Kacper Głowacki 43118

Mateusz Woźniak 43138

Informatyka E-commerce Developer 2023

**Wstęp:**

Naszym programem jest prosty sklep internetowy napisany w PHP Framework Symfony używający baz danych w celu przechowywania informacji o klientach oraz produktach. Do przechowywania projektu używamy GitLaba natomiast do planowania zadań Trello. W celu postawienia lokalnego serwera użyliśmy xampa’a. Potrzebowaliśmy też zainstalować composer, symfony. Aby uruchomić aplikację potrzebujemy php w wersji minimum 8.1, composer i symfony. Aplikacja jest napisana we wzorcu projektowym MVC – model view controller. Aplikacja ma być dostępna bez przerwy 24/7/365. Strona ma działać bardzo szybko – wszystkie operacje zajmują maksymalnie 1 s (Chyba, że ograniczeniem byłby kogoś wolny internet). Aplikacja ma działać na wszystkich przeglądarkach internetowych na systemach windows/linux/android.

Sklep internetowy posiada funkcjonalności CRUD – create, read, update, delete umożliwiające użytkownikowi z uprawnieniami administratora na tworzenie bazy produktów, odczytywania jej, aktualizowania oraz usuwania czyli pełnej edycji. Oprócz roli administatora mamy też rolę domyślną którą otrzymuje każdy zarejestrowany użytkownik. Użytkownik z tą rolą nie posiada uprawnień dodawania, usuwania oraz edytowania listy produktów może jedynie zakupić produkt z naszej oferty. Strona zawiera panel do rejestracji nowych użytkowników oraz stronę logowania istniejącego już klienta lub użytkownika z uprawnieniami administratora.

Użytkownik w tym systemie będzie mógł zakupić jakiś produkt wcześniej dodany przez administratora. Po kliknięciu na przycisk „Buy” użytkownik zostanie przekierowany do formularza w którym będzie poproszony o podanie swoich danych do wysyłki(Name, Surname, City, Post code, Address). Następnie po ponownym kliknięciu przycisku „Buy” użytkownik zostanie poinformowany o sfinalizowaniu procesu. W przypadku braku produktu w magazynie użytkownik zostanie o tym poinformowany przez system.

Po złożeniu zamówienia przez klienta zostanie ono przeniesione do tabeli „client\_info” w bazie danych gdzie zawarte są wszystkie informacje które przekazał nam użytkownik oraz numer produktu który zamówił z tabeli crud(tabela z produktami). Ta sama baza danych zawiera również tabele „user” z informacjami o zarejestrowanych użytkownikach oraz administratorach na stronie.

Oprócz interfejsu graficznego na stronie możemy wysyłać wszystkie zapytania rest api takie jak:

GET, POST, PUT, PATCH, DELETE.

Z api może korzystać tylko administrator.

Logowanie przez program postman wygląda następująco:

Pierwsze co musimy zrobić to wysłać GET request na stronę logowania naszej strony i z sekcji body skopiować: „\_csrf\_token”.

Następnie wchodzimy w sekcję Body ustawiamy opcję x-www-form-urlencoded.

Obraz zawierający tekst, numer, Czcionka, linia

Opis wygenerowany automatycznieI wpisujemy te same passy jakie używamy na naszej normalne stronie oraz csrf token.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznieZmieniamy typ żądania na POST i po kliknięciu przycisku send zostaniemy pozytywnie zalogowani.

**Lista produktów:**

Przykład żądania GET: Obraz zawierający tekst, oprogramowanie, numer, Ikona komputerowa

Opis wygenerowany automatycznie

Przykład żądania POST:

Obraz zawierający tekst, oprogramowanie, numer, Ikona komputerowa

Opis wygenerowany automatycznie

Przykład żądania PUT:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Przykład żądania PATCH:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Przykład żądania DELETE:

Obraz zawierający tekst, oprogramowanie, Ikona komputerowa, numer

Opis wygenerowany automatycznie

**Lista zamówień:**

Przykład żądania GET:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Przykład żądania POST:

Obraz zawierający tekst, oprogramowanie, numer, Ikona komputerowa

Opis wygenerowany automatycznie

Przykład żądania PUT:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Przykład żądania PATCH:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

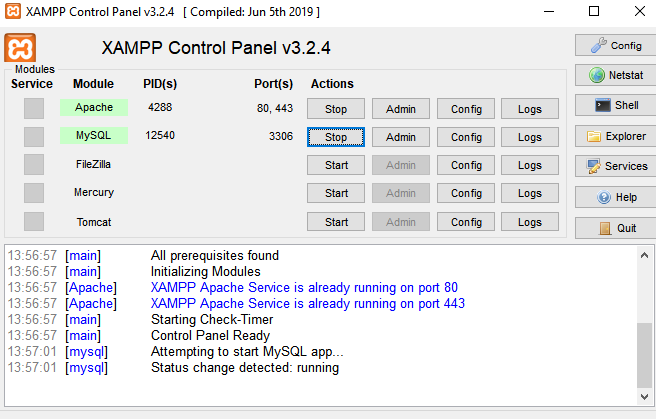
Przykład żądania DELETE:

Obraz zawierający tekst, oprogramowanie, Ikona komputerowa, zrzut ekranu

Opis wygenerowany automatycznie

**Instalacja:**

Aby zainstalować aplikację należy sklonować repozytorium: github.com/kacperjcb/ProjektSpecjalnosciowy2 lub pobrać plik zip z aplikacją i rozpakować go w dowolnym miejscu. Następnym krokiem jest uruchomienie xampp’a i włączenie Apache i MySQL.



Następnie wchodzimy w folder zawierający nasz projekt i wpisujemy następujące komendy:

Composer install – aby zainstalować wszystkie potrzebne pakiety dla naszego projektu

php bin/console doctrine:database:create – aby stworzyć bazę danych. Nazwę oraz typ bazy danych można zmienić w pliku .env. Można też użyć innej bazy np. postgress, Oracle, sqlite, mariadb.

Ale w naszym przykładzie używamy xampa i bazy MariaDB.

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznie

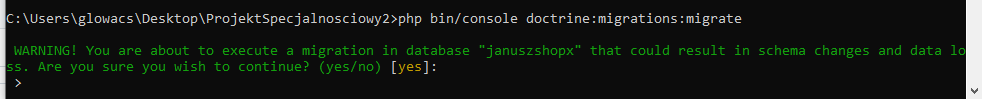
Następnie używamy komendy: php bin/console doctrine:migrations:migrate – aby zmigrować dane do bazy danych. (W przypadku innych typów baz danych niż mariadb/mysql wcześniej powinniśmy usunąć migrację i edytować plik .env aby obługiwał inną bazę danych i przed powyższą komendą powinniśmy najpierw stworzyć migrację następującą komendą: php bin/console make:migration.

Dokumentacja od doctrine orm którego użyliśmy znajduję się po tym linkiem:

https://symfony.com/doc/current/doctrine.html

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznieObraz zawierający zrzut ekranu, tekst, Czcionka

Opis wygenerowany automatycznie

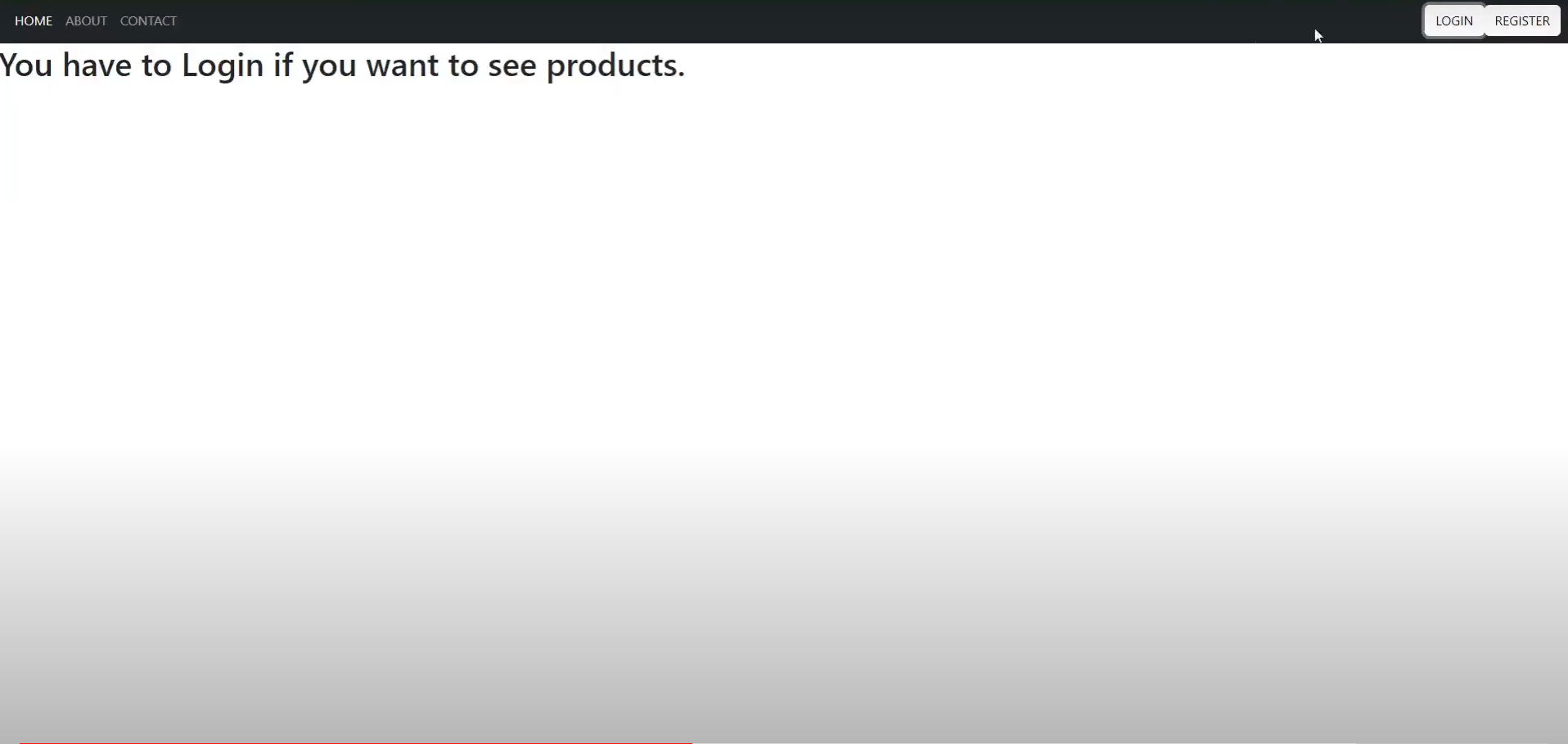
Teraz możemy uruchomić lokalny serwer symfony za pomocą komendy symfony serve -d lub symfony server:start

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznie

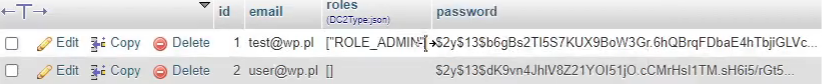
Nasza strona jest dostępna pod adresem 127.0.0.1:8000 lub localhost:8000.

Czas na stworzenie użytkownika. Wchodzimy więc na stronę i rejestrujemy się w prawym górnym rogu klikając „Register”.



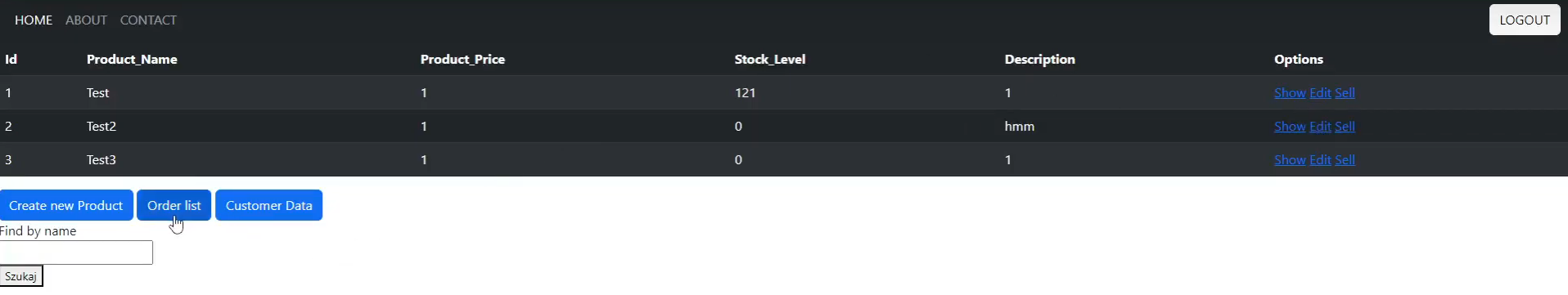
Jeśli chcemy nadać uprawnienia administratora – bez tego nie będziemy mogli dodać żadnego produktu. Wchodzimy do bazy danych i ustawiamy roles na [„ROLE\_ADMIN”].

Po ponownym zalogowaniu użytkownik w tym przypadku [test@wp.pl](mailto:test@wp.pl) dostanie wszystkie uprawnienia.



**Panel Administratora:**

Tak wygląda widok administratora po zalogowaniu wraz z przykładowymi produktami.

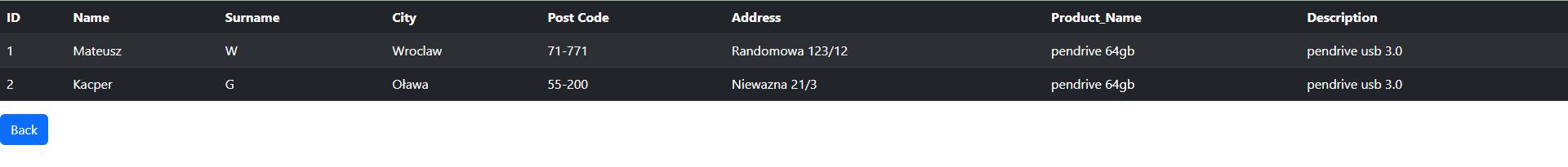


Możemy dodać nowy produkt klikając na przycisk – „Create new product”

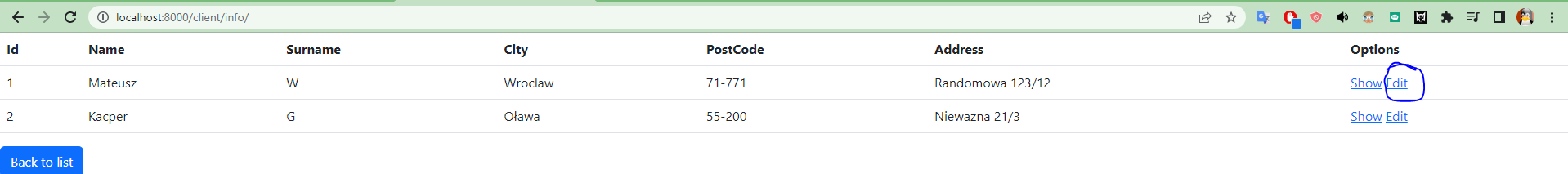
Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Wejść w listę zamówień

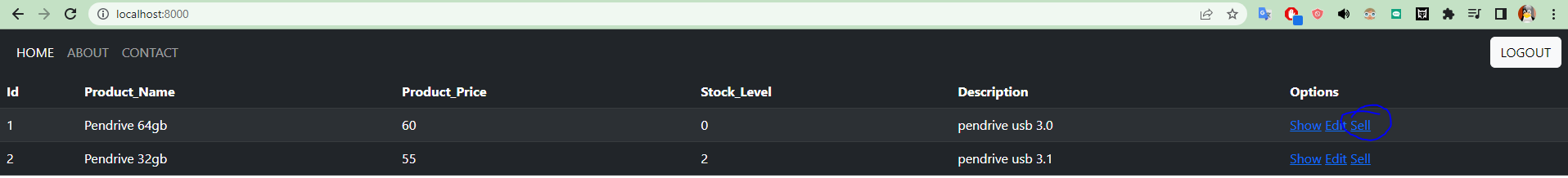


Oraz wejść w bazę klientów skąd również możemy edytować informację o nich

 Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Opcja sell zmniejsza Stan magazynowy o jeden i przekierowuje nas do formularza w którym podajemy dane kupującego. Czyli przykładowo ktoś się do nas zgłosił, żeby coś mu kupić i chcemy to wprowadzić w systemie i wprowadzamy to z perspektywy administratora. Ta podstrona różni się od podstrony którą widzi kupujący tylko tym, że ma dopisane – you are login as admin.

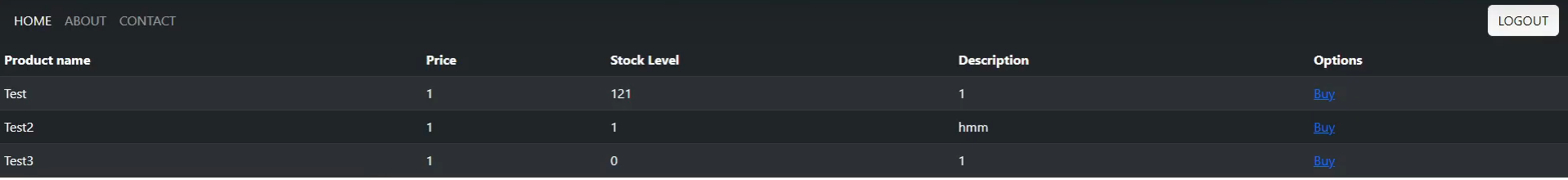


Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

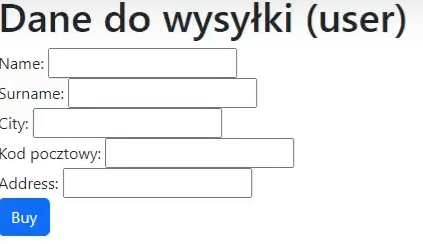
Opis wygenerowany automatycznie

**Panel Klienta:**

Z perspektywy klienta po zalogowaniu wygląda to następująco



Przycisk „Buy” znajduje się przy każdym produkcie. Jeśli chcemy zakupić jakiś produkt musimy kliknąć w odpowiedni przycisk przypisany do niego. Przekieruje nas to do formularza wysyłkowego dla klienta



Po ponownym kliknięciu „Buy” pojawi nam się komunikat z podziękowaniami za zakup.



Jeśli ilość produktów na stanie wynosi 0 to zostaniemy przekierowani do podstrony która nas informuje o tym, że nie ma produktów na stanie. Obraz zawierający tekst, oprogramowanie, zrzut ekranu

Opis wygenerowany automatycznie

**Omówienie kodu:**

**Struktura plików wygląda następująco:**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Plik .env w którym ustawiamy naszą bazę danych wygląda w ten sposób:**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Entity crud i Entity client info. Z tych plików orm tworzy bazę danych.**

**Pliki w większości są generowane automatycznie przez symfony.**

**Głowna zmiana w tych plikach to dodanie funkcji jsonSerialize która była potrzebna aby można było używać funkcji api. Bez niej przykładowy get request zwracałby tylko puste obiekty.**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający zrzut ekranu, tekst, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Kontrolery:**

**Crud kontroler zawiera przekierowania do podstron. Całą logikę strony głównej**

Obraz zawierający zrzut ekranu, tekst, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

**ClientInfo controller zawiera logikę strony w którym klient wprowadza dane:**

**Funkcja /new otrzymuje dzięki metodzie get, id produktu z głównej strony dzięki temu można później dojść do tego kto kupił jaki produkt.**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

**W ApiController zaimplementowane są wszystkie funkcję dla api crud oraz api clientinfo. Na górze pliku jest wprowadzona autentykacja aby tylko użytkownik posiadający rolę admin mógł korzystać z poniższych funkcji.**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

**Prametr OrderNumber zostaje przekazany jako ukryte pole w formularzu klienta. I przyjmuje on wartość id produktu.**

Obraz zawierający zrzut ekranu, tekst, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Formularz edycji danych klienta musi być lekko zmodyfikowany ponieważ przy edycji klienta wcześniej występował problem bo znikało pole OrderNumber**

Obraz zawierający zrzut ekranu, tekst, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Tak wygląda form2**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**A tak zwykły form:**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Do naszego projektu użyliśmy szablonów twig które umożliwiły nam na ograniczenie dostępu. Przykładowo tabelę orders może zobaczyć tylko administrator.**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Szablon głównej strony wygląda tak:**

**ROLE\_USER – jest to rola której nie trzeba wpisywać w bazę danych ponieważ każdy zarejestrowany użytkownik ją otrzymuje.**

**Jeśli ktoś odwiedza stronę i nie jest ani użytkownikiem ani administratorem to widzi tylko przycisk od rejestracji.**

**Jeśli użytkownik jest zalogowany jako role\_user albo administrator – każdy administrator posiada też oczywiście role usera to przycisk od rejestracji znika a pojawia się przycisk logout.**

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

**Na samym końcu pliku jest ustawione jaki widok powinien zobaczyć zalogowany użytkownik**

Obraz zawierający tekst, zrzut ekranu, wyświetlacz, oprogramowanie

Opis wygenerowany automatycznie

**Plik security.yaml wygląda następująco – można w nim dodać też kolejne role jeśli była by taka potrzeba.**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Funkcja która służy do tego aby wypisać użytkownika wraz z tym co zakupił znajduje się w pliku CrudRepository i wygląda następująco:**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

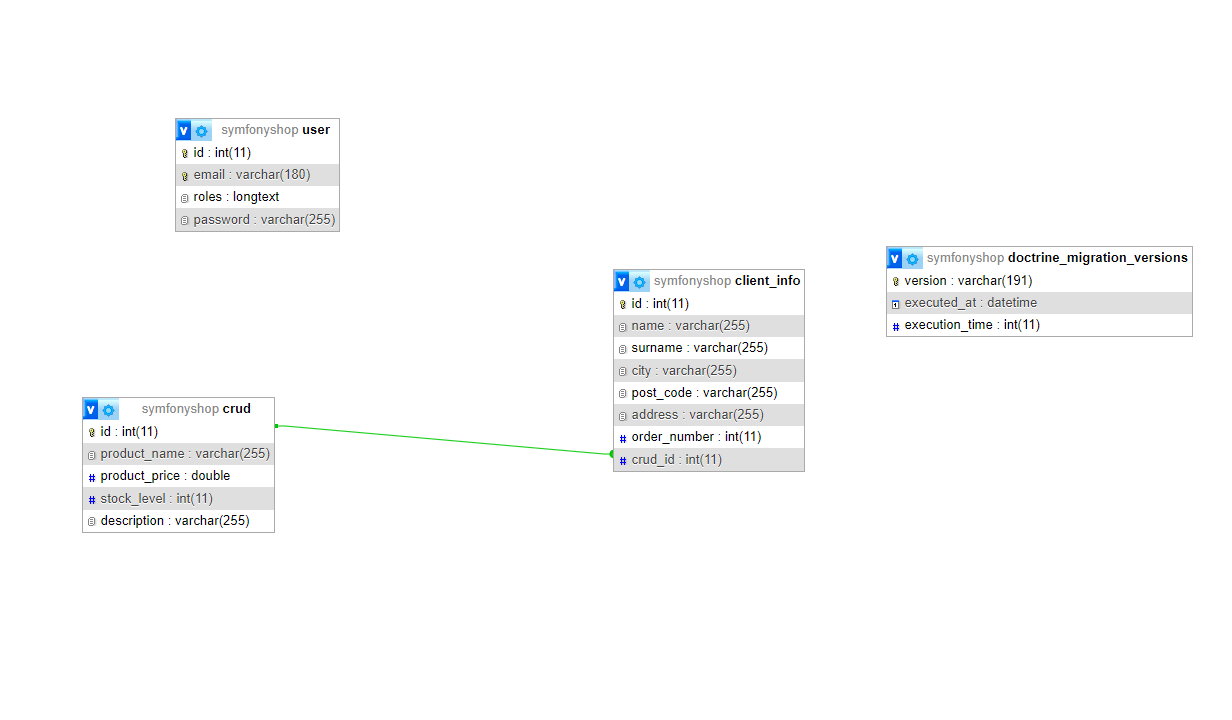
Opis wygenerowany automatycznie

**Strona posiada też wyszukiwarkę dzięki której można wyszukać jakiś produkt po nazwie. Gdy naciśniemy przycisk „search” to wyświetlą nam się tylko produkty które zawierają w nazwie tekst który wpisaliśmy – zarówno w przypadku gdy jesteśmy zalogowani jako admin jak i użytkownik.**

Obraz zawierający oprogramowanie, Oprogramowanie multimedialne, tekst, Oprogramowanie graficzne

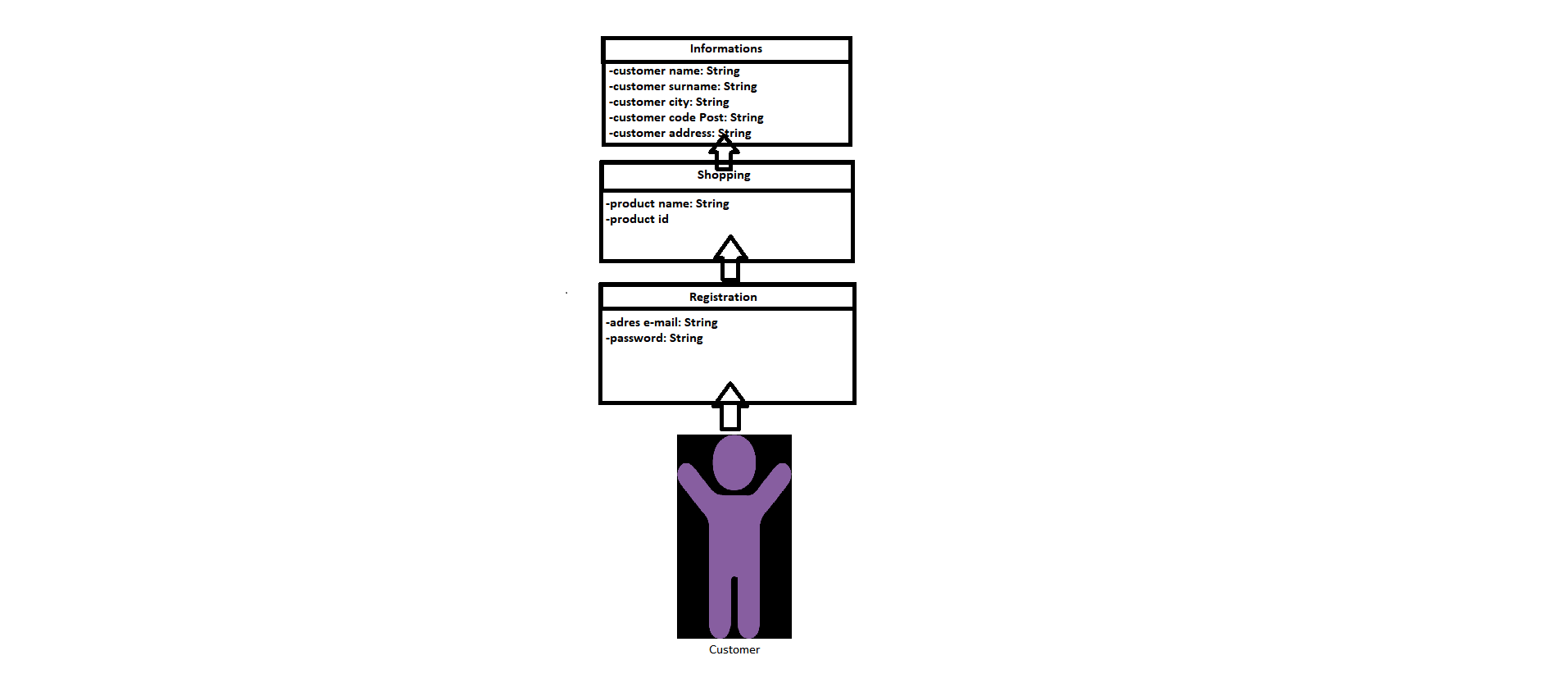
Opis wygenerowany automatycznie

**Diagram ERD:**



**Diagram UML:**

**Customer**



**Admin**

