

DOKUMENTACJA PROJEKTU

***E-COMMERCE W TECHNOLOGII C# I REACT***

Z PRZEDMIOTU

PROJEKTOWANIE WIELOWARSTWOWYCH APLIKACJI BIZNESOWYCH

Wykonali:

*WARSZAWA 2024*

Wstęp

Przegląd projektu Platforma handlu elektronicznego to nowoczesne rozwiązanie do zakupów online zaprojektowane w celu zapewnienia klientom intuicyjnych i płynnych zakupów. Projekt ten ma na celu opracowanie bogatej w funkcje, skalowalnej i bezpiecznej witryny e-commerce, w której użytkownicy mogą przeglądać produkty, dodawać je do swoich koszyków, dokonywać płatności i śledzić zamówienia.

Wykorzystane oprogramowanie

Visual studio

GitHub

Microsoft SQL Server

React.js

Wymagania funkcjonalne

1. Rejestracja i uwierzytelnianie użytkowników
2. Użytkownicy muszą mieć możliwość utworzenia konta
3. Użytkownicy muszą mieć możliwość logowania i wylogowywania się ze swoich kont.
4. Użytkownicy muszą mieć możliwość resetowania swoich haseł.
5. Katalog produktów
6. Witryna powinna wyświetlać obszerną listę produktów ze szczegółowymi opisami, zdjęciami, cenami i dostępnością.
7. Klienci powinni mieć możliwość wyszukiwania produktów przy użyciu różnych filtrów (np. kategorii, marki, zakresu cen).
8. Administratorzy powinni mieć możliwość dodawania, edytowania i usuwania produktów z katalogu.
9. Koszyk
10. Użytkownicy powinni mieć możliwość dodawania produktów do swoich koszyków.
11. Użytkownicy powinni mieć możliwość przeglądania zawartości swoich koszyków i aktualizowania ich ilości.
12. Użytkownicy powinni mieć możliwość usuwania produktów ze swoich koszyków.
13. Witryna powinna automatycznie obliczać łączną cenę produktów w koszyku.
14. Zarządzanie zamówieniami
15. Klienci powinni mieć możliwość składania zamówień i śledzenia ich statusu.
16. Administratorzy powinni mieć możliwość przeglądania, przetwarzania i zarządzania zamówieniami klientów.
17. Zamówienia powinny być wysyłane lub odbierane w fizycznej lokalizacji.

Wymagania niefunkcjonalne

1. Wydajność: Platforma powinna być w stanie obsłużyć co najmniej 1000 użytkowników jednocześnie bez degradacji wydajności.
2. Czas odpowiedzi: Czas odpowiedzi na żądania użytkowników nie powinien przekraczać 2 sekund. To zapewnia płynną i satysfakcjonującą interakcję użytkownika z platformą.
3. Dostępność: Platforma powinna być dostępna w 99.9xx%. To jest kluczowe dla e-commerce, ponieważ klienci mogą chcieć dokonywać zakupów o dowolnej porze dnia czy nocy.
4. Bezpieczeństwo: Platforma powinna zapewniać bezpieczne i niezawodne przetwarzanie płatności. Powinna również chronić dane osobowe użytkowników zgodnie z obowiązującymi przepisami o ochronie danych.
5. Skalowalność: Platforma powinna być łatwo skalowalna, aby móc obsługiwać rosnącą liczbę użytkowników i transakcji.
6. Użyteczność: Interfejs użytkownika powinien być intuicyjny i łatwy w użyciu, zarówno dla klientów, jak i sprzedawców. (baza na danej normie wiekowej, wziąć pod uwagę osoby niedowidzące itp)
7. Kompatybilność: Platforma powinna być kompatybilna z różnymi przeglądarkami i urządzeniami, aby umożliwić jak największej liczbie użytkowników korzystanie z niej.
8. Responsywność: Interfejs użytkownika powinien być responsywny, co oznacza, że powinien dobrze wyglądać i działać na różnych rozdzielczościach ekranu, od smartfonów po komputery stacjonarne.

Sposoby i metody testowania

Nasza strategia testowania obejmuje zarówno ręczne, jak i automatyczne metody testowania:

Testowanie ręczne

* Testowanie funkcjonalne: Obejmuje testowanie konkretnych przypadków użycia i historii użytkownika w celu zapewnienia funkcjonalności aplikacji.
* Testy użyteczności: Przeprowadzimy testy użyteczności z prawdziwymi użytkownikami, aby zidentyfikować wszelkie problemy w zakresie dostępności, łatwości użytkowania i ogólnego doświadczenia użytkownika.

Testowanie automatyczne

* Testy jednostkowe: Obejmuje testowanie poszczególnych komponentów lub funkcji bazy kodu w celu zapewnienia ich prawidłowego działania.
* Testy integracyjne: Testujemy interakcje między różnymi komponentami lub systemami w naszej aplikacji.
* Testy regresji: Będziemy okresowo ponownie uruchamiać testy automatyczne, aby zidentyfikować i naprawić wszelkie nowe problemy, które mogły zostać wprowadzone podczas rozwoju.

Testowanie ciągłe

* Będziemy przeprowadzać regularne przeglądy kodu, upewniając się, że wszystkie zmiany są dokładnie testowane przed scaleniem z główną gałęzią.
* Nasz serwer ciągłej integracji będzie uruchamiał testy automatyczne za każdym razem, gdy nowe zmiany zostaną przesłane do repozytorium.

Stosując się do tej kompleksowej strategii testowania, możemy zapewnić, że nasza platforma e-commerce dostarcza wysokiej jakości doświadczenia zakupowe dla naszych użytkowników, przy jednoczesnym Warstwa logiki biznesowej

Warstwa logiki biznesowej dla platformy e-commerce obejmuje następujące elementy:

1. Zarządzanie produktami: Powinny być metody umożliwiające dodawanie nowych produktów, edycję istniejących produktów i usuwanie produktów. Każdy produkt może mieć różne atrybuty, takie jak nazwa, opis, cena, zdjęcie i sprzedawca.
2. Zarządzanie zamówieniami: Powinny być metody umożliwiające tworzenie nowych zamówień, aktualizację statusu istniejących zamówień i anulowanie zamówień. Każde zamówienie może mieć różne atrybuty, takie jak lista produktów, łączna kwota, adres dostawy i status.
3. Zarządzanie użytkownikami: Powinny być metody umożliwiające rejestrację nowych użytkowników, logowanie i wylogowywanie, a także aktualizację profilu użytkownika. Każdy użytkownik może mieć różne atrybuty, takie jak nazwa użytkownika, hasło, adres e-mail i adres dostawy.
4. Obsługa płatności: Powinny być metody umożliwiające przetwarzanie płatności za zamówienia. To może obejmować integrację z zewnętrznym dostawcą usług płatniczych.
5. Wysyłka: Powinny być metody umożliwiające obsługę wysyłki zamówień. To może obejmować integrację z zewnętrznym dostawcą usług logistycznych.

Warstwa prezentacji

Warstwa prezentacji dla platformy e-commerce zawiera następujące elementy:

1. Strony produktów: Strony te powinny wyświetlać szczegółowe informacje o każdym produkcie, takie jak nazwa, opis, cena i zdjęcie. Powinny również umożliwiać klientom dodawanie produktu do koszyka.
2. Strona koszyka: Strona ta powinna wyświetlać wszystkie produkty dodane do koszyka przez klienta. Powinna umożliwiać klientom zmianę ilości każdego produktu w koszyku, usuwanie produktów z koszyka i składanie zamówienia.
3. Strony zamówień: Strony te powinny wyświetlać szczegółowe informacje o każdym zamówieniu, takie jak lista produktów, łączna kwota, status zamówienia i szczegóły dostawy. Powinny również umożliwiać klientom śledzenie statusu swojego zamówienia.
4. Strony użytkowników: Strony te powinny umożliwiać użytkownikom rejestrację, logowanie i wylogowywanie, a także aktualizację swojego profilu.

Baza danych

Baza danych w projekcie wygląda następująco:

1. Klienci:
2. KlientID (klucz podstawowy)
3. Imie
4. Nazwisko
5. Email
6. Hasło
7. Numer telefonu
8. adres
9. Miasto
10. Państwo
11. Kod pocztowy
12. Kraj
13. Produkty:
14. ProduktID (klucz podstawowy)
15. Nazwa
16. Opis
17. Cena
18. KategoriaID (klucz obcy, odwołuje się do tabeli Kategorie)
19. ImageURL
20. Kategorie:
21. kategoriaID(klucz podstawowy)
22. Nazwa
23. Zamówienia:
24. ZamówienieID (klucz podstawowy)
25. ZamawiającyID (klucz obcy, odwołuje się do tabeli Klienci)
26. Data zamówienia
27. Szczegóły zamówienia:
28. Szczegoly zamówienia ID (klucz podstawowy)
29. Zamowienie ID (klucz obcy, odwołuje się do tabeli Orders)
30. ProduktID (klucz obcy, odwołuje się do tabeli Products)
31. Ilość
32. Płatność:
33. PłatnośćID (klucz podstawowy)
34. ZamowienieID (klucz obcy, odwołuje się do tabeli Zamowienia)
35. Data Platnosci
36. Kwota
37. Metoda Platności ID (klucz obcy, odwołuje się do tabeli MetodyPlatnosci)
38. PaymentMethods:
39. Metoda platnosciID (klucz podstawowy)
40. Nazwa
41. Typ

Diagram klas UML

(w osobnym pliku, opisac co i jak)

Przykłady:

<https://creately.com/diagram/example/h8fya1lm1/ecommerce-uml-class-diagram>

<https://online.visual-paradigm.com/pl/community/share/ecommerce-app-l6ibu78xn>

Dokumentacja techniczna kodu

(GitHub)