach(button => {
  button.addEventListener('click', function() {
    const productId = this.getAttribute('data-produ
ct-id');
    removeProduct(productId);
  });
});

// Przycisk aktualizacji koszyka
if (updateCartButton) {
  updateCartButton.addEventListener('click', functi
on() {
    updateCart();
  });
}

// Przycisk zastosowania kuponu
if (applyDiscountButton) {
  applyDiscountButton.addEventListener('click', fun
ction() {
    applyCoupon();
  });
}

// Zmiana metody wysyłki
shippingMethods.forEach(method => {

```

```

method.addEventListener('change', function() {
  updateShipping(this.value);
});
});

// Funkcja aktualizacji ilości produktu
function updateQuantity(productId, quantity) {
  // Aktualizacja sumy dla produktu
  const item = document.querySelector(`.cart-item[data-product-id="${productId}"]`);
  const priceElement = item.querySelector('.product-price .price');
  const subtotalElement = item.querySelector('.product-subtotal .price');

  const price = parseFloat(priceElement.textContent.replace(/[^\d,\.]/g, '').replace(',', '.', ''));
  const subtotal = price * quantity;

  subtotalElement.textContent = formatPrice(subtotal);
}

// Aktualizacja koszyka na serwerze (Ajax)
fetch('update-cart.php', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/x-www-form-urlencoded',
  },
  body: `action=update&product_id=${productId}&quantity=${quantity}`
})
.then(response => response.json())
.then(data => {
  // Aktualizacja sumy koszyka
  updateCartTotals(data);
})
.catch(error => {
  console.error('Error updating cart:', error);
});

```

```

    });
}

// Funkcja usuwania produktu
function removeProduct(productId) {
    const item = document.querySelector(`.cart-item[data-product-id="${productId}"]`);

    // Animacja usuwania
    item.style.opacity = '0.5';

    // Usunięcie produktu z koszyka (Ajax)
    fetch('update-cart.php', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded',
        },
        body: `action=remove&product_id=${productId}`
    })
    .then(response => response.json())
    .then(data => {
        // Usunięcie elementu z DOM
        item.remove();

        // Aktualizacja sumy koszyka
        updateCartTotals(data);

        // Sprawdzenie, czy koszyk jest pusty
        if (data.item_count === 0) {
            showEmptyCartMessage();
        }
    })
    .catch(error => {
        console.error('Error removing product:', error);

        // Przywrócenie widoczności w przypadku błędu
        item.style.opacity = '1';
    });
;

```

```

    }

    // Funkcja aktualizacji koszyka
    function updateCart() {
        const updates = [];

        // Zebranie danych o wszystkich produktach
        quantityInputs.forEach(input => {
            const productId = input.getAttribute('data-product-id');
            const quantity = parseInt(input.value);

            if (!isNaN(quantity) && quantity > 0) {
                updates.push({ id: productId, quantity: quantity });
            }
        });

        // Aktualizacja całego koszyka (Ajax)
        fetch('update-cart.php', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json',
            },
            body: JSON.stringify({
                action: 'update_all',
                updates: updates
            })
        })
        .then(response => response.json())
        .then(data => {
            // Aktualizacja sumy koszyka
            updateCartTotals(data);

            // Wyświetlenie komunikatu o powodzeniu
            showMessage('Koszyk został zaktualizowany', 'success');
        })
        .catch(error => {

```



```

        console.error('Error updating cart:', error);
        showMessage('Wystąpił błąd podczas aktualizacji
koszyka', 'error');
    });
}

// Funkcja zastosowania kuponu
function applyCoupon() {
    const couponInput = document.querySelector('input
[name="coupon_code"]');
    const couponCode = couponInput.value.trim();

    if (!couponCode) {
        showMessage('Wprowadź kod kuponu', 'error');
        return;
    }

    // Wysłanie kodu kuponu do weryfikacji (Ajax)
    fetch('apply-coupon.php', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded',
        },
        body: `coupon_code=${encodeURIComponent(couponCode)}`
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            // Aktualizacja podsumowania koszyka
            updateCartTotals(data);
            showMessage(data.message, 'success');

            // Dodanie informacji o zastosowanym kuponie
            addCouponInfo(data.coupon);
        } else {
            showMessage(data.message, 'error');
        }
    })

```

```

    })
    .catch(error => {
        console.error('Error applying coupon:', error);
        showMessage('Wystąpił błąd podczas zastosowania
kuponu', 'error');
    });
}

// Funkcja aktualizacji metody wysyłki
function updateShipping(shippingMethod) {
    // Aktualizacja metody wysyłki (Ajax)
    fetch('update-shipping.php', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded',
        },
        body: `shipping_method=${encodeURIComponent(shi
ppingMethod)}`
    })
    .then(response => response.json())
    .then(data => {
        // Aktualizacja podsumowania koszyka
        updateCartTotals(data);
    })
    .catch(error => {
        console.error('Error updating shipping:', error
);
    });
}

// Funkcja aktualizacji sum koszyka
function updateCartTotals(data) {
    // Aktualizacja mini koszyka
    const cartCount = document.querySelector('.cart-c
ount');
    const cartTotal = document.querySelector('.cart-t
otal');

```

```

    if (cartCount) cartCount.textContent = data.item_
count;
    if (cartTotal) cartTotal.textContent = formatPric
e(data.total);

    // Aktualizacja podsumowania
    const subtotalElement = document.querySelector('.
cart-subtotal td');
    const shippingElement = document.querySelector('.
shipping td .shipping-methods');
    const totalElement = document.querySelector('.ord
er-total td strong');

    if (subtotalElement) subtotalElement.textContent
= formatPrice(data.subtotal);
    if (totalElement) totalElement.textContent = form
atPrice(data.total);

    // Aktualizacja opcji wysyłki, jeśli dostępne
    if (shippingElement && data.shipping_methods) {
        updateShippingOptions(data.shipping_methods, da
ta.selected_shipping);
    }
}

// Funkcja dodawania informacji o kuponie
function addCouponInfo(coupon) {
    // Sprawdzenie, czy informacja o kuponie już istn
ieje
    let couponRow = document.querySelector('.cart-dis
count');

    if (!couponRow) {
        // Tworzenie nowej wiersza dla kuponu
        const totalsTable = document.querySelector('.ca
rt-totals table');
        const shippingRow = document.querySelector('.sh
ipping');

```

```

couponRow = document.createElement('tr');
couponRow.className = 'cart-discount';

const couponTh = document.createElement('th');
couponTh.textContent = `Kupon: ${coupon.code}`;

const couponTd = document.createElement('td');
couponTd.textContent = `-${formatPrice(coupon.d
iscount)}`;

couponRow.appendChild(couponTh);
couponRow.appendChild(couponTd);

// Wstawienie przed wierszem wysyłki
if (shippingRow) {
    totalsTable.insertBefore(couponRow, shippingR
ow);
} else {
    totalsTable.appendChild(couponRow);
}
} else {
    // Aktualizacja istniejącego wiersza
    const couponTh = couponRow.querySelector('th');
    const couponTd = couponRow.querySelector('td');

    couponTh.textContent = `Kupon: ${coupon.code}`;
    couponTd.textContent = `-${formatPrice(coupon.d
iscount)}`;
}
}

// Funkcja aktualizacji opcji wysyłki
function updateShippingOptions(shippingMethods, sel
ectedMethod) {
    const shippingElement = document.querySelector('.
shipping td .shipping-methods');

    if (shippingElement) {
        shippingElement.innerHTML = '';
    }
}

```

```

shippingMethods.forEach((method, index) => {
    const li = document.createElement('li');

    const input = document.createElement('input')

    input.type = 'radio';
    input.name = 'shipping_method';
    input.id = `shipping_method_${index + 1}`;
    input.value = method.id;
    input.checked = method.id === selectedMethod;

    const label = document.createElement('label')

    label.htmlFor = `shipping_method_${index + 1}`;

    label.textContent = `${method.name}: ${method
.cost > 0 ? formatPrice(method.cost) : 'Darmowa wysył
ka'}`;

    li.appendChild(input);
    li.appendChild(label);
    shippingElement.appendChild(li);

    // Dodanie nasłuchiwanie zmiany
    input.addEventListener('change', function() {
        updateShipping(this.value);
    });
});
}

// Funkcja wyświetlania komunikatu o pustym koszyku
function showEmptyCartMessage() {
    const cartContent = document.querySelector('.cart
-content');

    if (cartContent) {
        cartContent.innerHTML = `

```

```

        <div class="empty-cart">
            <p>Twój koszyk jest pusty.</p>
            <a href="shop.php" class="continue-shopping
">Kontynuuj zakupy</a>
        </div>
    `;
}
}

// Funkcja wyświetlania komunikatów
function showMessage(message, type = 'info') {
    // Sprawdzenie, czy kontener komunikatów istnieje
    let messageContainer = document.querySelector('.c
art-messages');

    if (!messageContainer) {
        // Tworzenie kontenera komunikatów
        messageContainer = document.createElement('div'
);
        messageContainer.className = 'cart-messages';

        // Wstawienie przed koszykiem
        const cart = document.querySelector('.shopping-
cart');
        cart.parentNode.insertBefore(messageContainer,
cart);
    }

    // Tworzenie elementu komunikatu
    const messageElement = document.createElement('di
v');
    messageElement.className = `message message-${typ
e}`;
    messageElement.textContent = message;

    // Dodanie przycisku zamknięcia
    const closeButton = document.createElement('butto
n');
    closeButton.className = 'close-message';

```

```

closeButton.innerHTML = '&times;';
closeButton.addEventListener('click', function()
{
    messageElement.remove();
});

messageElement.appendChild(closeButton);
messageContainer.appendChild(messageElement);

// Automatyczne ukrycie po 5 sekundach
setTimeout(() => {
    messageElement.classList.add('fade-out');
    setTimeout(() => {
        messageElement.remove();
    }, 500);
}, 5000);
}

// Pomocnicza funkcja formatowania ceny
function formatPrice(price) {
    return new Intl.NumberFormat('pl-PL', {
        style: 'currency',
        currency: 'PLN'
    }).format(price);
}
});

```

4. Backend obsługi koszyka (PHP):

```

<?php
// Plik: cart-functions.php

// Rozpoczęcie sesji
session_start();

// Inicjalizacja koszyka, jeśli nie istnieje
if (!isset($_SESSION['cart'])) {
    $_SESSION['cart'] = [
        'items' => [],
        'item_count' => 0,
    ];
}

```

```

        'subtotal' => 0,
        'shipping' => 0,
        'coupon' => null,
        'discount' => 0,
        'total' => 0,
        'shipping_method' => 'flat_rate'
    ];
}

// Funkcja dodająca produkt do koszyka
function addToCart($product_id, $quantity = 1, $attributes = []) {
    global $db;

    // Sprawdzenie, czy produkt istnieje
    $product_query = "SELECT id, name, price, stock_quantity FROM products WHERE id = :id LIMIT 1";
    $stmt = $db->prepare($product_query);
    $stmt->bindParam(':id', $product_id);
    $stmt->execute();
    $product = $stmt->fetch(PDO::FETCH_ASSOC);

    if (!$product) {
        return [
            'success' => false,
            'message' => 'Produkt nie istnieje.'
        ];
    }

    // Sprawdzenie dostępności
    if ($product['stock_quantity'] < $quantity) {
        return [
            'success' => false,
            'message' => 'Nie ma wystarczającej ilości produktu w magazynie.'
        ];
    }

    // Tworzenie unikalnego klucza dla produktu (uwzg

```


Łączącego atrybuty)

```
$item_key = $product_id;

if (!empty($attributes)) {
    ksort($attributes); // Sortowanie atrybutów,
aby zapewnić spójny klucz
    $item_key .= '-' . md5(json_encode($attribute
s));
}

// Sprawdzenie, czy produkt już jest w koszyku
if (isset($_SESSION['cart'][$item_key]))
{
    // Aktualizacja ilości
    $_SESSION['cart'][$item_key]['quantity'] += $quantity;
} else {
    // Dodanie nowego produktu
    $_SESSION['cart'][$item_key] = [
        'id' => $product_id,
        'name' => $product['name'],
        'price' => $product['price'],
        'quantity' => $quantity,
        'attributes' => $attributes,
        'subtotal' => $product['price'] * $quantity
    ];
}

// Aktualizacja podsumowania koszyka
updateCartTotals();

return [
    'success' => true,
    'message' => 'Produkt został dodany do koszyk
a.',
    'cart' => $_SESSION['cart']
];
}
```

```

// Funkcja aktualizująca ilość produktu
function updateCartItemQuantity($item_key, $quantity)
{
    if (!isset($_SESSION['cart']['items'][$item_key]))
    ) {
        return [
            'success' => false,
            'message' => 'Produkt nie istnieje w koszyku.'
        ];
    }

    // Walidacja ilości
    if ($quantity <= 0) {
        // Usunięcie produktu, jeśli ilość jest zerowa lub ujemna
        return removeCartItem($item_key);
    }

    // Sprawdzenie dostępności w magazynie
    global $db;
    $product_id = $_SESSION['cart']['items'][$item_key]['id'];

    $product_query = "SELECT stock_quantity FROM products WHERE id = :id LIMIT 1";
    $stmt = $db->prepare($product_query);
    $stmt->bindParam(':id', $product_id);
    $stmt->execute();
    $product = $stmt->fetch(PDO::FETCH_ASSOC);

    if ($product && $product['stock_quantity'] < $quantity) {
        return [
            'success' => false,
            'message' => 'Nie ma wystarczającej ilości produktu w magazynie.',
            'available_quantity' => $product['stock_q

```

```

    'quantity']
    ];
}

// Aktualizacja ilości
$_SESSION['cart']['items'][$item_key]['quantity']
= $quantity;
$_SESSION['cart']['items'][$item_key]['subtotal']
= $_SESSION['cart']['items'][$item_key]['price'] * $
quantity;

// Aktualizacja podsumowania koszyka
updateCartTotals();

return [
    'success' => true,
    'message' => 'Ilość produktu została zaktuali
zowana.',
    'cart' => $_SESSION['cart']
];
}

// Funkcja usuwająca produkt z koszyka
function removeCartItem($item_key) {
    if (!isset($_SESSION['cart']['items'][$item_key])
) {
        return [
            'success' => false,
            'message' => 'Produkt nie istnieje w kosz
yku.'
        ];
    }

    // Usunięcie produktu
    unset($_SESSION['cart']['items'][$item_key]);

    // Aktualizacja podsumowania koszyka
    updateCartTotals();
}

```

```

return [
    'success' => true,
    'message' => 'Produkt został usunięty z koszy
ka.',
    'cart' => $_SESSION['cart']
];
}

// Funkcja czyszcząca koszyk
function clearCart() {
    $_SESSION['cart'] = [
        'items' => [],
        'item_count' => 0,
        'subtotal' => 0,
        'shipping' => 0,
        'coupon' => null,
        'discount' => 0,
        'total' => 0,
        'shipping_method' => 'flat_rate'
    ];

    return [
        'success' => true,
        'message' => 'Koszyk został wyczyszczony.',
        'cart' => $_SESSION['cart']
    ];
}

// Funkcja zastosowania kuponu
function applyCoupon($coupon_code) {
    global $db;

    // Usunięcie białych znaków
    $coupon_code = trim($coupon_code);

    if (empty($coupon_code)) {
        return [
            'success' => false,
            'message' => 'Wprowadź kod kuponu.'
        ];
    }
}

```

```

    ];
}

// Sprawdzenie, czy kupon istnieje
$query = "SELECT * FROM coupons WHERE code = :code AND active = 1
        AND (usage_limit = 0 OR usage_count < usage_limit)
        AND (expiry_date IS NULL OR expiry_date >= CURDATE())
        LIMIT 1";

$stmt = $db->prepare($query);
$stmt->bindParam(':code', $coupon_code);
$stmt->execute();
$coupon = $stmt->fetch(PDO::FETCH_ASSOC);

if (!$coupon) {
    return [
        'success' => false,
        'message' => 'Kupon jest nieważny lub wygasł.'
    ];
}

// Sprawdzenie minimalnej kwoty zamówienia
if ($coupon['minimum_amount'] > 0 && $_SESSION['cart']['subtotal'] < $coupon['minimum_amount']) {
    return [
        'success' => false,
        'message' => 'Aby użyć tego kuponu, minimalna kwota zamówienia musi wynosić ' . number_format($coupon['minimum_amount'], 2) . ' zł.'
    ];
}

// Obliczenie zniżki
$discount = 0;

```

```

    if ($coupon['type'] === 'percentage') {
        // Zniżka procentowa
        $discount = $_SESSION['cart']['subtotal'] * (
$coupon['amount'] / 100);
    } else {
        // Zniżka kwotowa
        $discount = min($coupon['amount'], $_SESSION[
'cart']['subtotal']);
    }

    // Zapisanie kuponu i zniżki
    $_SESSION['cart']['coupon'] = [
        'id' => $coupon['id'],
        'code' => $coupon['code'],
        'type' => $coupon['type'],
        'amount' => $coupon['amount'],
        'discount' => $discount
    ];

    $_SESSION['cart']['discount'] = $discount;

    // Aktualizacja podsumowania koszyka
    updateCartTotals();

    // Zwiększenie licznika użyć kuponu
    $update_query = "UPDATE coupons SET usage_count =
usage_count + 1 WHERE id = :id";
    $stmt = $db->prepare($update_query);
    $stmt->bindParam(':id', $coupon['id']);
    $stmt->execute();

    return [
        'success' => true,
        'message' => 'Kupon został zastosowany.',
        'coupon' => $_SESSION['cart']['coupon'],
        'cart' => $_SESSION['cart']
    ];
}

```

```

// Funkcja usuwająca kupon
function removeCoupon() {
    $_SESSION['cart']['coupon'] = null;
    $_SESSION['cart']['discount'] = 0;

    // Aktualizacja podsumowania koszyka
    updateCartTotals();

    return [
        'success' => true,
        'message' => 'Kupon został usunięty.',
        'cart' => $_SESSION['cart']
    ];
}

// Funkcja aktualizacji metody wysyłki
function updateShippingMethod($method) {
    global $db;

    // Pobranie dostępnych metod wysyłki
    $query = "SELECT * FROM shipping_methods WHERE id
= :id AND active = 1 LIMIT 1";
    $stmt = $db->prepare($query);
    $stmt->bindParam(':id', $method);
    $stmt->execute();
    $shipping_method = $stmt->fetch(PDO::FETCH_ASSOC)
;

    if (!$shipping_method) {
        return [
            'success' => false,
            'message' => 'Metoda wysyłki jest niepraw
idłowa.'
        ];
    }

    // Aktualizacja metody wysyłki
    $_SESSION['cart']['shipping_method'] = $shipping_
method['id'];

```

```
$_SESSION['cart']['shipping'] = calculateShipping($shipping_method);
```

```
// Aktualizacja podsumowania koszyka  
updateCartTotals();
```

```
return [  
    'success' => true,  
    'message' => 'Metoda wysyłki została zaktuali  
zowana.',  
    'cart' => $_SESSION['cart']  
];  
}
```

```
// Funkcja obliczania kosztu wysyłki
```

```
function calculateShipping($method) {  
    $subtotal = $_SESSION['cart']['subtotal'] - $_SES  
SION['cart']['discount'];
```

```
    // Darmowa wysyłka powyżej określonej kwoty  
    if ($method['free_shipping_threshold'] > 0 && $su  
btotal >= $method['free_shipping_threshold']) {  
        return 0;  
    }
```

```
    return $method['cost'];  
}
```

```
// Funkcja aktualizacji podsumowania koszyka
```

```
function updateCartTotals() {  
    // Obliczenie sumy produktów i liczby elementów  
    $item_count = 0;  
    $subtotal = 0;  
  
    foreach ($_SESSION['cart']['items'] as $item) {  
        $item_count += $item['quantity'];  
        $subtotal += $item['subtotal'];  
    }
```



```

$_SESSION['cart']['item_count'] = $item_count;
$_SESSION['cart']['subtotal'] = $subtotal;

// Obliczenie sumy końcowej
$total = $subtotal - $_SESSION['cart']['discount']
] + $_SESSION['cart']['shipping'];
$_SESSION['cart']['total'] = max(0, $total); // U
pewnienie się, że suma nie jest ujemna

return $_SESSION['cart'];
}

// Funkcja zwracająca dostępne metody wysyłki
function getShippingMethods() {
    global $db;

    $query = "SELECT * FROM shipping_methods WHERE ac
tive = 1 ORDER BY sort_order";
    $stmt = $db->prepare($query);
    $stmt->execute();
    $methods = $stmt->fetchAll(PDO::FETCH_ASSOC);

    $shipping_methods = [];
    foreach ($methods as $method) {
        $shipping_methods[] = [
            'id' => $method['id'],
            'name' => $method['name'],
            'description' => $method['description'],
            'cost' => $method['cost'],
            'free_shipping_threshold' => $method['fre
e_shipping_threshold']
        ];
    }

    return $shipping_methods;
}

// Funkcja zapisująca zamówienie
function createOrder($user_id, $shipping_address, $bi

```

```

lling_address, $payment_method) {
    global $db;

    // Sprawdzenie, czy koszyk nie jest pusty
    if (empty($_SESSION['cart']['items'])) {
        return [
            'success' => false,
            'message' => 'Koszyk jest pusty.'
        ];
    }

    try {
        // Rozpoczęcie transakcji
        $db->beginTransaction();

        // Generowanie numeru zamówienia
        $order_number = generateOrderNumber();

        // Zapisanie zamówienia
        $query = "INSERT INTO orders (order_number, u
ser_id, status, subtotal, discount, shipping, total,
shipping_method
, coupon_id, payment_method, created_at)
VALUES (:order_number, :user_id, 'p
ending', :subtotal, :discount, :shipping, :total,
:shipping_method, :coupon_i
d, :payment_method, NOW())";

        $stmt = $db->prepare($query);
        $stmt->bindParam(':order_number', $order_numbe
er);

        $stmt->bindParam(':user_id', $user_id);
        $stmt->bindParam(':subtotal', $_SESSION['cart
']['subtotal']);
        $stmt->bindParam(':discount', $_SESSION['cart
']['discount']);
        $stmt->bindParam(':shipping', $_SESSION['cart
']['shipping']);
        $stmt->bindParam(':total', $_SESSION['cart']['

```

```

'total']);
$stmt->bindParam(':shipping_method', $_SESSION[
N['cart']['shipping_method']);

$coupon_id = null;
if ($_SESSION['cart']['coupon']) {
    $coupon_id = $_SESSION['cart']['coupon']['
'id'];
}
$stmt->bindParam(':coupon_id', $coupon_id);
$stmt->bindParam(':payment_method', $payment_
method);
$stmt->execute();

$order_id = $db->lastInsertId();

// Zapisanie adresu wysyłki
$query = "INSERT INTO order_addresses (order_
id, type, first_name, last_name, company, address_1,
address
_2, city, state, postcode, country, phone, email)
VALUES (:order_id, 'shipping', :fir
st_name, :last_name, :company, :address_1,
:address_2, :city, :state,
:postcode, :country, :phone, :email)";

$stmt = $db->prepare($query);
$stmt->bindParam(':order_id', $order_id);
$stmt->bindParam(':first_name', $shipping_add
ress['first_name']);
$stmt->bindParam(':last_name', $shipping_addr
ess['last_name']);
$stmt->bindParam(':company', $shipping_addres
s['company']);
$stmt->bindParam(':address_1', $shipping_addr
ess['address_1']);
$stmt->bindParam(':address_2', $shipping_addr
ess['address_2']);
$stmt->bindParam(':city', $shipping_address['

```

```

city' ]];
    $stmt->bindParam(':state', $shipping_address[
'state' ]);
    $stmt->bindParam(':postcode', $shipping_addre
ss['postcode' ]);
    $stmt->bindParam(':country', $shipping_addres
s['country' ]);
    $stmt->bindParam(':phone', $shipping_address[
'phone' ]);
    $stmt->bindParam(':email', $shipping_address[
'email' ]);
    $stmt->execute();

    // Zapisanie adresu rozliczeniowego
    $query = "INSERT INTO order_addresses (order_
id, type, first_name, last_name, company, address_1,
                                address
_2, city, state, postcode, country, phone, email)
            VALUES (:order_id, 'billing', :firs
t_name, :last_name, :company, :address_1,
                    :address_2, :city, :state,
:postcode, :country, :phone, :email)";

    $stmt = $db->prepare($query);
    $stmt->bindParam(':order_id', $order_id);
    $stmt->bindParam(':first_name', $billing_addr
ess['first_name' ]);
    $stmt->bindParam(':last_name', $billing_addre
ss['last_name' ]);
    $stmt->bindParam(':company', $billing_address
['company' ]);
    $stmt->bindParam(':address_1', $billing_addre
ss['address_1' ]);
    $stmt->bindParam(':address_2', $billing_addre
ss['address_2' ]);
    $stmt->bindParam(':city', $billing_address['c
ity' ]);
    $stmt->bindParam(':state', $billing_address['
state' ]);

```

```

        $stmt->bindParam(':postcode', $billing_addresses['postcode']);
        $stmt->bindParam(':country', $billing_addresses['country']);
        $stmt->bindParam(':phone', $billing_addresses['phone']);
        $stmt->bindParam(':email', $billing_addresses['email']);
        $stmt->execute();

        // Zapisanie produktów
        foreach ($_SESSION['cart']['items'] as $item_key => $item) {
            $query = "INSERT INTO order_items (order_id, product_id, name, quantity, price, subtotal, attributes)
                    VALUES (:order_id, :product_id, :name, :quantity, :price, :subtotal, :attributes)";

            $stmt = $db->prepare($query);
            $stmt->bindParam(':order_id', $order_id);
            $stmt->bindParam(':product_id', $item['id']);

            $stmt->bindParam(':name', $item['name']);
            $stmt->bindParam(':quantity', $item['quantity']);
            $stmt->bindParam(':price', $item['price']);
            $stmt->bindParam(':subtotal', $item['subtotal']);

            $attributes_json = json_encode($item['attributes']);
            $stmt->bindParam(':attributes', $attributes_json);
            $stmt->execute();

            // Aktualizacja stanu magazynowego
            $query = "UPDATE products SET stock_quant

```

```

ity = stock_quantity - :quantity WHERE id = :id";
    $stmt = $db->prepare($query);
    $stmt->bindParam(':quantity', $item['quantity']);

    $stmt->bindParam(':id', $item['id']);
    $stmt->execute();
}

// Zatwierdzenie transakcji
$db->commit();

// Wyczyszczenie koszyka
clearCart();

return [
    'success' => true,
    'message' => 'Zamówienie zostało złożone
pomyślnie.',
    'order_id' => $order_id,
    'order_number' => $order_number
];
} catch (PDOException $e) {
    // Wycofanie transakcji w przypadku błędu
    $db->rollBack();

    error_log("Error creating order: " . $e->getMessage());

    return [
        'success' => false,
        'message' => 'Wystąpił błąd podczas skład
ania zamówienia. Spróbuj ponownie później.'
    ];
}

// Funkcja generująca numer zamówienia
function generateOrderNumber() {
    $prefix = 'ORD';

```

```

    $timestamp = time();
    $random = mt_rand(1000, 9999);

    return $prefix . $timestamp . $random;
}

```

Struktura bazy danych dla koszyka i zamówień:

-- Tabela produktów

```

CREATE TABLE products (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    slug VARCHAR(255) NOT NULL UNIQUE,
    description TEXT,
    price DECIMAL(10,2) NOT NULL,
    regular_price DECIMAL(10,2),
    sale_price DECIMAL(10,2),
    stock_quantity INT NOT NULL DEFAULT 0,
    stock_status ENUM('in_stock', 'out_of_stock', 'on_b
ackorder') DEFAULT 'in_stock',
    weight DECIMAL(10,2),
    dimensions JSON,
    featured BOOLEAN DEFAULT 0,
    status ENUM('published', 'draft', 'trash') DEFAULT
'published',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U
PDATE CURRENT_TIMESTAMP
);

```

-- Tabela metod wysyłki

```

CREATE TABLE shipping_methods (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    cost DECIMAL(10,2) NOT NULL DEFAULT 0,
    free_shipping_threshold DECIMAL(10,2) DEFAULT 0,
    active BOOLEAN DEFAULT 1,
    sort_order INT DEFAULT 0
);

```

-- Tabela kuponów rabatowych

```
CREATE TABLE coupons (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  code VARCHAR(50) NOT NULL UNIQUE,  
  type ENUM('percentage', 'fixed_amount') NOT NULL,  
  amount DECIMAL(10,2) NOT NULL,  
  minimum_amount DECIMAL(10,2) DEFAULT 0,  
  usage_limit INT DEFAULT 0,  
  usage_count INT DEFAULT 0,  
  active BOOLEAN DEFAULT 1,  
  expiry_date DATE,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

-- Tabela zamówień

```
CREATE TABLE orders (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  order_number VARCHAR(50) NOT NULL UNIQUE,  
  user_id INT NOT NULL,  
  status ENUM('pending', 'processing', 'on-hold', 'completed',  
  'cancelled', 'refunded', 'failed') DEFAULT 'pending',  
  subtotal DECIMAL(10,2) NOT NULL,  
  discount DECIMAL(10,2) DEFAULT 0,  
  shipping DECIMAL(10,2) DEFAULT 0,  
  total DECIMAL(10,2) NOT NULL,  
  shipping_method VARCHAR(50),  
  coupon_id INT,  
  payment_method VARCHAR(50) NOT NULL,  
  payment_status ENUM('pending', 'paid', 'failed') DEFAULT 'pending',  
  transaction_id VARCHAR(100),  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE  
  CURRENT_TIMESTAMP,  
  FOREIGN KEY (user_id) REFERENCES users(id),  
  FOREIGN KEY (shipping_method) REFERENCES shipping_methods(id),
```



```

FOREIGN KEY (coupon_id) REFERENCES coupons(id)
);

-- Tabela adresów zamówienia
CREATE TABLE order_addresses (
  id INT AUTO_INCREMENT PRIMARY KEY,
  order_id INT NOT NULL,
  type ENUM('billing', 'shipping') NOT NULL,
  first_name VARCHAR(100) NOT NULL,
  last_name VARCHAR(100) NOT NULL,
  company VARCHAR(100),
  address_1 VARCHAR(255) NOT NULL,
  address_2 VARCHAR(255),
  city VARCHAR(100) NOT NULL,
  state VARCHAR(100),
  postcode VARCHAR(20) NOT NULL,
  country VARCHAR(2) NOT NULL,
  phone VARCHAR(20),
  email VARCHAR(100) NOT NULL,
  FOREIGN KEY (order_id) REFERENCES orders(id) ON DEL
ETE CASCADE
);

-- Tabela elementów zamówienia
CREATE TABLE order_items (
  id INT AUTO_INCREMENT PRIMARY KEY,
  order_id INT NOT NULL,
  product_id INT NOT NULL,
  name VARCHAR(255) NOT NULL,
  quantity INT NOT NULL,
  price DECIMAL(10,2) NOT NULL,
  subtotal DECIMAL(10,2) NOT NULL,
  attributes JSON,
  FOREIGN KEY (order_id) REFERENCES orders(id) ON DEL
ETE CASCADE,
  FOREIGN KEY (product_id) REFERENCES products(id)
);

```

Dobre praktyki dla koszyków zakupowych: - Zapamiętuj zawartość koszyka między sesjami (lokalne przechowywanie/cookies/baza danych) - Umożliwiaj łatwe zwiększanie/zmniejszanie ilości produktów - Pokazuj podsumowanie cen (cena produktu, suma częściowa, podatek, wysyłka, rabaty, suma końcowa) - Wyświetlaj dostępność produktów (ilość w magazynie) - Umożliwiaj dodawanie kodów rabatowych - Obsługuj różne metody wysyłki i płatności - Zapewnij jasne komunikaty o błędach i potwierdzenia - Uwzględnij responsywność koszyka na urządzeniach mobilnych - Implementuj zabezpieczenia przed manipulacją danymi po stronie klienta - Aktualizuj koszyk asynchronicznie (AJAX), aby uniknąć przeładowywania strony

Dlaczego są ważne dla początkujących: - Koszyk zakupowy jest kluczowym elementem każdego sklepu internetowego - Dobrze zaprojektowany koszyk wpływa na współczynnik konwersji - Rozumienie działania koszyka pomaga zrozumieć przepływ danych między klientem a serwerem - Implementacja koszyka zakupowego wymaga integracji wielu aspektów programowania webowego

WYSZUKIWARKA

Wyszukiwarka to element strony umożliwiający użytkownikom szybkie znajdowanie treści poprzez wprowadzanie zapytań tekstowych, znacząco poprawiając ich doświadczenie i ułatwiając nawigację.

Kluczowe cechy: - Umożliwia szybkie znajdowanie zawartości na stronie - Obsługuje różne rodzaje zapytań (tekstowe, filtry) - Wyświetla trafne i posortowane wyniki - Oferuje podpowiedzi i korekty - Wspiera zaawansowane funkcje (autouzupełnianie, filtry, sortowanie)

Komponenty wyszukiwarki:

1. Interfejs wyszukiwarki (HTML/CSS):

```

<!-- Prosta wyszukiwarka w nagłówku -->
<div class="search-container">
  <form action="search.php" method="get" class="search-form">
    <input type="text" name="q" placeholder="Szukaj.." class="search-input" autocomplete="off">
    <button type="submit" class="search-button">
      <svg class="search-icon" width="16" height="16" viewBox="0 0 24 24">
        <path d="M15.5 14h-.79l-.28-.27a6.5 6.5 0 0 0 1.48-5.34c-.47-2.78-2.79-5-5.59-5.34a6.505 6.505 0 0 0 -7.27 7.27c.34 2.8 2.56 5.12 5.34 5.59a6.5 6.5 0 0 0 5.34-1.48l.27.28v.79l4.25 4.25c.41.41 1.08.41 1.49 0 .41-.41.41-1.08 0-1.49L15.5 14zm-6 0C7.01 14 5 11.99 5 9.5S7.01 5 9.5 5 14 7.01 14 9.5 11.99 14 9.5 14z" />
      </svg>
    </button>

    <!-- Kontener dla odpowiedzi -->
    <div class="search-suggestions" style="display: none;">
      <ul class="suggestions-list"></ul>
    </div>
  </form>
</div>

<!-- Zaawansowana wyszukiwarka na stronie wyników -->
<div class="advanced-search">
  <h2>Wyniki wyszukiwania</h2>

  <div class="search-header">
    <div class="search-stats">
      Znalezione <span class="results-count">42</span>
    > wyników dla zapytania "<span class="search-query">przykład</span>"
    </div>

    <div class="search-tools">

```

```

<div class="search-filters">
  <label for="category-filter">Kategoria:</label>

  <select id="category-filter" name="category">
    <option value="">Wszystkie kategorie</option>

    <option value="1">Kategoria 1</option>
    <option value="2">Kategoria 2</option>
    <option value="3">Kategoria 3</option>
  </select>

  <label for="date-filter">Data:</label>
  <select id="date-filter" name="date">
    <option value="">Dowolna data</option>
    <option value="day">Ostatnie 24 godziny</option>

    <option value="week">Ostatni tydzień</option>

    <option value="month">Ostatni miesiąc</option>

    <option value="year">Ostatni rok</option>
  </select>
</div>

<div class="search-sort">
  <label for="sort-order">Sortuj wg:</label>
  <select id="sort-order" name="sort">
    <option value="relevance">Trafności</option>

    <option value="date">Daty (od najnowszych)</option>
    <option value="date_asc">Daty (od najstarszych)</option>
    <option value="title">Tytułu (A-Z)</option>
    <option value="title_desc">Tytułu (Z-A)</option>
  </select>
</div>
</div>

```

```

</div>

<div class="search-results">
  <!-- Przykładowy wynik wyszukiwania -->
  <div class="search-result">
    <h3 class="result-title">
      <a href="article.php?id=123">Przykładowy tytuł
      artykułu z wyróżnionym <mark>przykład</mark>em słow
      a kluczowego</a>
    </h3>
    <div class="result-meta">
      <span class="result-date">12.03.2025</span> |

      <span class="result-category">Kategoria 1</span> |

      <span class="result-author">Jan Kowalski</span>
    </div>
    <div class="result-excerpt">
      Lorem ipsum dolor sit amet, consectetur adipi
      scing elit. Nullam in ipsum ut velit aliquam vulputat
      e. In hac habitasse platea dictumst. Sed eget <mark>p
      rzykład</mark> mauris, vitae tempus nisi. Praesent eu
      ismod...
    </div>
  </div>

  <!-- Więcej wyników wyszukiwania... -->
</div>

<!-- Paginacja -->
<div class="search-pagination">
  <a href="#" class="pagination-prev disabled">&laquo;
  uo; Poprzednia</a>
  <div class="pagination-pages">
    <a href="#" class="page active">1</a>
    <a href="#" class="page">2</a>
    <a href="#" class="page">3</a>
    <span class="dots">...</span>
  </div>
</div>

```

```

        <a href="#" class="page">10</a>
    </div>
    <a href="#" class="pagination-next">Następna &raq
uo;</a>
</div>

<!-- Brak wyników -->
<div class="no-results" style="display: none;">
    <h3>Nie znaleziono wyników dla zapytania "<span c
lass="search-query">przykład</span>"</h3>
    <p>Sugestie:</p>
    <ul>
        <li>Sprawdź pisownię wyszukiwanego hasła</li>
        <li>Spróbuj użyć innych słów</li>
        <li>Użyj bardziej ogólnych słów kluczowych</li>
        <li>Zmniejsz liczbę filtrów</li>
    </ul>
</div>
</div>

```

2. CSS dla wyszukiwarki:

```

/* Prosta wyszukiwarka */
.search-container {
    position: relative;
    max-width: 400px;
    margin: 0 auto;
}

.search-form {
    display: flex;
    align-items: center;
}

.search-input {
    width: 100%;
    padding: 10px 40px 10px 15px;
    border: 1px solid #ddd;
    border-radius: 50px;
    font-size: 14px;
}

```

```
    transition: border-color 0.3s, box-shadow 0.3s;
}

.search-input:focus {
    outline: none;
    border-color: #4285f4;
    box-shadow: 0 0 0 2px rgba(66, 133, 244, 0.2);
}

.search-button {
    position: absolute;
    right: 5px;
    top: 50%;
    transform: translateY(-50%);
    background: transparent;
    border: none;
    cursor: pointer;
    padding: 8px;
}

.search-icon {
    fill: #5f6368;
}

.search-button:hover .search-icon {
    fill: #4285f4;
}

/* Podpowiedzi */
.search-suggestions {
    position: absolute;
    top: 100%;
    left: 0;
    right: 0;
    background: white;
    border: 1px solid #ddd;
    border-top: none;
    border-radius: 0 0 8px 8px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}
```

```
    z-index: 100;
    margin-top: -5px;
}

.suggestions-list {
    list-style: none;
    padding: 0;
    margin: 0;
}

.suggestions-list li {
    padding: 10px 15px;
    cursor: pointer;
    transition: background-color 0.2s;
}

.suggestions-list li:hover,
.suggestions-list li.active {
    background-color: #f1f3f4;
}

.suggestions-list mark {
    background: none;
    font-weight: bold;
    color: #4285f4;
}

/* Zaawansowana wyszukiwarka */
.advanced-search {
    max-width: 800px;
    margin: 0 auto;
    padding: 20px;
}

.search-header {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-between;
    align-items: center;
```



```
margin-bottom: 20px;
padding-bottom: 15px;
border-bottom: 1px solid #eee;
}

.search-stats {
margin-bottom: 10px;
color: #666;
}

.results-count,
.search-query {
font-weight: bold;
}

.search-tools {
display: flex;
flex-wrap: wrap;
gap: 15px;
}

.search-filters,
.search-sort {
display: flex;
align-items: center;
flex-wrap: wrap;
gap: 8px;
}

.search-filters label,
.search-sort label {
margin-right: 5px;
font-size: 14px;
}

.search-filters select,
.search-sort select {
padding: 6px 10px;
border: 1px solid #ddd;
}
```

```
border-radius: 4px;
background-color: white;
}

/* Wyniki wyszukiwania */
.search-results {
  margin-bottom: 30px;
}

.search-result {
  margin-bottom: 25px;
  padding-bottom: 20px;
  border-bottom: 1px solid #f0f0f0;
}

.search-result:last-child {
  border-bottom: none;
}

.result-title {
  margin: 0 0 8px 0;
  font-size: 18px;
}

.result-title a {
  color: #1a0dab;
  text-decoration: none;
}

.result-title a:hover {
  text-decoration: underline;
}

.result-meta {
  margin-bottom: 8px;
  font-size: 13px;
  color: #70757a;
}
```

```

.result-excerpt {
  font-size: 14px;
  line-height: 1.5;
  color: #4d5156;
}

.result-excerpt mark,
.result-title mark {
  background-color: rgba(255, 213, 0, 0.3);
  padding: 0 2px;
}

/* Paginacja */
.search-pagination {
  display: flex;
  justify-content: center;
  align-items: center;
  margin-top: 30px;
}

.pagination-prev,
.pagination-next {
  padding: 8px 12px;
  border: 1px solid #ddd;
  border-radius: 4px;
  color: #1a0dab;
  text-decoration: none;
}

.pagination-prev.disabled,
.pagination-next.disabled {
  color: #70757a;
  border-color: #f0f0f0;
  pointer-events: none;
}

.pagination-pages {
  display: flex;
  margin: 0 10px;
}

```

```

}

.page,
.dots {
  padding: 8px 12px;
  margin: 0 2px;
  color: #70757a;
  text-decoration: none;
}

.page:hover {
  background-color: #f1f3f4;
  border-radius: 4px;
}

.page.active {
  background-color: #4285f4;
  color: white;
  border-radius: 4px;
}

/* Brak wyników */
.no-results {
  padding: 20px;
  background-color: #f8f9fa;
  border-radius: 8px;
}

.no-results h3 {
  margin-top: 0;
  color: #202124;
}

.no-results ul {
  padding-left: 20px;
  line-height: 1.6;
}

/* Responsywność */

```

```

@media (max-width: 576px) {
  .search-header {
    flex-direction: column;
    align-items: flex-start;
  }

  .search-tools {
    flex-direction: column;
    width: 100%;
  }

  .search-filters,
  .search-sort {
    width: 100%;
    margin-bottom: 10px;
  }

  .search-filters select,
  .search-sort select {
    width: 100%;
  }
}

```

3. JavaScript dla autouzupełniania:

```

document.addEventListener('DOMContentLoaded', function() {
  const searchInput = document.querySelector('.search-input');
  const suggestionsContainer = document.querySelector('.search-suggestions');
  const suggestionsList = document.querySelector('.suggestions-list');

  let currentSuggestions = [];
  let selectedSuggestionIndex = -1;

  // Następowanie wprowadzania tekstu
  searchInput.addEventListener('input', debounce(function() {

```

```

const query = this.value.trim();

if (query.length < 2) {
  hideSuggestions();
  return;
}

fetchSuggestions(query);
}, 300));

// Obsługa klawiatury
searchInput.addEventListener('keydown', function(e)
{
  // Jeśli podpowiedzi nie są widoczne, nie rób nic
  if (suggestionsContainer.style.display === 'none'
) {
    return;
  }

  switch(e.key) {
    case 'ArrowDown':
      e.preventDefault();
      navigateSuggestions('down');
      break;
    case 'ArrowUp':
      e.preventDefault();
      navigateSuggestions('up');
      break;
    case 'Enter':
      if (selectedSuggestionIndex >= 0) {
        e.preventDefault();
        selectSuggestion(selectedSuggestionIndex);
      }
      break;
    case 'Escape':
      hideSuggestions();
      break;
  }
});

```

```

// Ukrywanie podpowiedzi po kliknięciu poza polem w
// wyszukiwania
document.addEventListener('click', function(e) {
    if (!searchInput.contains(e.target) && !suggestionsContainer.contains(e.target)) {
        hideSuggestions();
    }
});

// Funkcja pobierająca podpowiedzi
function fetchSuggestions(query) {
    // Przykładowa implementacja z użyciem fetch
    fetch(`suggestions.php?q=${encodeURIComponent(query)}`)
        .then(response => response.json())
        .then(data => {
            currentSuggestions = data;
            displaySuggestions(query, data);
        })
        .catch(error => {
            console.error('Error fetching suggestions:',
error);
            hideSuggestions();
        });

    // Alternatywna wersja z twardymi danymi (dla dem
    onstracji)
    /*
    const demoSuggestions = [
        query + " przykład 1",
        query + " przykład 2",
        query + " przykładowy produkt",
        "przykład " + query,
        "najlepszy " + query
    ];

    currentSuggestions = demoSuggestions;
    displaySuggestions(query, demoSuggestions);

```

```

    */
}

// Funkcja wyświetlająca podpowiedzi
function displaySuggestions(query, suggestions) {
    if (suggestions.length === 0) {
        hideSuggestions();
        return;
    }

    // Wyczyszczenie listy podpowiedzi
    suggestionsList.innerHTML = '';

    // Wypełnienie listy podpowiedziami
    suggestions.forEach((suggestion, index) => {
        const li = document.createElement('li');

        // Wyróżnienie zapytania w podpowiedzi
        const highlightedText = highlightQuery(suggestion, query);
        li.innerHTML = highlightedText;

        // Dodanie zdarzenia kliknięcia
        li.addEventListener('click', function() {
            selectSuggestion(index);
        });

        // Dodanie zdarzenia najechania myszą
        li.addEventListener('mouseenter', function() {
            selectedSuggestionIndex = index;
            updateSelectedSuggestion();
        });

        suggestionsList.appendChild(li);
    });

    // Pokazanie kontenera podpowiedzi
    suggestionsContainer.style.display = 'block';
}

```



```

// Resetowanie indeksu wybranej podpowiedzi
selectedSuggestionIndex = -1;
}

// Funkcja wyróżniająca zapytanie w tekście podpowi
edzi
function highlightQuery(text, query) {
  const regex = new RegExp(`(${escapeRegExp(query)})`
), 'gi');
  return text.replace(regex, '<mark>$1</mark>');
}

// Funkcja nawigacji po podpowiedziach za pomocą st
rzatek
function navigateSuggestions(direction) {
  if (direction === 'down') {
    selectedSuggestionIndex = Math.min(selectedSugg
estionIndex + 1, currentSuggestions.length - 1);
  } else if (direction === 'up') {
    selectedSuggestionIndex = Math.max(selectedSugg
estionIndex - 1, -1);
  }

  updateSelectedSuggestion();
}

// Funkcja aktualizująca zaznaczoną podpowieź
function updateSelectedSuggestion() {
  const suggestions = suggestionsList.querySelector
All('li');

  // Usunięcie klasy active ze wszystkich podpowied
zi
  suggestions.forEach(suggestion => {
    suggestion.classList.remove('active');
  });

  // Dodanie klasy active do wybranej podpowiedzi
  if (selectedSuggestionIndex >= 0) {

```

```

        suggestions[selectedSuggestionIndex].classList.
add('active');

        // Aktualizacja pola wyszukiwania, ale bez wysy
łania formularza
        searchInput.value = currentSuggestions[selected
SuggestionIndex];
    } else {
        // Przywrócenie oryginalnego zapytania
        searchInput.value = searchInput.dataset.origina
lQuery || searchInput.value;
    }
}

// Funkcja wybierająca odpowiedź
function selectSuggestion(index) {
    searchInput.value = currentSuggestions[index];
    hideSuggestions();

    // Opcjonalnie: automatyczne wysłanie formularza
    searchInput.form.submit();
}

// Funkcja ukrywająca odpowiedzi
function hideSuggestions() {
    suggestionsContainer.style.display = 'none';
    selectedSuggestionIndex = -1;
}

// Funkcja zabezpieczająca znaki specjalne w wyraże
niach regularnych
function escapeRegExp(string) {
    return string.replace(/[\.*+?^${}()|[\]\\"/g, '\\$
&');
}

// Funkcja ograniczająca liczbę wywołań funkcji
function debounce(func, wait) {
    let timeout;

```

```

return function() {
  const context = this, args = arguments;
  clearTimeout(timeout);
  timeout = setTimeout(() => {
    func.apply(context, args);
  }, wait);
};
}
});

```

4. JavaScript dla strony wyników wyszukiwania:

```

document.addEventListener('DOMContentLoaded', function() {
  // Elementy DOM
  const searchForm = document.querySelector('.search-form');
  const searchInput = document.querySelector('.search-input');
  const categoryFilter = document.getElementById('category-filter');
  const dateFilter = document.getElementById('date-filter');
  const sortOrder = document.getElementById('sort-order');
  const resultsContainer = document.querySelector('.search-results');
  const pagination = document.querySelector('.search-pagination');

  // Parametry URL
  const urlParams = new URLSearchParams(window.location.search);
  const query = urlParams.get('q') || '';
  const page = parseInt(urlParams.get('page')) || 1;
  const category = urlParams.get('category') || '';
  const date = urlParams.get('date') || '';
  const sort = urlParams.get('sort') || 'relevance';

  // Ustawienie wartości filtrów zgodnie z parametrami

```

```

i URL
searchInput.value = query;
if (categoryFilter) categoryFilter.value = category
;
if (dateFilter) dateFilter.value = date;
if (sortOrder) sortOrder.value = sort;

// Następowanie zmian filtrów
if (categoryFilter) {
    categoryFilter.addEventListener('change', updateSearch);
}

if (dateFilter) {
    dateFilter.addEventListener('change', updateSearch);
}

if (sortOrder) {
    sortOrder.addEventListener('change', updateSearch);
}

// Funkcja aktualizująca wyniki wyszukiwania po zmianie filtrów
function updateSearch() {
    // Tworzenie nowego URL z aktualnymi parametrami
    const newUrl = new URL(window.location.href);
    const params = newUrl.searchParams;

    // Aktualizacja parametrów
    if (query) params.set('q', query);
    if (categoryFilter && categoryFilter.value) {
        params.set('category', categoryFilter.value);
    } else {
        params.delete('category');
    }

    if (dateFilter && dateFilter.value) {

```

```

        params.set('date', dateFilter.value);
    } else {
        params.delete('date');
    }

    if (sortOrder && sortOrder.value) {
        params.set('sort', sortOrder.value);
    } else {
        params.delete('sort');
    }

    // Zresetowanie strony przy zmianie filtrów
    params.set('page', '1');

    // Przekierowanie do nowego URL
    window.location.href = newUrl.toString();
}

// Obsługa paginacji
if (pagination) {
    const pageLinks = pagination.querySelectorAll('.p
age');
    const prevLink = pagination.querySelector('.pagin
ation-prev');
    const nextLink = pagination.querySelector('.pagin
ation-next');

    // Dodanie zdarzeń do linków stron
    pageLinks.forEach(link => {
        link.addEventListener('click', function(e) {
            e.preventDefault();

            const pageNum = parseInt(this.textContent);
            navigateToPage(pageNum);
        });
    });

    // Dodanie zdarzeń do linków poprzednia/następna
    if (prevLink && !prevLink.classList.contains('dis

```

```

abled')) {
    prevLink.addEventListener('click', function(e)
    {
        e.preventDefault();

        navigateToPage(page - 1);
    });
}

if (nextLink && !nextLink.classList.contains('dis
abled')) {
    nextLink.addEventListener('click', function(e)
    {
        e.preventDefault();

        navigateToPage(page + 1);
    });
}
}

```

// Funkcja nawigacji do określonej strony

```

function navigateToPage(pageNum) {
    const newUrl = new URL(window.location.href);
    newUrl.searchParams.set('page', pageNum);

    window.location.href = newUrl.toString();
}

```

// Funkcja aktualizująca widoczność komunikatu "brak wyników"

```

function updateNoResultsMessage() {
    const noResultsElem = document.querySelector('.no
-results');
    const resultsCount = document.querySelector('.res
ults-count');
    const searchQueryElements = document.querySelecto
rAll('.search-query');

    if (noResultsElem && resultsCount) {

```

```

    );

    if (count === 0) {
        noResultsElem.style.display = 'block';
        resultsContainer.style.display = 'none';
        pagination.style.display = 'none';
    } else {
        noResultsElem.style.display = 'none';
        resultsContainer.style.display = 'block';
        pagination.style.display = 'flex';
    }
}

// Aktualizacja zapytania we wszystkich elementach
searchQueryElements.forEach(elem => {
    elem.textContent = query;
});

// Wywołanie funkcji aktualizującej
updateNoResultsMessage();
});

```

5. Backend wyszukiwarki (PHP):

```

<?php
// Plik: search.php

// Połączenie z bazą danych
require_once 'config.php';
require_once 'functions.php';

// Parametry wyszukiwania
$query = isset($_GET['q']) ? trim($_GET['q']) : '';
$page = isset($_GET['page']) && is_numeric($_GET['page']) ? (int)$_GET['page'] : 1;
$category = isset($_GET['category']) ? $_GET['category'] : '';

```

```

$date_filter = isset($_GET['date']) ? $_GET['date'] :
    '';
$sort = isset($_GET['sort']) ? $_GET['sort'] : 'relev
ance';

// Parametry paginacji
$results_per_page = 10;
$offset = ($page - 1) * $results_per_page;

// Inicjalizacja wyników
$results = [];
$total_results = 0;

// Jeśli zapytanie jest puste, przekieruj na stronę g
łówną
if (empty($query)) {
    header('Location: index.php');
    exit;
}

// Wykonanie wyszukiwania, tylko jeśli istnieje zapyt
anie
if (!empty($query)) {
    // Przygotowanie zapytania SQL
    $sql_conditions = [];
    $sql_params = [];

    // Podstawowe wyszukiwanie w tytule i treści
    $sql_conditions[] = "(title LIKE ? OR content LIK
E ?)";
    $sql_params[] = "%$query%";
    $sql_params[] = "%$query%";

    // Dodanie filtra kategorii, jeśli wybrany
    if (!empty($category)) {
        $sql_conditions[] = "category_id = ?";
        $sql_params[] = $category;
    }
}

```



```

// Dodanie filtra daty, jeśli wybrany
if (!empty($date_filter)) {
    switch ($date_filter) {
        case 'day':
            $sql_conditions[] = "published_at >=
DATE_SUB(NOW(), INTERVAL 1 DAY)";
            break;
        case 'week':
            $sql_conditions[] = "published_at >=
DATE_SUB(NOW(), INTERVAL 1 WEEK)";
            break;
        case 'month':
            $sql_conditions[] = "published_at >=
DATE_SUB(NOW(), INTERVAL 1 MONTH)";
            break;
        case 'year':
            $sql_conditions[] = "published_at >=
DATE_SUB(NOW(), INTERVAL 1 YEAR)";
            break;
    }
}

// łączenie warunków
$sql_where = implode(' AND ', $sql_conditions);

// Sortowanie wyników
$sql_order = "";
switch ($sort) {
    case 'date':
        $sql_order = "ORDER BY published_at DESC"
;
        break;
    case 'date_asc':
        $sql_order = "ORDER BY published_at ASC";
        break;
    case 'title':
        $sql_order = "ORDER BY title ASC";
        break;
    case 'title_desc':

```

```

        $sql_order = "ORDER BY title DESC";
        break;
    default:
        // Sortowanie wg trafności (domyślne)
        // Wyniki zawierające zapytanie w tytule
        mają wyższą wagę
        $sql_order = "ORDER BY (
            CASE
                WHEN title LIKE ? THEN 3
                WHEN title LIKE ? THEN 2
                ELSE 1
            END
        ) DESC, published_at DESC";
        array_push($sql_params, "%$query%", "% $q
query %");
    }

    // Zapytanie SQL zliczające całkowitą liczbę wyni
ków
    $count_sql = "SELECT COUNT(*) FROM articles WHERE
$sql_where";
    $stmt = $db->prepare($count_sql);

    // Bindowanie parametrów
    for ($i = 0; $i < count($sql_params); $i++) {
        $stmt->bindValue($i + 1, $sql_params[$i]);
    }

    $stmt->execute();
    $total_results = $stmt->fetchColumn();

    // Zapytanie SQL pobierające wyniki dla bieżącej
strony
    $search_sql = "SELECT id, title, excerpt, content
, category_id, author_id, published_at
        FROM articles
        WHERE $sql_where
        $sql_order
        LIMIT $offset, $results_per_page";

```

```

$stmt = $db->prepare($search_sql);

// Bindowanie parametrów
for ($i = 0; $i < count($sql_params); $i++) {
    $stmt->bindValue($i + 1, $sql_params[$i]);
}

$stmt->execute();
$results = $stmt->fetchAll(PDO::FETCH_ASSOC);

// Przetwarzanie wyników
foreach ($results as &$result) {
    // Pobieranie dodatkowych informacji
    $result['category_name'] = getCategoryName($result['category_id']);
    $result['author_name'] = getAuthorName($result['author_id']);

    // Tworzenie fragmentu tekstu z wyróżnionym zapytaniem
    $result['snippet'] = generateSnippet($result['content'], $query);

    // Wyróżnienie zapytania w tytule
    $result['highlighted_title'] = highlightQuery($result['title'], $query);

    // Formatowanie daty
    $result['formatted_date'] = date('d.m.Y', strtotime($result['published_at']));

    // Generowanie URL artykułu
    $result['url'] = 'article.php?id=' . $result['id'];
}

// Funkcje pomocnicze

```

```
function getCategoryName($category_id) {
    global $db;

    $stmt = $db->prepare("SELECT name FROM categories
WHERE id = ?");
    $stmt->bindParam(1, $category_id);
    $stmt->execute();

    return $stmt->fetchColumn() ?: 'Bez kategorii';
}
```

```
function getAuthorName($author_id) {
    global $db;

    $stmt = $db->prepare("SELECT CONCAT(first_name, '
', last_name) AS name FROM users WHERE id = ?");
    $stmt->bindParam(1, $author_id);
    $stmt->execute();

    return $stmt->fetchColumn() ?: 'Nieznany autor';
}
```

```
function generateSnippet($content, $query) {
    // Usunięcie znaczników HTML
    $text = strip_tags($content);

    // Wyszukanie pozycji zapytania w tekście
    $position = stripos($text, $query);

    if ($position === false) {
        // Jeśli zapytanie nie występuje dokładnie, s
        próbuj znaleźć podobne
        $words = explode(' ', $query);
        foreach ($words as $word) {
            if (strlen($word) > 3) { // Ignoruj krót
kie słowa
                $pos = stripos($text, $word);
                if ($pos !== false) {
                    $position = $pos;
                }
            }
        }
    }
}
```

```

        break;
    }
}

// Jeśli nie znaleziono, użyj początku tekstu
if ($position === false) {
    $position = 0;
}

// Określenie zakresu fragmentu
$start = max(0, $position - 100);
$length = 200;

// Dostosowanie początku, aby zaczynał się od poc
zątku słowa
if ($start > 0) {
    $prev_space = strrpos(substr($text, 0, $start
), ' ');
    if ($prev_space !== false) {
        $start = $prev_space + 1;
    }
}

// Pobieranie fragmentu
$snippet = substr($text, $start, $length);

// Dodanie wielokropka na początku, jeśli nie zac
zynamy od początku tekstu
if ($start > 0) {
    $snippet = '...' . $snippet;
}

// Dodanie wielokropka na końcu, jeśli fragment n
ie kończy się na końcu tekstu
if ($start + $length < strlen($text)) {
    $snippet = $snippet . '...';
}

```

```

// Wyróżnienie zapytania
return highlightQuery($snippet, $query);
}

function highlightQuery($text, $query) {
    // Przygotowanie wyrażenia regularnego
    $words = explode(' ', $query);
    $patterns = [];

    foreach ($words as $word) {
        if (strlen($word) > 2) { // Ignoruj bardzo k
            $pattern = preg_quote($word, '/');
            $patterns[] = "/\b($pattern)\b/i"; // Ty
        }
    }

    // Zastąpienie wszystkich wystąpień tagiem <mark>
    if (!empty($patterns)) {
        $text = preg_replace($patterns, '<mark>$1</ma
rk>', $text);
    }

    return $text;
}

// Obliczenie danych paginacji
$total_pages = ceil($total_results / $results_per_pag
e);
$max_pages_to_show = 5;
?>

```

6. PHP dla funkcji autouzupełniania:

```

<?php
// Plik: suggestions.php

// Połączenie z bazą danych

```

```

require_once 'config.php';

// Pobranie zapytania
$query = isset($_GET['q']) ? trim($_GET['q']) : '';
$limit = isset($_GET['limit']) ? (int)$_GET['limit']
: 10;

// Sprawdzenie, czy zapytanie nie jest puste
if (empty($query) || strlen($query) < 2) {
    echo json_encode([]);
    exit;
}

// Tablica na sugestie
$suggestions = [];

try {
    // Wyszukiwanie w popularnych zapytaniach
    $sql = "SELECT query, count FROM search_history
            WHERE query LIKE ?
            ORDER BY count DESC
            LIMIT ?";

    $stmt = $db->prepare($sql);
    $stmt->bindValue(1, "%$query%");
    $stmt->bindValue(2, $limit, PDO::PARAM_INT);
    $stmt->execute();

    $history_results = $stmt->fetchAll(PDO::FETCH_ASSOC);

    foreach ($history_results as $result) {
        $suggestions[] = $result['query'];
    }

    // Jeśli mamy mniej sugestii niż limit, dodaj sug
    estie z tytułów artykułów
    if (count($suggestions) < $limit) {
        $remaining = $limit - count($suggestions);
    }
}

```

```

$sql = "SELECT title FROM articles
        WHERE title LIKE ?
        ORDER BY published_at DESC
        LIMIT ?";

$stmt = $db->prepare($sql);
$stmt->bindValue(1, "%$query%");
$stmt->bindValue(2, $remaining, PDO::PARAM_INT);

$stmt->execute();

$title_results = $stmt->fetchAll(PDO::FETCH_ASSOC);

foreach ($title_results as $result) {
    // Sprawdzenie, czy sugestia już istnieje
    if (!in_array($result['title'], $suggestions)) {
        $suggestions[] = $result['title'];
    }
}

// Jeśli nadal mamy mniej sugestii niż limit, dodaj sugestie z nazw kategorii
if (count($suggestions) < $limit) {
    $remaining = $limit - count($suggestions);

    $sql = "SELECT name FROM categories
            WHERE name LIKE ?
            LIMIT ?";

    $stmt = $db->prepare($sql);
    $stmt->bindValue(1, "%$query%");
    $stmt->bindValue(2, $remaining, PDO::PARAM_INT);

    $stmt->execute();

```



```

        $category_results = $stmt->fetchAll(PDO::FETCH_
H_ASSOC);

        foreach ($category_results as $result) {
            // Sprawdzenie, czy sugestia już istnieje
            if (!in_array($result['name'], $suggestions
ns)) {

                $suggestions[] = $result['name'];
            }
        }
    }

    // Zapisanie zapytania w historii
    logSearchQuery($query);

    // Zwrócenie wyników jako JSON
    echo json_encode($suggestions);
} catch (PDOException $e) {
    // W przypadku błędu zwracamy pustą tablicę
    error_log("Error in suggestions.php: " . $e->getM
essage());
    echo json_encode([]);
}

// Funkcja zapisująca zapytanie w historii
function logSearchQuery($query) {
    global $db;

    try {
        // Sprawdzenie, czy zapytanie już istnieje
        $sql = "SELECT id, count FROM search_history
WHERE query = ?";
        $stmt = $db->prepare($sql);
        $stmt->bindValue(1, $query);
        $stmt->execute();

        $result = $stmt->fetch(PDO::FETCH_ASSOC);

        if ($result) {

```

```

        // Aktualizacja licznika
        $sql = "UPDATE search_history SET count =
count + 1, last_searched = NOW() WHERE id = ?";
        $stmt = $db->prepare($sql);
        $stmt->bindValue(1, $result['id']);
        $stmt->execute();
    } else {
        // Dodanie nowego zapytania
        $sql = "INSERT INTO search_history (query
, count, first_searched, last_searched)
VALUES (?, 1, NOW(), NOW())";
        $stmt = $db->prepare($sql);
        $stmt->bindValue(1, $query);
        $stmt->execute();
    }
} catch (PDOException $e) {
    error_log("Error logging search query: " . $e
->getMessage());
}
}
?>

```

Struktura bazy danych dla wyszukiwarki:

```

-- Tabela historii wyszukiwań
CREATE TABLE search_history (
    id INT AUTO_INCREMENT PRIMARY KEY,
    query VARCHAR(255) NOT NULL,
    count INT NOT NULL DEFAULT 1,
    first_searched TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_searched TIMESTAMP DEFAULT CURRENT_TIMESTAMP O
N UPDATE CURRENT_TIMESTAMP,
    UNIQUE KEY (query)
);

-- Tabela synonimów (dla zaawansowanego wyszukiwania)
CREATE TABLE search_synonyms (
    id INT AUTO_INCREMENT PRIMARY KEY,
    term VARCHAR(255) NOT NULL,
    synonym VARCHAR(255) NOT NULL,

```

UNIQUE KEY (term, synonym)

);

Dobre praktyki dla wyszukiwarek: - Implementuj autouzupełnianie dla lepszego doświadczenia użytkownika - Aktualizuj wyniki w czasie rzeczywistym (Ajax) dla szybszej interakcji - Wyróżniaj zapytanie w wynikach wyszukiwania - Umożliwiaj filtrowanie i sortowanie wyników - Implementuj korektę błędów i sugestie alternatywnych zapytań - Optymalizuj zapytania do bazy danych dla szybszych wyników - Uwzględniaj synonimy i pokrewne terminy - Zapewnij paginację dla dużej liczby wyników - Zapisuj historię wyszukiwań w celu poprawy sugestii - Uwzględniaj popularne i trafne wyniki na początku - Testuj wyszukiwarkę z różnymi zapytaniami i brzegowymi przypadkami

Zaawansowane techniki wyszukiwania: - **Indeksowanie pełnotekstowe** - specjalne indeksy w bazie danych optymalizujące wyszukiwanie tekstu - **Stemming** - sprowadzanie słów do rdzenia (np. “pisanie”, “pisał”, “piszę” do “pis”) - **Analiza n-gramów** - badanie sekwencji sąsiadujących słów - **Wagi i punktacja trafności** - przypisywanie różnych wag różnym polom (np. tytuł vs. treść) - **Wyszukiwanie rozmyte (fuzzy search)** - znajdowanie podobnych terminów, nawet z błędami - **Geolokalizacja** - uwzględnianie lokalizacji użytkownika w wynikach - **Wyszukiwanie fasetowe** - dynamiczne filtry oparte na atrybutach wyników - **Silniki wyszukiwania** - Elasticsearch, Solr, Algolia dla zaawansowanych potrzeb

Dlaczego są ważne dla początkujących: - Wyszukiwarka to kluczowy element nawigacji na stronach z dużą ilością treści - Dobrze działająca wyszukiwarka znacząco poprawia doświadczenie użytkownika - Implementacja wyszukiwania łączy frontend z backendem - Nauka tworzenia wyszukiwarek rozwija umiejętności pracy z bazami danych i JavaScript

KOMENTARZE

System komentarzy umożliwia użytkownikom interakcję z treścią witryny, wyrażanie opinii i prowadzenie dyskusji, co zwiększa zaangażowanie i buduje społeczność wokół strony.

Kluczowe cechy: - Możliwość dodawania, edytowania i usuwania komentarzy - Hierarchiczna struktura (odpowiedzi na komentarze) - Moderacja i zgłaszanie nieodpowiednich treści - Powiadomienia o odpowiedziach - Integracja z systemem użytkowników

Komponenty systemu komentarzy:

1. Interfejs komentarzy (HTML/CSS):

```
<section class="comments-section" id="comments">
  <h3 class="comments-title">Komentarze (15)</h3>

  <!-- Formularz dodawania komentarza -->
  <div class="comment-form-container">
    <form class="comment-form" id="comment-form">
      <div class="form-row">
        
        <div class="form-fields">
          <textarea name="comment_content" id="commen
t_content" rows="3" placeholder="Napisz komentarz..."
required></textarea>

          <!-- Pola dla niezalogowanych użytkowników
-->
          <div class="guest-fields" style="display: n
one;">
            <div class="input-group">
              <input type="text" name="author_name" p
laceholder="Imię" required>
              <input type="email" name="author_email"
placeholder="Email (nie będzie widoczny)" required>
            </div>
          </div>
        </div>
      </div>
    </form>
  </div>
```

```

        </div>
    </div>

    <div class="form-actions">
        <input type="hidden" name="parent_id" value="0">
        <input type="hidden" name="post_id" value="123">
        <button type="button" class="btn btn-cancel" style="display: none;">Anuluj</button>
        <button type="submit" class="btn btn-submit">Dodaj komentarz</button>
    </div>
</form>
</div>

<!-- Lista komentarzy -->
<div class="comments-list">
    <!-- Komentarz główny -->
    <div class="comment" id="comment-1">
        <div class="comment-avatar">
            
        </div>

        <div class="comment-content">
            <div class="comment-header">
                <span class="comment-author">Jan Kowalski</span>
                <span class="comment-date">2 godziny temu</span>
            </div>

            <div class="comment-text">
                <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam eget felis eget nunc lobortis mattis aliquam faucibus purus in.</p>
            </div>
        </div>
    </div>

```

```

<div class="comment-actions">
  <button class="action-btn like-btn" data-co
mment-id="1">
    <span class="like-icon">👍</span>
    <span class="like-count">12</span>
  </button>
  <button class="action-btn reply-btn" data-c
omment-id="1">Odpowiedz</button>
  <button class="action-btn report-btn" data-
comment-id="1">Zgłoś</button>
</div>

```

```

<!-- Formularz odpowiedzi (początkowo ukryty)
-->
<div class="reply-form-container" id="reply-f
orm-1" style="display: none;">
  <!-- Ten formularz będzie klonowany i wypet
niany dynamicznie przez JavaScript -->
</div>

```

```

<!-- Odpowiedzi na komentarz -->
<div class="replies">
  <!-- Odpowiedź na komentarz -->
  <div class="comment reply" id="comment-2">
    <div class="comment-avatar">
      
    </div>

```

```

    <div class="comment-content">
      <div class="comment-header">
        <span class="comment-author">Anna Now
ak</span>
        <span class="comment-date">1 godzinę
temu</span>
      </div>

```

```

      <div class="comment-text">
        <p>Sed do eiusmod tempor incidunt u

```

t labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation.</p>

</div>

<div class="comment-actions">

<button class="action-btn like-btn" data-comment-id="2">

👍

5

</button>

<button class="action-btn reply-btn" data-comment-id="2">Odpowiedz</button>

<button class="action-btn report-btn" data-comment-id="2">Zgłoś</button>

</div>

<!-- Formularz odpowiedzi (początkowo ukryty) -->

<div class="reply-form-container" id="reply-form-2" style="display: none;"></div>

</div>

</div>

<!-- Więcej odpowiedzi... -->

</div>

</div>

</div>

<!-- Więcej komentarzy głównych... -->

<!-- Przykład zgłoszonego komentarza -->

<div class="comment flagged" id="comment-3">

<div class="comment-avatar">

</div>

<div class="comment-content">

<div class="comment-header">

```

        <span class="comment-author">Anonimowy</span>
    </span>
    <span class="comment-date">3 godziny temu</span>
    <span class="comment-status">Komentarz zgłoszony do moderacji</span>
</div>

<div class="comment-text blurred">
    <p>Treść tego komentarza została ukryta do czasu weryfikacji przez moderatora.</p>
</div>
</div>
</div>
</div>

<!-- Paginacja komentarzy -->
<div class="comments-pagination">
    <a href="#" class="prev-page disabled">&laquo; Po
przednie</a>
    <span class="page-info">Strona 1 z 3</span>
    <a href="#" class="next-page">Następne &raquo;</a>
</div>
</section>

<!-- Modalne okno zgłaszania komentarza -->
<div class="modal report-modal" id="report-modal" style="display: none;">
    <div class="modal-content">
        <div class="modal-header">
            <h4>Zgłoś komentarz</h4>
            <button class="close-modal">&times;</button>
        </div>

        <div class="modal-body">
            <form id="report-form">
                <input type="hidden" name="comment_id" id="report_comment_id">

```



```

<div class="form-group">
  <label>Powód zgłoszenia:</label>
  <select name="reason" required>
    <option value="">Wybierz powód...</option>
    <option value="spam">Spam</option>
    <option value="offensive">Obrażliwa treść
  </option>
    <option value="harassment">Nękanie</option>
    <option value="misinformation">Nieprawdziwe informacje</option>
    <option value="other">Inny powód</option>
  </select>
</div>

<div class="form-group">
  <label>Dodatkowe informacje (opcjonalnie):</label>
  <textarea name="additional_info" rows="3">
</textarea>
</div>

<div class="form-actions">
  <button type="button" class="btn btn-cancel close-modal">Anuluj</button>
  <button type="submit" class="btn btn-submit">Zgłoś</button>
</div>
</form>
</div>
</div>
</div>

```

2. CSS dla systemu komentarzy:

```

/* Sekcja komentarzy */
.comments-section {
  margin-top: 40px;
}

```

```
border-top: 1px solid #eee;
padding-top: 30px;
}

.comments-title {
  margin-bottom: 20px;
  font-size: 1.5rem;
}

/* Formularz komentarza */
.comment-form-container {
  margin-bottom: 30px;
}

.comment-form {
  background-color: #f9f9f9;
  border-radius: 8px;
  padding: 15px;
}

.form-row {
  display: flex;
  gap: 15px;
}

.comment-avatar {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  object-fit: cover;
}

.form-fields {
  flex: 1;
}

textarea {
  width: 100%;
  padding: 12px;
```

```
border: 1px solid #ddd;
border-radius: 4px;
resize: vertical;
min-height: 80px;
font-family: inherit;
font-size: 14px;
}

.guest-fields {
  margin-top: 10px;
}

.input-group {
  display: flex;
  gap: 10px;
}

.input-group input {
  flex: 1;
  padding: 10px;
  border: 1px solid #ddd;
  border-radius: 4px;
  font-size: 14px;
}

.form-actions {
  display: flex;
  justify-content: flex-end;
  margin-top: 10px;
  gap: 10px;
}

.btn {
  padding: 8px 16px;
  border-radius: 4px;
  font-size: 14px;
  cursor: pointer;
  border: none;
  font-weight: 500;
}
```

```
}

.btn-submit {
  background-color: #4285f4;
  color: white;
}

.btn-cancel {
  background-color: #f1f1f1;
  color: #333;
}

/* Lista komentarzy */
.comments-list {
  margin-bottom: 30px;
}

.comment {
  display: flex;
  gap: 15px;
  margin-bottom: 20px;
  padding-bottom: 20px;
  border-bottom: 1px solid #f1f1f1;
}

.comment:last-child {
  border-bottom: none;
}

.comment-content {
  flex: 1;
}

.comment-header {
  margin-bottom: 8px;
}

.comment-author {
  font-weight: 600;
}
```

```
    font-size: 15px;
}

.comment-date {
    color: #777;
    font-size: 13px;
    margin-left: 10px;
}

.comment-status {
    color: #ff6b6b;
    font-size: 13px;
    margin-left: 10px;
    font-style: italic;
}

.comment-text {
    line-height: 1.5;
    color: #333;
    font-size: 14px;
}

.comment-text p {
    margin: 0 0 10px 0;
}

.comment-text p:last-child {
    margin-bottom: 0;
}

.comment-actions {
    margin-top: 10px;
    display: flex;
    gap: 15px;
}

.action-btn {
    background: none;
    border: none;
}
```

```
    color: #666;
    font-size: 13px;
    cursor: pointer;
    padding: 0;
    display: flex;
    align-items: center;
}

.action-btn:hover {
    color: #4285f4;
}

.like-btn {
    color: #666;
}

.like-btn.liked {
    color: #4285f4;
}

.like-icon {
    margin-right: 5px;
    font-size: 16px;
}

/* Odpowiedzi na komentarze */
.replies {
    margin-top: 15px;
    margin-left: 20px;
    border-left: 2px solid #f1f1f1;
    padding-left: 20px;
}

.reply {
    padding-bottom: 15px;
    margin-bottom: 15px;
}

/* Zgłoszone komentarze */
```

```
.flagged .comment-text.blurred {
    color: #999;
    font-style: italic;
}

/* Paginacja komentarzy */
.comments-pagination {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-top: 30px;
}

.prev-page,
.next-page {
    color: #4285f4;
    text-decoration: none;
}

.prev-page.disabled,
.next-page.disabled {
    color: #ccc;
    pointer-events: none;
}

.page-info {
    color: #666;
    font-size: 14px;
}

/* Modalne okno */
.modal {
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.5);
    display: flex;
```

```
    justify-content: center;
    align-items: center;
    z-index: 1000;
}

.modal-content {
    background-color: white;
    border-radius: 8px;
    width: 90%;
    max-width: 500px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.15);
}

.modal-header {
    padding: 15px 20px;
    border-bottom: 1px solid #eee;
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.modal-header h4 {
    margin: 0;
    font-size: 18px;
}

.close-modal {
    background: none;
    border: none;
    font-size: 22px;
    cursor: pointer;
    color: #999;
}

.modal-body {
    padding: 20px;
}

.form-group {
```



```

margin-bottom: 15px;
}

.form-group label {
  display: block;
  margin-bottom: 5px;
  font-weight: 500;
}

.form-group select,
.form-group textarea {
  width: 100%;
  padding: 10px;
  border: 1px solid #ddd;
  border-radius: 4px;
  font-size: 14px;
}

/* Responsywność */
@media (max-width: 576px) {
  .form-row {
    flex-direction: column;
    gap: 10px;
  }

  .comment-avatar {
    width: 30px;
    height: 30px;
  }

  .input-group {
    flex-direction: column;
  }

  .replies {
    margin-left: 10px;
    padding-left: 10px;
  }
}

```

3. JavaScript dla funkcji komentarzy:

```
document.addEventListener('DOMContentLoaded', function() {
    // Elementy DOM
    const commentForm = document.getElementById('comment-form');
    const commentsSection = document.querySelector('.comments-section');
    const replyButtons = document.querySelectorAll('.reply-btn');
    const likeButtons = document.querySelectorAll('.like-btn');
    const reportButtons = document.querySelectorAll('.report-btn');
    const reportModal = document.getElementById('report-modal');
    const reportForm = document.getElementById('report-form');
    const closeModalButtons = document.querySelectorAll('.close-modal');
    const paginationLinks = document.querySelectorAll('.comments-pagination a');

    // Stan komentarzy (do przechowywania lokalnej kopii komentarzy, jeśli to potrzebne)
    let commentsState = {
        postId: document.querySelector('input[name="postId"]').value,
        page: 1,
        perPage: 10,
        activeReplyForm: null
    };

    // Funkcja inicjalizacji formularza odpowiedzi
    function initReplyForm(commentId) {
        // Zamknięcie wcześniej otwartego formularza
        if (commentsState.activeReplyForm) {
            const oldContainer = document.getElementById(`reply-form-${commentsState.activeReplyForm}`);

```

```

    oldContainer.style.display = 'none';
    oldContainer.innerHTML = '';
}

// Otwarcie nowego formularza
const replyContainer = document.getElementById(`reply-form-${commentId}`);

// Jeśli kliknięto ten sam przycisk ponownie, zamknij formularz
if (commentsState.activeReplyForm === commentId) {
    commentsState.activeReplyForm = null;
    return;
}

// Klonowanie głównego formularza komentarzy
const originalForm = commentForm.cloneNode(true);
originalForm.id = `reply-form-${commentId}`;

// Modyfikacja formularza do odpowiedzi
const textArea = originalForm.querySelector('text area');
textArea.setAttribute('placeholder', 'Napisz odpowiedź...');
textArea.value = '';

// Ustawienie rodzica komentarza
const parentInput = originalForm.querySelector('input[name="parent_id"]');
parentInput.value = commentId;

// Dodanie przycisku anulowania
const cancelButton = originalForm.querySelector('.btn-cancel');
cancelButton.style.display = 'block';

// Zmiana tekstu przycisku
const submitButton = originalForm.querySelector('

```

```

.btn-submit');
submitButton.textContent = 'Odpowiedz';

// Dodanie nastuchiwania zdarzeń dla przycisków
submitButton.addEventListener('click', function(e
) {
    e.preventDefault();
    submitComment(originalForm);
});

cancelButton.addEventListener('click', function(e
) {
    e.preventDefault();
    replyContainer.style.display = 'none';
    commentsState.activeReplyForm = null;
});

// Wyświetlenie formularza odpowiedzi
replyContainer.innerHTML = '';
replyContainer.appendChild(originalForm);
replyContainer.style.display = 'block';

// Ustawienie focusu na textarea
textArea.focus();

// Aktualizacja stanu
commentsState.activeReplyForm = commentId;
}

// Funkcja dodawania komentarza/odpowiedzi
function submitComment(form) {
    // Pobieranie danych z formularza
    const formData = new FormData(form);

    // Walidacja
    const content = formData.get('comment_content').t
rim();
    if (!content) {
        showError(form, 'Komentarz nie może być pusty.'

```

```

    );
    return;
}

// Początek obsługi spinera ładowania
const submitButton = form.querySelector('.btn-submit');
const originalText = submitButton.textContent;
submitButton.disabled = true;
submitButton.textContent = 'Wysyłanie...';

// Wysłanie komentarza (ajax)
fetch('add-comment.php', {
    method: 'POST',
    body: formData
})
.then(response => {
    if (!response.ok) {
        throw new Error('Wystąpił błąd podczas dodawania komentarza.');
    }
    return response.json();
})
.then(data => {
    if (data.success) {
        // Wyczyszczenie formularza
        form.reset();

        // Jeśli to odpowiedź, ukrycie formularza odpowiedzi
        if (form.id !== 'comment-form') {
            const replyContainer = form.parentElement;
            replyContainer.style.display = 'none';
            commentsState.activeReplyForm = null;
        }

        // Odświeżenie listy komentarzy lub dodanie nowego komentarza do DOM
        if (data.html) {

```

```

        // Opcja 1: Serwer zwraca gotowy HTML komen
        tarza
        addCommentToDOM(data.html, formData.get('pa
        rent_id'));
    } else {
        // Opcja 2: Odświeżenie całej listy komenta
        rzy
        loadComments();
    }

    // Pokazanie komunikatu sukcesu
    showSuccess('Komentarz został dodany pomyślni
    e!');
    } else {
        // Pokazanie błędu
        showError(form, data.message || 'Nie udało si
        ę dodać komentarza.');
```

```

    }
  })
  .catch(error => {
    console.error('Error:', error);
    showError(form, error.message);
  })
  .finally(() => {
    // Przywrócenie przycisku
    submitButton.disabled = false;
    submitButton.textContent = originalText;
  });
}

```

```

// Funkcja dodająca nowy komentarz do DOM
function addCommentToDOM(html, parentId) {
  if (parentId && parentId !== '0') {
    // Dodanie jako odpowiedź
    const parentComment = document.getElementById(`
    comment-${parentId}`);
    let replies = parentComment.querySelector('.rep
    lies');
```

```

    // Jeśli nie ma jeszcze kontenera na odpowiedzi
    , utwórz go
    if (!replies) {
        replies = document.createElement('div');
        replies.className = 'replies';
        parentComment.querySelector('.comment-content')
            .appendChild(replies);
    }

    // Dodanie odpowiedzi na początek
    replies.insertAdjacentHTML('afterbegin', html);
} else {
    // Dodanie jako główny komentarz
    const commentsList = document.querySelector('.comments-list');
    commentsList.insertAdjacentHTML('afterbegin', html);
}

// Aktualizacja licznika komentarzy
updateCommentCount(1);

// Dodanie następowania zdarzeń dla nowego komentarza
const newComment = document.querySelector('.comment:first-child');
initializeCommentActions(newComment);
}

// Funkcja aktualizująca liczbę komentarzy
function updateCommentCount(change) {
    const title = document.querySelector('.comments-title');
    const countPattern = /\((\d+)\)/;
    const match = title.textContent.match(countPattern);

    if (match) {
        const currentCount = parseInt(match[1]);
    }
}

```

```

        const newCount = currentCount + change;
        title.textContent = title.textContent.replace(c
ountPattern, `${newCount}`);
    }
}

// Funkcja obsługująca polubienie komentarza
function likeComment(button) {
    const commentId = button.getAttribute('data-commen
t-id');
    const likeCount = button.querySelector('.like-cou
nt');
    const currentLikes = parseInt(likeCount.textConte
nt);

    // Sprawdzenie, czy komentarz jest już polubiony
    const isLiked = button.classList.contains('liked'
);

    // Żądanie do serwera
    fetch('like-comment.php', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/x-www-form-urle
n
coded',
        },
        body: `comment_id=${commentId}&action=${isLiked
? 'unlike' : 'like'}`
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            // Aktualizacja UI
            if (isLiked) {
                button.classList.remove('liked');
                likeCount.textContent = currentLikes - 1;
            } else {
                button.classList.add('liked');
                likeCount.textContent = currentLikes + 1;
            }
        }
    })
}

```



```

    }
  } else {
    showError(null, data.message || 'Nie udało si
ę zmienić polubienia.');
```

}

```

  })
  .catch(error => {
    console.error('Error:', error);
    showError(null, 'Wystąpił błąd podczas zmiany p
olubienia.');
```

});
 }
}

```

// Funkcja otwierająca modal zgłoszenia komentarza
function openReportModal(commentId) {
  const reportCommentIdInput = document.getElementB
yId('report_comment_id');
  reportCommentIdInput.value = commentId;

  // Wyświetlenie modalu
  reportModal.style.display = 'flex';
}

// Funkcja zamykająca modal
function closeModal() {
  reportModal.style.display = 'none';
  reportForm.reset();
}

// Funkcja zgłaszająca komentarz
function reportComment(form) {
  const formData = new FormData(form);

  // Walidacja
  const reason = formData.get('reason');
  if (!reason) {
    showError(form, 'Wybierz powód zgłoszenia.');
```

return;
 }
}

```

// Dezaktywacja przycisku
const submitButton = form.querySelector('.btn-submit');
const originalText = submitButton.textContent;
submitButton.disabled = true;
submitButton.textContent = 'Wysyłanie...';

// Wysłanie zgłoszenia
fetch('report-comment.php', {
  method: 'POST',
  body: formData
})
.then(response => response.json())
.then(data => {
  if (data.success) {
    // Zamknięcie modalu
    closeModal();

    // Pokazanie komunikatu sukcesu
    showSuccess('Komentarz został zgłoszony do moderacji.');
```

// Opcjonalnie: oznaczenie komentarza jako zgłoszonego w UI

```

    const commentId = formData.get('comment_id');
    const reportedComment = document.getElementById(`comment-${commentId}`);

    if (reportedComment && data.hide_comment) {
      reportedComment.classList.add('flagged');
      const commentText = reportedComment.querySelector('.comment-text');
      commentText.classList.add('blurred');
      commentText.innerHTML = '<p>Treść tego komentarza została ukryta do czasu weryfikacji przez moderatora.</p>';

      // Ukrycie przycisków akcji

```

```

        const actions = reportedComment.querySelector(
or('.comment-actions');
        if (actions) actions.style.display = 'none'
;

        // Dodanie statusu
        const header = reportedComment.querySelector(
r('.comment-header');
        const status = document.createElement('span
');

        status.className = 'comment-status';
        status.textContent = 'Komentarz zgłoszony d
o moderacji';
        header.appendChild(status);
    }
    } else {
        showError(form, data.message || 'Nie udało si
ę zgłosić komentarza.');
```

}

```

    })
    .catch(error => {
        console.error('Error:', error);
        showError(form, 'Wystąpił błąd podczas zgłaszan
ia komentarza.');
```

}

```

    .finally(() => {
        // Przywrócenie przycisku
        submitButton.disabled = false;
        submitButton.textContent = originalText;
    });
}

// Funkcja ładująca komentarze (paginacja)
function loadComments(page = 1) {
    const postId = commentsState.postId;

    // Pokazanie Loadera
    const commentsSection = document.querySelector('.
comments-section');
```

```

commentsSection.classList.add('loading');

// Pobranie komentarzy z serwera
fetch(`get-comments.php?post_id=${postId}&page=${
page}`)
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            // Aktualizacja listy komentarzy
            const commentsList = document.querySelector(
                '.comments-list');
            commentsList.innerHTML = data.html;

            // Aktualizacja paginacji
            updatePagination(data.pagination);

            // Aktualizacja liczby komentarzy
            const title = document.querySelector('.comm
ents-title');
            title.textContent = `Komentarze (${data.tot
al_comments})`;

            // Inicjalizacja akcji dla nowych komentarz
y
            initializeAllComments();

            // Aktualizacja stanu
            commentsState.page = page;
        } else {
            showError(null, data.message || 'Nie udało
            się załadować komentarzy.');
```

```

        // Ukrycie loadera
        commentsSection.classList.remove('loading');
    });
}

// Funkcja aktualizująca paginację
function updatePagination(pagination) {
    const paginationContainer = document.querySelector(
        '.comments-pagination');

    if (pagination.total_pages <= 1) {
        paginationContainer.style.display = 'none';
        return;
    }

    const prevPage = paginationContainer.querySelector(
        '.prev-page');
    const nextPage = paginationContainer.querySelector(
        '.next-page');
    const pageInfo = paginationContainer.querySelector(
        '.page-info');

    // Aktualizacja informacji o stronie
    pageInfo.textContent = `Strona ${pagination.current_page} z ${pagination.total_pages}`;

    // Aktualizacja przycisków poprzednia/następna
    if (pagination.current_page <= 1) {
        prevPage.classList.add('disabled');
    } else {
        prevPage.classList.remove('disabled');
        prevPage.setAttribute('data-page', pagination.c
            urrent_page - 1);
    }

    if (pagination.current_page >= pagination.total_p
        ages) {
        nextPage.classList.add('disabled');
    } else {

```

```

    nextPage.classList.remove('disabled');
    nextPage.setAttribute('data-page', pagination.c
urrent_page + 1);
}

paginationContainer.style.display = 'flex';
}

// Funkcja pokazująca komunikat błędu
function showError(form, message) {
    // Jeśli formularz jest określony, pokaż błąd prz
y formularzu
    if (form) {
        // Usunięcie wcześniejszych komunikatów
        const existingError = form.querySelector('.erro
r-message');
        if (existingError) {
            existingError.remove();
        }

        // Dodanie nowego komunikatu
        const errorDiv = document.createElement('div');
        errorDiv.className = 'error-message';
        errorDiv.textContent = message;

        const formActions = form.querySelector('.form-a
ctions');
        form.insertBefore(errorDiv, formActions);

        // Automatyczne usunięcie po czasie
        setTimeout(() => {
            errorDiv.remove();
        }, 5000);
    } else {
        // Pokaż ogólny komunikat
        const notification = document.createElement('di
v');
        notification.className = 'notification error';
        notification.textContent = message;
    }
}

```

```

document.body.appendChild(notification);

// Automatyczne usunięcie po czasie
setTimeout(() => {
    notification.classList.add('fadeout');

    setTimeout(() => {
        notification.remove();
    }, 500);
}, 5000);
}
}

// Funkcja pokazująca komunikat sukcesu
function showSuccess(message) {
    const notification = document.createElement('div'
);
    notification.className = 'notification success';
    notification.textContent = message;

    document.body.appendChild(notification);

    // Automatyczne usunięcie po czasie
    setTimeout(() => {
        notification.classList.add('fadeout');

        setTimeout(() => {
            notification.remove();
        }, 500);
    }, 5000);
}

// Funkcja inicjalizująca akcje dla komentarza
function initializeCommentActions(comment) {
    // Przyciski odpowiedzi
    const replyBtn = comment.querySelector('.reply-bt
n');
    if (replyBtn) {

```

```

        replyBtn.addEventListener('click', function() {
            const commentId = this.getAttribute('data-comment-id');
            initReplyForm(commentId);
        });
    }

    // Przyciski polubienia
    const likeBtn = comment.querySelector('.like-btn');
    if (likeBtn) {
        likeBtn.addEventListener('click', function() {
            likeComment(this);
        });
    }

    // Przyciski zgłoszenia
    const reportBtn = comment.querySelector('.report-btn');
    if (reportBtn) {
        reportBtn.addEventListener('click', function() {
            const commentId = this.getAttribute('data-comment-id');
            openReportModal(commentId);
        });
    }

    // Funkcja inicjalizująca wszystkie komentarze
    function initializeAllComments() {
        const comments = document.querySelectorAll('.comment');
        comments.forEach(comment => {
            initializeCommentActions(comment);
        });
    }

    // Nasłuchiwanie zdarzeń

```



```

// Główny formularz komentarzy
if (commentForm) {
  commentForm.addEventListener('submit', function
(e) {
    e.preventDefault();
    submitComment(this);
  });
}

// Przyciski odpowiedzi
replyButtons.forEach(button => {
  button.addEventListener('click', function() {
    const commentId = this.getAttribute('data-com
ment-id');
    initReplyForm(commentId);
  });
});

// Przyciski polubienia
likeButtons.forEach(button => {
  button.addEventListener('click', function() {
    likeComment(this);
  });
});

// Przyciski zgłoszenia
reportButtons.forEach(button => {
  button.addEventListener('click', function() {
    const commentId = this.getAttribute('data-com
ment-id');
    openReportModal(commentId);
  });
});

// Formularz zgłoszenia
if (reportForm) {
  reportForm.addEventListener('submit', function(
e) {
    e.preventDefault();

```

```

        reportComment(this);
    });
}

// Przyciski zamykania modalu
closeModalButtons.forEach(button => {
    button.addEventListener('click', closeModal);
});

// Zamykanie modalu przy kliknięciu poza jego zaw
artościq
reportModal.addEventListener('click', function(e)
{
    if (e.target === this) {
        closeModal();
    }
});

// Paginacja
paginationLinks.forEach(link => {
    link.addEventListener('click', function(e) {
        if (this.classList.contains('disabled')) retu
rn;

        e.preventDefault();
        const page = parseInt(this.getAttribute('data
-page')) || 1;
        loadComments(page);

        // Przewinięcie do sekcji komentarzy
commentsSection.scrollIntoView({ behavior: 's
mooth' });
    });
});

// Inicjalizacja wszystkich komentarzy
initializeAllComments();

// CSS dla powiadomień

```

```
const style = document.createElement('style');
style.textContent = `
.notification {
  position: fixed;
  top: 20px;
  right: 20px;
  padding: 15px 20px;
  border-radius: 4px;
  font-size: 14px;
  z-index: 1000;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
  transition: opacity 0.3s, transform 0.3s;
}

.notification.success {
  background-color: #4CAF50;
  color: white;
}

.notification.error {
  background-color: #f44336;
  color: white;
}

.notification.fadeout {
  opacity: 0;
  transform: translateY(-10px);
}

.error-message {
  color: #f44336;
  font-size: 14px;
  margin-top: 10px;
}

.comments-section.loading::before {
  content: '';
  position: absolute;
  top: 0;
}
```

```

    left: 0;
    right: 0;
    bottom: 0;
    background-color: rgba(255, 255, 255, 0.7);
    z-index: 10;
}

.comments-section.loading::after {
    content: '';
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    width: 40px;
    height: 40px;
    border: 4px solid #f3f3f3;
    border-top: 4px solid #3498db;
    border-radius: 50%;
    animation: spin 1s linear infinite;
    z-index: 11;
}

@keyframes spin {
    0% { transform: translate(-50%, -50%) rotate(
0deg); }
    100% { transform: translate(-50%, -50%) rotat
e(360deg); }
}
`;

document.head.appendChild(style);
});

```

4. Backend obsługi komentarzy (PHP):

```

<?php
// Plik: add-comment.php

// Połączenie z bazą danych
require_once 'config.php';

```

```
require_once 'functions.php';
```

```
// Sprawdzenie, czy żądanie jest typu POST
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    echo json_encode([
        'success' => false,
        'message' => 'Nieprawidłowe żądanie.'
    ]);
    exit;
}
```

```
// Podstawowa walidacja
```

```
$post_id = isset($_POST['post_id']) ? intval($_POST['post_id']) : 0;
$parent_id = isset($_POST['parent_id']) ? intval($_POST['parent_id']) : 0;
$content = isset($_POST['comment_content']) ? trim($_POST['comment_content']) : '';
```

```
if (empty($content)) {
    echo json_encode([
        'success' => false,
        'message' => 'Treść komentarza nie może być pusta.'
    ]);
    exit;
}
```

```
if (empty($post_id)) {
    echo json_encode([
        'success' => false,
        'message' => 'Nieprawidłowy identyfikator posta.'
    ]);
    exit;
}
```

```
// Sprawdzenie, czy post istnieje
```

```
$post_query = "SELECT id FROM posts WHERE id = ?";
```

```

$stmt = $db->prepare($post_query);
$stmt->bindParam(1, $post_id);
$stmt->execute();

if ($stmt->rowCount() === 0) {
    echo json_encode([
        'success' => false,
        'message' => 'Post nie istnieje.'
    ]);
    exit;
}

// Sprawdzenie, czy komentarz nadrzędny istnieje (jeś
// li to odpowiedź)
if ($parent_id > 0) {
    $parent_query = "SELECT id FROM comments WHERE id
    = ? AND post_id = ?";
    $stmt = $db->prepare($parent_query);
    $stmt->bindParam(1, $parent_id);
    $stmt->bindParam(2, $post_id);
    $stmt->execute();

    if ($stmt->rowCount() === 0) {
        echo json_encode([
            'success' => false,
            'message' => 'Komentarz nadrzędny nie ist
nieje.'
        ]);
        exit;
    }
}

// Sprawdzenie, czy użytkownik jest zalogowany
$user_id = null;
$author_name = null;
$author_email = null;

if (isset($_SESSION['user_id'])) {
    $user_id = $_SESSION['user_id'];
}

```

```

// Pobranie danych użytkownika
$user_query = "SELECT username, email FROM users
WHERE id = ?";
$stmt = $db->prepare($user_query);
$stmt->bindParam(1, $user_id);
$stmt->execute();

$user_data = $stmt->fetch(PDO::FETCH_ASSOC);

if ($user_data) {
    $author_name = $user_data['username'];
    $author_email = $user_data['email'];
} else {
    // Nieprawidłowy ID użytkownika
    echo json_encode([
        'success' => false,
        'message' => 'Nieprawidłowe dane użytkown
ika.'
    ]);
    exit;
}
} else {
    // Użytkownik niezalogowany - sprawdzenie wymaga
nych pól
    $author_name = isset($_POST['author_name']) ? trim($_POST['author_name']) : '';
    $author_email = isset($_POST['author_email']) ? trim($_POST['author_email']) : '';

    if (empty($author_name) || empty($author_email))
    {
        echo json_encode([
            'success' => false,
            'message' => 'Imię i email są wymagane dla niezalogowanych użytkowników.'
        ]);
        exit;
    }
}

```

```

// Walidacja email
if (!filter_var($author_email, FILTER_VALIDATE_EMAIL)) {
    echo json_encode([
        'success' => false,
        'message' => 'Podaj prawidłowy adres email.'
    ]);
    exit;
}

// Sprawdzenie, czy komentarz nie jest spamem
$is_spam = checkForSpam($content, $author_name, $author_email);

// Zapisanie komentarza w bazie danych
try {
    $query = "INSERT INTO comments (post_id, parent_id, user_id, author_name, author_email, content, status, ip_address, created_at)
              VALUES (?, ?, ?, ?, ?, ?, ?, ?, NOW())"
;

    $status = $is_spam ? 'pending' : 'approved';
    $ip_address = $_SERVER['REMOTE_ADDR'];

    $stmt = $db->prepare($query);
    $stmt->bindParam(1, $post_id);
    $stmt->bindParam(2, $parent_id);
    $stmt->bindParam(3, $user_id);
    $stmt->bindParam(4, $author_name);
    $stmt->bindParam(5, $author_email);
    $stmt->bindParam(6, $content);
    $stmt->bindParam(7, $status);
    $stmt->bindParam(8, $ip_address);

```



```

$stmt->execute();

$comment_id = $db->lastInsertId();

// Aktualizacja licznika komentarzy dla posta
if ($status === 'approved') {
    $update_query = "UPDATE posts SET comment_cou
nt = comment_count + 1 WHERE id = ?";
    $stmt = $db->prepare($update_query);
    $stmt->bindParam(1, $post_id);
    $stmt->execute();
}

// Generowanie HTML dla nowego komentarza (jeśli
nie jest spamem)
$html = '';

if ($status === 'approved') {
    $html = generateCommentHtml([
        'id' => $comment_id,
        'author_name' => $author_name,
        'content' => $content,
        'created_at' => date('Y-m-d H:i:s'),
        'parent_id' => $parent_id,
        'user_id' => $user_id
    ]);

    // Wysłanie powiadomienia o nowym komentarzu
    if ($parent_id > 0) {
        sendCommentNotification($parent_id, $comm
ent_id);
    }
}

echo json_encode([
    'success' => true,
    'message' => $status === 'approved' ? 'Koment
arz został dodany.' : 'Komentarz został dodany i ocze
kuje na zatwierdzenie.',

```

```

        'comment_id' => $comment_id,
        'status' => $status,
        'html' => $html
    ]);
} catch (PDOException $e) {
    error_log("Error adding comment: " . $e->getMessage());

    echo json_encode([
        'success' => false,
        'message' => 'Wystąpił błąd podczas dodawania
komentarza. Spróbuj ponownie później.'
    ]);
}

// Funkcja sprawdzająca, czy komentarz jest spamem
function checkForSpam($content, $author_name, $author_email) {
    // Podstawowe reguły dla spamu
    $spam_words = ['viagra', 'cialis', 'casino', 'poker', 'loan', 'free money', 'guaranteed'];

    // Sprawdzenie treści
    foreach ($spam_words as $word) {
        if (stripos($content, $word) !== false) {
            return true;
        }
    }

    // Sprawdzenie ilości linków (zbyt wiele linków może wskazywać na spam)
    $url_count = preg_match_all('/(https?:\/\/\/[^\s]+)\/', $content, $matches);
    if ($url_count > 3) {
        return true;
    }

    // Sprawdzenie znanych domen spammerskich
    $spam_domains = ['spam.com', 'scam.ru', 'sketchy.

```

```

net'];
$email_domain = substr(strrchr($author_email, "@"
), 1);

    if (in_array($email_domain, $spam_domains)) {
        return true;
    }

    // Opcjonalnie: integracja z usługami antyspamowy
    mi jak Akismet
    // if (checkWithAkismet($content, $author_name, $
    author_email)) {
        // return true;
        // }

    return false;
}

// Funkcja generująca HTML komentarza
function generateCommentHtml($comment) {
    $avatar_url = $comment['user_id'] ? "avatars/user
{$comment['user_id']}.jpg" : "avatars/default.png";
    $is_reply = $comment['parent_id'] > 0 ? ' reply'
: '';
    $time_ago = timeAgo($comment['created_at']);

    $html = <<<HTML
    <div class="comment{$is_reply}" id="comment-{$com
ment['id']}">
        <div class="comment-avatar">
            
        </div>

        <div class="comment-content">
            <div class="comment-header">
                <span class="comment-author">{$comment['aut
hor_name']}</span>
                <span class="comment-date">{$time_ago}</spa

```

n>

```
</div>

<div class="comment-text">
  <p>{$comment['content']}</p>
</div>

<div class="comment-actions">
  <button class="action-btn like-btn" data-comment-id="{ $comment['id'] }">
    <span class="like-icon">👍</span>
    <span class="like-count">0</span>
  </button>
  <button class="action-btn reply-btn" data-comment-id="{ $comment['id'] }">Odpowiedz</button>
  <button class="action-btn report-btn" data-comment-id="{ $comment['id'] }">Zgłoś</button>
</div>

<div class="reply-form-container" id="reply-form-{$comment['id']}" style="display: none;"></div>
HTML;
```

```
// Jeśli jest to komentarz główny, dodaj kontener na odpowiedzi
if ($comment['parent_id'] == 0) {
  $html .= '<div class="replies"></div>';
}

$html .= '</div></div>';

return $html;
}
```

```
// Funkcja wysyłająca powiadomienie o odpowiedzi
function sendCommentNotification($parent_id, $new_comment_id) {
  global $db;
```

```

// Pobieranie danych komentarza nadrzędnego
$query = "SELECT c.author_email, c.author_name, c
.user_id, p.title, p.slug
        FROM comments c
        JOIN posts p ON c.post_id = p.id
        WHERE c.id = ?";

$stmt = $db->prepare($query);
$stmt->bindParam(1, $parent_id);
$stmt->execute();

$parent_comment = $stmt->fetch(PDO::FETCH_ASSOC);

if (!$parent_comment) {
    return;
}

// Pobieranie danych nowego komentarza
$query = "SELECT author_name, content FROM commen
ts WHERE id = ?";
$stmt = $db->prepare($query);
$stmt->bindParam(1, $new_comment_id);
$stmt->execute();

$new_comment = $stmt->fetch(PDO::FETCH_ASSOC);

if (!$new_comment) {
    return;
}

// Generowanie treści emaila
$subject = "Nowa odpowiedź na Twój komentarz";

$post_url = "https://example.com/{ $parent_comment
['slug'] }#comment-{ $new_comment_id }";

$message = "Witaj { $parent_comment['author_name']
},\n\n";
$message .= "{ $new_comment['author_name'] } odpowi

```

```

edział na Twój komentarz w artykule \"{$parent_commen
t['title']}\".\n\n";
$message .= "Treść odpowiedzi:\n";
$message .= "\"{$new_comment['content']}\":\n\n";
$message .= "Aby zobaczyć wszystkie komentarze i
odpowiedzieć, kliknij poniższy link:\n";
$message .= $post_url . "\n\n";
$message .= "Pozdrawiamy,\nZespół Example.com";

$headers = "From: noreply@example.com\r\n";
$headers .= "Reply-To: noreply@example.com\r\n";

// Wysłanie emaila
mail($parent_comment['author_email'], $subject, $
message, $headers);

// Jeśli użytkownik jest zalogowany, można też do
dać powiadomienie w systemie
if ($parent_comment['user_id']) {
    $query = "INSERT INTO notifications (user_id,
type, related_id, content, created_at)
VALUES (?, 'comment_reply', ?, ?, N
OW())";

    $content = "{$new_comment['author_name']} odp
owiedział na Twój komentarz w artykule \"{$parent_com
ment['title']}\":\".";

    $stmt = $db->prepare($query);
    $stmt->bindParam(1, $parent_comment['user_id'
]);
    $stmt->bindParam(2, $new_comment_id);
    $stmt->bindParam(3, $content);
    $stmt->execute();
}

// Funkcja zwracająca czas od komentarza w przyjaznym
formacie

```

```

function timeAgo($datetime) {
    $time = strtotime($datetime);
    $now = time();
    $diff = $now - $time;

    if ($diff < 60) {
        return "przed chwilą";
    } elseif ($diff < 3600) {
        $mins = floor($diff / 60);
        return $mins . " " . ngettext("minutę", "minut", $mins) . " temu";
    } elseif ($diff < 86400) {
        $hours = floor($diff / 3600);
        return $hours . " " . ngettext("godzinę", "godzin", $hours) . " temu";
    } elseif ($diff < 604800) {
        $days = floor($diff / 86400);
        return $days . " " . ngettext("dzień", "dni", $days) . " temu";
    } elseif ($diff < 2592000) {
        $weeks = floor($diff / 604800);
        return $weeks . " " . ngettext("tydzień", "tygodni", $weeks) . " temu";
    } else {
        return date("d.m.Y", $time);
    }
}

```

// Funkcja wspomagająca wybór odpowiedniej formy

```

function ngettext($singular, $plural, $count) {
    if ($count == 1) {
        return $singular;
    } else if ($count % 10 >= 2 && $count % 10 <= 4 &
    & ($count % 100 < 10 || $count % 100 >= 20)) {
        return substr($plural, 0, -1) . 'y'; // dla 2
        , 3, 4, 22, 23, 24, itd.
    } else {
        return $plural;
    }
}

```

```
}  
?>
```

5. Struktura bazy danych dla komentarzy:

-- Tabela komentarzy

```
CREATE TABLE comments (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  post_id INT NOT NULL,  
  parent_id INT DEFAULT 0,  
  user_id INT,  
  author_name VARCHAR(100) NOT NULL,  
  author_email VARCHAR(100) NOT NULL,  
  author_website VARCHAR(100),  
  content TEXT NOT NULL,  
  status ENUM('pending', 'approved', 'spam', 'trash')  
  DEFAULT 'pending',  
  ip_address VARCHAR(45),  
  agent VARCHAR(255),  
  likes INT DEFAULT 0,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U  
PDATE CURRENT_TIMESTAMP,  
  FOREIGN KEY (post_id) REFERENCES posts(id) ON DELET  
E CASCADE,  
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELET  
E SET NULL,  
  INDEX (post_id),  
  INDEX (parent_id),  
  INDEX (status)  
);
```

-- Tabela polubień komentarzy

```
CREATE TABLE comment_likes (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  comment_id INT NOT NULL,  
  user_id INT,  
  ip_address VARCHAR(45) NOT NULL,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  FOREIGN KEY (comment_id) REFERENCES comments(id) ON
```



```
DELETE CASCADE,  
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE,  
  UNIQUE KEY (comment_id, user_id),  
  UNIQUE KEY (comment_id, ip_address, user_id),  
  INDEX (comment_id)  
);
```

-- Tabela zgłoszeń komentarzy

```
CREATE TABLE comment_reports (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  comment_id INT NOT NULL,  
  user_id INT,  
  reason ENUM('spam', 'offensive', 'harassment', 'misinformation', 'other') NOT NULL,  
  additional_info TEXT,  
  status ENUM('pending', 'reviewed', 'rejected') DEFAULT 'pending',  
  ip_address VARCHAR(45),  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
  FOREIGN KEY (comment_id) REFERENCES comments(id) ON DELETE CASCADE,  
  FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE SET NULL,  
  INDEX (comment_id),  
  INDEX (status)  
);
```

-- Tabela powiadomień

```
CREATE TABLE notifications (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  user_id INT NOT NULL,  
  type ENUM('comment_reply', 'comment_like', 'comment_approved', 'mention') NOT NULL,  
  related_id INT,  
  content TEXT NOT NULL,  
  is_read BOOLEAN DEFAULT FALSE,
```

```
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE,  
INDEX (user_id),  
INDEX (is_read)  
);
```

Panel administracyjny dla systemu komentarzy:

System komentarzy powinien mieć również panel administracyjny, umożliwiający moderatorom zarządzanie komentarzami:

1. **Lista komentarzy** - z możliwością filtrowania po statusie, dacie, postach
2. **Widok szczegółowy komentarza** - z informacjami o autorze, IP, treści
3. **Akcje na komentarzach:**
 - Zatwierdzanie komentarzy
 - Oznaczanie jako spam
 - Usuwanie komentarzy
 - Edycja treści
 - Blokowanie użytkowników
4. **Ustawienia komentarzy:**
 - Włączanie/wyłączanie komentarzy
 - Automatyczne zatwierdzanie komentarzy (dla zalogowanych użytkowników)
 - Moderacja słów kluczowych (automatyczne filtrowanie)
 - Ustawienia powiadomień

Integracja z zewnętrznymi systemami komentarzy:

Alternatywą dla własnego systemu komentarzy jest integracja z zewnętrznymi usługami, takimi jak:

1. **Disqus** - popularna platforma komentarzy z wieloma funkcjami:

```
<div id="disqus_thread"></div>
<script>
  var disqus_config = function () {
    this.page.url = PAGE_URL; // Replace PAGE_URL
    with your page's canonical URL
    this.page.identifier = PAGE_IDENTIFIER; // Replace PAGE_IDENTIFIER with your page's unique identifier
  };

  (function() {
    var d = document, s = d.createElement('script');
    s.src = 'https://EXAMPLE.disqus.com/embed.js';
    s.setAttribute('data-timestamp', +new Date());
    (d.head || d.body).appendChild(s);
  })();
</script>
```

2. **Facebook Comments** - system komentarzy zintegrowany z Facebookiem:

```
<div id="fb-root"></div>
<script async defer crossorigin="anonymous" src="https://connect.facebook.net/pl_PL/sdk.js#xfbml=1&version=v10.0&appId=YOUR_APP_ID"></script>

<div class="fb-comments" data-href="PAGE_URL" data-width="100%" data-numposts="10"></div>
```

3. **Commento** - lekka i zorientowana na prywatność platforma komentarzy:

```
<div id="commento"></div>
<script src="https://cdn.commento.io/js/commento.js">
</script>
```

Dobre praktyki dla systemów komentarzy: - Implementuj mechanizmy antyspamowe (CAPTCHA, filtrowanie treści) - Umożliwiaj moderację i zgłaszanie nie stosownych treści - Zawsze wymagaj potwierdzenia usuwania komentarzy - Zachowaj responsywność interfejsu komentarzy na urządzeniach mobilnych - Dbaj o dostępność (używaj odpowiednich etykiet, wsparcie dla klawiatury) - Rozważ implementację markdown lub prostego formatowania tekstu - Implementuj powiadomienia o odpowiedziach na komentarze - Stosuj przyjazne formaty czasu (np. "2 godziny temu" zamiast pełnej daty) - Umożliw sortowanie komentarzy (najnowsze, najstarsze, najpopularniejsze) - Ogranicz zagnieżdżenie odpowiedzi (max 3-4 poziomy) dla czytelności

Dlaczego są ważne dla początkujących: - System komentarzy zwiększa interaktywność strony - Buduje społeczność wokół treści - Dostarcza dodatkowej wartości poprzez wkład użytkowników - Implementacja komentarzy łączy wiele aspektów programowania (frontend, backend, bazy danych, AJAX) - Jest dobrym projektem do nauki zarządzania stanem, interakcji użytkownika i komunikacji klient-serwer

9. Aspekty prawne i formalne

Polityka prywatności Polityka prywatności to dokument informujący użytkowników o gromadzeniu, przetwarzaniu i ochronie ich danych osobowych na stronie internetowej, będący prawnym wymogiem w większości krajów.

Kluczowe cechy: - Informuje o zbieranych danych osobowych i celu ich przetwarzania - Opisuje wykorzystanie plików cookie i innych technologii śledzących - Informuje o prawach użytkowników dotyczących ich danych - Wyjaśnia procedury bezpieczeństwa i ochrony danych - Określa okres przechowywania danych

Elementy dobrej polityki prywatności:

1. Wprowadzenie i informacje o administratorze danych:

Polityka Prywatności

1. Wprowadzenie

Niniejsza Polityka Prywatności określa zasady przetwarzania i ochrony danych osobowych, które są przekazywane przez Użytkowników podczas korzystania ze strony internetowej [nazwa strony] dostępnej pod adresem [adres strony].

Administratorem danych osobowych jest [nazwa firmy/osoby] z siedzibą w [adres], NIP: [NIP], REGON: [REGON], wpisana do rejestru przedsiębiorców Krajowego Rejestru Sądowego prowadzonego przez Sąd Rejonowy dla [miasto], [nr KRS] ("Administrator").

Kontakt z Administratorem jest możliwy pod adresem e-mail: [adres e-mail] lub pisemnie na adres siedziby Administratora.

2. Zakres zbieranych danych:

2. Jakie dane zbieramy?

2.1 Dane podawane dobrowolnie

Podczas korzystania z naszej strony internetowej może sz zostać poproszony o podanie niektórych danych osobowych, w szczególności poprzez wypełnienie formularzy dostępnych na stronie, w tym:

- Formularza kontaktowego (imię, adres e-mail);
- Formularza rejestracji konta (imię i nazwisko, adres e-mail, hasło);
- Formularza zamówienia (imię i nazwisko, adres dostawy, adres e-mail, numer telefonu, dane płatności).

2.2 Dane zbierane automatycznie

Podczas korzystania z naszej strony internetowej, aut

omatycznie zbieramy niektóre dane, w tym:

- Adres IP;
- Informacje o urządzeniu i przeglądarce;
- Informacje o sposobie korzystania ze strony (odwiedzone podstrony, czas spędzony na stronie);
- Informacje o lokalizacji;
- Inne informacje zbierane przez pliki cookies i podobne technologie.

3. Cel i podstawa prawna przetwarzania danych:

3. W jakim celu i na jakiej podstawie przetwarzamy dane?

Twoje dane osobowe będą przetwarzane w następujących celach:

3.1 Realizacja umowy lub podjęcie działań przed zawarciem umowy (art. 6 ust. 1 lit. b RODO)

- Obsługa Twojego konta w serwisie;
- Realizacja zamówienia i świadczenie usług;
- Obsługa zapytań przez formularz kontaktowy.

3.2 Wypełnienie obowiązków prawnych (art. 6 ust. 1 lit. c RODO)

- Wystawianie i przechowywanie faktur oraz dokumentów księgowych;
- Rozpatrywanie reklamacji;
- Realizacja obowiązków wynikających z przepisów podatkowych i rachunkowych.

3.3 Realizacja prawnie uzasadnionych interesów Administratora (art. 6 ust. 1 lit. f RODO)

- Marketing własnych produktów i usług;
- Analityka internetowa i statystyki;
- Zapewnienie bezpieczeństwa IT;
- Ustalenie, dochodzenie i obrona roszczeń;
- Dostosowanie strony do potrzeb użytkowników.

3.4 Zgoda (art. 6 ust. 1 lit. a RODO)

- Wysyłka newslettera i komunikacji marketingowej;
- Wykorzystanie plików cookies i podobnych technologii;
- Inne cele marketingowe, na które wyrazisz zgodę.

4. Przekazywanie danych stronom trzecim:

4. Komu udostępniamy Twoje dane?

Twoje dane osobowe możemy udostępniać następującym kategoriom odbiorców:

4.1 Podmioty przetwarzające dane w naszym imieniu

- Dostawcy usług hostingowych i IT;
- Dostawcy usług księgowych i płatności elektronicznych;
- Dostawcy usług kurierskich i pocztowych;
- Dostawcy systemów analitycznych i marketingowych;
- Dostawcy usług wspierających obsługę klienta.

4.2 Inni administratorzy danych

- Partnerzy biznesowi – w zakresie niezbędnym do realizacji usług;
- Portale społecznościowe (np. Facebook, Instagram) – w przypadku korzystania z wtyczek społecznościowych;
- Organy publiczne – w przypadkach określonych przez przepisy prawa.

4.3 Przekazywanie danych poza EOG

Niektóre z podmiotów, którym przekazujemy dane, mogą mieć siedziby poza Europejskim Obszarem Gospodarczym (EOG). Przekazujemy dane poza EOG tylko wtedy, gdy jest to konieczne, i z zapewnieniem odpowiedniego stopnia ochrony, w szczególności poprzez:

- Przekazanie danych do państw, co do których Komisja Europejska wydała decyzję stwierdzającą odpowiedni stopień ochrony;
- Stosowanie standardowych klauzul umownych zatwierdzonych przez Komisję Europejską.

onych przez Komisję Europejską;
- Stosowanie wiążących reguł korporacyjnych.

5. Prawa użytkowników:

5. Jakie masz prawa?

W związku z przetwarzaniem Twoich danych osobowych, przysługują Ci następujące prawa:

5.1 Prawo dostępu do danych

Masz prawo uzyskać od nas potwierdzenie, czy przetwarzamy Twoje dane osobowe, a jeżeli tak, masz prawo uzyskać dostęp do nich oraz informacje o celach przetwarzania, kategoriach danych, odbiorcach lub kategoriach odbiorców, planowanym okresie przechowywania, źródle pozyskania oraz innych informacjach wymaganych przez RODO.

5.2 Prawo do sprostowania danych

Masz prawo żądać niezwłocznego sprostowania nieprawidłowych danych osobowych lub uzupełnienia niekompletnych danych.

5.3 Prawo do usunięcia danych ("prawo do bycia zapomnianym")

Masz prawo żądać usunięcia Twoich danych osobowych, jeżeli zachodzi jedna z okoliczności określonych w art. 17 RODO.

5.4 Prawo do ograniczenia przetwarzania

Masz prawo żądać ograniczenia przetwarzania Twoich danych osobowych w przypadkach określonych w art. 18 RODO.

5.5 Prawo do przenoszenia danych

Masz prawo otrzymać dane osobowe, które nam dostarczyłeś, w ustrukturyzowanym, powszechnie używanym formacie nadającym się do odczytu maszynowego oraz prawo przesyłania tych danych innemu administratorowi.

5.6 Prawo do sprzeciwu

Masz prawo, z przyczyn związanych z Twoją szczególną sytuacją, wnieść sprzeciw wobec przetwarzania Twoich danych osobowych opartego na prawnie uzasadnionym interesie (art. 6 ust. 1 lit. f RODO), w tym profilowaniu. Masz również prawo wnieść sprzeciw wobec przetwarzania Twoich danych na potrzeby marketingu bezpośredniego.

5.7 Prawo do cofnięcia zgody

Jeżeli przetwarzanie odbywa się na podstawie zgody, masz prawo cofnąć zgodę w dowolnym momencie bez wpływu na zgodność z prawem przetwarzania, którego dokonano na podstawie zgody przed jej cofnięciem.

5.8 Prawo do wniesienia skargi

Masz prawo wnieść skargę do organu nadzorczego, w szczególności w państwie członkowskim swojego zwykłego pobytu, swojego miejsca pracy lub miejsca popełnienia domniemanego naruszenia. W Polsce organem nadzorczym jest Prezes Urzędu Ochrony Danych Osobowych (ul. Stawki 2, 00-193 Warszawa).

6. Pliki cookie i inne technologie:

6. Pliki cookies i podobne technologie

6.1 Czym są pliki cookies?

Cookies to małe pliki tekstowe umieszczane na Twoim urządzeniu (komputerze, tablecie, smartfonie) podczas odwiedzania naszej strony internetowej. Umożliwiają one prawidłowe funkcjonowanie strony, zapamiętywanie Twoich preferencji oraz dostosowanie zawartości do Twoich potrzeb.

6.2 Jakie rodzaje cookies stosujemy?

Cookies niezbędne

Są niezbędne do prawidłowego funkcjonowania strony in

internetowej i nie mogą być wyłączone. Umożliwiają korzystanie z podstawowych funkcji strony, takich jak logowanie, zapamiętywanie zawartości koszyka czy ustawień prywatności.

Cookies analityczne

Pozwalają nam analizować sposób korzystania z naszej strony internetowej, co umożliwia jej ulepszanie i optymalizację. Zbierają informacje o liczbie odwiedzających, źródłach ruchu, odwiedzanych podstronach i czasie spędzonym na stronie.

Cookies funkcjonalne

Służą do zapamiętywania Twoich wyborów i preferencji na stronie, co zwiększa wygodę korzystania z niej.

Cookies marketingowe

Służą do wyświetlania reklam dostosowanych do Twoich zainteresowań na naszej stronie i na innych stronach internetowych, które odwiedzasz.

6.3 Jak długo przechowujemy cookies?

Stosujemy zarówno cookies sesyjne (usuwane po zamknięciu przeglądarki) jak i trwałe (przechowywane przez określony czas na Twoim urządzeniu).

6.4 Inne technologie

Oprócz plików cookies, możemy stosować inne podobne technologie, w tym:

- Sygnalizatory sieci web (web beacons);
- Piksele śledzące;
- Skrypty śledzące;
- Local Storage.

6.5 Zarządzanie cookies

Możesz kontrolować i zarządzać cookies w ustawieniach przeglądarki. Możesz skonfigurować przeglądarkę tak, aby akceptowała wszystkie cookies, odrzucała wszystkie cookies lub powiadamiała o umieszczaniu cookies. W

łączenie niektórych cookies może ograniczyć dostęp do wybranych funkcji strony.

Informacje o zarządzaniu cookies w najpopularniejszych przeglądarkach:

- [Chrome](<https://support.google.com/chrome/answer/95647?hl=pl>)
- [Firefox](<https://support.mozilla.org/pl/kb/usuwanie-ciasteczek>)
- [Safari](<https://support.apple.com/pl-pl/guide/safari/sfri11471/mac>)
- [Edge](<https://support.microsoft.com/pl-pl/microsoft-edge/usuwanie-pliku-cookie-w-programie-microsoft-edge-63947406-40ac-c3b8-57b9-2a946a29ae09>)

7. Bezpieczeństwo danych i okres przechowywania:

7. Bezpieczeństwo danych osobowych

Administrator danych stosuje odpowiednie środki techniczne i organizacyjne, aby zapewnić ochronę przetwarzanych danych osobowych, w szczególności:

- Szyfrowanie danych (wykorzystanie protokołu SSL/TLS);
- Pseudonimizację i anonimizację danych;
- Kontrolę dostępu i zarządzanie uprawnieniami;
- Regularne testowanie, mierzenie i ocenianie skuteczności środków bezpieczeństwa;
- Szkolenia personelu z zakresu ochrony danych;
- Procedury reagowania na incydenty bezpieczeństwa.

8. Okres przechowywania danych

Przechowujemy dane osobowe tylko tak długo, jak jest to niezbędne do celów, dla których są przetwarzane, lub do czasu wycofania zgody. W szczególności:

- Dane związane z kontem użytkownika: do czasu usunięcia konta;
- Dane związane z realizacją umowy: przez okres obowiązywania umowy, a po jej zakończeniu przez okres niez

- będny do dochodzenia roszczeń i wykonania obowiązków prawnych (np. podatkowych, księgowych);
- Dane przetwarzane na podstawie zgody: do momentu wycofania zgody;
 - Dane przetwarzane na podstawie prawnie uzasadnionego interesu: do czasu wniesienia skutecznego sprzeciwu.

Po upływie tych okresów dane są usuwane lub poddawane anonimizacji.

8. Postanowienia końcowe:

9. Zmiany Polityki Prywatności

Administrator zastrzega sobie prawo do wprowadzania zmian w Polityce Prywatności, co może być spowodowane:

- zmianą przepisów prawa;
- rozwojem technologii;
- zmianą zasad korzystania z serwisu;
- rozwojem oferty.

O każdej zmianie Polityki Prywatności użytkownicy będą informowani poprzez zamieszczenie odpowiedniej informacji na stronie internetowej lub drogą e-mailową.

10. Dane kontaktowe

W przypadku jakichkolwiek pytań, uwag lub wniosków do tyczących naszej Polityki Prywatności lub przetwarzania danych osobowych, prosimy o kontakt:

[Nazwa firmy/osoby]

Adres: [adres]

E-mail: [adres e-mail do kontaktu ws. danych osobowych]

Telefon: [numer telefonu]

Data ostatniej aktualizacji: [data]

Implementacja w HTML:

```
<!DOCTYPE html>
<html lang="pl">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width
, initial-scale=1.0">
  <title>Polityka Prywatności - Nazwa Firmy</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      line-height: 1.6;
      color: #333;
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
    }
    h1 {
      color: #2c3e50;
      border-bottom: 2px solid #ecf0f1;
      padding-bottom: 10px;
    }
    h2 {
      color: #3498db;
      margin-top: 30px;
    }
    h3 {
      color: #2980b9;
    }
    p, ul {
      margin-bottom: 15px;
    }
    a {
      color: #3498db;
      text-decoration: none;
    }
    a:hover {
      text-decoration: underline;
```

```

}
.container {
    background-color: #f9f9f9;
    padding: 30px;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}
.toc {
    background-color: #ecf0f1;
    padding: 15px;
    border-radius: 5px;
    margin-bottom: 30px;
}
.toc h3 {
    margin-top: 0;
}
.last-updated {
    font-style: italic;
    color: #7f8c8d;
    margin-top: 40px;
}
@media (max-width: 600px) {
    body {
        padding: 10px;
    }
    .container {
        padding: 15px;
    }
}
</style>
</head>
<body>
    <div class="container">
        <h1>Polityka Prywatności</h1>

        <div class="toc">
            <h3>Spis treści</h3>
            <ol>
                <li><a href="#intro">Wprowadzenie</a>

```

```

</li>
        <li><a href="#data-collection">Jakie
dane zbieramy?</a></li>
        <li><a href="#purpose">W jakim celu i
na jakiej podstawie przetwarzamy dane?</a></li>
        <li><a href="#sharing">Komu udostępni
amy Twoje dane?</a></li>
        <li><a href="#rights">Jakie masz praw
a?</a></li>
        <li><a href="#cookies">Pliki cookies
i podobne technologie</a></li>
        <li><a href="#security">Bezpieczeństw
o danych osobowych</a></li>
        <li><a href="#retention">Okres przech
owywania danych</a></li>
        <li><a href="#changes">Zmiany Polityk
i Prywatności</a></li>
        <li><a href="#contact">Dane kontaktow
e</a></li>
    </ol>
</div>

<!-- Tutaj umieść pełną treść polityki prywat
ności -->

<p class="last-updated">Data ostatniej aktual
izacji: 15.04.2025</p>
</div>
</body>
</html>

```

Dobre praktyki dla polityki prywatności: - Używaj prostego, zrozumiałego języka zamiast żargonu prawniczego - Strukturyzuj dokument za pomocą nagłówków i sekcji dla łatwiejszego czytania - Dodaj spis treści dla długich dokumentów - Regularnie aktualizuj w przypadku zmian w przepisach lub praktykach przetwarzania danych - Umieść wyraźny link do polityki prywatności w stopce strony oraz innych strategicznych miejscach - Zawsze informuj

użytkowników o istotnych zmianach w polityce prywatności -
Zapewnij możliwość łatwego kontaktu w przypadku pytań
dotyczących danych osobowych - Dostosuj politykę do specyfiki
Twojej strony (e-commerce, blog, forum) - Konsultuj się z
prawnikiem specjalizującym się w RODO i prawie internetowym

Dlaczego jest ważna dla początkujących: - Jest wymogiem
prawnym w większości krajów (RODO w UE, CCPA w Kalifornii, etc.)
- Buduje zaufanie użytkowników do strony - Chroni przed
potencjalnymi karami i postępowaniami prawnymi - Jest niezbędna
w przypadku zbierania jakichkolwiek danych osobowych - Stanowi
podstawę dla uzyskania świadomej zgody użytkowników

REGULAMIN

Regulamin strony internetowej to dokument określający zasady
korzystania z witryny, prawa i obowiązki użytkowników oraz
administratora, stanowiący umowę między stronami.

Kluczowe cechy: - Definiuje zasady korzystania z serwisu - Określa
prawa i obowiązki użytkowników i właścicieli - Wyjaśnia procedury
zakupowe w przypadku sklepów - Reguluje kwestie własności
intelektualnej - Wyznacza sposoby rozwiązywania sporów

Elementy dobrego regulaminu:

1. Postanowienia ogólne:

Regulamin serwisu [nazwa serwisu]

1. Postanowienia ogólne

1.1. Niniejszy Regulamin określa zasady korzystania z
serwisu internetowego [nazwa serwisu], dostępnego po
d adresem [adres strony] („Serwis”).

1.2. Właścicielem i administratorem Serwisu jest [nazwa
firmy/osoby], [forma prawna], z siedzibą w [adres]
, NIP: [NIP], REGON: [REGON], wpisana do rejestru prz

edsiębiorców Krajowego Rejestru Sądowego prowadzonego przez Sąd Rejonowy [miasto], [nr KRS] („Administrator”).

1.3. Kontakt z Administratorem jest możliwy:

- elektronicznie na adres e-mail: [adres e-mail]
- telefonicznie pod numerem: [nr telefonu]
- pisemnie na adres siedziby Administratora

1.4. Regulamin jest udostępniony nieodpłatnie za pośrednictwem Serwisu w formie umożliwiającej jego pozyskanie, odtwarzanie i utrwalanie. Użytkownik może pobrać Regulamin w formie pliku PDF tutaj: [link].

1.5. Akceptacja Regulaminu jest dobrowolna, ale konieczna do korzystania z Serwisu.

2. Definicje:

2. Definicje

W niniejszym Regulaminie następujące pojęcia będą miały znaczenie określone poniżej:

2.1. Serwis – serwis internetowy [nazwa serwisu], prowadzony przez Administratora, dostępny pod adresem [adres strony].

2.2. Użytkownik – osoba fizyczna, osoba prawna lub jednostka organizacyjna nieposiadająca osobowości prawnej, korzystająca z Serwisu.

2.3. Konto – zbiór danych i ustawień Użytkownika w systemie informatycznym Serwisu, do którego ma dostęp po zalogowaniu się.

2.4. Treści – wszelkie dane, informacje, materiały, elementy graficzne, zdjęcia, oprogramowanie i inne elementy udostępniane lub publikowane w ramach Serwisu przez Administratora lub Użytkowników.

2.5. RODO – Rozporządzenie Parlamentu Europejskiego i Rady (UE) 2016/679 z dnia 27 kwietnia 2016 r. w sprawie ochrony osób fizycznych w związku z przetwarzaniem danych osobowych i w sprawie swobodnego przepływu takich danych oraz uchylenia dyrektywy 95/46/WE.

3. Warunki korzystania z serwisu:

3. Warunki korzystania z Serwisu

3.1. Dostęp do Serwisu jest bezpłatny, chyba że wyraźnie wskazano inaczej.

3.2. Do korzystania z Serwisu niezbędne jest:

- urządzenie z dostępem do Internetu;
- przeglądarka internetowa: [wymienić wspierane przez przeglądarki, np. Chrome (wersja 90+), Firefox (wersja 88+), Safari (wersja 14+), Edge (wersja 90+)];
- włączona obsługa JavaScript i cookies;
- aktywne konto e-mail (w przypadku rejestracji).

3.3. Administrator podejmuje działania w celu zapewnienia prawidłowego funkcjonowania Serwisu, w zakresie wynikającym z aktualnej wiedzy technicznej. W przypadku błędów w działaniu Serwisu, Użytkownik może poinformować o tym Administratora.

3.4. Administrator zastrzega sobie prawo do:

- czasowego wyłączenia Serwisu lub jego części w celu konserwacji, ulepszeń lub naprawy błędów;
- dodawania i usuwania funkcjonalności Serwisu;
- zmiany wyglądu i układu Serwisu.

3.5. W przypadku planowanych przerw technicznych, Administrator dołoży starań, by poinformować o nich Użytkowników z odpowiednim wyprzedzeniem.

4. Rejestracja i konto użytkownika:

4. Rejestracja i konto użytkownika

4.1. Niektóre funkcje Serwisu mogą być dostępne tylko dla zarejestrowanych Użytkowników posiadających Konto.

4.2. Rejestracja w Serwisie jest dobrowolna i bezpłatna.

4.3. W celu rejestracji Konta, Użytkownik powinien wypełnić formularz rejestracyjny, podając wymagane dane i wyrażając zgodę na postanowienia niniejszego Regulaminu oraz Polityki Prywatności.

4.4. Po wypełnieniu formularza rejestracyjnego, na podany adres e-mail zostanie wysłana wiadomość z linkiem aktywacyjnym. Kliknięcie linku aktywacyjnego jest niezbędne do ukończenia procesu rejestracji.

4.5. Użytkownik może mieć tylko jedno Konto w Serwisie.

4.6. Użytkownik zobowiązany jest do:

- podania prawdziwych i aktualnych danych podczas rejestracji;
- aktualizowania danych w przypadku ich zmiany;
- zachowania w tajemnicy hasła dostępu do Konta;
- nieudostępniania Konta osobom trzecim.

4.7. Użytkownik ponosi pełną odpowiedzialność za:

- treść podanych przez siebie danych;
- działania i zaniechania wykonane za pośrednictwem Konta;
- zachowanie w tajemnicy hasła dostępu do Konta;
- szkody powstałe w wyniku nieprawidłowego korzystania z Konta.

4.8. Administrator może usunąć Konto Użytkownika lub ograniczyć dostęp do niego, jeżeli Użytkownik:

- narusza postanowienia Regulaminu;
- podejmuje działania na szkodę innych Użytkowników, osób trzecich lub Administratora;
- podał nieprawdziwe dane podczas rejestracji;
- nie korzystał z Konta przez okres dłuższy niż 12 miesięcy.

4.9. Użytkownik może w każdej chwili usunąć Konto, wysyłając stosowny wniosek do Administratora na adres e-mail: [adres e-mail] lub korzystając z odpowiedniej funkcji w ustawieniach Konta.

5. Zasady publikowania treści (dla serwisów z treściami tworzonymi przez użytkowników):

5. Zasady publikowania treści przez Użytkowników

5.1. Użytkownik może publikować w Serwisie własne Treści, w ramach funkcjonalności udostępnionych przez Administratora.

5.2. Publikując Treści, Użytkownik oświadcza, że:

- jest ich autorem lub posiada odpowiednie prawa do ich publikacji;
- Treści nie naruszają praw osób trzecich, w tym praw autorskich, praw własności przemysłowej, dóbr osobistych;
- Treści nie zawierają materiałów bezprawnych, w szczególności o charakterze obraźliwym, rasistowskim, pornograficznym lub nawołującym do nienawiści.

5.3. Zabronione jest publikowanie Treści, które:

- naruszają przepisy prawa, dobre obyczaje lub normy moralne;
- zawierają treści pornograficzne, erotyczne, wulgarnie, obraźliwe, rasistowskie;
- nawołują do nienawiści, przemocy lub dyskryminacji;
- naruszają prawa osób trzecich, w tym prawa autorskie, prawa własności przemysłowej;
- wprowadzają w błąd, zawierają nieprawdziwe informacje.

je;

- mają charakter spamu lub niezamówionej reklamy;
- zawierają wirusy lub inne szkodliwe oprogramowanie;
- służą promocji działalności konkurencyjnej wobec Administratora.

5.4. Administrator nie monitoruje ani nie weryfikuje Treści publikowanych przez Użytkowników przed ich opublikowaniem. Administrator zastrzega sobie jednak prawo do:

- moderowania i usuwania Treści naruszających Regulamin;
- blokowania możliwości publikowania Treści przez Użytkowników naruszających Regulamin;
- modyfikacji Treści w celu dostosowania ich do wymogów technicznych Serwisu.

5.5. Administrator nie ponosi odpowiedzialności za Treści publikowane przez Użytkowników.

5.6. Użytkownik publikujący Treści w Serwisie udziela Administratorowi niewyłącznej, nieodpłatnej, nieograniczonej terytorialnie licencji na korzystanie z tych Treści w zakresie niezbędnym do świadczenia usług w ramach Serwisu, w szczególności do:

- przechowywania Treści na serwerach;
- wyświetlania Treści innym Użytkownikom;
- modyfikacji Treści w zakresie niezbędnym do dostosowania ich do wymogów technicznych Serwisu.

5.7. Licencja jest udzielana na czas publikacji Treści w Serwisie i wygasa po ich usunięciu przez Użytkownika.

6. Warunki sprzedaży (dla sklepów internetowych):

6. Warunki sprzedaży

6.1. Za pośrednictwem Serwisu, Administrator prowadzi sprzedaż towarów i/lub usług ("Produkty").

6.2. Informacje o Produktach, w tym ceny, opisy i dostępność, stanowią zaproszenie do zawarcia umowy w rozumieniu art. 71 Kodeksu cywilnego.

6.3. Ceny Produktów są podane w złotych polskich i zawierają podatek VAT, chyba że wyraźnie wskazano inaczej.

6.4. Koszty dostawy są podawane przy składaniu zamówienia i doliczane do jego wartości.

6.5. Zawarcie umowy sprzedaży:

- W celu złożenia zamówienia, Użytkownik wybiera Produkt, a następnie przechodzi do procesu zamówienia, podając wymagane dane i wybierając metodę dostawy i płatności.
- Po złożeniu zamówienia, Użytkownik otrzymuje potwierdzenie jego przyjęcia drogą elektroniczną.
- Umowa sprzedaży zostaje zawarta z chwilą potwierdzenia przez Administratora przyjęcia zamówienia do realizacji.

6.6. Metody płatności:

- Płatność elektroniczna (przelew online, BLIK, karty płatnicze)
- Przelew tradycyjny
- Płatność za pobraniem
- [inne dostępne metody]

6.7. Metody dostawy:

- Przesyłka kurierska
- Paczkomat
- Odbiór osobisty
- [inne dostępne metody]

6.8. Termin realizacji zamówienia wynosi [X-Y] dni roboczych od momentu zaksięgowania płatności lub od momentu złożenia zamówienia w przypadku płatności za pob

ranie.

6.9. W przypadku braku możliwości realizacji zamówienia, Administrator niezwłocznie poinformuje o tym Użytkownika, proponując alternatywne rozwiązanie lub zwrot wpłaconych środków.

7. Reklamacje i zwroty (dla sklepów internetowych):

7. Reklamacje i zwroty

7.1. Produkty oferowane w Serwisie objęte są [X] miesięczną gwarancją producenta i/lub rękojmią sprzedawcy zgodnie z przepisami prawa.

7.2. Odstąpienie od umowy (zwrot):

- Użytkownik będący konsumentem może odstąpić od umowy zawartej na odległość bez podania przyczyny w terminie 14 dni od dnia otrzymania Produktu.
- Aby skorzystać z prawa odstąpienia od umowy, Użytkownik powinien poinformować Administratora o swojej decyzji poprzez jednoznaczne oświadczenie (na przykład pismo wysłane pocztą, faksem lub pocztą elektroniczną).
- Użytkownik może skorzystać z wzoru formularza odstąpienia od umowy, dostępnego [tutaj], jednak nie jest to obowiązkowe.
- Termin do odstąpienia od umowy wygasa po upływie 14 dni od dnia, w którym Użytkownik wszedł w posiadanie rzeczy lub w którym osoba trzecia inna niż przewoźnik i wskazana przez Użytkownika weszła w posiadanie rzeczy.
- W przypadku odstąpienia od umowy, Administrator zwraca wszystkie otrzymane od Użytkownika płatności, w tym koszty dostarczenia rzeczy (z wyjątkiem dodatkowych kosztów wynikających z wybranego przez Użytkownika sposobu dostarczenia innego niż najtańszy zwykły sposób dostarczenia oferowany przez Administratora), niezwłocznie, a w każdym przypadku nie później niż 14 dni od dnia, w którym Administrator został poinformowany

o decyzji Użytkownika o wykonaniu prawa odstąpienia od umowy.

- Zwrot płatności zostanie dokonany przy użyciu takich samych sposobów płatności, jakie zostały użyte przez Użytkownika w pierwotnej transakcji, chyba że Użytkownik wyraźnie zgodził się na inne rozwiązanie.

- Administrator może wstrzymać się ze zwrotem płatności do czasu otrzymania rzeczy lub do czasu dostarczenia dowodu jej odesłania, w zależności od tego, które zdarzenie nastąpi wcześniej.

- Użytkownik powinien odesłać rzecz do Administratora niezwłocznie, a w każdym razie nie później niż 14 dni od dnia, w którym poinformował Administratora o odstąpieniu od umowy.

- Użytkownik ponosi bezpośrednie koszty zwrotu rzeczy.

- Użytkownik odpowiada tylko za zmniejszenie wartości rzeczy wynikające z korzystania z niej w sposób inny niż było to konieczne do stwierdzenia charakteru, cech i funkcjonowania rzeczy.

7.3. Prawo odstąpienia od umowy nie przysługuje Użytkownikowi w odniesieniu do umów:

- o świadczenie usług, jeżeli Administrator wykonał w pełni usługę za wyraźną zgodą Użytkownika, który został poinformowany przed rozpoczęciem świadczenia, że po spełnieniu świadczenia przez Administratora utraci prawo odstąpienia od umowy;

- w której przedmiotem świadczenia jest rzecz nieprefabrykowana, wyprodukowana według specyfikacji Użytkownika lub służąca zaspokojeniu jego zindywidualizowanych potrzeb;

- w której przedmiotem świadczenia jest rzecz ulegająca szybkiemu zepsuciu lub mająca krótki termin przydatności do użycia;

- w której przedmiotem świadczenia jest rzecz dostarczona w zapieczętowanym opakowaniu, której po otwarciu opakowania nie można zwrócić ze względu na ochronę zdrowia lub ze względów higienicznych, jeżeli opakowanie

ie zostało otwarte po dostarczeniu;

- w której przedmiotem świadczenia są nagrania dźwiękowe lub wizualne albo programy komputerowe dostarczane w zabezpieczonym opakowaniu, jeżeli opakowanie zostało otwarte po dostarczeniu;
- o dostarczanie treści cyfrowych, które nie są zapisane na nośniku materialnym, jeżeli spełnianie świadczenia rozpoczęło się za wyraźną zgodą Użytkownika przed upływem terminu do odstąpienia od umowy i po poinformowaniu go przez Administratora o utracie prawa odstąpienia od umowy.

7.4. Reklamacje:

- Użytkownik ma prawo do złożenia reklamacji, jeśli Produkt ma wadę fizyczną lub prawną (rękojmia).
- Reklamacja powinna zawierać dane Użytkownika, informacje o Produkcie i zamówieniu, opis wady oraz żądania Użytkownika (np. naprawa, wymiana, obniżenie ceny, zwrot pieniędzy).
- Reklamację można złożyć drogą elektroniczną na adres: [adres e-mail] lub pisemnie na adres siedziby Administratora.
- Administrator rozpatrzy reklamację w terminie 14 dni od jej otrzymania i poinformuje Użytkownika o swojej decyzji tą samą drogą, którą reklamacja została złożona.
- W przypadku uznania reklamacji, Administrator, w zależności od żądania Użytkownika, wymieni Produkt na wolny od wad, usunie wadę, obniży cenę lub zwróci pieniądze.
- Koszty związane z odesłaniem reklamowanego Produktu do Administratora ponosi Administrator.
- W przypadku odrzucenia reklamacji, Administrator uzasadni swoją decyzję.

7.5. Pozasądowe sposoby rozpatrywania reklamacji i dochodzenia roszczeń:

- Użytkownik będący konsumentem ma możliwość skorzystania z pozasądowych sposobów rozpatrywania reklamacji

i dochodzenia roszczeń, w tym:

- * złożenia skargi za pośrednictwem unijnej platformy internetowej ODR, dostępnej pod adresem: <http://ec.europa.eu/consumers/odr/>

- * zwrócenia się do stałego polubownego sądu konsumenckiego

- * zwrócenia się do wojewódzkiego inspektora Inspekcji Handlowej

- * uzyskania bezpłatnej pomocy powiatowego (miejskiego) rzecznika konsumentów

- Szczegółowe informacje o pozasądowych sposobach rozpatrywania reklamacji i dochodzenia roszczeń dostępne są w siedzibach oraz na stronach internetowych powiatowych (miejskich) rzeczników konsumentów, organizacji społecznych, do których zadań statutowych należy ochrona konsumentów, Wojewódzkich Inspektoratów Inspekcji Handlowej oraz pod adresem <https://www.uokik.gov.pl>

8. Własność intelektualna:

8. Własność intelektualna

8.1. Wszelkie prawa własności intelektualnej do Serwisu i jego elementów (w tym oprogramowania, układu funkcjonalnego, elementów graficznych, baz danych i Treści) należą do Administratora lub podmiotów, z którymi Administrator zawarł stosowne umowy.

8.2. Korzystanie z Serwisu nie oznacza nabycia jakichkolwiek praw do Serwisu ani jego elementów. Bez uprzedniej zgody Administratora, Użytkownik nie może:

- kopiować, modyfikować, rozpowszechniać ani transmitować żadnej części Serwisu;

- używać znaków towarowych, logotypów ani innych elementów charakterystycznych dla Serwisu;

- dekompilować, dezasemblować ani dokonywać inżynierii wstecznej jakiegokolwiek części Serwisu;

- usuwać informacji o prawach autorskich, znakach towarowych lub innych informacji o prawach własności z S

erwisu;

- korzystać z Serwisu w sposób naruszający prawa własności intelektualnej osób trzecich.

8.3. Administrator udziela Użytkownikowi niewyłącznej, nieprzenoszalnej, odwoływalnej licencji na korzystanie z Serwisu wyłącznie na potrzeby prywatne, niekomercyjne i zgodne z Regulaminem.

8.4. Wszelkie znaki towarowe, logotypy, nazwy firm itp. wyświetlane w Serwisie podlegają ochronie prawnej i należą do ich prawowitych właścicieli.

8.5. W przypadku powzięcia przez Administratora wiarygodnej informacji, że Użytkownik narusza prawa własności intelektualnej Administratora lub osób trzecich, Administrator ma prawo usunąć kwestionowane treści lub zablokować dostęp do Serwisu Użytkownikowi naruszającemu te prawa.

9. Odpowiedzialność:

9. Odpowiedzialność

9.1. Administrator dołoży wszelkich starań, aby Serwis działał prawidłowo i bez zakłóceń, jednak nie gwarantuje ciągłości działania Serwisu i nie ponosi odpowiedzialności za przerwy w jego funkcjonowaniu.

9.2. Administrator nie ponosi odpowiedzialności za:

- szkody wynikłe z korzystania z Serwisu w sposób nie zgodny z Regulaminem lub przepisami prawa;
- szkody wynikłe z podania przez Użytkownika nieprawdziwych, nieaktualnych lub niepełnych danych;
- szkody wynikłe z działania lub zaniechania osób trzecich, za które Administrator nie ponosi odpowiedzialności;
- szkody wynikłe z działania siły wyższej;
- szkody wynikłe z działania wirusów komputerowych lub innego szkodliwego oprogramowania;

- szkody wynikłe z niedostępności Serwisu spowodowane j przyczynami niezależnymi od Administratora;
- treści publikowane przez Użytkowników;
- utratę danych spowodowaną awarią sprzętu, systemu l ub innymi okolicznościami niezależnymi od Administrat ora.

9.3. Odpowiedzialność Administratora wobec Użytkownik a jest ograniczona do wysokości rzeczywiście poniesio nej szkody i nie obejmuje utraconych korzyści.

9.4. Ograniczenia odpowiedzialności nie dotyczą szkód wyrządzonych przez Administratora umyślnie oraz przy padków, w których ograniczenie odpowiedzialności jest niedopuszczalne zgodnie z bezwzględnie obowiązującym i przepisami prawa.

9.5. Użytkownik ponosi pełną odpowiedzialność za:

- korzystanie z Serwisu w sposób niezgodny z Regulami nem lub przepisami prawa;
- treści publikowane przez siebie w Serwisie;
- szkody wyrządzone osobom trzecim w związku z korzyst aniem z Serwisu;
- nieprawidłowe, niepełne lub nieaktualne dane podane podczas rejestracji.

10. Zmiany regulaminu i postanowienia końcowe:

10. Zmiany Regulaminu

10.1. Administrator zastrzega sobie prawo do zmiany R egulaminu w dowolnym czasie z ważnych przyczyn, takie h jak:

- zmiana przepisów prawa mających wpływ na treść Regu laminu;
- zmiana funkcjonalności Serwisu;
- zmiana sposobu świadczenia usług;
- zmiana danych Administratora;
- względy bezpieczeństwa.

10.2. Zmiany Regulaminu będą publikowane na stronie Serwisu. O zmianach Regulaminu Administrator poinformuje Użytkowników posiadających Konto za pośrednictwem wiadomości e-mail, z co najmniej 14-dniowym wyprzedzeniem.

10.3. Jeżeli Użytkownik nie akceptuje zmian Regulaminu, powinien zaprzestać korzystania z Serwisu lub usunąć Konto przed dniem wejścia zmian w życie.

10.4. Korzystanie przez Użytkownika z Serwisu po wprowadzeniu zmian Regulaminu oznacza ich akceptację.

11. Postanowienia końcowe

11.1. Regulamin oraz umowy zawierane na jego podstawie podlegają prawu polskiemu.

11.2. W przypadku, gdy Użytkownik jest konsumentem, spory będą rozstrzygane przez sąd właściwy zgodnie z przepisami kodeksu postępowania cywilnego.

11.3. W przypadku, gdy Użytkownik nie jest konsumentem, spory będą rozstrzygane przez sąd właściwy miejscowo dla siedziby Administratora.

11.4. W sprawach nieuregulowanych Regulaminem zastosowanie mają odpowiednie przepisy prawa polskiego, w szczególności: Kodeksu cywilnego, ustawy o świadczeniu usług drogą elektroniczną, ustawy o prawach konsumenta, przepisy o ochronie danych osobowych.

11.5. Nieważność lub bezskuteczność któregośkolwiek z postanowień Regulaminu nie powoduje nieważności lub bezskuteczności pozostałych postanowień.

11.6. Regulamin wchodzi w życie z dniem [data].

Implementacja w HTML:

```

<!DOCTYPE html>
<html lang="pl">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width
, initial-scale=1.0">
  <title>Regulamin - Nazwa Firmy</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      line-height: 1.6;
      color: #333;
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
    }
    h1 {
      color: #2c3e50;
      border-bottom: 2px solid #ecf0f1;
      padding-bottom: 10px;
    }
    h2 {
      color: #3498db;
      margin-top: 30px;
    }
    h3 {
      color: #2980b9;
    }
    p, ul, ol {
      margin-bottom: 15px;
    }
    a {
      color: #3498db;
      text-decoration: none;
    }
    a:hover {
      text-decoration: underline;
    }
    .container {

```

```

        background-color: #f9f9f9;
        padding: 30px;
        border-radius: 5px;
        box-shadow: 0 2px 5px rgba(0,0,0,0.1);
    }
    .toc {
        background-color: #ecf0f1;
        padding: 15px;
        border-radius: 5px;
        margin-bottom: 30px;
    }
    .toc h3 {
        margin-top: 0;
    }
    .last-updated {
        font-style: italic;
        color: #7f8c8d;
        margin-top: 40px;
    }
    @media (max-width: 600px) {
        body {
            padding: 10px;
        }
        .container {
            padding: 15px;
        }
    }
</style>
</head>
<body>
    <div class="container">
        <h1>Regulamin serwisu</h1>

        <div class="toc">
            <h3>Spis treści</h3>
            <ol>
                <li><a href="#general">Postanowienia
ogólne</a></li>
                <li><a href="#definitions">Definicje<

```

```

/a></li>
<li><a href="#conditions">Warunki kor
zystania z Serwisu</a></li>
<li><a href="#account">Rejestracja i
konto użytkownika</a></li>
<li><a href="#content">Zasady publiko
wania treści</a></li>
<li><a href="#sales">Warunki sprzedaż
y</a></li>
<li><a href="#complaints">Reklamacje
i zwroty</a></li>
<li><a href="#intellectual">Własność
intelektualna</a></li>
<li><a href="#liability">Odpowiedzial
ność</a></li>
<li><a href="#changes">Zmiany Regulam
inu</a></li>
<li><a href="#final">Postanowienia ko
ńcowe</a></li>
</ol>
</div>

<!-- Tutaj umieść pełną treść regulaminu -->

<p class="last-updated">Data ostatniej aktual
izacji: 15.04.2025</p>
</div>
</body>
</html>

```

Dobre praktyki dla regulaminu: - Dostosuj regulamin do specyfiki Twojej strony (sklep, forum, portal informacyjny) - Używaj prostego, zrozumiałego języka, unikając nadmiernego żargonu prawniczego - Strukturyzuj dokument nagłówkami i numeracją dla łatwiejszego odniesienia - Uwzględnij wszystkie aspekty korzystania z witryny i oferowanych usług - Zawsze informuj użytkowników o zmianach w regulaminie z odpowiednim wyprzedzeniem - Umożliw łatwy dostęp do regulaminu z każdej podstrony (np. link w stopce) -

Zadbaj o możliwość pobrania regulaminu w formacie PDF -
Regularnie aktualizuj w przypadku zmian prawnych lub
funkcjonalnych serwisu - Konsultuj się z prawnikiem przy tworzeniu
i aktualizacji regulaminu

Dlaczego jest ważny dla początkujących: - Stanowi umowę
między właścicielem strony a użytkownikami - Określa prawa i
obowiązki obu stron - Chroni przed potencjalnymi roszczeniami i
sporami - Jest wymagany prawnie w przypadku sklepów
internetowych - Buduje zaufanie użytkowników dzięki
transparentnym zasadom

INFORMACJE O PLIKACH COOKIE

Informacja o plikach cookie to komunikat informujący
użytkowników o wykorzystywaniu plików cookie i innych
technologii śledzących na stronie, wymagany przez przepisy prawa
w wielu krajach.

Kluczowe cechy: - Informuje o rodzajach używanych plików cookie
- Wyjaśnia cel ich wykorzystania - Umożliwia wyrażenie zgody lub
odmowy - Zawiera instrukcje zarządzania plikami cookie - Jest
widoczna przy pierwszych odwiedzinach strony

Elementy dobrej informacji o plikach cookie:

1. Podstawowe informacje:

Polityka dotycząca plików cookie

Ta strona korzysta z plików cookie, aby zapewnić najl
epsze doświadczenia podczas korzystania z naszej witr
yny. Kontynuując przeglądanie strony bez zmiany ustaw
ień przeglądarki dotyczących plików cookie, wyrażasz
zgoda na ich wykorzystanie. Możesz jednak zmienić ust
awienia dotyczące plików cookie w dowolnym momencie.

2. Czym są pliki cookie:

Czym są pliki cookie?

Pliki cookie to małe pliki tekstowe przechowywane na Twoim urządzeniu (komputerze, tablecie lub telefonie) przez przeglądarkę internetową podczas przeglądania stron internetowych. Umożliwiają one stronie "zapamiętanie" Twoich działań i preferencji (np. język, rozmiar czcionki i inne preferencje wyświetlania) przez pewien okres, dzięki czemu nie musisz ich ponownie wprowadzać za każdym razem, gdy wracasz na stronę lub przeglądasz różne podstrony.

3. Rodzaje plików cookie:

Jakich plików cookie używamy?

Na naszej stronie wykorzystujemy następujące rodzaje plików cookie:

Niezbędne pliki cookie

Są konieczne do prawidłowego funkcjonowania strony. Umożliwiają korzystanie z podstawowych funkcji, takich jak bezpieczeństwo, zarządzanie siecią, dostęp do zabezpieczonych obszarów. Bez tych plików cookie strona nie może prawidłowo funkcjonować.

Analityczne pliki cookie

Pozwalają nam zrozumieć, w jaki sposób odwiedzający korzystają z naszej strony. Zbierają informacje o liczbie odwiedzających, źródłach ruchu, odwiedzanych podstronach. Pomagają nam ulepszać stronę i monitorować jej wydajność. Wykorzystujemy do tego narzędzia takie jak Google Analytics.

Funkcjonalne pliki cookie

Umożliwiają zapamiętanie Twoich preferencji i ustawień, takich jak język, region, preferencje dotyczące wyglądu. Dzięki nim nie musisz ponownie konfigurować strony przy każdych odwiedzinach.

Marketingowe pliki cookie

Służą do śledzenia odwiedzających na stronach internetowych. Ich celem jest wyświetlanie reklam, które są odpowiednio dopasowane i angażujące dla poszczególnych użytkowników, a przez to bardziej wartościowe dla wydawców i reklamodawców zewnętrznych.

4. Zarządzanie plikami cookie:

Jak zarządzać plikami cookie?

Większość przeglądarek internetowych automatycznie akceptuje pliki cookie. Możesz jednak zmienić ustawienia przeglądarki tak, aby usunąć lub zablokować pliki cookie.

Poniżej znajdziesz informacje, jak zarządzać plikami cookie w najpopularniejszych przeglądarkach:

- **Chrome**: Menu > Ustawienia > Zaawansowane > Prywatność i bezpieczeństwo > Ustawienia witryn > Pliki cookie i dane witryn. [Więcej informacji](<https://support.google.com/chrome/answer/95647?hl=pl>)
- **Firefox**: Menu > Opcje > Prywatność i bezpieczeństwo > Ciasteczka i dane witryn. [Więcej informacji](<https://support.mozilla.org/pl/kb/usuwanie-ciasteczek>)
- **Safari**: Preferencje > Prywatność > Pliki cookie i dane witryn. [Więcej informacji](<https://support.apple.com/pl-pl/guide/safari/sfri11471/mac>)
- **Edge**: Menu > Ustawienia > Prywatność, wyszukiwanie i usługi > Wyczyść dane przeglądania > Wybierz, co chcesz wyczyścić. [Więcej informacji](<https://support.microsoft.com/pl-pl/microsoft-edge/usuwanie-plik%C3%B3w-cookie-w-programie-microsoft-edge-63947406-40ac-c3b8-57b9-2a946a29ae09>)

- ****Opera****: Preferencje > Zaawansowane > Prywatność i bezpieczeństwo > Ustawienia treści > Pliki cookie. [Więcej informacji](https://help.opera.com/pl/latest/web-preferences/#cookies)

Pamiętaj, że ograniczenie plików cookie może wpłynąć na funkcjonalność naszej strony.

Pliki cookie podmiotów zewnętrznych

Nasza strona korzysta również z plików cookie dostarczanych przez podmioty zewnętrzne, w tym:

1. ****Google Analytics**** - do analizy ruchu na stronie
2. ****Facebook**** - do integracji z mediami społecznościowymi i remarketingu
3. ****[Inne używane serwisy]****

Aby dowiedzieć się więcej o tym, jak te podmioty wykorzystują pliki cookie, zapoznaj się z ich politykami prywatności.

5. Aktualizacje i kontakt:

Aktualizacje polityki cookie

Regularnie przeglądamy i aktualizujemy naszą politykę dotyczącą plików cookie. Ostatnia aktualizacja miała miejsce [data].

Kontakt

W przypadku pytań dotyczących naszej polityki plików cookie, prosimy o kontakt:

[Nazwa firmy/osoby]
E-mail: [adres e-mail]
Telefon: [numer telefonu]

Przykład banneru informującego o plikach cookie:

```

<div id="cookie-banner" class="cookie-banner">
  <div class="cookie-content">
    <p>
      Ta strona wykorzystuje pliki cookie, aby zapewnić
      najlepsze doświadczenie.
      Dowiedz się więcej w naszej <a href="/cookie-policy">Polityce plików cookie</a>.
    </p>
    <div class="cookie-buttons">
      <button id="cookie-accept-all" class="btn btn-primary">Akceptuj wszystkie</button>
      <button id="cookie-settings" class="btn btn-secondary">Ustawienia</button>
      <button id="cookie-reject" class="btn btn-outline">Odrzuć opcjonalne</button>
    </div>
  </div>
</div>

<!-- Panel ustawień plików cookie (początkowo ukryty) -->
<div id="cookie-settings-panel" class="cookie-settings-panel" style="display: none;">
  <div class="settings-content">
    <h3>Ustawienia plików cookie</h3>
    <p>Możesz określić, które rodzaje plików cookie chcesz akceptować:</p>

    <div class="cookie-options">
      <div class="cookie-option">
        <input type="checkbox" id="essential-cookies" checked disabled>
        <label for="essential-cookies">Niezbędne (wymagane)</label>
        <p>Są konieczne do prawidłowego funkcjonowania strony.</p>
      </div>

      <div class="cookie-option">

```

```

        <input type="checkbox" id="analytics-cookies"
checked>
        <label for="analytics-cookies">Analityczne</l
abel>
        <p>Pomagają nam zrozumieć, jak użytkownicy ko
rzystają z naszej strony.</p>
    </div>

    <div class="cookie-option">
        <input type="checkbox" id="functional-cookies
" checked>
        <label for="functional-cookies">Funkcjonalne<
/label>
        <p>Zapamiętują Twoje wybory i preferencje na
stronie.</p>
    </div>

    <div class="cookie-option">
        <input type="checkbox" id="marketing-cookies"
>
        <label for="marketing-cookies">Marketingowe</
label>
        <p>Służą do wyświetlania dopasowanych reklam.
</p>
    </div>
</div>

<div class="settings-buttons">
    <button id="save-preferences" class="btn btn-pr
imary">Zapisz preferencje</button>
    <button id="close-settings" class="btn btn-seco
ndary">Zamknij</button>
</div>
</div>
</div>

```

CSS dla banneru cookie:

```

.cookie-banner {
    position: fixed;

```

```
bottom: 0;
left: 0;
right: 0;
background-color: #fff;
box-shadow: 0 -2px 10px rgba(0, 0, 0, 0.1);
z-index: 9999;
padding: 15px;
display: none; /* Domyślnie ukryty, pokazywany prze
z JavaScript */
}
```

```
.cookie-content {
  max-width: 1200px;
  margin: 0 auto;
  display: flex;
  flex-wrap: wrap;
  justify-content: space-between;
  align-items: center;
  gap: 20px;
}
```

```
.cookie-content p {
  margin: 0;
  flex: 1;
  min-width: 300px;
}
```

```
.cookie-buttons {
  display: flex;
  gap: 10px;
  flex-wrap: wrap;
}
```

```
.btn {
  padding: 8px 16px;
  border-radius: 4px;
  font-size: 14px;
  cursor: pointer;
  border: none;
}
```

```
font-weight: 500;
}

.btn-primary {
background-color: #4CAF50;
color: white;
}

.btn-secondary {
background-color: #f1f1f1;
color: #333;
}

.btn-outline {
background-color: transparent;
border: 1px solid #ddd;
color: #333;
}

/* Panel ustawień cookie */
.cookie-settings-panel {
position: fixed;
top: 0;
left: 0;
right: 0;
bottom: 0;
background-color: rgba(0, 0, 0, 0.5);
z-index: 10000;
display: flex;
justify-content: center;
align-items: center;
}

.settings-content {
background-color: white;
padding: 30px;
border-radius: 8px;
max-width: 600px;
width: 90%;
}
```



```
    max-height: 80vh;
    overflow-y: auto;
}

.cookie-options {
    margin: 20px 0;
}

.cookie-option {
    margin-bottom: 15px;
    padding-bottom: 15px;
    border-bottom: 1px solid #eee;
}

.cookie-option:last-child {
    border-bottom: none;
}

.cookie-option label {
    font-weight: 600;
    margin-left: 8px;
}

.cookie-option p {
    margin: 5px 0 0 25px;
    font-size: 13px;
    color: #666;
}

.settings-buttons {
    display: flex;
    justify-content: space-between;
    margin-top: 20px;
}

/* Responsywność */
@media (max-width: 576px) {
    .cookie-content {
        flex-direction: column;
    }
}
```

```

    gap: 15px;
}

.cookie-buttons {
    width: 100%;
    justify-content: center;
}

.settings-content {
    padding: 20px;
}
}

```

JavaScript dla banneru cookie:

```

document.addEventListener('DOMContentLoaded', function() {
    const cookieBanner = document.getElementById('cookie-banner');
    const cookieSettingsPanel = document.getElementById('cookie-settings-panel');
    const acceptAllButton = document.getElementById('cookie-accept-all');
    const settingsButton = document.getElementById('cookie-settings');
    const rejectButton = document.getElementById('cookie-reject');
    const savePreferencesButton = document.getElementById('save-preferences');
    const closeSettingsButton = document.getElementById('close-settings');

    // Checkboxy
    const analyticsCheckbox = document.getElementById('analytics-cookies');
    const functionalCheckbox = document.getElementById('functional-cookies');
    const marketingCheckbox = document.getElementById('marketing-cookies');

```

```

// Sprawdzenie, czy użytkownik już dokonał wyboru
const cookieConsent = getCookie('cookie_consent');

if (!cookieConsent) {
  // Pokazanie banneru, jeśli użytkownik nie dokonał
  // jeszcze wyboru
  cookieBanner.style.display = 'block';
} else {
  // Zastosowanie zapisanych preferencji
  applyStoredPreferences(JSON.parse(cookieConsent))
;
}

// Obsługa przycisku "Akceptuj wszystkie"
acceptAllButton.addEventListener('click', function(
) {
  const preferences = {
    essential: true,
    analytics: true,
    functional: true,
    marketing: true
  };

  savePreferences(preferences);
  cookieBanner.style.display = 'none';
});

// Obsługa przycisku "Odrzuć opcjonalne"
rejectButton.addEventListener('click', function() {
  const preferences = {
    essential: true,
    analytics: false,
    functional: false,
    marketing: false
  };

  // Aktualizacja checkboxów w panelu ustawień
  analyticsCheckbox.checked = false;
  functionalCheckbox.checked = false;

```

```

marketingCheckbox.checked = false;

savePreferences(preferences);
cookieBanner.style.display = 'none';
});

// Obsługa przycisku "Ustawienia"
settingsButton.addEventListener('click', function()
{
    cookieSettingsPanel.style.display = 'flex';
});

// Obsługa przycisku "Zapisz preferencje"
savePreferencesButton.addEventListener('click', function() {
    const preferences = {
        essential: true, // Zawsze włączone
        analytics: analyticsCheckbox.checked,
        functional: functionalCheckbox.checked,
        marketing: marketingCheckbox.checked
    };

    savePreferences(preferences);
    cookieSettingsPanel.style.display = 'none';
    cookieBanner.style.display = 'none';
});

// Obsługa przycisku "Zamknij"
closeSettingsButton.addEventListener('click', function() {
    cookieSettingsPanel.style.display = 'none';
});

// Zamykanie panelu ustawień przy kliknięciu poza jego zawartością
cookieSettingsPanel.addEventListener('click', function(e) {
    if (e.target === this) {
        cookieSettingsPanel.style.display = 'none';
    }
});

```

```

    }
  });

  // Funkcja zapisująca preferencje
  function savePreferences(preferences) {
    // Zapisanie preferencji w cookie (ważne przez 6
    // miesiące)
    setCookie('cookie_consent', JSON.stringify(preferences), 180);

    // Zastosowanie preferencji
    applyPreferences(preferences);
  }

  // Funkcja stosująca preferencje do skryptów strony
  function applyPreferences(preferences) {
    // Przykład: aktywowanie skryptów analitycznych,
    // jeśli dozwolone
    if (preferences.analytics) {
      enableAnalytics();
    } else {
      disableAnalytics();
    }

    // Przykład: aktywowanie skryptów marketingowych,
    // jeśli dozwolone
    if (preferences.marketing) {
      enableMarketing();
    } else {
      disableMarketing();
    }
  }

  // Funkcja stosująca zapisane preferencje
  function applyStoredPreferences(preferences) {
    // Aktualizacja checkboxów w panelu ustawień
    analyticsCheckbox.checked = preferences.analytics
;
    functionalCheckbox.checked = preferences.function

```

```

al;
marketingCheckbox.checked = preferences.marketing
;

// Zastosowanie preferencji do skryptów
applyPreferences(preferences);
}

// Funkcje pomocnicze do obsługi cookies
function setCookie(name, value, days) {
    const date = new Date();
    date.setTime(date.getTime() + (days * 24 * 60 * 6
0 * 1000));
    const expires = "expires=" + date.toUTCString();
    document.cookie = name + "=" + value + ";" + expi
res + ";path=/;SameSite=Strict";
}

function getCookie(name) {
    const nameEQ = name + "=";
    const ca = document.cookie.split(';');
    for(let i = 0; i < ca.length; i++) {
        let c = ca[i];
        while (c.charAt(0) === ' ') c = c.substring(1,
c.length);
        if (c.indexOf(nameEQ) === 0) return c.substring
(nameEQ.length, c.length);
    }
    return null;
}

// Funkcja aktywująca skrypty analityczne
function enableAnalytics() {
    // Przykład: Google Analytics
    window.dataLayer = window.dataLayer || [];
    function gtag(){dataLayer.push(arguments);}
    gtag('js', new Date());
    gtag('config', 'G-XXXXXXXXXX', { 'anonymize_ip':
true });
}

```

```

// Dodanie skryptu Google Analytics
const script = document.createElement('script');
script.async = true;
script.src = 'https://www.googletagmanager.com/gtag/js?id=G-XXXXXXXXXX';
document.head.appendChild(script);
}

// Funkcja dezaktywująca skrypty analityczne
function disableAnalytics() {
    // Usunięcie skryptów Google Analytics
    window['ga-disable-G-XXXXXXXXXX'] = true;
}

// Funkcja aktywująca skrypty marketingowe
function enableMarketing() {
    // Przykład: Facebook Pixel
    !function(f,b,e,v,n,t,s)
    {if(f.fbq)return;n=f.fbq=function(){n.callMethod?
    n.callMethod.apply(n,arguments):n.queue.push(arguments)};
    if(!f._fbq)f._fbq=n;n.push=n;n.loaded=!0;n.version='2.0';
    n.queue=[];t=b.createElement(e);t.async=!0;
    t.src=v;s=b.getElementsByTagName(e)[0];
    s.parentNode.insertBefore(t,s)}(window, document,
'script',
'https://connect.facebook.net/en_US/fbevents.js')
;
    fbq('init', 'XXXXXXXXXXXX');
    fbq('track', 'PageView');
}

// Funkcja dezaktywująca skrypty marketingowe
function disableMarketing() {
    // Dezaktywacja Facebook Pixel
    window.fbq = function() {};

```



```
}  
});
```

Dobre praktyki dla informacji o plikach cookie: - Zamieść wyraźny banner z informacją o plikach cookie przy pierwszej wizycie - Umożliw użytkownikom wybór rodzajów plików cookie, które akceptują - Nie blokuj dostępu do strony dla użytkowników, którzy nie zaakceptowali cookie - Przechowuj udzielone zgody w bezpieczny sposób - Zapewnij możliwość zmiany preferencji w dowolnym momencie - Używaj prostego, zrozumiałego języka wyjaśniającego, czym są pliki cookie - Aktualizuj politykę cookie przy wprowadzaniu nowych technologii śledzących - Zapewnij zgodność z lokalnymi przepisami (RODO w UE, CCPA w Kalifornii) - Testuj banner na różnych urządzeniach i przeglądarkach

Dlaczego jest ważna dla początkujących: - Jest wymogiem prawnym w wielu krajach, szczególnie w UE - Buduje zaufanie użytkowników poprzez transparentność - Chroni przed potencjalnymi karami za nieprzestrzeganie przepisów - Stanowi część dobrej praktyki w zakresie ochrony prywatności użytkowników

DOSTĘPNOŚĆ (WCAG)

Dostępność (Web Content Accessibility Guidelines - WCAG) to zestaw wytycznych dotyczących tworzenia stron internetowych dostępnych dla wszystkich użytkowników, w tym osób z niepełnosprawnościami, zapewniających równy dostęp do informacji i funkcjonalności.

Kluczowe cechy: - Zapewnia dostępność dla osób z różnymi niepełnosprawnościami - Poprawia ogólną użyteczność strony dla wszystkich użytkowników - Zwiększa zgodność z wymogami prawnymi i standardami - Dociera do szerszej grupy odbiorców - Poprawia SEO i wydajność strony

Główne zasady WCAG:

1. **Percepcja** - informacje i komponenty interfejsu użytkownika muszą być przedstawiane w sposób dostępny dla zmysłów użytkowników.

1. Percepcja

1.1. Alternatywy tekstowe

Zapewnij alternatywy tekstowe dla treści nietekstowych.

- Wszystkie obrazy muszą posiadać atrybut alt.
- Materiały wideo powinny mieć napisy.
- Elementy dekoracyjne powinny mieć pusty alt lub być zaimplementowane jako tło CSS.

Przykład:

```
```html
<!-- Dobra praktyka -->

<!-- Obraz dekoracyjny -->

<!-- Złożony wykres -->

<p>Opis wykresu: Sprzedaż wzrosła o 15% w 2021, 22% w 2022...</p>
```

## 1.2. MEDIA ZALEŻNE OD CZASU

Zapewnij alternatywy dla mediów zależnych od czasu.

- Filmy powinny mieć napisy dla osób niesłyszących.
- Materiały audio powinny mieć transkrypcje.
- Filmy zawierające istotne informacje wizualne powinny mieć audiodeskrypcję.

## 1.3. ADAPTOWALNOŚĆ

Twórz treści, które można prezentować na różne sposoby bez utraty informacji.

- Używaj semantycznego HTML (nagłówki, listy, tabele z odpowiednimi nagłówkami).
- Nie przekazuj informacji wyłącznie za pomocą koloru, kształtu czy lokalizacji.
- Zachowaj logiczną kolejność treści w kodzie HTML.

## 1.4. ROZRÓŻNIALNOŚĆ

Ułatw użytkownikom widzenie i słyszenie treści.

- Zapewnij wystarczający kontrast między tekstem a tłem (minimum 4.5:1 dla normalnego tekstu).
- Umożliw powiększanie tekstu do 200% bez utraty funkcjonalności.
- Nie używaj dźwięków, które automatycznie odtwarzają się dłużej niż 3 sekundy.
- Zapewnij alternatywne widoki dla treści przedstawionych jako tekst na obrazie.

2. **Funkcjonalność** - komponenty interfejsu i nawigacji muszą być możliwe do użycia.

## 2. Funkcjonalność

### 2.1. DOSTĘPNOŚĆ Z KLAWIATURY

Zapewnij dostępność wszystkich funkcji za pomocą klawiatury.

- Wszystkie interaktywne elementy muszą być dostępne za pomocą klawiatury.

- Nie twórz pułapek klawiaturowych (sytuacji, gdzie użytkownik nie może opuścić elementu za pomocą klawiatury).
- Zapewnij widoczny fokus dla elementów aktywowanych klawiaturą.

Przykład:

```
/* Widoczny fokus */
:focus {
 outline: 3px solid #4299e1;
}
```

```
/* Nigdy nie usuwaj całkowicie obrysu fokusu bez zapewnienia alternatywy */
:focus:not(:focus-visible) {
 outline: none;
 box-shadow: 0 0 0 3px rgba(66, 153, 225, 0.6);
}
```

## 2.2. WYSTARCZAJĄCA ILOŚĆ CZASU

Zapewnij użytkownikom wystarczającą ilość czasu na przeczytanie i skorzystanie z treści.

- Unikaj ograniczeń czasowych lub zapewnij możliwość ich wyłączenia.
- Umożliw zatrzymanie, wstrzymanie lub ukrycie elementów poruszających się automatycznie.
- Ostrzegaj przed automatowym wylogowaniem i pozwól na przedłużenie sesji.

## 2.3. ATAKI PADACZKI

Nie projektuj treści w sposób, który może powodować ataki padaczki.

- Unikaj szybko błyskających elementów (nie więcej niż trzy błyski w ciągu jednej sekundy).
- Zapewnij możliwość wyłączenia animacji i efektów wyświetlania.

## 2.4. NAWIGOWANIE

Zapewnij metody pomagające użytkownikom nawigować, znaleźć treści i określić, gdzie się znajdują.

- Używaj jasnych, opisowych tytułów stron.
- Implementuj linki pomijające powtarzające się bloki nawigacyjne.
- Zapewnij logiczną kolejność nawigacji za pomocą klawiatury.
- Zapewnij więcej niż jeden sposób zlokalizowania strony w serwisie (wyszukiwarka, mapa strony, nawigacja).
- Używaj opisowych tekstów linków (zamiast “kliknij tutaj” użyj “dowiedz się więcej o dostępności”).

## 2.5. METODY WPROWADZANIA DANYCH

Ułatw użytkownikom obsługę funkcji za pomocą różnych urządzeń wejściowych.

- Zapewnij alternatywę dla gestów złożonych i aktywacji przez ściskanie.
- Projektuj komponenty tak, aby można je było używać za pomocą dotyku (odpowiednia wielkość obszaru kliknięcia).
- Umożliw działanie za pomocą ruchu urządzenia (np. potrząsania), ale zapewnij też alternatywę.

3. **\*\*Zrozumiałość\*\*** - informacje i obsługa interfejsu użytkownika muszą być zrozumiałe.

## 3. Zrozumiałość

### 3.1. CZYTELNOŚĆ

Uczyń treść tekstu możliwą do odczytania i zrozumienia.

- Określ język strony za pomocą atrybutu lang w HTML.
- Oznaczaj fragmenty w innym języku.
- Wyjaśniaj skomplikowane słowa i żargon.
- Zapewnij wymowę słów, które mogą być niejasne bez znajomości wymowy.

Przykład:

```
<html lang="pl">
 <head>...</head>
 <body>
 <p>To jest tekst po polsku.</p>
 <p>Fragment w innym języku: This
is in English</p>
 </body>
</html>
```

### 3.2. PRZEWIDYWALNOŚĆ

Spraw, by strony internetowe wyświetlały się i działały w przewidywalny sposób.

- Nie inicjuj zmian kontekstu bez świadomej akcji użytkownika.
- Zachowaj spójną nawigację w obrębie serwisu.
- Identyfikuj elementy o tej samej funkcjonalności w spójny sposób.
- Nie zmieniaj ustawień formularza automatycznie przy zmianie pola (np. automatyczne przesyłanie formularza).

### 3.3. POMOC PRZY WPROWADZANIU INFORMACJI

Pomagaj użytkownikom unikać błędów i je poprawiać.

- Identyfikuj błędy wprowadzania danych i opisuj je tekstowo.
- Zapewnij etykiety i instrukcje dla pól formularzy.
- Sugeruj poprawki w przypadku wykrycia błędów.
- Oferuj możliwość weryfikacji, potwierdzenia i korekty informacji przed finalnym zatwierdzeniem, szczególnie w przypadku nieodwracalnych transakcji.

4. **\*\*Solidność\*\*** - treść musi być wystarczająco solidna, aby mogła być interpretowana przez różne przeglądarki, w tym technologie wspomagające.

## 4. Solidność

### 4.1. KOMPATYBILNOŚĆ

Maksymalizuj kompatybilność z obecnymi i przyszłymi programami użytkowników, w tym z technologiami wspomagającymi.

- Stosuj poprawny HTML (dobrze sformowany kod zgodny ze standardami).
- Zapewnij pełne nazwy, role i wartości dla niestandardowych komponentów interfejsu.
- Używaj ARIA (Accessible Rich Internet Applications) dla dynamicznych treści i zaawansowanych kontrolek interfejsu.

Przykład użycia atrybutów ARIA:

```
<button aria-expanded="false" aria-controls="menu-list">
 Menu
</button>
<ul id="menu-list" aria-labelledby="menu-button" role
```

```

="menu" hidden>
<li role="menuitem">Home
<li role="menuitem">About
<li role="menuitem">Contact<
/li>


```

**\*\*Przykłady wdrażania dostępności:\*\***

1. **\*\*Formularze dostępne:\*\***

```

``html
<!-- Formularz dostępny -->
<form>
 <div class="form-group">
 <label for="name">Imię i nazwisko</label>
 <input type="text" id="name" name="name" required
 aria-required="true">
 <div id="name-error" class="error-message" role="
 alert" aria-live="assertive"></div>
 </div>

 <div class="form-group">
 <label for="email">Adres e-mail</label>
 <input type="email" id="email" name="email" requi
 red aria-required="true"
 aria-describedby="email-hint">
 <div id="email-hint" class="hint">Format: nazwa@d
 omena.pl</div>
 <div id="email-error" class="error-message" role=
 "alert" aria-live="assertive"></div>
 </div>

 <fieldset>
 <legend>Preferowany kontakt</legend>

 <div class="form-check">
 <input type="radio" id="contact-email" name="co
 ntact-pref" value="email">

```

```

 <label for="contact-email">E-mail</label>
 </div>

 <div class="form-check">
 <input type="radio" id="contact-phone" name="contact-pref" value="phone">
 <label for="contact-phone">Telefon</label>
 </div>
</fieldset>

<div class="form-group">
 <label for="message">Wiadomość</label>
 <textarea id="message" name="message" rows="5" aria-required="true" required></textarea>
</div>

<button type="submit">Wyślij</button>
</form>

```

## 2. Tabele dostępne:

```

<!-- Tabela dostępna -->
<table>
 <caption>Zestawienie przychodów w latach 2022-2024<
/caption>
 <thead>
 <tr>
 <th scope="col">Kwartał</th>
 <th scope="col">2022</th>
 <th scope="col">2023</th>
 <th scope="col">2024</th>
 </tr>
 </thead>
 <tbody>
 <tr>
 <th scope="row">Q1</th>
 <td>10 000 zł</td>
 <td>12 000 zł</td>
 <td>15 000 zł</td>
 </tr>
 </tbody>

```



```

<tr>
 <th scope="row">Q2</th>
 <td>11 500 zł</td>
 <td>13 500 zł</td>
 <td>16 200 zł</td>
</tr>
<tr>
 <th scope="row">Q3</th>
 <td>9 800 zł</td>
 <td>14 200 zł</td>
 <td>17 500 zł</td>
</tr>
<tr>
 <th scope="row">Q4</th>
 <td>12 500 zł</td>
 <td>15 800 zł</td>
 <td>19 200 zł</td>
</tr>
</tbody>
<tfoot>
 <tr>
 <th scope="row">Suma</th>
 <td>43 800 zł</td>
 <td>55 500 zł</td>
 <td>67 900 zł</td>
 </tr>
</tfoot>
</table>

```

### 3. Dostępne menu mobilne (hamburger):

```

<header>
 <div class="logo">

 </div>

 <nav>

```

```

<button class="menu-toggle" aria-expanded="false"
aria-controls="main-menu">
 Menu

</button>

<ul id="main-menu" class="nav-menu" hidden>
 Strona główna

 <button class="submenu-toggle" aria-expanded=
"false" aria-controls="products-menu">
 Produkty
 <span class="dropdown-icon" aria-hidden="tr
ue">▼
 </button>
 <ul id="products-menu" class="submenu" hidden
>
 Kategoria
1
 Kategoria
2
 Kategoria
3

 O nas
 Kontakt

</nav>
</header>

<script>
 // JavaScript dla dostępnego menu
 document.addEventListener('DOMContentLoaded', funct
ion() {
 const menuToggle = document.querySelector('.menu-
toggle');

```

```

const mainMenu = document.getElementById('main-menu');
const submenuToggles = document.querySelectorAll(
 '.submenu-toggle');

// Obsługa przycisku hamburger menu
menuToggle.addEventListener('click', function() {
 const expanded = this.getAttribute('aria-expanded') === 'true';
 this.setAttribute('aria-expanded', !expanded);
 mainMenu.hidden = expanded;
});

// Obsługa podmenu
submenuToggles.forEach(toggle => {
 toggle.addEventListener('click', function() {
 const expanded = this.getAttribute('aria-expanded') === 'true';
 this.setAttribute('aria-expanded', !expanded);

 const submenuId = this.getAttribute('aria-controls');
 const submenu = document.getElementById(submenuId);
 submenu.hidden = expanded;
 });
});

// Obsługa klawisza Escape
document.addEventListener('keyup', function(e) {
 if (e.key === 'Escape') {
 // Zamknij menu główne, jeśli otwarte
 if (menuToggle.getAttribute('aria-expanded') === 'true') {
 menuToggle.click();
 }

 // Zamknij wszystkie podmenu

```

```
submenuToggles.forEach(toggle => {
 if (toggle.getAttribute('aria-expanded') ==
 = 'true') {
 toggle.click();
 }
});
});
});
});
</script>
```

**Testowanie dostępności:** - Weryfikacja z użyciem narzędzi, takich jak Lighthouse, WAVE, Axe - Przegląd kodu HTML pod kątem poprawnego semantycznego znacznika - Sprawdzanie kontrastu kolorów - Testowanie nawigacji klawiaturowej - Testowanie z czytnikami ekranowymi (np. NVDA, JAWS, VoiceOver) - Przeprowadzanie testów użyteczności z osobami z niepełnosprawnościami

**Dobre praktyki dla dostępności:** - Używaj semantycznego HTML zgodnie z jego przeznaczeniem (nagłówki, listy, przyciski) - Zapewnij wystarczający kontrast między tekstem a tłem - Upewnij się, że wszystkie funkcje są dostępne za pomocą klawiatury - Dodawaj alternatywne teksty dla obrazów i innych treści nietekstowych - Używaj etykiet dla pól formularzy i informuj o błędach - Zapewnij widoczny fokus dla elementów interaktywnych - Unikaj polegania wyłącznie na kolorze do przekazywania informacji - Testuj stronę z różnymi technologiami wspomagającymi - Wdrażaj ARIA tylko tam, gdzie jest to niezbędne - Projektuj responsywnie z myślą o różnych urządzeniach i preferencjach użytkownika

**Dlaczego jest ważna dla początkujących:** - Zwiększa zasięg strony, docierając do większej liczby użytkowników - Jest wymagana prawnie w wielu krajach i dla wielu organizacji - Poprawia SEO i ogólną użyteczność strony - Stanowi fundament etycznego i inkluzywnego projektowania - Buduje pozytywny wizerunek marki/firmy

# 10. Optymalizacja

## SEO (SEARCH ENGINE OPTIMIZATION)

SEO to zestaw praktyk mających na celu poprawę widoczności strony w wynikach wyszukiwania poprzez optymalizację jej treści, struktury i powiązań, co prowadzi do wyższych pozycji w wynikach wyszukiwarek i zwiększonego ruchu organicznego.

### Kluczowe elementy SEO:

#### 1. Podstawowe elementy techniczne:

##### ## 1. Podstawy techniczne SEO

##### ### 1.1. Meta tagi

Najważniejsze meta tagi dla SEO:

```
```html
<head>
  <!-- Tytuł strony (50-60 znaków) -->
  <title>Opisowy i zawierający słowa kluczowe tytuł s
trony | Nazwa firmy</title>

  <!-- Meta opis (150-160 znaków) -->
  <meta name="description" content="Zwiążły i przekon
ujący opis zawartości strony, zachęcający do kliknięc
ia w wynikach wyszukiwania.">

  <!-- Znacznik językowy -->
  <meta http-equiv="content-language" content="pl">

  <!-- Znacznik viewport dla responsywności -->
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">

  <!-- Kontrola indeksowania i śledzenia linków -->
  <meta name="robots" content="index, follow">

  <!-- Link kanoniczny -->
```

```
<link rel="canonical" href="https://przyklad.pl/strona">
</head>
```

1.2. STRUKTURA URL

Optymalne URL-e powinny być: - Krótkie i opisowe - Zawierające słowa kluczowe - Wykorzystujące myślniki zamiast podkreślników - Unikające parametrów i specjalnych znaków

Przykład dobrego URL:

<https://przyklad.pl/kategoria/nazwa-artykułu>

1.3. STRUKTURA NAGŁÓWKÓW

Hierarchia nagłówków powinna odzwierciedlać strukturę treści: - Używaj tylko jednego nagłówka H1 na stronie (zazwyczaj tytuł strony/artykułu) - Stosuj nagłówki H2 dla głównych sekcji - Używaj nagłówków H3-H6 dla podsekcji - Zawieraj słowa kluczowe w nagłówkach

Przykład:

```
<h1>Najlepsze praktyki SEO dla początkujących</h1>
```

```
<h2>Optymalizacja treści</h2>
```

```
<p>Treść związana z optymalizacją treści...</p>
```

```
<h2>Optymalizacja techniczna</h2>
```

```
<h3>Meta tagi</h3>
```

```
<p>Informacje o meta tagach...</p>
```

```
<h3>Struktura URL</h3>
```

```
<p>Wskazówki dotyczące struktury URL...</p>
```

1.4. XML SITEMAP

Mapa witryny w formacie XML pomagająca wyszukiwarkom w indeksowaniu stron:

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitema
p/0.9">
  <url>
    <loc>https://przyklad.pl/</loc>
    <lastmod>2025-04-01</lastmod>
    <changefreq>weekly</changefreq>
    <priority>1.0</priority>
  </url>
  <url>
    <loc>https://przyklad.pl/o-nas</loc>
    <lastmod>2025-03-15</lastmod>
    <changefreq>monthly</changefreq>
    <priority>0.8</priority>
  </url>
  <!-- Więcej URL-i -->
</urlset>
```

1.5. PLIK ROBOTS.TXT

Plik informujący roboty wyszukiwarek, które części witryny można indeksować:

```
User-agent: *
Disallow: /admin/
Disallow: /prywatne/
Allow: /
```

```
Sitemap: https://przyklad.pl/sitemap.xml
```

2. **Optymalizacja treści:**

2. Optymalizacja treści

2.1. BADANIE SŁÓW KLUCZOWYCH

Podstawa strategii SEO: - Identyfikacja słów i fraz, których używają potencjalni odbiorcy - Analiza wolumenu wyszukiwań i konkurencji - Uwzględnienie słów kluczowych o długim ogonie (long-tail) - Grupowanie słów kluczowych według intencji użytkownika - Regularne aktualizowanie badań

2.2. TWORZENIE TREŚCI PRZYJAZNEJ DLA SEO

Zasady tworzenia treści: - Wysokiej jakości, oryginalna i wartościowa treść - Odpowiednia długość (min. 300 słów, najlepiej 1000+ dla ważnych stron) - Naturalne umieszczanie słów kluczowych (bez przesycenia) - Użycie głównego słowa kluczowego w: - Tytule strony (H1) - Pierwszym akapicie - Co najmniej jednym nagłówku H2 - Meta opisie - Używanie synonimów i powiązanych fraz - Grupowanie treści w logiczne sekcje z nagłówkami - Aktualne i regularnie aktualizowane informacje

2.3. PRZYJAZNE DLA UŻYTKOWNIKA I SEO FORMATOWANIE

- Krótkie akapity (2-3 zdania)
- Użycie list wypunktowanych i numerowanych
- Wyróżnianie ważnych fragmentów bold/kursywą
- Dodawanie multimediów (obrazy, wideo, infografiki)
- Dodawanie alt tagów do obrazów ze słowami kluczowymi
- Stosowanie odpowiedniego interliniowania i odstępów
- Przejrzysta typografia

2.4. LINKOWANIE WEWNĘTRZNE

- Tworzenie logicznej struktury linków wewnętrznych
- Łączenie powiązanych treści

- Używanie opisowych tekstów linków (anchor text)
- Tworzenie stron kategorii i archiwów
- Linkowanie do ważnych stron z menu nawigacyjnego
- Używanie tagów do grupowania treści

2.5. REGULARNE AKTUALIZACJE

- Aktualizacja istniejących treści nowymi informacjami
- Poprawa i rozszerzanie popularnych artykułów
- Usuwanie lub przekierowanie nieaktualnych treści
- Dodawanie nowych treści zgodnie z kalendarzem wydawniczym

3. **Optymalizacja techniczna:**

3. Optymalizacja techniczna

3.1. SZYBKOŚĆ ŁADOWANIA STRONY

Kluczowe czynniki dla szybkości: - Optymalizacja obrazów (rozmiar, format) - Minimalizacja CSS i JavaScript - Eliminacja blokującego renderowanie JS i CSS - Wykorzystanie cachowania przeglądarki - Kompresja GZIP/Brotli - Wykorzystanie CDN (Content Delivery Network) - Optymalizacja bazy danych - Wykorzystanie lazy loading dla obrazów i wideo

Narzędzia do testowania: - Google PageSpeed Insights - GTmetrix - WebPageTest

3.2. RESPONSYWNY DESIGN

- Strona dostosowana do wszystkich urządzeń
- Mobile-first approach (podejście mobile-first)
- Testy na różnych urządzeniach i przeglądarkach
- Odpowiedni rozmiar przycisków i elementów interaktywnych

- Czytelne czcionki na małych ekranach

3.3. BEZPIECZEŃSTWO (HTTPS)

- Certyfikat SSL dla całej witryny
- Przekierowanie HTTP na HTTPS
- Zabezpieczenie przed atakami typu injection
- Regularne aktualizacje CMS i wtyczek
- Silne hasła i dwuskładnikowe uwierzytelnianie

3.4. DANE STRUKTURALNE (SCHEMA.ORG)

Przykładowy kod JSON-LD dla strony firmowej:

```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@type": "LocalBusiness",
  "name": "Nazwa Twojej Firmy",
  "image": "https://przyklad.pl/logo.jpg",
  "address": {
    "@type": "PostalAddress",
    "streetAddress": "ul. Przykładowa 123",
    "addressLocality": "Warszawa",
    "postalCode": "00-001",
    "addressCountry": "PL"
  },
  "telephone": "+48123456789",
  "email": "kontakt@przyklad.pl",
  "url": "https://przyklad.pl",
  "openingHoursSpecification": {
    "@type": "OpeningHoursSpecification",
    "dayOfWeek": [
      "Monday",
      "Tuesday",
      "Wednesday",
      "Thursday",
      "Friday"
    ]
  }
},
```

```
"opens": "09:00",  
"closes": "17:00"
```

```
}  
}
```

```
</script>
```

3.5. OPTIMALIZACJA DLA CORE WEB VITALS

- LCP (Largest Contentful Paint) - czas ładowania największego elementu
- FID (First Input Delay) - czas do pierwszej interakcji
- CLS (Cumulative Layout Shift) - stabilność układu wizualnego

Rozwiązania: - Optymalizacja LCP: szybkie ładowanie głównych obrazów i treści - Optymalizacja FID: minimalizacja blokującego JS, dzielenie kodu - Optymalizacja CLS: określanie rozmiarów obrazów/wideo, rezerwowanie miejsca dla dynamicznych elementów

4. **Budowa linków i autorytetu:**

4. Budowa linków i autorytetu

4.1. NATURALNE POZYSKIWANIE LINKÓW

Strategie pozyskiwania jakościowych linków: - Tworzenie wartościowych, unikalnych treści (linkbait) - Tworzenie narzędzi online i kalkulatorów - Publikowanie badań i raportów branżowych - Tworzenie kompleksowych poradników - Infografiki i materiały wizualne - Wywiady z ekspertami branżowymi

4.2. MIEJSCA DO POZYSKIWANIA LINKÓW

- Katalogi branżowe
- Strony partnerów biznesowych
- Lokalne katalogi firm

- Profil firmy w Google Business Profile
- Media społecznościowe
- Fora branżowe i Q&A (np. Quora)
- Strony organizacji branżowych

4.3. BUDOWANIE AUTORYTETU (E-A-T)

E-A-T (Expertise, Authoritativeness, Trustworthiness): - Tworzenie szczegółowych stron “O nas” z informacjami o firmie - Publikowanie biogramów autorów z kredencjami - Zdobywanie recenzji i opinii klientów - Prezentowanie certyfikatów i nagród - Informacje kontaktowe i polityki prywatności - Aktywna obecność w mediach społecznościowych - Publikacje w uznanych mediach branżowych

4.4. MONITOROWANIE PROFILU LINKÓW

- Regularne sprawdzanie linków przychodzących
- Identyfikacja i disavowal (odrzuć) szkodliwych linków
- Analiza linków konkurencji
- Śledzenie utraconych linków i ich odzyskiwanie

Narzędzia: - Google Search Console - Ahrefs - SEMrush - Majestic

5. ****Monitorowanie i analityka:****

5. Monitorowanie i analityka

5.1. KONFIGURACJA PODSTAWOWYCH NARZĘDZI

Niezbędne narzędzia: - Google Search Console - monitorowanie indeksacji i ruchu z wyszukiwarki - Google Analytics - analiza zachowań użytkowników - Narzędzia do monitorowania pozycji słów kluczowych

5.2. KLUCZOWE METRYKI DO ŚLEDZENIA

- Ruch organiczny
- CTR (Click-Through Rate) w wynikach wyszukiwania
- Pozycje słów kluczowych
- Współczynnik odrzuceń (Bounce Rate)
- Czas spędzony na stronie
- Głębokość wizyt (liczba odwiedzonych stron)
- Wskaźnik konwersji
- Szybkość ładowania strony

5.3. REGULARNE AUDYTY SEO

Co sprawdzać podczas audytu: - Błędy indeksowania - Powolne strony - Duplikaty treści - Brakujące meta tagi - Problemy z dostępnością mobilną - Błędy 404 - Stare/zdezaktualizowane treści - Brakujące alt tagi obrazów

5.4. DOSTOSOWYWANIE STRATEGII

- Identyfikacja najlepiej działających treści
- Analiza trendów w słowach kluczowych
- Optymalizacja stron z niską wydajnością
- Dostosowywanie strategii do zmian w algorytmach wyszukiwarek
- A/B testy elementów strony wpływających na SEO

****SEO dla różnych typów stron:****

6. SEO dla różnych typów stron

6.1. SKLEPY INTERNETOWE

- Optymalizacja stron kategorii i produktów
- Unikalne opisy produktów
- Stosowanie struktury breadcrumbs

- Schema.org dla produktów i recenzji
- Zarządzanie duplikatami treści
- Zarządzanie dostępnością produktów (out of stock)
- Optymalizacja procesów filtrowania i wyszukiwania

6.2. BLOGI

- Regularne publikacje nowych treści
- Odpowiednia kategoryzacja
- Wewnętrzne linkowanie pomiędzy powiązаныmi artykułami
- Schema.org dla artykułów
- Optymalizacja dla featured snippets
- Tworzenie evergreen content (zawsze aktualnych treści)

6.3. STRONY LOKALNE

- Optymalizacja Google Business Profile
- Spójne NAP (Nazwa, Adres, Telefon) na wszystkich platformach
- Lokalne słowa kluczowe
- Pozyskiwanie recenzji klientów
- Lokalne backlinki
- Schema.org dla firm lokalnych
- Optymalizacja pod wyszukiwania “near me”

6.4. STRONY FIRMOWE

- Przejrzysta struktura strony
- Szczegółowe informacje o usługach/produktach
- Studia przypadków i portfolio
- Strony eksperckie i poradniki
- Referencje i opinie klientów
- FAQ odpowiadające na popularne pytania

****Dobre praktyki SEO:****

- Twórz treści przede wszystkim dla użytkowników, nie dla wyszukiwarek
- Unikaj technik "black hat" (przesycanie słowami kluczowymi, ukryte teksty, etc.)
- Regularnie monitoruj i aktualizuj swoją stronę
- Dostosowuj się do zmian algorytmów wyszukiwarek
- Optymalizuj pod kątem wyszukiwania mobilnego i głosowego
- Stawiaj na długofalową strategię zamiast szybkich rozwiązań
- Koncentruj się na szybkości, użyteczności i wartościowych treściach
- Testuj różne podejścia i stale uczysz się na podstawie wyników
- Analizuj działania konkurencji, ale unikaj kopiowania
- Zachowuj równowagę między optymalizacją a użytecznością

****Dlaczego SEO jest ważne dla początkujących:****

- Zwiększa organiczną widoczność strony w wyszukiwarkach
- Przyciąga ukierunkowany ruch bez płatnej reklamy
- Buduje długoterminową wartość i stabilny ruch
- Zwiększa wiarygodność i autorytet w danej branży
- Poprawia doświadczenie użytkownika i użyteczność strony
- Pozwala konkurować z większymi firmami przy mniejszym budżecie
- Stanowi fundament innych działań marketingowych online

Szybkość ładowania

Szybkość ładowania to kluczowy element wydajności strony, wpływający na doświadczenie użytkownika, współczynniki konwersji i pozycje w wynikach wyszukiwania.

****Kluczowe czynniki:****

- Wpływa na zadowolenie użytkowników i współczynniki

konwersji

- Jest ważnym czynnikiem rankingowym dla wyszukiwarek
- Szczególnie istotna na urządzeniach mobilnych
- Wpływa na Core Web Vitals i ogólną wydajność strony
- Redukuje współczynnik odrzuceń

****Główne obszary optymalizacji:****

1. ****Optymalizacja serwera:****

1. Optymalizacja serwera

1.1. HOSTING

Wybór odpowiedniego hostingu: - Serwer dedykowany lub VPS dla większych stron - Hosting współdzielony tylko dla małych stron z niewielkim ruchem - Serwery z SSD zamiast HDD - Lokalizacja serwera bliska docelowym użytkownikom - Wystarczające zasoby (RAM, CPU, przepustowość)

1.2. KONFIGURACJA SERWERA

- Włączenie kompresji (GZIP, Brotli)
- Włączenie cachowania HTTP
- Optymalizacja bazy danych
- Aktualizacja oprogramowania serwera
- Wdrożenie HTTP/2 lub HTTP/3
- Wykorzystanie CDN (Content Delivery Network)

1.3. PRZYKŁADOWA KONFIGURACJA APACHE (.HTACCESS)

Włączenie kompresji GZIP

```
<IfModule mod_deflate.c>
```

```
    AddOutputFilterByType DEFLATE text/html text/plain  
    text/xml text/css application/javascript application/  
    json  
</IfModule>
```

Cache-Control


```

<IfModule mod_expires.c>
    ExpiresActive On
    ExpiresByType image/jpg "access plus 1 year"
    ExpiresByType image/jpeg "access plus 1 year"
    ExpiresByType image/gif "access plus 1 year"
    ExpiresByType image/png "access plus 1 year"
    ExpiresByType image/webp "access plus 1 year"
    ExpiresByType image/svg+xml "access plus 1 year"
    ExpiresByType text/css "access plus 1 month"
    ExpiresByType application/javascript "access plus 1
month"
    ExpiresByType text/html "access plus 1 day"
</IfModule>

```

1.4. PRZYKŁADOWA KONFIGURACJA NGINX

```

# Włączenie kompresji GZIP
gzip on;
gzip_types text/plain text/css application/json appli
cation/javascript text/xml application/xml applicatio
n/xml+rss text/javascript;
gzip_min_length 1000;
gzip_comp_level 5;

# Cache-Control
location ~* \.(jpg|jpeg|png|gif|ico|css|js|webp|svg)$
{
    expires 1y;
    add_header Cache-Control "public, max-age=3153600
0";
}

```

2. **Optymalizacja ładowania zasobów:**

2. Optymalizacja ładowania zasobów

2.1. MINIMALIZACJA ŻĄDAŃ HTTP

- Łączenie plików CSS i JavaScript
- Wykorzystanie CSS Sprites dla ikon

- Wbudowywanie mniejszych obrazów jako data URI
- Implementacja lazy loading dla obrazów i wideo

2.2. OPTIMALIZACJA CSS I JAVASCRIPT

- Minifikacja (usuwanie zbędnych spacji, komentarzy, itp.)
- Usuwanie nieużywanego kodu
- Priorytetyzacja ładowania krytycznego CSS
- Opóźnione ładowanie nieistotnego JavaScript
- Asynchroniczne ładowanie skryptów zewnętrznych

Przykład asynchronicznego ładowania:

```
<!-- Asynchroniczne ładowanie skryptu Google Analytic  
s -->  
<script async src="https://www.googletagmanager.com/g  
tag/js?id=G-XXXXXXX"></script>  
  
<!-- Opóźnione ładowanie skryptu (defer) -->  
<script defer src="non-critical.js"></script>
```

2.3. OPTIMALIZACJA OBRAZÓW

- Wybór odpowiedniego formatu (WebP, AVIF, JPEG, PNG)
- Zmniejszenie rozmiaru pliku bez utraty jakości
- Responsywne obrazy (srcset, rozmiary)
- Leniwe ładowanie obrazów
- Określanie wymiarów obrazów w HTML
- Używanie CDN dla obrazów

Przykład responsywnych obrazów:

```
<img  
  srcset="  
    image-320w.jpg 320w,  
    image-480w.jpg 480w,  
    image-800w.jpg 800w  
  ">
```

```
sizes="(max-width: 480px) 100vw,  
(max-width: 960px) 50vw,  
800px"  
src="image-800w.jpg"  
alt="Opis obrazu"  
loading="lazy"  
width="800"  
height="600"
```

>

2.4. WYKORZYSTANIE CDN

Zalety Content Delivery Network: - Przechowywanie kopii statycznych zasobów na globalnej sieci serwerów - Zmniejszenie odległości między serwerem a użytkownikiem - Rozłożenie obciążenia serwera - Dodatkowa warstwa cachowania - Ochrona przed atakami DDoS

Popularne CDN: - Cloudflare - Amazon CloudFront - Akamai - Google Cloud CDN - Fastly

3. **Monitorowanie i testowanie wydajności**

3. Monitorowanie i testowanie wydajności

3.1. NARZĘDZIA DO TESTOWANIA

Podstawowe narzędzia: - Google PageSpeed Insights - Lighthouse (wbudowany w Chrome DevTools) - WebPageTest - GTmetrix - Pingdom - Chrome DevTools (zakładka Performance) - Core Web Vitals report (Google Search Console)

3.2. KLUCZOWE METRYKI WYDAJNOŚCIOWE

Core Web Vitals: - LCP (Largest Contentful Paint) - czas ładowania największego elementu widocznego, cel < 2.5s - FID (First Input

Delay) - czas do pierwszej interakcji, cel < 100ms - CLS (Cumulative Layout Shift) - stabilność układu wizualnego, cel < 0.1

Dodatkowe metryki: - TTFB (Time to First Byte) - czas do pierwszego bajtu, cel < 200ms - FCP (First Contentful Paint) - pierwszy widoczny element, cel < 1.8s - TTI (Time to Interactive) - czas do interaktywności, cel < 3.8s - Total Blocking Time (TBT) - całkowity czas blokowania - Total Page Size - całkowity rozmiar strony, cel < 2MB

3.3. WATERFALL ANALIZA

- Identyfikacja wąskich gardeł za pomocą wykresu kaskadowego
- Sprawdzanie kolejności ładowania zasobów
- Identyfikacja zasobów blokujących renderowanie
- Sprawdzanie długości czasu odpowiedzi serwera

3.4. CIĄGŁE MONITOROWANIE

- Regularne testy wydajności (co najmniej raz w miesiącu)
- Automatyczne testy przy wdrożeniach (CI/CD)
- Monitorowanie w czasie rzeczywistym (Real User Monitoring)
- Śledzenie zmian w wydajności po aktualizacjach

4. ****Optymalizacja kodu frontendowego:****

4. Optymalizacja kodu frontendowego

4.1. OPTYMALIZACJA HTML

- Używaj semantycznego HTML5
- Utrzymuj płaską strukturę DOM
- Usuń zbędne znaczniki i komentarze
- Używaj lazy loading dla obrazów i iframe
- Unikaj nadmiaru zagnieżdżonych elementów

4.2. OPTYMALIZACJA CSS

- Użyj CSS Grid i Flexbox zamiast float
- Unikaj selektorów uniwersalnych i głęboko zagnieżdżonych
- Używaj kaskady CSS efektywnie
- Rozważ CSS-in-JS lub atomic CSS dla dużych aplikacji
- Optymalizuj animacje (używaj transform i opacity)
- Zastosuj Critical CSS (inline dla “above the fold”)

4.3. OPTYMALIZACJA JAVASCRIPT

- Używaj nowoczesnego JavaScript (ES6+)
- Implementuj code-splitting dla dużych aplikacji
- Unikaj blokującego renderowanie JS
- Używaj Event Delegation zamiast wielu listenerów
- Optymalizuj manipulacje DOM (unikaj reflow i repaint)
- Wykorzystaj Web Workers dla zadań intensywnych obliczeniowo
- Implementuj wirtualizację dla długich list

4.4. OPTYMALIZACJA RENDEROWANIA

- Minimalizuj reflow i repaint
- Używaj transform zamiast manipulacji położeniem
- Optymalizuj Critical Rendering Path
- Wdrażaj progresywne renderowanie
- Używaj will-change dla elementów z animacjami
- Unikaj zbyt wielu cieni, gradientów i efektów

****Dobre praktyki dla szybkości ładowania:****

- Priorytetyzuj treści widoczne “above the fold”
- Testuj stronę na różnych urządzeniach i prędkościach internetu
- Regularnie monitoruj i optymalizuj wydajność
- Implementuj progresywne ulepszanie (progressive enhancement)

- Usuń nieużywane zasoby i kod
- Utrzymuj równowagę między estetyką a wydajnością
- Wykorzystuj preloading dla kluczowych zasobów
- Stosuj techniki SPA (Single Page Application) tylko gdy konieczne
- Optymalizuj obrazy przed uploadem na serwer
- Ustalaj budżety wydajnościowe dla każdej strony

****Dlaczego jest ważna dla początkujących:****

- Bezpośrednio wpływa na wrażenia użytkowników
- Jest kluczowym czynnikiem dla SEO
- Zwiększa współczynniki konwersji i ogranicza porzucenia
- Poprawia dostępność na urządzeniach mobilnych i wolnych łączach
- Zapewnia przewagę konkurencyjną
- Stanowi fundament dobrego UX (User Experience)

Bezpieczeństwo

Bezpieczeństwo strony internetowej to zespół praktyk mających na celu ochronę witryny, danych użytkowników i infrastruktury przed zagrożeniami, zapewniając poufność, integralność i dostępność informacji.

****Kluczowe obszary bezpieczeństwa:****

1. ****Podstawowe zabezpieczenia:****

1. Podstawowe zabezpieczenia

1.1. PROTOKÓŁ HTTPS

Wdrażanie SSL/TLS: - Zainstalowanie certyfikatu SSL (Let's Encrypt, komercyjne certyfikaty) - Przekierowanie całego ruchu HTTP na HTTPS - Implementacja HSTS (HTTP Strict Transport Security) - Aktualizacja wewnętrznych linków na HTTPS - Używanie bezpiecznych cookie (flagi Secure i HttpOnly)

Przykład konfiguracji HSTS w Apache:

```
<IfModule mod_headers.c>
    Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
</IfModule>
```

Przykład konfiguracji HSTS w Nginx:

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains; preload" always;
```

1.2. BEZPIECZNE HASŁA I UWIERZYTELNIANIE

- Wymaganie silnych haseł (min. 12 znaków, małe i wielkie litery, cyfry, znaki specjalne)
- Bezpieczne przechowywanie haseł (bcrypt, Argon2)
- Implementacja uwierzytelniania dwuskładnikowego (2FA)
- Ograniczenie liczby prób logowania
- Bezpieczne procesy odzyskiwania hasła
- Regularny wymuszanie zmiany haseł dla kont administratorów

2. **Zabezpieczenia aplikacji webowej:**

2. Zabezpieczenia aplikacji webowej

2.1. OCHRONA PRZED TYPOWYMI ATAKAMI

- **XSS (Cross-Site Scripting)** - sanityzacja danych wprowadzanych przez użytkowników, escape danych wyjściowych
- **CSRF (Cross-Site Request Forgery)** - używanie tokenów CSRF w formularzach
- **SQL Injection** - używanie parametryzowanych zapytań, ORM
- **File Upload Attacks** - walidacja typów plików, skanowanie złośliwego oprogramowania

- **CORS (Cross-Origin Resource Sharing)** - prawidłowa konfiguracja nagłówków
- **Clickjacking** - implementacja nagłówków X-Frame-Options
- **MITM (Man-in-the-Middle)** - konsekwentne używanie HTTPS

2.2. IMPLEMENTACJA CSP (CONTENT SECURITY POLICY)

Przykład nagłówka CSP:

```
Content-Security-Policy: default-src 'self'; script-src 'self' https://trusted-cdn.com; style-src 'self' https://trusted-cdn.com; img-src 'self' data:; connect-src 'self' https://api.example.com; font-src 'self'; object-src 'none'; media-src 'self'; frame-src 'self';
```

2.3. BEZPIECZNA KONFIGURACJA COOKIE

Przykład bezpiecznego cookie:

```
Set-Cookie: session=123; Path=/; Secure; HttpOnly; SameSite=Strict
```

2.4. WALIDACJA DANYCH

- Sprawdzanie wszystkich danych wejściowych po stronie serwera
- Ograniczanie typów akceptowanych plików
- Filtrowanie specjalnych znaków z danych wprowadzanych przez użytkowników
- Używanie białych list zamiast czarnych dla zaakceptowanych wartości
- Walidacja po stronie klienta dla UX, ale nigdy jako jedyne zabezpieczenie

3. **Zarządzanie i monitoring:**

3. Zarządzanie i monitoring

3.1. REGULARNIE AKTUALIZACJE

- Szybkie wdrażanie poprawek bezpieczeństwa
- Aktualizacja CMS, wtyczek i bibliotek
- Monitorowanie biuletynów bezpieczeństwa
- Plan reakcji na zagrożenia bezpieczeństwa
- Testowanie aktualizacji przed wdrożeniem produkcyjnym

3.2. ZARZĄDZANIE UPRAWNIENIAMI

- Przestrzeganie zasady najmniejszych uprawnień
- Różne poziomy dostępu dla różnych ról użytkowników
- Okresowe przeglądy uprawnień
- Natychmiastowe usuwanie kont byłych pracowników
- Monitorowanie podejrzanych działań administratorów

3.3. AUDYTY I SKANOWANIE BEZPIECZEŃSTWA

- Regularne skanowanie podatności (narzędzia SAST i DAST)
- Okresowe testy penetracyjne
- Przegląd kodu pod kątem bezpieczeństwa
- Monitorowanie logów serwera i aplikacji
- Wdrożenie systemów wykrywania włamań (IDS/IPS)

3.4. ZAPASOWE KOPIE DANYCH

- Regularne automatyczne kopie zapasowe
- Przechowywanie kopii w różnych lokalizacjach
- Testowanie procesu przywracania danych
- Szyfrowanie kopii zapasowych
- Strategia przywracania danych po ataku ransomware

4. **Ochrona infrastruktury:**

4. Ochrona infrastruktury

4.1. BEZPIECZEŃSTWO SERWERA

- Aktualne oprogramowanie serwera
- Wyłączenie niepotrzebnych usług i portów
- Konfiguracja firewalla
- Bezpieczna konfiguracja SSH (klucze zamiast haseł, zmiana domyślnego portu)
- Regularne aktualizacje systemu operacyjnego
- Używanie kont z ograniczonymi uprawnieniami

4.2. OCHRONA PRZED ATAKAMI DDOS

- Implementacja filtrowania ruchu na poziomie sieci
- Użycie usług anti-DDoS (Cloudflare, Akamai, etc.)
- Monitorowanie wzorców ruchu
- Plan reakcji na ataki

4.3. KONFIGURACJA DNS

- Używanie DNSSEC dla zabezpieczenia DNS
- Implementacja SPF, DKIM i DMARC dla zabezpieczenia poczty
- Monitorowanie zmian DNS
- Bezpieczna konfiguracja rejestratora domen

4.4. ZABEZPIECZENIA BAZY DANYCH

- Ograniczenie dostępu z zewnątrz
- Używanie silnych haseł
- Regularne aktualizacje
- Minimalne uprawnienia dla kont aplikacji
- Szyfrowanie wrażliwych danych
- Audyty zapytań i dostępu

5. Zgodność z przepisami i standardy

5.1. REGULACJE DOTYCZĄCE OCHRONY DANYCH

- RODO (GDPR) w Unii Europejskiej
- CCPA w Kalifornii
- Lokalne przepisy o ochronie danych
- PCI DSS dla transakcji kartami płatniczymi
- HIPAA dla danych medycznych (USA)

5.2. STANDARDY BEZPIECZEŃSTWA

- OWASP Top 10 (najpopularniejsze zagrożenia aplikacji webowych)
- CIS Controls (Critical Security Controls)
- ISO 27001 (system zarządzania bezpieczeństwem informacji)
- NIST Cybersecurity Framework

5.3. DOKUMENTACJA BEZPIECZEŃSTWA

- Polityka bezpieczeństwa informacji
- Procedury reagowania na incydenty
- Plan ciągłości działania
- Dokumentacja zgodności z przepisami
- Rejestr czynności przetwarzania danych (RODO)

****Przykłady implementacji zabezpieczeń:****

1. Bezpieczny formularz kontaktowy:

```
```php
<?php
// Implementacja zabezpieczeń formularza

// 1. Walidacja i sanityzacja danych
$name = isset($_POST['name']) ? filter_var(trim($_POS
```

```

T['name']), FILTER_SANITIZE_STRING) : '';
$email = isset($_POST['email']) ? filter_var(trim($_POST['email']), FILTER_SANITIZE_EMAIL) : '';
$message = isset($_POST['message']) ? htmlspecialchars(trim($_POST['message'])) : '';

// 2. Walidacja email
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
 die("Nieprawidłowy adres email");
}

// 3. Sprawdzenie tokena CSRF
session_start();
if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
 die("Błąd weryfikacji formularza (CSRF)");
}

// 4. Ochrona przed spamem (prosty honeypot)
if (!empty($_POST['phone'])) { // Pole ukryte CSS - w
 // wypełnione tylko przez boty
 exit; // Cicha porażka - nie informuj bota
}

// 5. Ograniczenie liczby wysłanych formularzy
$ip = $_SERVER['REMOTE_ADDR'];
$timestamp = time();
$file = 'form_submissions.txt';

if (file_exists($file)) {
 $submissions = json_decode(file_get_contents($file), true);

 // Usunięcie starszych niż 1 godzina
 foreach ($submissions as $key => $submission) {
 if ($timestamp - $submission['time'] > 3600)
 {
 unset($submissions[$key]);
 }
 }
}

```

```

 }

 // Sprawdzenie liczby wysłań z danego IP
 $count = 0;
 foreach ($submissions as $submission) {
 if ($submission['ip'] === $ip) {
 $count++;
 }
 }

 if ($count >= 5) {
 die("Przekroczono limit wysłanych formularzy.
 Spróbuj ponownie później.");
 }
} else {
 $submissions = [];
}

// Dodanie nowego wysłania
$submissions[] = [
 'ip' => $ip,
 'time' => $timestamp
];
file_put_contents($file, json_encode($submissions));

// 6. Bezpieczne wysłanie e-maila
$to = 'admin@example.com';
$subject = 'Nowa wiadomość z formularza kontaktowego'
;
$email_message = "Nadawca: $name\nEmail: $email\n\nWiadomość: \n$message";
$headers = "From: webmaster@example.com\r\n" .
 "Reply-To: $email\r\n" .
 "X-Mailer: PHP/" . phpversion();

if (mail($to, $subject, $email_message, $headers)) {
 echo "Wiadomość została wysłana!";
} else {
 echo "Wystąpił błąd podczas wysyłania wiadomości.

```

```
"}
}
?>
```

```
<!-- HTML formularza z zabezpieczeniami -->
<form method="post" action="contact.php">
 <!-- Pole-pułapka dla botów (ukryte przez CSS) -->
 <input type="text" name="phone" style="display:none">

 <!-- Token CSRF -->
 <input type="hidden" name="csrf_token" value="<?php
echo $_SESSION['csrf_token']; ?>">

 <!-- Standardowe pola formularza -->
 <div class="form-group">
 <label for="name">Imię i nazwisko</label>
 <input type="text" id="name" name="name" required>
 </div>

 <div class="form-group">
 <label for="email">Adres e-mail</label>
 <input type="email" id="email" name="email" required>
 </div>

 <div class="form-group">
 <label for="message">Wiadomość</label>
 <textarea id="message" name="message" rows="5" required></textarea>
 </div>

 <!-- reCAPTCHA -->
 <div class="g-recaptcha" data-sitekey="YOUR_SITE_KEY"></div>
```

```
<button type="submit">Wyślij wiadomość</button>
</form>
```

## 2. Nagłówki zabezpieczające:

```
<?php
// Przykładowe nagłówki zabezpieczające PHP
header("Content-Security-Policy: default-src 'self';
script-src 'self' https://trusted-cdn.com; style-src
'self' https://trusted-cdn.com; img-src 'self' data;;
");
header("X-Frame-Options: DENY");
header("X-XSS-Protection: 1; mode=block");
header("X-Content-Type-Options: nosniff");
header("Referrer-Policy: strict-origin-when-cross-ori
gin");
header("Permissions-Policy: geolocation=(), microphon
e=(), camera=()");
header("Strict-Transport-Security: max-age=31536000;
includeSubDomains; preload");

// Ukrywanie informacji o serwerze
header_remove("X-Powered-By");
?>
```

## 3. Bezpieczne przechowywanie haseł:

```
<?php
// Bezpieczne hashowanie hasła przy rejestracji
function registerUser($username, $password, $email) {
 // Minimum 8 znaków, litery, cyfry i znaki specja
 lne
 if (!preg_match('/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)
(?=.*[!@#$%^&*?])[A-Za-z\d@!%*?&]{8,}$/', $password))
 {
 return "Hasło nie spełnia wymagań bezpieczeństwa";
 }

 // Generowanie bezpiecznego hashu hasła
 $hash = password_hash($password, PASSWORD_DEFAULT
```

```

);

// Zapisanie w bazie danych
$db = new PDO('mysql:host=localhost;dbname=mydb',
'username', 'password');
$stmt = $db->prepare("INSERT INTO users (username
, password_hash, email) VALUES (?, ?, ?)");
return $stmt->execute([$username, $hash, $email])
;
}

// Bezpieczne sprawdzanie hasła przy logowaniu
function loginUser($username, $password) {
 $db = new PDO('mysql:host=localhost;dbname=mydb',
 'username', 'password');

 // Pobieranie hashu hasła
 $stmt = $db->prepare("SELECT id, password_hash FR
OM users WHERE username = ?");
 $stmt->execute([$username]);
 $user = $stmt->fetch(PDO::FETCH_ASSOC);

 if ($user && password_verify($password, $user['pa
ssword_hash'])) {
 // Logowanie udane
 session_start();
 $_SESSION['user_id'] = $user['id'];
 return true;
 }

 // Logowanie nieudane
 return false;
}

```

**Dobre praktyki dla bezpieczeństwa:** - Traktuj bezpieczeństwo jako proces ciągły, nie jednorazowe zadanie - Stosuj zasadę najniższych uprawnień - Regularnie aktualizuj wszystkie komponenty strony - Implementuj mechanizmy kontroli dostępu - Weryfikuj i sanityzuj wszystkie dane wejściowe - Używaj



szyfrowania dla wrażliwych danych - Nie ufaj danym od użytkownika (waliduj po stronie serwera) - Implementuj wielowarstwową ochronę (defense in depth) - Edukuj zespół w zakresie bezpieczeństwa - Miej gotowy plan reakcji na incydenty bezpieczeństwa

**Dlaczego jest ważne dla początkujących:** - Chroni dane użytkowników i zaufanie do witryny - Zapobiega włamaniom i atakom - Może uchronić przed konsekwencjami prawnymi i finansowymi - Buduje wiarygodność marki/firmy - Zapewnia zgodność z regulacjami (np. RODO) - Stanowi podstawę dla innych aspektów strony (nie można mieć dobrego UX bez bezpieczeństwa)

## Podsumowanie

Tworzenie nowoczesnych stron internetowych to złożony proces wymagający zrozumienia wielu różnych aspektów — od kodu i struktury, przez elementy wizualne i interaktywne, aż po kwestie prawne, dostępność i bezpieczeństwo.

### KLUCZOWE ELEMENTY UDANEJ STRONY INTERNETOWEJ:

#### 1. **Solidna podstawa techniczna**

- Poprawny, semantyczny HTML
- Dobrze zorganizowany CSS
- Wydajny JavaScript
- Właściwa struktura plików

#### 2. **Przemyślany design i UX**

- Spójna kolorystyka i typografia
- Intuicyjna nawigacja
- Przejrzysta hierarchia informacji
- Responsywność na wszystkich urządzeniach

#### 3. **Funkcjonalność i interaktywność**

- Formularze przyjazne dla użytkownika
- Skuteczna wyszukiwarka

- System komentarzy (gdy potrzebny)
- Koszyk zakupowy (dla e-commerce)
- 4. **Bezpieczeństwo i prywatność**
  - Protokół HTTPS
  - Zabezpieczenia przed popularnymi atakami
  - Zgodność z przepisami (RODO)
  - Przejrzysta polityka prywatności
- 5. **Dostępność i inkluzywność**
  - Zgodność z wytycznymi WCAG
  - Wsparcie dla technologii wspomagających
  - Alternatywne metody dostępu do treści
  - Testowanie z różnymi użytkownikami
- 6. **Optymalizacja i wydajność**
  - Szybkie ładowanie strony
  - Optymalizacja dla wyszukiwarek (SEO)
  - Wydajność na urządzeniach mobilnych
  - Monitorowanie i analiza danych

## WSKAZÓWKI DLA POCZĄTKUJĄCYCH TWÓRCÓW STRON INTERNETOWYCH:

1. **Zacznij od podstaw**
  - Opanuj HTML, CSS i podstawy JavaScript
  - Zrozum zasady responsywnego designu
  - Naucz się korzystać z narzędzi deweloperskich w przeglądarce
2. **Praktykuj regularnie**
  - Buduj małe projekty dla zdobycia doświadczenia
  - Eksperymentuj z różnymi technikami i bibliotekami
  - Analizuj i odtwarzaj elementy ze stron, które ci się podobają
3. **Ucz się od społeczności**

- Korzystaj z platform jak MDN, CSS-Tricks, freeCodeCamp
- Dołącz do forów i grup dyskusyjnych
- Śledź trendy i najlepsze praktyki

#### **4. Postaw na dostępność i bezpieczeństwo od początku**

- Nie traktuj ich jako dodatki, lecz jako fundamentalne aspekty
- Testuj z różnymi użytkownikami i urządzeniami
- Wykorzystuj narzędzia do audytu dostępności i bezpieczeństwa

#### **5. Ciągłe się rozwijaj**

- Branża webowa zmienia się dynamicznie
- Regularnie aktualizuj swoją wiedzę
- Eksperymentuj z nowymi technologiami, zachowując balans między innowacją a sprawdzonymi rozwiązaniami

Pamiętaj, że tworzenie stron internetowych to zarówno sztuka, jak i nauka — wymaga kreatywności, umiejętności rozwiązywania problemów oraz technicznej precyzji. Najlepsze strony internetowe łączą estetyczny design z funkcjonalnością, dostępnością i wydajnością, tworząc kompleksowe doświadczenie dla wszystkich użytkowników.

**Sukces w tworzeniu stron internetowych wymaga nie tylko umiejętności technicznych, ale również zrozumienia potrzeb użytkowników i celów biznesowych — dlatego warto regularnie poszerzać wiedzę w tych obszarach, aby tworzyć strony, które nie tylko dobrze wyglądają, ale również skutecznie realizują swoje zadania.**

# Zasoby do dalszej nauki

## DOKUMENTACJA I PORADNIKI

- MDN Web Docs (<https://developer.mozilla.org/>)
- W3Schools (<https://www.w3schools.com/>)
- CSS-Tricks (<https://css-tricks.com/>)
- web.dev by Google (<https://web.dev/>)
- Smashing Magazine (<https://www.smashingmagazine.com/>)

## NARZĘDZIA

- Chrome DevTools
- Lighthouse (audit narzędzie)
- Can I Use (<https://caniuse.com/>)
- Google PageSpeed Insights
- WAVE Web Accessibility Evaluation Tool

## SPOŁECZNOŚCI

- Stack Overflow
- GitHub
- Dev.to
- CodePen
- Reddit (r/webdev, r/web\_design)

## KURSY ONLINE

- freeCodeCamp
- Codecademy
- Udemy
- Frontend Masters
- Coursera

## NEWSLETTERY

- CSS Weekly
- JavaScript Weekly

- 
- Frontend Focus
  - Web Design Weekly

Powodzenia w tworzeniu stron internetowych! Pamiętaj, że najlepsza nauka przychodzi poprzez praktykę i nieustanne doskonalenie umiejętności.