Warsztaty IV
sklep internetowy

v.1.1

Coders Lab
SZKOŁA PROGRAMOWANIA

Cel warsztatów

- Celem warsztatów jest napisanie prostego sklepu internetowego.
- Projekt ma być napisany obiektowo i do każdej z klas mają być napisane testy.
- Projekt jest grupowy (zespoły dwuosobowe).

Funkcjonalności aplikacji

Aplikacja ma implementować funkcjonalności przedstawione na tym slajdzie i na następnym.

Przedmioty
W sklepie mają być przedmioty mające swoją nazwę, opis, cenę.

Zdjęcia przedmiotów
Każdy przedmiot może mieć kilka zdjęć.
Zdjęcie w MySQL powinno trzymać ścieżkę do zdjęcia na dysku i id przedmiotu, do którego się odnosi.

Użytkownicy
Sklep powinien mieć możliwość założenia konta do robienia zakupów (bez tego robienie zakupów nie powinno być możliwe).
Użytkownik powinien mieć następujące dane:

- imię,
- nazwisko,
- email,
- hasło,
- adres do wysyłki.

Użytkownik ma mieć również możliwości zobaczenia historii swoich zakupów.

Funkcjonalności aplikacji

Zamówienie

Zatwierdzony koszyk zostaje przetworzony w zamówienie mające następujące dane:

- autor zamówienia,
- status zamówienia (oczekujące, złożone, opłacone, zrealizowane),
- zamówione przedmioty i ich liczba.

Zamówienie oczekujące to zamówienie jeszcze niezatwierdzone przez użytkownika (czyli jego koszyk).

Administratorzy

Strona ma implementować konta administratorów (inna tabelka niż użytkowników). Mają oni mieć tylko maila i hasło.

Strony, które musi mieć aplikacja

Strona główna naszego sklepu
Ma mieć miejsce do zalogowania się, link do rejestracji, menu z wszystkimi grupami przedmiotów i karuzelę z kilkoma wybranymi przedmiotami.

Strona rejestracji
Strona do rejestracji użytkownika. Ma pobierać wszystkie informacje o użytkowniku.

Panel użytkownika
Strona ta ma mieć informacje o użytkowniku, dawać opcje zmiany tych informacji, pokazywać wszystkie poprzednie zakupy tego użytkownika. Użytkownik może zobaczyć tylko swój panel.

Strona zamówienia
Ta strona ma pokazywać wszystkie informacje na temat zamówienia. Użytkownik może widzieć tylko swoje zamówienia.

Strona przedmiotu
Na tej stronie wyświetla się opis przedmiotu oraz jego zdjęcia w postaci karuzeli. Jest też możliwość dodania przedmiotu do koszyka obecnie zalogowanego użytkownika.

Strony, które musi mieć aplikacja

Strona koszyka

Na tej stronie powinny być następujące możliwości dotyczące przedmiotów znajdujących się w koszyku:

- wyświetlenie wszystkich przedmiotów,
- zmiana liczby przedmiotów,
- całkowite usunięcie przedmiotów,
- wyświetlenie łącznej kwoty zamówienia.

Strona zamówienia

Strona musi przyjąć wszystkie informacje dotyczące zamówienia:

- przedmioty i ich liczba,
- dane użytkownika (w tym adres do wysyłki),
- całkowita kwota zamówienia,
- informacje dotyczące płatności.

Panel administracyjny

Ma on być dostępny po wejściu na podstronę /panel.
Strona główna panelu ma mieć linki do zarządzania następującymi elementami:

- grupami,
- przedmiotami,
- użytkownikami,
- zamówieniami.

Zarządzanie grupami
Administrator ma mieć możliwość dodawania i usuwania grup dla przedmiotów.

Zarządzanie przedmiotami
Administrator powinien mieć możliwość dodania lub usunięcia przedmiotów.

Zarządzanie użytkownikami
Administrator powinien mieć możliwość zobaczenia danych użytkownika, wszystkich jego zakupów i usunięcia takiego użytkownika.

Zarządzanie zamówieniami
Administrator powinien widzieć wszystkie zamówienia (pogrupowane po ich aktualnym statusie), móc zmienić status zamówienia, usunąć dane zamówienie i wysłać wiadomość do danego klienta na temat tego zamówienia.

Podstawowy layout stron głównych zostanie zaprezentowany na zajęciach.

Przygotowanie

Zadanie 1. Przygotowanie

- Dobierzcie się w pary.
- W parze załóżcie jedno repozytorium na GitHubie (musi się znajdować na jednym z kont – obojętnie którym).
- Podepnijcie swoje nowe projekty do repozytorium i zobaczcie, czy działa (np. przez dodanie pliku readme na GitHubie i ściągnięciu go na oba komputery).

Zadanie 2. Połączenie do bazy danych

- Na jednym z komputerów wspólnie stwórzcie plik z połączeniem do bazy danych.
- Dane do połączenia każdy powinien trzymać w osobnym pliku (np. config.php), który nie może się znajdować w repozytorium (użyjcie do tego .gitignore).

Klasy

Zadanie 3a. Klasa przedmiotu

- Stwórz (w katalogu /src) klasę dla przedmiotu. Przedmiot ma mieć nazwę, cenę, opis, i dostępność (liczbę na stanie – int).
- Przygotuj wszystkie funkcje, które mogą być przydatne dla tej klasy.
- Przygotuj relację jeden do wielu z tabelką, która trzyma zdjęcia w bazie danych.

Zadanie 3b. Klasa użytkownika

- Stwórz (w katalogu /src) klasę dla użytkownika.
- Użytkownik ma mieć wszystkie typowe informacje: imię, nazwisko, mail i hasło.
- Przygotuj wszystkie potrzebne funkcje przydatne dla tej klasy.

Synchronizacja

Zadanie 3c. Synchronizacja

- Wyślij swoje commity do repozytorium i pobierzcie kod drugiej osoby.
- Pamiętajcie o modyfikacji swoich baz danych.
- Przetestuj kod drugiej osoby (przez mały skrypt, który tworzy obiekt danej klasy i na niej działa).
- Sprawdźcie, czy wszystko jest ok – jeżeli nie, to wspólnie znajdźcie błąd. Następnie do repozytorium dodajcie fix (kod naprawiający ten błąd).
- Jeżeli ukończysz swoje funkcjonalności wcześniej niż druga osoba, to pomóż jej w pracy.

Klasa administratora

Zadanie 4a. Klasa administratora i wiadomości

- Stwórz klasę dla administratora.
- Ma on mieć nazwę, mail i hasło.
- Przygotuj wszystkie funkcje, które mogą być przydatne dla tej klasy.
- Przygotuj klasę wiadomości – powinna mieć ona tekst wiadomości i użytkownika, do którego jest skierowana.
- Wiadomości w naszym systemie mają służyć do powiadamiania użytkownika o różnych sytuacjach (np. wysłaniu do niego paczki, przyjęciu zamówienia itp.).
- Administrator powinien mieć możliwość wysyłania wiadomości.
- Dodaj do klasy User funkcje, które zwracają wszystkie wysłane do niego wiadomości.

Klasa zamówienia

Zadanie 4b. Klasa zamówienia

- Stwórz klasę dla zamówienia.
- Ma ona mieć następujące relacje:
  - jeden do wielu z użytkownikiem,
  - wiele do wielu z przedmiotami.
- Zamówienie ma mieć swój stan (niezłożone, złożone, opłacone, zrealizowane).
- Dodaj do klasy User funkcję zwracające jego koszyk i wszystkie zamówienia (poza koszykiem).

Klasa zamówienia

Zadanie 4c. Synchronizacja

- Wyślij swoje commity do repozytorium i pobierzcie kod drugiej osoby.
- Pamiętajcie o modyfikacji swoich baz danych.
- Przetestuj kod drugiej osoby (przez mały skrypt, który tworzy obiekt danej klasy i na niej działa).
- Sprawdźcie, czy wszystko jest ok – jeżeli nie, to wspólnie znajdźcie błąd. Następnie do repozytorium dodajcie fix (kod naprawiający ten błąd).
- Jeżeli ukończysz swoje funkcjonalności wcześniej niż druga osoba, to pomóż jej w pracy.

Panel administracyjny

Zadanie 5a. Panel administracyjny

- Stwórz panel administracyjny (w katalogu /admin) według podanych wcześniej wytycznych.

Zadanie 5b. Sklep

- Stwórz sklep (w katalogu /shop) według podanych wcześniej wytycznych.
- Jeżeli ukończysz swoje funkcjonalności wcześniej niż druga osoba, to pomóż jej w pracy.

Synchronizacja

Zadanie 5c. Synchronizacja

- Wyślij swoje commity do repozytorium i pobierzcie kod drugiej osoby.
- Pamiętajcie o modyfikacji swoich baz danych.
- Przetestuj kod drugiej osoby (przez mały skrypt, który tworzy obiekt danej klasy i na niej działa).
- Sprawdźcie, czy wszystko jest ok – jeżeli nie, to wspólnie znajdźcie błąd. Następnie do repozytorium dodajcie fix (kod naprawiający ten błąd).
- Jeżeli ukończysz swoje funkcjonalności wcześniej niż druga osoba, to pomóż jej w pracy.

Epilog

- Pod koniec pamiętajcie o tym, żeby osoba, która nie ma repozytorium, zrobiła fork na swoje konto GitHuba.
- Dzięki temu będzie miała takie samo repozytorium u siebie.
- Osoba, która to zrobi, musi też pamiętać o zmianie adresu **origin** w swoim repozytorium, dzięki temu będzie commitować do repozytorium swojego, a nie do koleżanki lub kolegi.