**09.12.2024**

**Przedmiot:**

**Projekt Internetowe bazy danych**

**Temat:**

**Sklep internetowy odzieżowy(PerfectLook)**

**Grupa:**

Sebastian Wołoszyn

Dominik Duma

Michał Marczak

**Prowadzący:**

Dr inż. Roman Ptak

## Spis treści

1. Cel i zakres pracy
   1. Cel projektu
   2. Zakres pracy
2. Technologie wykorzystane do realizacji
3. Projektowanie interfejsu użytkownika (UI/UX)
4. Harmonogram realizacji projektu
5. Kosztorys
6. Analiza SWOT z przeliczeniem czynników wagowych
   1. Charakterystyka organizacji
   2. Cele i metodologia analizy SWOT
   3. Identyfikacja i opis czynników

6.3.1. Mocne strony (Strengths)

6.3.2. Słabe strony (Weaknesses)

6.3.3. Szanse (Opportunities)

6.3.4. Zagrożenia (Threats)

* 1. Ważność czynników SWOT
  2. Analiza powiązań SWOT/TOWS

6.5.1. Analiza powiązań SWOT

6.5.2. Analiza powiązań TOWS

* 1. Wnioski oraz wybór strategii i uzasadnienie

1. Architektura systemu

7.1 Serwer i konfiguracja aplikacji

7.2 Routing

7.3 Moduły routingu

7.4 Połączenie z bazą danych

7.5 Szablony EJS

7.6 Skrypty JavaScript

7.7 Przykładowe pliki i ich funkcje

1. Diagramy

8.1 Diagram UML

8.2 Diagram bazy danych

1. Testy

9.1 Scenariusze testowe

9.2 Testy Lighthouse

9.2.1 Testy głównej witryny

9.2.2 Testy strony magazynu

1. Wnioski
2. Literatura
3. Spis

12.1 Spis tabel

12.2 Spis rysunków

1. **Cel i zakres pracy**

**1.1 Cel projektu**

Głównym celem projektu jest stworzenie sklepu internetowego, który zautomatyzuje proces sprzedaży i zarządzania zamówieniami. Dzięki temu praca po stronie sprzedawcy będzie prostsza, a klienci zyskają wygodę podczas robienia zakupów. Kluczowe jest ograniczenie ręcznego przetwarzania danych, takich jak aktualizowanie stanów magazynowych czy przetwarzanie zamówień, co w praktyce pozwoli zaoszczędzić czas i zminimalizować błędy. Sklep internetowy ma działać przez całą dobę, co umożliwi klientom zakupy w dowolnym momencie, niezależnie od lokalizacji.

Dodatkowo, projekt zakłada stworzenie systemu, który będzie wspierał sprzedawców i administratorów w zarządzaniu produktami, zamówieniami oraz danymi klientów. Dzięki intuicyjnemu panelowi administracyjnemu sprzedawcy będą mogli łatwo kontrolować całą działalność sklepu, od aktualizowania oferty po przetwarzanie reklamacji.

* 1. **Zakres pracy**

Zakres pracy obejmuje stworzenie aplikacji internetowej, która będzie dostosowana do trzech różnych typów użytkowników: klientów niezalogowanych, klientów zalogowanych oraz administratorów sklepu. Każda z tych ról ma przypisane konkretne funkcje, które definiują sposób korzystania z systemu.

Klienci niezalogowani będą mieli dostęp wyłącznie do przeglądania oferty sklepu. Będą mogli korzystać z wyszukiwarki oraz filtrować produkty według różnych kryteriów, takich jak kategoria, cena czy dostępność. Ponadto będą mogli zobaczyć szczegółowe opisy produktów, zdjęcia oraz inne informacje, które mogą pomóc w podjęciu decyzji zakupowej. Jednak osoby niezalogowane nie będą mogły dodawać produktów do koszyka ani składać zamówień – te funkcje będą dostępne dopiero po założeniu konta i zalogowaniu się.

Dla klientów zalogowanych dostępny będzie pełen zakres funkcjonalności zakupowych. Po zalogowaniu użytkownicy będą mogli dodawać produkty do koszyka, realizować zamówienia i śledzić ich status. Dodatkowo zalogowani klienci będą mieli dostęp do historii swoich zakupów, co pozwoli im sprawdzić szczegóły wcześniejszych zamówień. Mogą także zarządzać swoimi danymi osobowymi, na przykład zmieniać adresy dostawy czy edytować dane kontaktowe. To wszystko ma na celu zapewnienie klientom większej kontroli nad procesem zakupowym i zwiększenie ich zaangażowania.

Najbardziej rozbudowaną rolę w systemie będzie miał administrator, który pełni jednocześnie funkcję sprzedawcy. To właśnie on będzie zarządzał całym sklepem od strony operacyjnej. Administrator będzie miał możliwość dodawania nowych produktów, co obejmuje wprowadzenie ich opisów, zdjęć, cen oraz dostępności w magazynie. Będzie mógł również edytować istniejące produkty, na przykład zmieniając ich ceny lub stan magazynowy, a także usuwać te, które nie są już dostępne. Kolejną ważną funkcją będzie zarządzanie zamówieniami – administrator będzie mógł przeglądać zamówienia składane przez klientów, zmieniać ich statusy (na przykład oznaczać zamówienie jako wysłane lub anulowane) i wprowadzać ewentualne poprawki.

Administrator będzie miał także pełną kontrolę nad użytkownikami. W razie potrzeby będzie mógł usuwać konta klientów, edytować ich dane osobowe lub rozwiązywać problemy techniczne związane z dostępem do kont. Te funkcje mają zapewnić sprawne działanie sklepu i umożliwić sprzedawcy szybkie reagowanie na różne sytuacje.

Technicznie zakres pracy obejmuje stworzenie responsywnej strony internetowej, która będzie dobrze działać zarówno na komputerach, jak i urządzeniach mobilnych. Projekt zakłada również opracowanie bazy danych, która będzie przechowywać informacje o produktach, zamówieniach i użytkownikach. Aplikacja ma być skalowalna i wydajna, co pozwoli na jej rozwój w miarę zwiększającej się liczby klientów i zamówień. Całość ma być oparta na nowoczesnych technologiach, które zapewnią płynność działania i łatwość w obsłudze zarówno dla klientów, jak i administratorów.

1. **Technologie wykorzystywane do realizacji**

Do realizacji projektu sklepu internetowego wybrano następujące technologie, które zapewnią skalowalność, wydajność i łatwość rozbudowy:

**Frontend (interfejs użytkownika):**

* + **HTML5 i CSS3** – do stworzenia struktury i stylu strony internetowej.
  + **JavaScript** – do dodania interaktywności na stronie (np. obsługa koszyka, filtrowanie produktów, paginacja).
  + **EJS (Embedded JavaScript)** – do dynamicznego renderowania stron internetowych, co pozwala na ładowanie danych z serwera bez konieczności przeładowywania całej strony.

**Backend (serwer i logika biznesowa):**

* + **Node.js** – serwer po stronie backendu, umożliwiający szybką i skalowalną obsługę zapytań. Node.js pozwala na utrzymanie spójności języka JavaScript zarówno po stronie klienta, jak i serwera.
  + **Express.js** – framework webowy oparty na Node.js, który ułatwia budowanie struktury aplikacji, zarządzanie trasami (routing) i integrację z bazą danych.

**Baza danych:**

* + **MySQL** – relacyjna baza danych, w której będą przechowywane informacje o produktach, zamówieniach, użytkownikach, a także stany magazynowe. MySQL zapewnia niezawodne zarządzanie danymi i pozwala na łatwe skalowanie wraz z rozwojem sklepu.

**Repozytorium kodu:**

* + **GitHub** – narzędzie do zarządzania wersjami kodu i współpracy w zespole. GitHub umożliwia śledzenie zmian, tworzenie branchy dla nowych funkcji oraz zarządzanie zgłoszeniami błędów i poprawek.

1. **Projektowanie interfejsu użytkownika (UI/UX)**

Pierwszym krokiem będzie zaprojektowanie interfejsu użytkownika za pomocą narzędzia **Figma**. Umożliwi to wizualizację wyglądu strony, jej funkcjonalności i struktury. Projektowanie interfejsu powinno uwzględniać:

* Stronę główną, która umożliwi przeglądanie kategorii produktów, oferty specjalne i nowości.
* Stronę produktu z dokładnym opisem, zdjęciami, wyborem rozmiaru i koloru oraz przyciskiem „Dodaj do koszyka”.
* Funkcje koszyka zakupów oraz proces finalizacji zamówienia (checkout).
* Formularze logowania i rejestracji użytkowników.

Na tym etapie musimy również przemyśleć responsywność strony, aby była dobrze dostosowana do urządzeń mobilnych, ponieważ wielu klientów będzie korzystać ze sklepu na telefonach i tabletach.

**Projektowanie architektury aplikacji i bazy danych**

Następnym krokiem będzie zaprojektowanie struktury aplikacji oraz bazy danych MySQL. Musimy opracować odpowiednią architekturę, która umożliwi płynne działanie sklepu oraz zarządzanie danymi. Baza danych powinna obejmować:

* Tabele produktów z informacjami o nazwie, cenie, opisie, dostępności i zdjęciach.
* Tabele użytkowników z danymi logowania oraz danymi osobistymi (adresy, zamówienia).
* Tabele zamówień, gdzie będą zapisywane informacje o złożonych transakcjach i statusie wysyłki.

Dobrze zaprojektowana baza danych pozwoli na szybki dostęp do informacji oraz bezproblemowe zarządzanie stanami magazynowymi i zamówieniami.

**Repozytorium kodu**

W ramach pracy zespołowej wykorzystamy **GitHub** jako platformę do przechowywania kodu. Pozwoli nam to na efektywną współpracę, śledzenie zmian oraz zarządzanie wersjami projektu. Repozytorium będzie zorganizowane w sposób umożliwiający podział pracy na moduły:

* **Frontend** – foldery z kodem HTML, CSS, JavaScript oraz szablonami EJS.
* **Backend** – kod Node.js i Express do obsługi zapytań serwera.
* **Baza danych** – skrypty SQL do tworzenia i aktualizacji tabel w MySQL.

1. **Harmonogram realizacji projektu**

* **Tydzień 1-2: Przygotowanie projektu**
  + Określenie celu, zakresu, technicznej wykonalności systemu oraz estymacja kosztów.
  + Opracowanie opisu projektu, uwzględniając analizę SWOT (lub analizę celów, problemów i kluczowych czynników sukcesu - CSF).
  + Utworzenie repozytorium projektu na platformie GitHub.
  + Podział obowiązków w grupie i określenie odpowiedzialności.
* **Tydzień 3-4: Planowanie i projektowanie**
  + Opracowanie zbioru wymagań funkcjonalnych i niefunkcjonalnych oraz stworzenie dokumentacji UML.
  + Projektowanie interfejsu użytkownika (UI) w narzędziu Figma.
  + Opracowanie struktury bazy danych w MySQL.
  + Przygotowanie podstawowych szablonów frontendowych i pierwszych widoków.
* **Tydzień 5-6: Implementacja backendu i integracja z bazą danych**
  + Implementacja logiki backendowej w Node.js i Express.
  + Połączenie aplikacji z bazą danych i implementacja podstawowych funkcji, takich jak dodawanie produktów i zarządzanie użytkownikami.
* **Tydzień 7-8: Testowanie funkcjonalności**
  + Testowanie kluczowych funkcji aplikacji (np. dodawanie do koszyka, składanie zamówień).
  + Testowanie responsywności i wydajności aplikacji na różnych urządzeniach i przeglądarkach.
  + Przeprowadzenie testów obciążeniowych w celu oceny wydajności.
* **Tydzień 9-10: Wdrożenie na środowisko produkcyjne**
  + Ostateczne testy przed publikacją.
  + Przygotowanie prezentacji na forum publicznym lub przed prowadzącym.
  + Omówienie poszczególnych etapów projektu, prezentacja osiągniętych wyników oraz dokumentacji projektu.

1. **Kosztorys**

Tabela . Koszt stworzenia sklepu internetowego

|  |  |  |
| --- | --- | --- |
| **Etap** | **szacunkowe godziny** | **szacunkowy koszt pln** |
| Przygotowanie projektu | 10h | 1 200,00 zł |
| Planowanie i projektowanie | 50h | 6 000,00 zł |
| Implementacja backendu | 45h | 5 400,00 zł |
| Testowanie | 20h | 2 400,00 zł |
| Wdrożenie | 15h | 1 800,00 zł |
| **Podsumowanie** | **140h** | **16 800,00 zł** |

**Wliczone koszty amortyzacja i inne podczas tworzenia strony(miesiąc):**

* Zużycie prądu 90,00 PLN
* Koszty eksploatacji sprzętu : 255,00 PLN
* Dieta pracownika: 255,00 PLN
* Licencje na oprogramowanie: 300,00 PLN

Tabela . Koszt utrzymania sklepu internetowego

|  |  |  |
| --- | --- | --- |
| **Licencje** | **koszt miesięczny** | **koszt roczny** |
| domena | 20,44 zł | 245,28 zł |
| serwer(hosting) | 184,50 zł | 2214,00 zł |
| **Podsumowanie** | **204,94 zł** | **2459,28 zł** |

\*Ceny opierane o koszty na stronie „home.pl”

**Amortyzacja i inne koszty podczas tworzenia strony:**

1. **Analiza SWOT**

**6.1. Charakterystyka organizacji**

Analizowany podmiot to sklep internetowy specjalizujący się w sprzedaży odzieży, który wyróżnia się szerokim asortymentem dostosowanym do potrzeb różnych grup klientów, zarówno lokalnych, jak i globalnych. Oferta obejmuje ubrania na każdą okazję, od odzieży codziennej po bardziej formalne i stylowe kreacje. Dzięki obecności online firma unika kosztów związanych z utrzymywaniem fizycznych lokalizacji, co pozwala na lepszą kontrolę kosztów operacyjnych.

Sklep działa w pełni w trybie całodobowym, umożliwiając klientom zakupy w dowolnym czasie. Dzięki zastosowaniu zautomatyzowanych systemów zarządzania zamówieniami i magazynem, proces realizacji zakupów jest szybki i efektywny. Automatyzacja pozwala również na minimalizację błędów, takich jak problemy z dostępnością produktów, co wpływa na wyższą satysfakcję klientów.

**6.2. Cele i metodologia analizy SWOT**

Celem przeprowadzenia analizy SWOT jest wsparcie strategicznego zarządzania rozwojem sklepu internetowego z odzieżą. Poprzez identyfikację kluczowych czynników wewnętrznych i zewnętrznych, analiza umożliwia wskazanie mocnych i słabych stron działalności oraz szans i zagrożeń związanych z dynamicznie zmieniającym się rynkiem modowym.

Metodologia analizy SWOT obejmuje:

* Identyfikację czynników wewnętrznych i zewnętrznych: Przeprowadzono szczegółową ocenę elementów związanych z funkcjonowaniem sklepu oraz otoczeniem biznesowym.
* Ocena ważności poszczególnych czynników: Każdy zidentyfikowany czynnik został przypisany odpowiedniej wadze, co pozwala na określenie jego wpływu na działalność firmy.
* Zastosowanie macierzy TOWS: Dzięki powiązaniu mocnych i słabych stron z szansami i zagrożeniami, wypracowano konkretne strategie mające na celu maksymalizację potencjału firmy oraz minimalizację ryzyka.

Analiza SWOT pozwala nie tylko na diagnozę obecnej sytuacji sklepu, ale również na wypracowanie kierunków dalszego rozwoju, które są zgodne z potrzebami rynku oraz oczekiwaniami klientów. Jest to nieodzowny element planowania strategicznego, szczególnie w kontekście dynamicznie rozwijającej się branży e-commerce.

**6.3. Identyfikacja i opis czynników**

6.3.1 Mocne strony:

* **Dostępność całodobowa -** Sklep internetowy jest w stanie przedstawiać klientowi swoją ofertę w każdej chwili oraz pozwala mu złożyć zamówienie w dowolnym momencie
* **Niższe koszty operacyjne -** Sklep internetowy nie potrzebuje fizycznych punktów sprzedaży, co niweluje koszty wynajmu i innych wydatków związanych z prowadzeniem sklepów stacjonarnych
* **Automatyzacja procesów** –system zarządzania zamówieniami i stanami magazynowymi.

6.3.2 Słabe strony:

* **Brak możliwości fizycznego przymierzenia -** Przez brak fizycznych sklepów, klient nie jest w stanie przymierzyć na sobie wybranego produktu. Musi on polegać na dostosowaniu wymiarów podanych na stronie do pomiarów wykonanych indywidualnie
* **Koszty zwrotów i reklamacji -** Z uwagi na brak możliwości sprawdzenia produktu przed kupnem, jest większe prawdopodobieństwo zwrotów i reklamacji przez klientów, co liczy się z dodatkowymi kosztami
* **Uzależnienie od technologii i logistyki -** Problemy techniczne oraz związane z dostawą mogą prowadzić do niezadowolenia klientów
* **Brak lokalnej rozpoznawalności marki** – trudność w budowaniu lojalności klientów w początkowej fazie działalności.

6.3.3 Szanse:

* **Większe zapotrzebowanie na zakupy online -** Obecnie więcej ludzi (zwłaszcza po latach pandemi) preferują robienie zakupów bez wychodzenia z domu, co skutkuje większą liczbą potencjalnych klientów
* **Współczesny marketing -** Współprace z influencerami pozwolą na dotarcie do większego grona osob, szczególnie do osób młodych
* **Nowoczesne metody płatności –** wprowadzenie takich opcji jak BLIK, Apple Pay.

6.3.4 Zagrożenia:

* **Zmiana zachowań konsumenckich -** Klienci coraz częściej poszukują produktów ekologicznych i etycznie produkowanych, co może być wyzwaniem dla marek oferujących standardowe produkty odzieżowe
* **Wysoka konkurencja -** Istnienie wielu podobnych sklepów wymusza dostosowanie ceny i jakości produktów do konkurencji
* **Problemy logistyczne** – opóźnienia w dostawach negatywnie wpływają na satysfakcję klientów.
* **Ryzyko cyberataków** – konieczność ochrony danych klientów.

**6.4 Ważność czynników SWOT**

Tabela . Ważność czynników SWOT

|  |  |
| --- | --- |
| **Czynniki analizy SWOT** | **Waga** |
| **Mocne strony** | 1.00 |
| Dostępność całodobowa | 0.35 |
| Niższe koszty operacyjne | 0.30 |
| Automatyzacja procesów | 0.35 |
| **Słabe strony** | 1.00 |
| Brak fizycznego kontaktu z produktem | 0.30 |
| Koszty zwrotów i reklamacji | 0.25 |
| Uzależnienie od technologii | 0.25 |
| Brak lokalnej rozpoznawalności marki | 0.20 |
| **Szanse** | 1.00 |
| Wzrost popularności zakupów online | 0.40 |
| Możliwość współpracy z influencerami | 0.35 |
| Nowoczesne metody płatności | 0.25 |
| **Zagrożenia** | 1.00 |
| Silna konkurencja | 0.35 |
| Zmiany preferencji konsumenckich | 0.30 |
| Problemy logistyczne | 0.25 |
| Ryzyko cyberataków | 0.10 |

**6.5 Analiza powiązań SWOT/TOWS**

6.5.1 Analiza powiązań SWOT

Czy określona mocna strona pozwala wykorzystać daną szansę?

Tabela . Szanse mocnych stron

|  |  |  |  |
| --- | --- | --- | --- |
| **Szanse / Mocne strony** | **Dostępność całodobowa** | **Niższe koszty operacyjne** | **Automatyzacja procesów** |
| Większe zapotrzebowanie na zakupy online | 1 | 1 | 1 |
| Współczesny marketing | 1 | 0 | 0 |
| Nowoczesne metody płatności | 1 | 1 | 1 |

Liczba interakcji: 7

Czy określona mocna strona pozwala ograniczyć dane zagrożenie?

Tabela . Zagrożenia mocnych stron

|  |  |  |  |
| --- | --- | --- | --- |
| **Zagrożenia / Mocne strony** | **Dostępność całodobowa** | **Niższe koszty operacyjne** | **Automatyzacja procesów** |
| Zmiana zachowań konsumenckich | 1 | 1 | 1 |
| Wysoka konkurencja | 0 | 1 | 1 |
| Problemy logistyczne | 0 | 0 | 1 |
| Ryzyko cyberataków | 0 | 0 | 1 |

Liczba interakcji: 6

6.5.2 Analiza powiązań TOWS

Czy określona słaba strona ogranicza możliwość wykorzystania danej szansy?

Tabela . Szanse słabych stron

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Szanse / Słabe strony** | **Brak możliwości fizycznego przymierzenia** | **Koszty zwrotów i reklamacji** | **Uzależnienie od technologii i logistyki** | **Brak lokalnej rozpoznawalności marki** |
| Większe zapotrzebowanie na zakupy online | 1 | 0 | 1 | 0 |
| Współczesny marketing | 0 | 0 | 1 | 1 |
| Nowoczesne metody płatności | 0 | 0 | 1 | 0 |

Liczba interakcji: 6

Czy określona słaba strona potęguje dane zagrożenie?

Tabela . Zagrożenia słabych stron

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Zagrożenia / Słabe strony** | **Brak możliwości fizycznego przymierzenia** | **Koszty zwrotów i reklamacji** | **Uzależnienie od technologii i logistyki** | **Brak lokalnej rozpoznawalności marki** |
| Zmiana zachowań konsumenckich | 1 | 1 | 0 | 0 |
| Wysoka konkurencja | 0 | 0 | 1 | 1 |
| Problemy logistyczne | 0 | 1 | 1 | 0 |
| Ryzyko cyberataków | 0 | 0 | 1 | 0 |

Liczba interakcji: 7

**6.6. Wnioski oraz wybór strategii i uzasadnienie**

Na podstawie przeprowadzonej analizy SWOT i TOWS zidentyfikowano kluczowe obszary, które wymagają strategicznego zarządzania w celu rozwoju sklepu internetowego z odzieżą.

**Wnioski:**

**6.6.1 Wykorzystanie mocnych stron w kontekście szans:**

* + Dostępność całodobowa oraz automatyzacja procesów stanowią podstawę do dynamicznego rozwoju sklepu, szczególnie w kontekście rosnącego zapotrzebowania na zakupy online.
  + Współpraca z influencerami oraz inwestycja w nowoczesne metody płatności (np. BLIK, Apple Pay) mogą znacząco zwiększyć zasięg i przyciągnąć młodszych klientów.

**6.6.2 Ograniczenie słabych stron i minimalizacja zagrożeń:**

* + Brak możliwości fizycznego przymierzenia odzieży należy kompensować szczegółowymi opisami produktów, dokładnymi tabelami rozmiarów oraz polityką łatwych zwrotów.
  + Niezbędne jest wdrożenie zaawansowanych zabezpieczeń technologicznych w celu ochrony danych klientów i minimalizacji ryzyka cyberataków.
  + Należy budować lokalną rozpoznawalność marki poprzez działania marketingowe skierowane na identyfikację wizualną i zaangażowanie społeczności lokalnych.
    1. **Adaptacja do zmieniających się zachowań konsumenckich:**

Klienci coraz częściej oczekują produktów ekologicznych i etycznie produkowanych. Dodanie do oferty linii odzieży zrównoważonej może przyciągnąć nowych klientów i zwiększyć konkurencyjność.

1. **Architektura systemów**

**Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie**

Rysunek . Drzewo projektu PerfectLook

**7.1. Serwer i konfiguracja aplikacji**

* **Plik server.js:**
  + Tworzy instancję aplikacji Express.
  + Dodaje middleware compression do kompresji odpowiedzi HTTP dla szybszego ładowania stron.
  + Konfiguruje helmet, aby dodać nagłówki bezpieczeństwa i chronić aplikację przed zagrożeniami.
  + Ustawia Politykę Bezpieczeństwa Treści (CSP), wykorzystując nonce do obsługi bezpiecznego wykonywania skryptów.
  + Używa session z MemoryStore do zarządzania sesjami użytkowników, przechowywaniem sesji i ich wygaśnięciem.
  + Dodaje middleware body-parser do obsługi żądań POST z formularzy i danych JSON.
  + Serwuje pliki statyczne z katalogu /public.
  + Konfiguruje silnik szablonów EJS oraz ustawia ścieżki widoków na katalog /public.
  + Ładuje główny router z pliku routings.js.
  + Uruchamia serwer na porcie 3000 i loguje adres lokalny aplikacji.



Rysunek . Plik server.js

**7.2. Routing**

* **Plik routings.js:**
* Importuje i używa różnych modułów routingu, które obsługują różne części aplikacji.
* Każdy moduł routingu jest odpowiedzialny za określone funkcjonalności, np. cartRoutes obsługuje koszyk, loginRoutes obsługuje logowanie, itd.

Obraz zawierający tekst, elektronika, wyświetlacz, zrzut ekranu

Opis wygenerowany automatycznie

Rysunek . Plik routings.js

**7.3. Moduły routingu**

* **Przykładowe moduły routingu**:
  + **routes/cart.js**: Obsługuje operacje związane z koszykiem, takie jak dodawanie produktów do koszyka, przetwarzanie płatności i tworzenie zamówień.
  + **routes/login.js**: Obsługuje logowanie i wylogowywanie użytkowników.
  + **routes/admin.js**: Obsługuje operacje administracyjne, takie jak zarządzanie produktami, kurierami i zamówieniami.
  + **routes/faq.js**: Obsługuje wyświetlanie stron FAQ, takich jak regulamin, polityka prywatności i cookies.
  + **routes/payment.js**: Obsługuje proces płatności, w tym etapy płatności i weryfikację danych.

**7.4. Połączenie z bazą danych**

* Plik **dbConnect.js**:
* Tworzy połączenie z bazą danych MySQL za pomocą modułu mysql.
* Eksportuje połączenie, które jest używane w różnych modułach routingu do wykonywania zapytań do bazy danych.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

Opis wygenerowany automatycznie

Rysunek . Plik dbConnect.js

**7.5. Szablony EJS**

* Pliki w folderze **public**:
* Zawierają szablony EJS, które są renderowane przez serwer i wysyłane do klienta.
* Przykłady:
  + **public/my-orders.ejs**: Wyświetla zamówienia użytkownika.
  + **public/payment\_stage2.ejs**: Obsługuje drugi etap płatności.
  + **public/admin-options.ejs**: Wyświetla opcje administracyjne.
  + **public/regulations.ejs**: Wyświetla regulamin sklepu.
  + **public/policy.ejs**: Wyświetla politykę prywatności.



Rysunek . Plik my-orders.js

**7.6. Skrypty JavaScript**

* Pliki w folderze **js**:
* Zawierają skrypty JavaScript, które są uruchamiane po stronie klienta.
* Przykłady:
  + **public/js/carusel.js**: Obsługuje karuzelę produktów.
  + **public/js/paymentStage2.js**: Obsługuje logikę drugiego etapu płatności.
  + **public/js/list.js**: Obsługuje logikę wyświetlania listy użytkownika.

Obraz zawierający tekst, zrzut ekranu, wyświetlacz, oprogramowanie

Opis wygenerowany automatycznie

Rysunek . Plik list.js

**7.7. Przykładowe pliki i ich funkcje**

* **public/my-orders.ejs**:
  + Wyświetla listę zamówień użytkownika.
  + Używa zmiennej orders do iteracji i wyświetlania szczegółów każdego zamówienia.
  + Zawiera nagłówki tabeli i przyciski akcji dla każdego zamówienia.
* **routes/cart.js**:
  + Obsługuje dodawanie produktów do koszyka.
  + Przetwarza płatności i tworzy zamówienia w bazie danych.
  + Usuwa produkty z koszyka po złożeniu zamówienia.
* **routes/login.js**:
  + Obsługuje logowanie użytkowników.
  + Sprawdza dane logowania w bazie danych.
  + Ustawia sesję użytkownika po pomyślnym zalogowaniu.
* **routes/admin.js**:
  + Obsługuje operacje administracyjne, takie jak dodawanie, edytowanie i usuwanie produktów.
  + Zarządza kurierami i zamówieniami.

1. **Diagramy**

**8.1. Diagram UML**

Prezentowany diagram składa się trzech aktorów: użytkownika niezalogowanego, użytkownika zalogowanego oraz sprzedawcę.

Użytkownik niezalogowany może przeglądać produkty i zarejestrować się lub zalogować, aby uzyskać pełny dostęp.

Użytkownik zalogowany ma możliwość przeglądania produktów, dodawania ich do koszyka, składania zamówień, dokonywania płatności oraz śledzenia statusu zamówienia.

Sprzedawca zarządza produktami, zamówieniami oraz ich statusami, zapewniając sprawne funkcjonowanie systemu.

Obraz zawierający diagram, tekst, krąg, zrzut ekranu

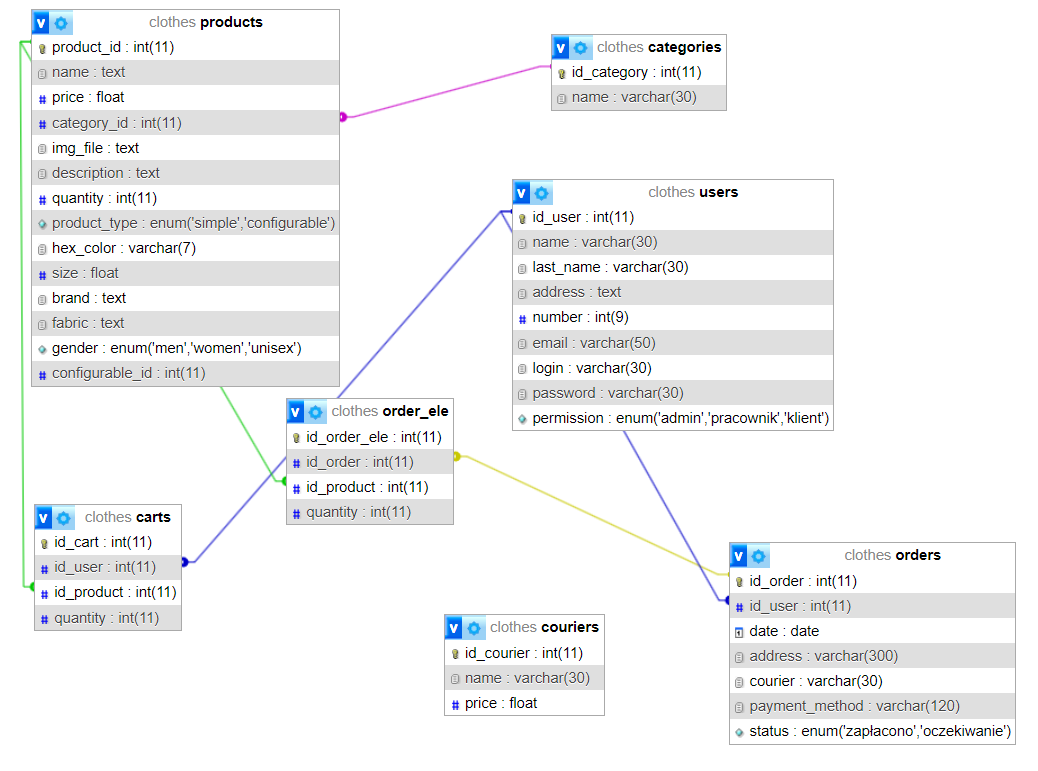
Opis wygenerowany automatycznie

Rysunek . Diagram UML

**8.2. Diagram bazy danych**

W bazie danych sklepu internetowego główną tabelą jest tabela users, która przechowuje dane o użytkownikach. Jest ona połączona z tabelami carts i orders, dzięki czemu można przypisać do użytkownika jego koszyk oraz zamówienia.

Tabela products jest powiązana z tabelą categories, co pozwala przypisać produkt do odpowiedniej kategorii. Produkty trafiają do koszyków (carts) i zamówień, a szczegóły zamówień, takie jak ilość czy rodzaj produktów, są zapisane w tabeli order\_ele. Relacje między tabelami są oparte na kluczach obcych, co pozwala na automatyczne usuwanie lub aktualizowanie danych, jeśli zmienią się rekordy w powiązanych tabelach. Wszystko jest tak połączone, żeby dane zawsze były spójne i logicznie powiązane.



Rysunek . Diagram bazy danych

* 1. **Testy**

**9.1. Scenariusze testowe**

Przeprowadziliśmy scenariusze testowe polegające na sprawdzaniu rejestracji użytkownika używając do tego Selenium.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Rysunek . Zrzut ekranu poprawne dane logowania

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Rysunek . Zrzut ekranu błędny login i hasło

**9.2 Testy Lighthouse**

9.2.1 Testy głownej witryny

Strona (http://127.0.0.1:3000)została oceniona pozytywnie wyniki w Lighthouse, z solidną wydajnością (94), dostępnością (95), najlepszymi praktykami (89), i SEO (91). Kluczowe pozytywy to szybki czas ładowania pierwszego elementu (1.4 s), brak przesunięć układu (CLS = 0), oraz brak opóźnień w renderowaniu spowodowanych przez skrypty blokujące. Dodatkowo, strona wykorzystuje nowoczesne technologie, takie jak obrazy WebP, oraz ma zoptymalizowaną strukturę zasobów i krótki czas odpowiedzi serwera.

**Obraz zawierający tekst, zrzut ekranu, design

Opis wygenerowany automatycznie**

Rysunek . Testy głównej witryny Lighthouse

9.2.2 Testy strony magazynu

Ta strona osiąga doskonały wynik 99 w kategorii wydajności w narzędziu Lighthouse, co świadczy o wyjątkowo zoptymalizowanej strukturze. Kluczowe metryki, takie jak "First Contentful Paint" (1.4 s) i "Largest Contentful Paint" (2.0 s), są na bardzo wysokim poziomie, zapewniając szybkie ładowanie treści. Brak przesunięć układu (CLS = 0) gwarantuje stabilność wizualną strony. Wyróżnia się również brak czasu blokowania (0 ms) i krótki czas renderowania.

Obraz zawierający tekst, zrzut ekranu, Oprogramowanie multimedialne, oprogramowanie

Opis wygenerowany automatycznie

Rysunek . Testy witryny magazynu Lighthouse

**10.Wnioski**

Projekt sklepu internetowego zakończył się sukcesem w zakresie założonych celów technicznych. Powstała aplikacja, która automatyzuje procesy związane z zarządzaniem zamówieniami, stanami magazynowymi i obsługą użytkowników. System został zaprojektowany z myślą o różnych typach użytkowników – niezalogowanych, zalogowanych oraz administratorach – co pozwala na elastyczne dostosowanie funkcjonalności do ich potrzeb.

W szczególności należy podkreślić skalowalność i wydajność aplikacji, które udało się osiągnąć dzięki zastosowaniu takich technologii jak Node.js, Express oraz MySQL. Strona została zaprojektowana w sposób responsywny, co umożliwia korzystanie z niej na różnych urządzeniach, takich jak komputery i telefony, bez utraty jakości użytkowania.

Przeprowadzone testy, w tym testy Lighthouse, potwierdziły wysoki poziom techniczny aplikacji. Krótki czas ładowania stron, stabilność wizualna i brak istotnych opóźnień w renderowaniu wskazują na optymalną strukturę kodu oraz dobrze zaplanowaną architekturę systemu. Zastosowanie technologii takich jak obrazy WebP i kompresja HTTP dodatkowo wpłynęło na efektywność działania aplikacji.

Warto rozważyć dalsze rozwijanie projektu pod kątem funkcjonalności. Można wprowadzić bardziej zaawansowane zarządzanie stanami magazynowymi, takie jak automatyczne alerty o niskim poziomie zapasów, czy bardziej szczegółowe statystyki sprzedaży dostępne w panelu administracyjnym. Dodatkowo integracja z systemami raportowania błędów i monitorowania wydajności, takimi jak Sentry lub New Relic, może pomóc w szybkim wykrywaniu i rozwiązywaniu problemów technicznych.

Pod względem zabezpieczeń sugerujemy regularne aktualizacje technologii, które zostały wykorzystane, aby zapewnić ochronę przed nowymi zagrożeniami. Warto także wdrożyć dodatkowe mechanizmy ochrony danych, takie jak szyfrowanie haseł użytkowników z użyciem nowoczesnych algorytmów, oraz dokładniejsze logowanie aktywności administracyjnych.

Podsumowując, projekt w obecnym stanie spełnia swoje podstawowe założenia i może być dalej rozwijany w kierunku bardziej zaawansowanego systemu zarządzania sklepem internetowym. Stworzona aplikacja to solidna baza technologiczna, na której można budować kolejne moduły i funkcjonalności zgodnie z przyszłymi potrzebami użytkowników.

**11. Literatura**

**Lighthouse - Narzędzie do audytu aplikacji internetowych**

1. Dokumentacja Google Lighthouse: https://developer.chrome.com/docs/lighthouse/ (Data dostępu: 11 grudnia 2024)
2. Dokumentacja Express.js: <https://expressjs.com/> (Data dostępu: 11 grudnia 2024)

**Wątki na Stack Overflow o Node.js:**

* 1. "How to set up Express.js with Node.js": <https://stackoverflow.com/questions/16753912/how-to-set-up-express-js-with-node-js> (Data dostępu: 11 grudnia 2024)
  2. "Session management in Express.js": <https://stackoverflow.com/questions/40851383/session-management-in-express-js> (Data dostępu: 11 grudnia 2024)

**Bazy danych – MySQL**

1. Oficjalna dokumentacja MySQL: <https://dev.mysql.com/doc/> (Data dostępu: 11 grudnia 2024)
2. Przewodnik po MySQL: https://www.w3schools.com/mysql/ (Data dostępu: 11 grudnia 2024)

**Wątki na Stack Overflow:**

* 1. "Best practices for MySQL database design": <https://stackoverflow.com/questions/26515700/best-practices-for-mysql-database-design> (Data dostępu: 11 grudnia 2024)
  2. "MySQL performance optimization tips": <https://stackoverflow.com/questions/16756034/mysql-performance-optimization-tips> (Data dostępu: 11 grudnia 2024)

**EJS (Embedded JavaScript) - Renderowanie dynamicznych stron**

1. Oficjalna dokumentacja EJS: <https://ejs.co/> (Data dostępu: 11 grudnia 2024)
2. Tutorial EJS na W3Schools: https://www.w3schools.com/nodejs/nodejs\_ejs.asp (Data dostępu: 11 grudnia 2024)

**Helmet.js i bezpieczeństwo w aplikacjach internetowych**

1. Oficjalna dokumentacja Helmet.js: https://helmetjs.github.io/ (Data dostępu: 11 grudnia 2024)
2. Wątek na Stack Overflow: "How to configure Helmet for best security": <https://stackoverflow.com/questions/42663319/how-to-configure-helmet-for-best-security> (Data dostępu: 11 grudnia 2024)

**Kompresja HTTP - middleware compression**

1. Dokumentacja middleware compression: http://expressjs.com/en/resources/middleware/compression.html (Data dostępu: 11 grudnia 2024)
2. Wątek na Stack Overflow: "How to enable gzip compression in Express.js": <https://stackoverflow.com/questions/20128317/how-to-enable-gzip-compression-in-express-js> (Data dostępu: 11 grudnia 2024)

**Testowanie i automatyzacja testów**

Selenium WebDriver dokumentacja: https://www.selenium.dev/documentation/webdriver/ (Data dostępu: 11 grudnia 2024)

1. Jak pisać testy w Node.js: https://blog.logrocket.com/a-quick-and-complete-guide-to-mocha-testing-d07dd42d7ed0/ (Data dostępu: 11 grudnia 2024)

**Responsywność i optymalizacja frontendowa**

1. Jak tworzyć responsywne strony: <https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Responsive_Design> (Data dostępu: 11 grudnia 2024)
2. Optymalizacja obrazów w WebP: https://developers.google.com/speed/webp (Data dostępu: 11 grudnia 2024)

**12.Spis**

**12.1 Spis tabel**

Tabela 1. Koszt stworzenia sklepu internetowego str. 7

Tabela 2. Koszt utrzymania sklepu internetowego str. 7

Tabela 3. Ważność czynników SWOT str. 9

Tabela 4. Szanse mocnych stron str. 10

Tabela 5. Zagrożenia mocnych stron str. 10

Tabela 6. Szanse słabych stron str. 10

Tabela 7. Zagrożenia słabych stron str. 11

**12.2 Spis rysunków**

Rysunek 1. Drzewo projektu PerfectLook str. 12

Rysunek 2. Plik server.js str. 13

Rysunek 3. Plik routings.js str. 14

Rysunek 4. Plik dbConnect.js str. 15

Rysunek 5. Plik my-orders.js str. 16

Rysunek 6. Plik list.js str. 17

Rysunek 7. Diagram UML str. 18

Rysunek 8. Diagram bazy danych str. 19

Rysunek 9. Zrzut ekranu poprawne dane logowania str. 20

Rysunek 10. Zrzut ekranu błędny login i hasło str. 20

Rysunek 11. Testy głównej witryny Lighthouse str. 21

Rysunek 12. Testy witryny magazynu Lighthouse str. 22