

Państwowa Wyższa Szkoła Zawodowa
im. Witelona w Legnicy

Sprawozdanie z projektu grupowego aplikacji
internetowej „PizzaOrders”

Projektowanie i programowanie systemów internetowych II

Prowadzący:

mgr inż. Krzysztof Rewak

Konrad Faliński

Filip Fonał

Mateusz Lencki

Adam Pankiewicz

Informatyka

Programowanie aplikacji mobilnych i internetowych

17 stycznia 2019

1 Opis funkcjonalny

Pizza orders to platforma internetowa przeznaczona dla restauracji typu pizzeria i ich klientów. Główną funkcjonalnością aplikacji jest graficzny kreator pizzy, w którym klient danej pizzerii może edytować już istniejące w menu rodzaje pizz lub może stworzyć całkowicie nowe kompozycje, wybierając odpowiednie pozycje z listy dostępnych w danej restauracji składników. Kreator pizzy ma postać edytora typu drag & drop, w którym klient przeciąga składniki na ciasto na którym automatycznie jest prezentowany końcowy wygląd tworzonej pizzy.

Oprócz kreatora pizzy aplikacja wspomaga proces zamawiania jedzenia w restauracji. Po cząwszy od złożenia zamówienia na stworzoną pizzę, poprzez odebranie zamówienia w kuchni, statusowanie przez kucharza postępów w przetwarzaniu zamówienia i skończywszy na dostarczeniu pizzy do klienta. Zmiany w statusach zamówienia są pokazywane klientowi w czasie rzeczywistym.

Właściciel pizzerii może zarejestrować swoją restaurację w aplikacji, dodać kucharzy i managerów magazynu. Jest możliwość zdefiniowania menu restauracji. Po wypełnieniu wszystkich danych o restauracji można włączyć możliwość zamawiania pizzy przez klientów, zgłaszając tym samym zgłoszenie do administratora systemu aby ten opublikował naszą restaurację. Od momentu zatwierdzenia restauracji przez administratora jest ona widoczna w wyszukiwarce pizzerii a klienci mogą zamówić jedzenie.

Kolejną ważną funkcjonalnością jest panel zarządcy. Właściciel lub zarządca pizzerii ma dostęp do modułu, w którym można określać dostępność danych składników w restauracji i ich cenę. Składniki, które zostaną oznaczone jako dostępne są prezentowane końcowemu użytkownikowi w kreatorze pizzy. Mają one wtedy taką cenę jaką ustali zarządca restauracji.

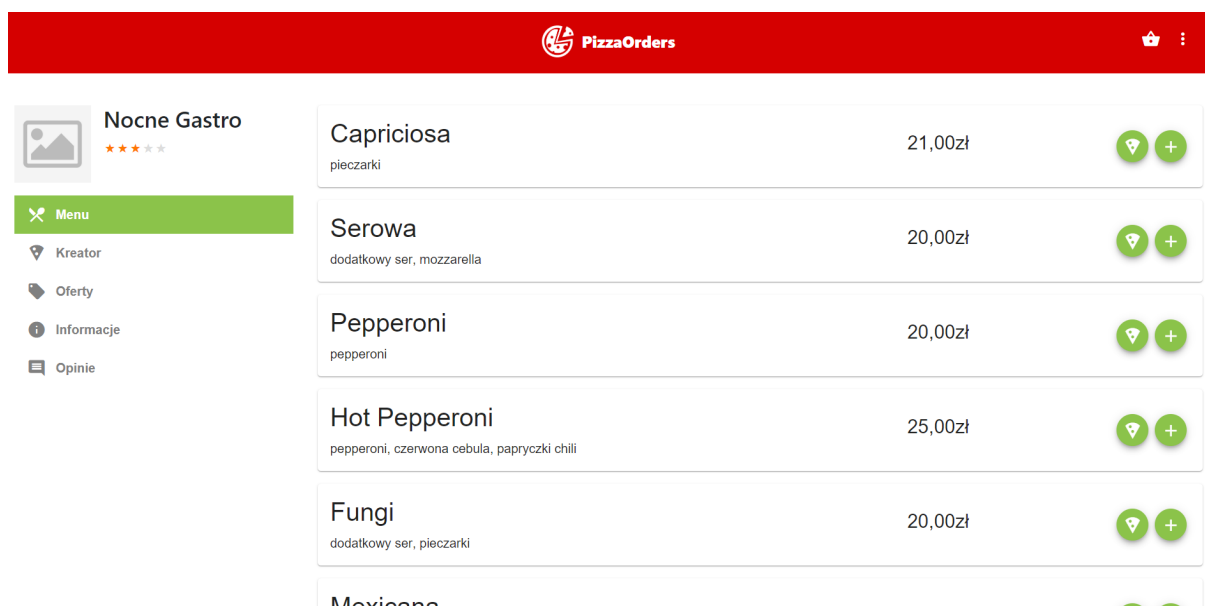
Następnym elementem systemu jest panel administracyjny, w którym osoba o szczególnych uprawnieniach może zarządzać kluczowymi danymi w systemie takimi jak lista definicji składników pizzy, listy restauracji, użytkowników. Ma dostęp do logów i narzędzi analizy ruchu.

Na następnych stronach przedstawiono zrzuty najważniejszych ekranów systemu, które ukazują finalny wygląd aplikacji.

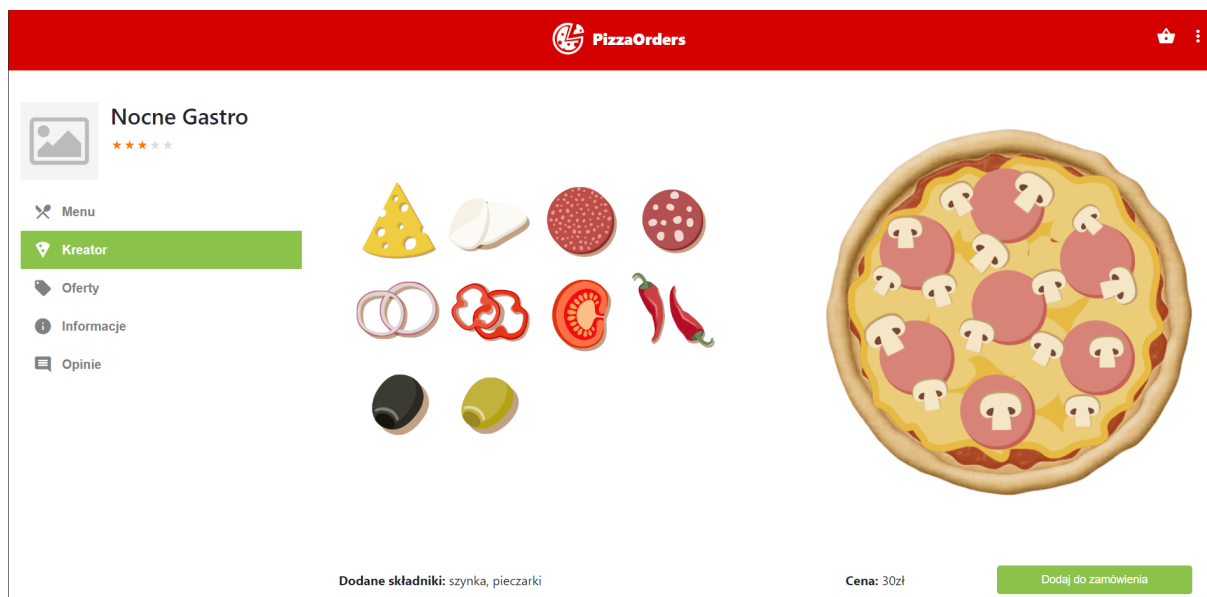
Główny widok aplikacji. Wyszukiwarka restauracji.



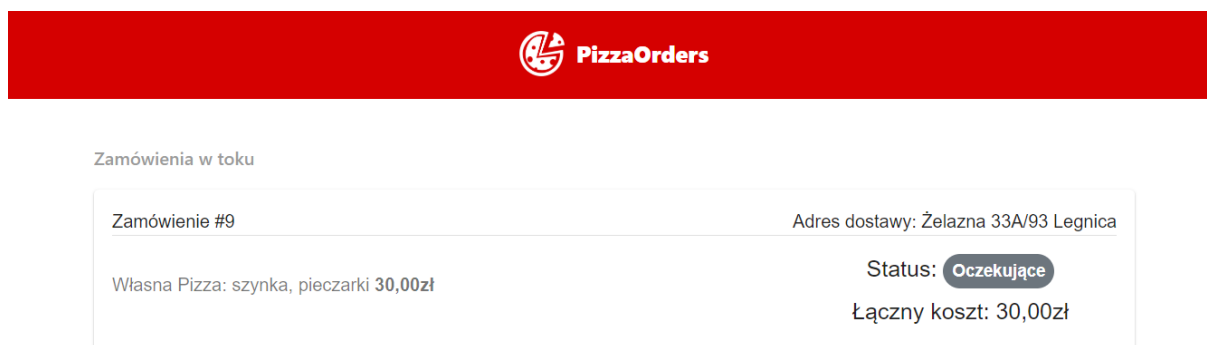
Menu pojedynczej restauracji.



Kreator własnej pizzy.



Widok ze śledzeniem zamówienia.



Widok panelu kucharza z aktywnymi zamówieniami.

Pizza Station 21

Kokpit

Zamówienia

Magazyn

Menu

Pracownicy

Dane restauracji

Wyjście

Wyloguj

Wszystkie

Oczekujące

Przyjęte

W realizacji

Dostarczane

Gotowe

Zamówienie #6

Oczekujące

Koszt: 43,00

Adres dostawy: Sowińskiego Józefa 69A/24 Sosnowiec

Telefon: 123123123

Zmodyfikowana 'Mexicana': salami, czerwona papryka, papryczki chili + (salami, czerwona papryka, papryczki chili)

Cena: 43,00

Zmień status

Zamówienie #8

Oczekujące

Koszt: 20,00

Adres dostawy: Sowińskiego Józefa 69A/24 Sosnowiec

Telefon: 123123123

Pizza 'Capriciosa': szynka, pieczarki

Cena: 20,00

Zmień status

Widok zarządcy magazynu z dostępnymi składnikami.

Pizza Station 21

Kokpit

Zamówienia

Magazyn

Menu

Pracownicy

Dane restauracji


Wyjście

Wyloguj

Dostępne składniki

Składniki

Dodaj składnik




dodatkowy ser

Dostępny

Cena (zł)

4

Zapisz




szynka

Dostępny

Cena (zł)

4

Zapisz




mozzarella

Dostępny

Cena (zł)

7

Zapisz




salami

Dostępny

Cena (zł)

7

Zapisz




pepperoni

Dostępny

Cena (zł)

5

Zapisz




czerwona cebula

Dostępny

Cena (zł)

6

Zapisz




czerwona papryka

Dostępny

Cena (zł)

4













































Zapisz



Cena (zł)

5

Panel administratora do zarządzania definicjami składników.

Administracja					
<div><div></div><div>Kokpit</div><div></div><div>Restauracje</div><div></div><div>Użytkownicy</div><div></div><div>Składniki</div><div></div><div>Logi</div><div></div><div>Wyjście</div><div></div><div>Wyloguj</div></div>	Szukaj		Dodaj składnik		
	ID	Obrazek	Miniaturka	Nazwa	
	1			dodatkowy ser	 
	2			szynka	 
	3			mozzarella	 
	4			salami	 
	5			pepperoni	 
	6			czerwona cebula	 
	7			czerwona papryka	 
	8			pieczarki	 
	9			pomidory	 
	10			papryczki chili	 
	11			oliwki czarne	 

2 Opis technologiczny

Projekt jest zrealizowany w postaci aplikacji internetowej z wykorzystaniem różnorodnych technologii internetowych. Warstwa backendu i warstwa prezentacji to dwa osobne aplikacje, które wymieniają dane między sobą za pomocą udokumentowanego API. Dlatego też opis technologiczny można podzielić na dwie poniższe sekcje.

2.1 Backend

Backend aplikacji został zrealizowany z użyciem języka programowania PHP 7. Do innych technologii i narzędzi wykorzystanych w projekcie należą:

- Laravel 5.7 - framework języka PHP.
- Baza danych MySQL.
- Pusher.

2.2 Frontend

Przy tworzeniu aplikacji frontendowej wykorzystano:

- Angular 7.1.
- Material 7.2.
- Leaflet 1.3

3 Instrukcja uruchomienia

3.1 Backend

- Pobranie projektu z repozytorium i pobranie zależności:

```
git clone https://github.com/ppsi2-pizza-orders ...  
.../pizza-orders-api
```

```
cd pizza-orders-api
```

```
composer install
```

- Konfiguracja pliku .env:

```
cp .env.example .env
```

W pliku .env należy ustawić odpowiednie dane dostępu do bazy danych itp.

```
php artisan key:generate
```

- Uruchomienie kontenerów dockerowych:

```
docker-compose up -d
```

- Migracja tabel w bazie danych i seed początkowych danych:

```
php artisan migrate
```

```
php artisan db:seed
```

- Wygenerowanie klucza JWT AUTH:

```
php artisan jwt:secret
```

- Aplikacja powinna być dostępna pod adresem:

```
localhost:8080
```


3.2 Frontend

- Pobranie projektu z repozytorium:

```
git clone https://github.com/ppsi2-pizza-orders ...  
.../pizza-orders-frontend  
cd pizza-orders-frontend
```

- Budowa kontnera dockerowego:

```
docker-compose build
```

- Uruchomienie aplikacji:

```
docker-compose up
```

- Budowanie aplikacji:

```
docker exec -it pizza-orders-frontend bash  
ng build
```

- Aplikacja jest dostępna pod adresem:

```
localhost:4200
```

4 Instrukcja uruchomienia testów i opis testowanych funkcjonalności

Pisząc testy, skorzystaliśmy z narzędzia PHPUnit do testowania backendu naszej aplikacji. Większość z udokumentowanych endpointów została przetestowana za pomocą testów integracyjnych. Przetestowane zostały zarówno scenariusze domyślne jak i progowe, które powinny zostać odpowiednio obsłużone przez naszą aplikację. W celu uruchomienia testów należy odpowiednio skonfigurować środowisko testowe.

- Należy skonfigurować plik `.env.testing` i ustawić w nim odpowiednie dostępy do bazy testowej:

```
cp .env .env.testing
```

- Migracja tabel do bazy testowej i uzupełnianie jej danymi testowymi:

```
php artisan migrate --database=mysql_testing  
php artisan db:seed --database=mysql_testing
```

- Uruchomienie wszystkich testów:

```
php vendor/phpunit/phpunit/phpunit
```

5 Wnioski projektowe

Odseparowanie backendu od warstwy prezentacji w postaci tworzenia dwóch oddzielnych aplikacji w projektach bardziej złożonych od najprostszych stron internetowych jest dobrym rozwiązaniem i rozsądnym podejściem do rozwijania nowoczesnych aplikacji internetowych. Takie rozwiązanie pozwala na lepszą koordynację prac między dwoma działami deweloperskimi zajmującymi się osobno wyglądem aplikacji i implementacją logiki biznesowej. Wymienianie danych w aplikacji za pomocą ustandaryzowanego i udokumentowanego API ułatwia pracę programistom frontendu jak i backendu. Człowiek odpowiedzialny za wygląd aplikacji, otrzymując dokumentację API nie musi zastanawiać się nad szczegółami implementacyjnymi funkcjonalności backendu i tak samo programista wewnątrz systemu zwolniony jest z odpowiedzialności dostosowywania swojego kodu do wymagań frontendu. Podejście to pozwala również na połączenie innych aplikacji klienckich, takich jak np. aplikacja mobilna, z już stworzonym API.

Programowanie reaktywnych aplikacji frontendowych z użyciem różnorodnych framework'ów, takich jak używany przez nas Angular, jest dużym ułatwieniem dla programisty. Korzystanie z gotowych mechanizmów zdarzeń, obserwatorów, dwustronnego wiązania zmiennych sprawia, że taka aplikacja powstaje szybciej w porównaniu do tradycyjnego podejścia stosowanego w przeszłości - ręcznego modyfikowania drzewa DOM dokumentu HTML za pomocą czystego języka JavaScript.

Stosowanie, już sprawdzonych wzorców i technik również pozwala na szybsze i łatwiejsze tworzenie zaawansowanych aplikacji internetowych. Wykorzystany przez nas PHP-owy framework Laravel implementuje szeroko stosowany wzorzec MVC - Model, View, Controller. Stawia on pewne ramy, których trzymanie się ułatwia pisanie kodu wysokiej jakości. Dzięki stosowaniu paradygmatu Inversion of Control, który implementujemy stosując wstrzykiwanie odpowiednich zależności do używanych przez nas klas, zwiększa modularność i skalowalność całości tworzonej aplikacji.

6 Linki.

Opublikowana aplikacja:

<https://pizzaorders.pl/>

Repozytoria projektu:

<https://github.com/ppsi2-pizza-orders>

Dokumentacja API:

<https://api.pizzaorders.pl/api/documentation>