Car Rental – aplikacja webowa do zarządzania wypożyczalnią samochodów

Konrad Rejman
Mateusz Matyaszek

Spis treści

- Główne założenia aplikacji
- Technologie wykorzystane w projekcie
- Schematy UML
- Środowisko
- Uruchomienie aplikacji

Główne założenia aplikacji

Aplikacja służy do rezerwacji i wypożyczenia samochodów online. Klient za pośrednictwem strony internetowej ma dostęp do informacji na temat samochodów które może wypożyczyć oraz ich aktualną dostępność.

Aplikacji ma przeprowadzić użytkownika przez cały proces rezerwacji samochodów.

Witryna pozwala na tworzenie kont internetowych, aby użytkownik nie musiał wprowadzać za każdym razem swoich danych osobistych podczas głównego procesu.

- Użytkownik niezalogowany:

  Osoba niezalogowana ma dostęp jedynie o przeglądu oferowanych samochodów

- Użytkownik zalogowany:

  Po zalogowaniu użytkownik oprócz funkcji użytkownika niezalogowanego otrzymuje dodatkowe możliwości. Dzięki nim ma dostęp do podglądu danych dotyczących jego konta z możliwością ich edycji oraz do wypożyczenia samochodu.

- Administrator:

  Jako osoba z największymi uprawnieniami administrator ma dostęp do wszystkich informacji w bazie danych. Dzięki uprawnieniom może finalizować wypożyczenie wydając lub odbierając samochód od klienta.

Aplikacja może znaleźć zastosowanie w branży usługowej, gdzie pozwoli. Umożliwi lepsze zarządzanie biznesem i zoptymalizuje koszty jego. A poprzez wykorzystanie go do stworzenia profilu klienta umożliwi późniejsze przedstawienie mu bardziej sprecyzowanej indywidualnej oferty.

Technologie wykorzystane w projekcie:

- Python
- Django
- Docker
- Jenkins
- Github

Encje

CarModel

- id- integer NOT NULL PRIMARY KEY autoincrement
- mark – char(200)
- model – char(200)
- sort – char(200)
- doors – char(1)
- desc – text
- price – decimal(6)

Car

- id – integer NOT NULL PRIMARY KEY autoincrement
- plate - char(7)
- color – char(100)
- carModel – integer NOT NULL foreign key
- available - boolean

Rental

- Id – integer NOT NULL PRIMARY KEY autoincrement
- User – integer NOT NULL foreign key
- Car – integer NOT NULL foreign key
- dateOfRental – date
- dateOfReturn – date

Contact

- Id – integer NOT NULL PRIMARY KEY autoincrement
- User – integer NOT NULL foreign key
- Street – char(100)
- City – char(100)
- Code – char(6)
- Phone – char(9)

User

- Username -varcher
- first_name - varchar
- last_name – varchar
- password – varchar
- email – varchar

Schematy UML

Diagram przypadków użycia

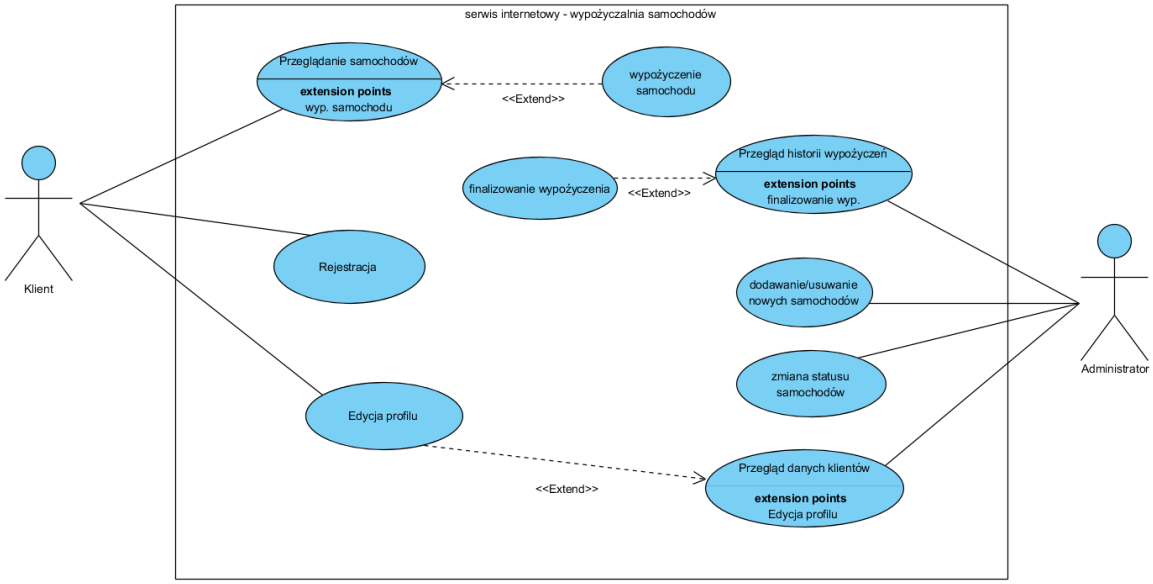


Diagram stanów

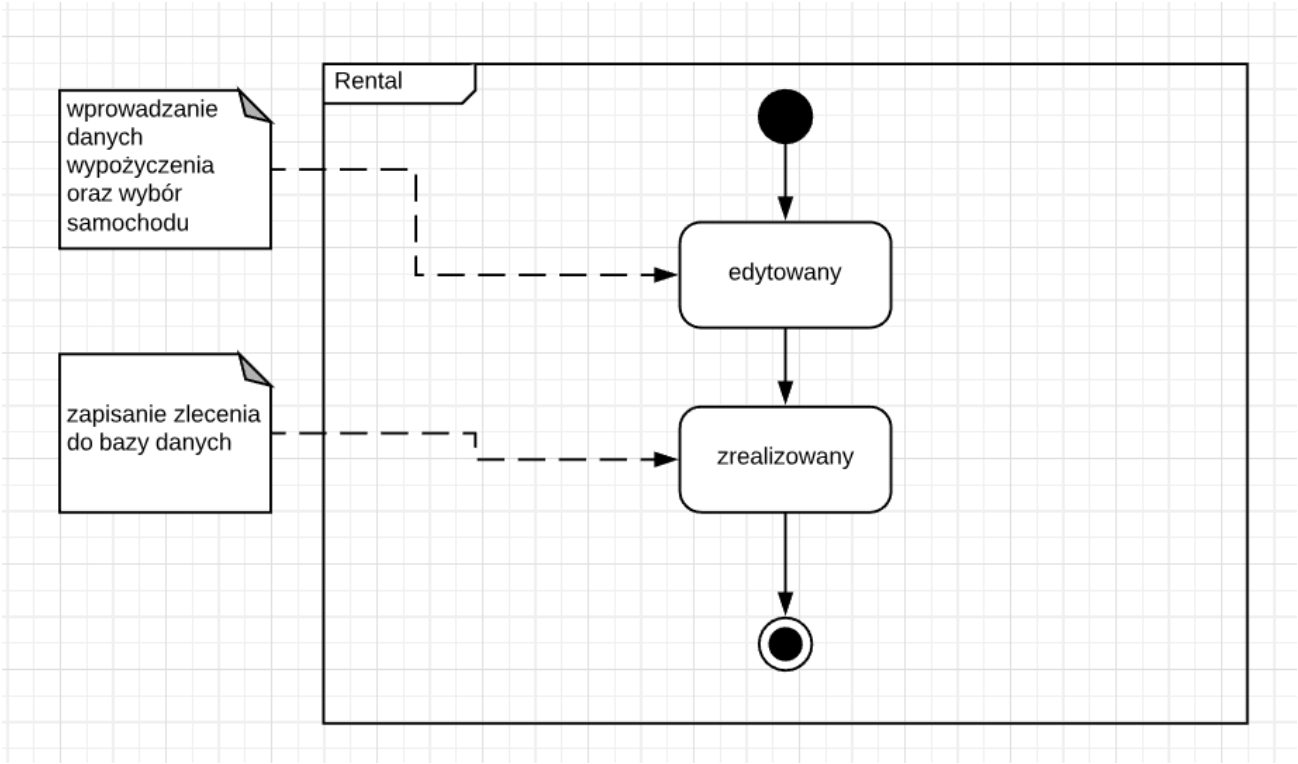
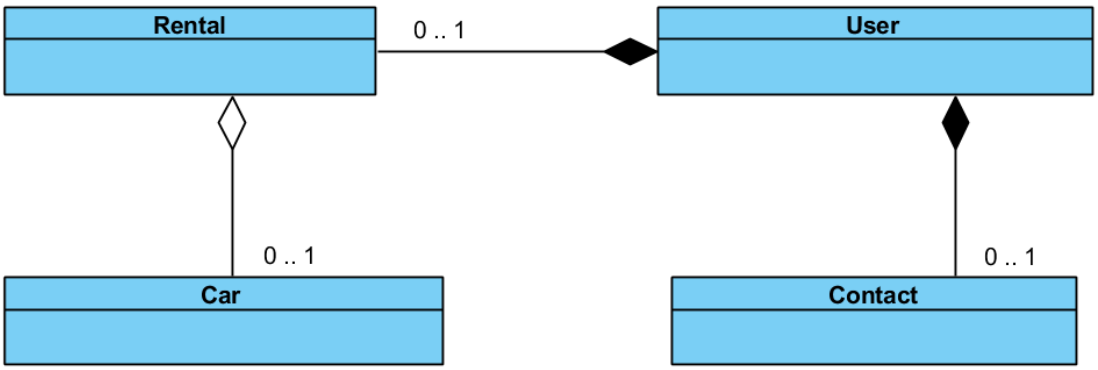


Diagram klas



Car:

Obiekty tego typu są tworzone na podstawie informacji z baz danych. Na stronie internetowej użytkownik może wyświetlić poszczególne informacje o pojeździe.

User:

Po uzupełnieniu danych osobowych podczas rejestracji lub logowaniu w serwisie – dane te są zgrupowane w obiekcie User.

Rental:

Obiekt ten posiada w sobie kompletne dane na temat rezerwacji samochodu. Dane te są zapisywane w bazie.

Contact:

Dane kontaktowe przypisywane do użytkowników

Diagram sekwencji

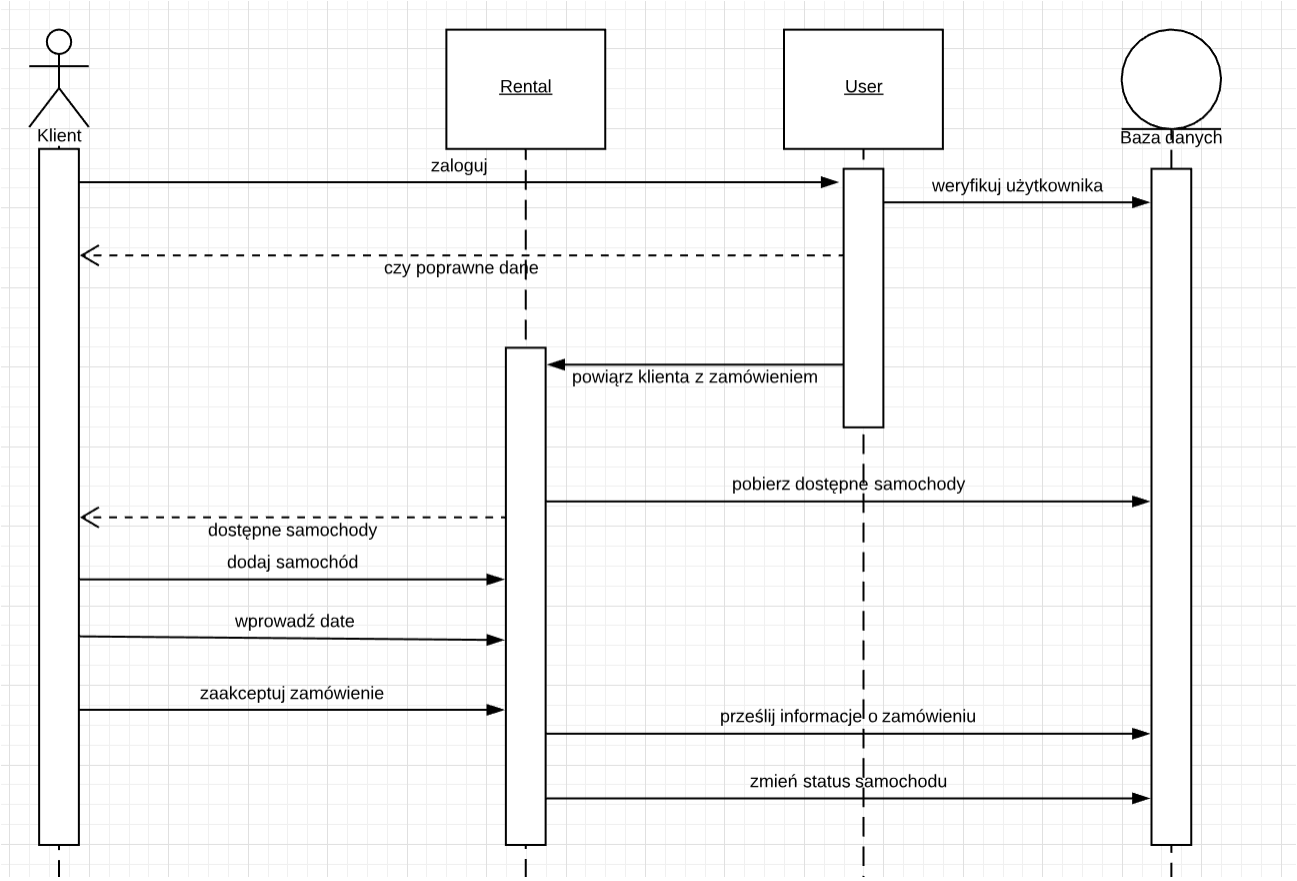


Diagram aktywności

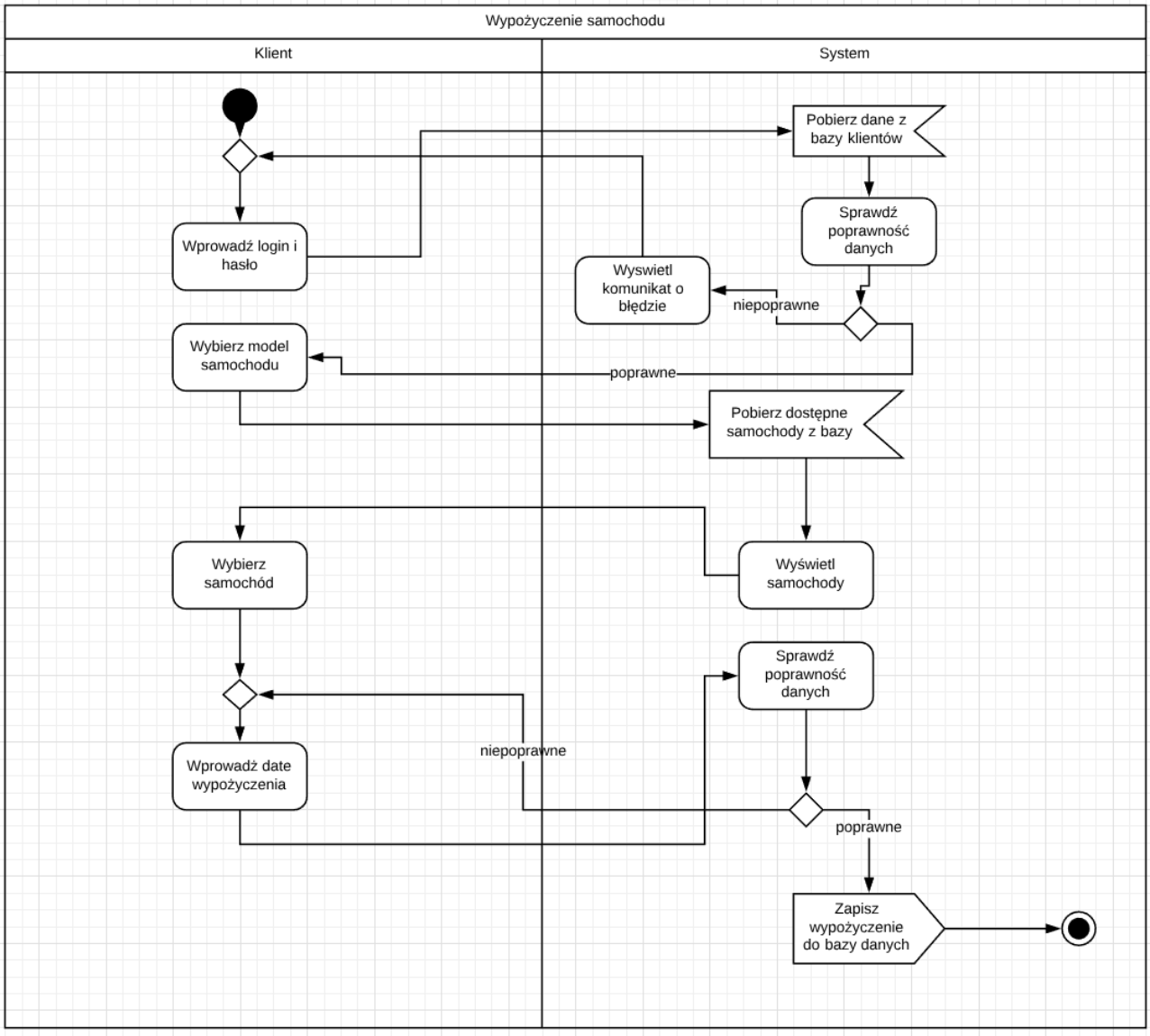


Diagram komponentów

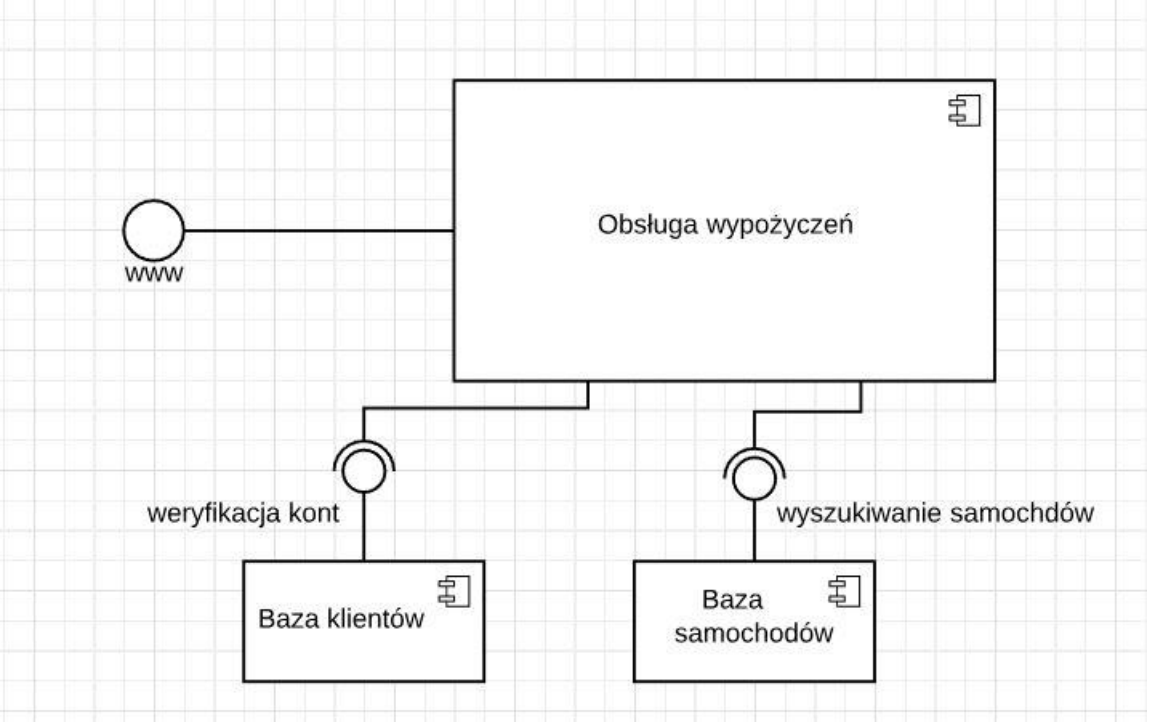
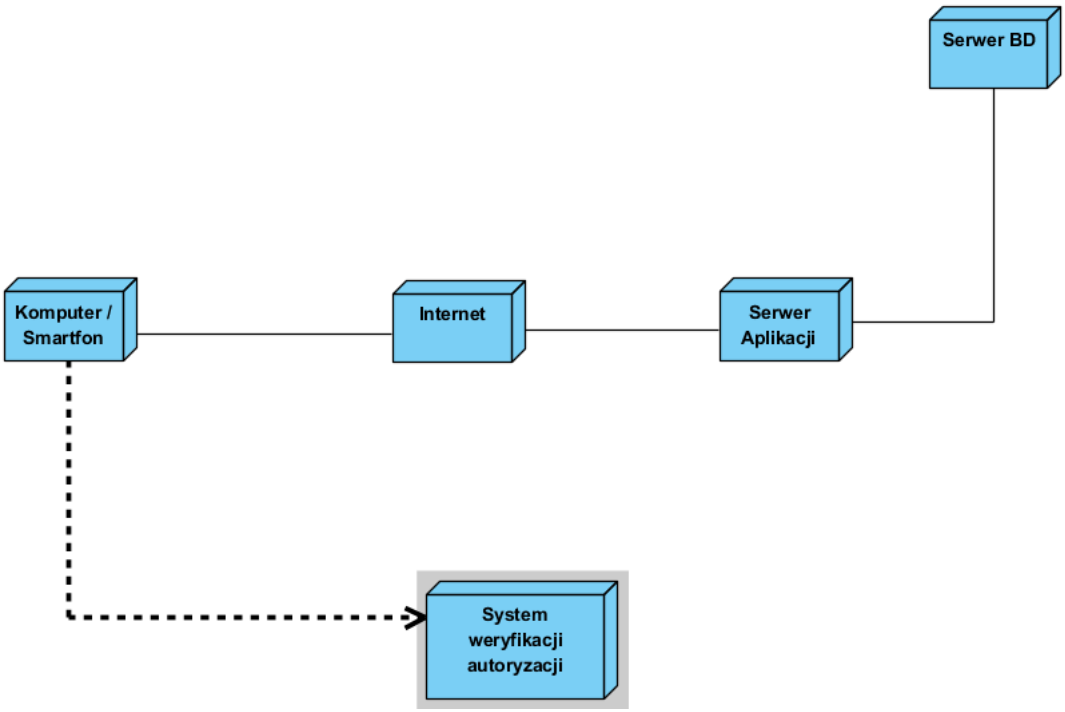


Diagram wdrożenia



Szczegółowy przypadek użycia

Przypadek użycia	**Logowanie**
Opis	Użytkownik loguje się
Warunki wstępne	Użytkownik posiada konto
Warunki powodzenia	Użytkownik poda prawidłowe dane
Warunki niepowodzenia	Użytkownik poda nieprawidłowe dane
Aktorzy	Klient
Ścieżka postępowania	1. Użytkownik wchodzi na stronę internetową. 2. Użytkownik naciska przycisk „Logowanie” 3. Użytkownik podaje prawidłowe dane 4. Użytkownik naciska przycisk "Zaloguj" 5. System weryfikuje dane jako poprawne 6. Użytkownik zostaje zalogowany
Alternatywna ścieżka	1. Użytkownik wchodzi na stronę internetową. 2. Użytkownik naciska przycisk „Logowanie” 3. Użytkownik podaje nieprawidłowe dane 4. System weryfikuje dane jako niepoprawne i zgłasza komunikat o błędach 5. Użytkownik poprawia dane 6. Użytkownik naciska przycisk "Zaloguj” 7. Użytkownik zostaje zalogowany

Przypadek użycia	**Wypożyczenie samochodu**
Opis	Użytkownik dokonuje wypożyczenia samochodu.
Warunki wstępne	Użytkownik posiada konto i jest zalogowany
Warunki powodzenia	Użytkownik poda prawidłowe dane.
Warunki niepowodzenia	Użytkownik poda nieprawidłowe dane.
Aktorzy	Klient
Ścieżka postępowania	1. Użytkownik wchodzi na stronę internetową. 2. Użytkownik loguje się 3. Użytkownik wybiera interesujący go samochód z listy dostępnych egzemplarzy. 4. Użytkownik podaje okres wypożyczenia samochodu. 5. Użytkownik podaje prawidłowe dane. 6. Użytkownik naciska przycisk „Zarezerwuj”. 7. System weryfikuje dane jako poprawne. 8. Samochód zostaje zarezerwowany.
Alternatywna ścieżka	1. Użytkownik wchodzi na stronę internetową i się loguje 2. Użytkownik wybiera interesujący go samochód z listy dostępnych egzemplarzy. 3. Użytkownik podaje okres wypożyczenia samochodu. 4. Użytkownik podaje nieprawidłowe dane 5. Użytkownik naciska przycisk „Zarezerwuj”. 6. System weryfikuje dane jako niepoprawne i wyświetla komunikat. 7. Użytkownik poprawia dane

	8. Użytkownik naciska przycisk „Zarezerwuj” 9. Samochód zostaje zarezerwowany.

Opis funkcjonalności:

Po zalogowaniu użytkownik znajduję się, na stronie z listą modeli samochodów. Użytkownik może on przy każdym modelu kliknąć na przycisk “Sprawdź”. W przypadku kiedy użytkownik jest zalogowany wyświetlą mu się dostępne samochody do wypożyczenia.

Gdy użytkownik wybierze “wypożycz” zostanie przeniesiony na odpowiednią stronę gdzie wyświetlone będą pola z okresami wypożyczenia.

Po wypełnieniu poprawnie pól z datami i przyciśnięciu przycisku „Zarezerwuj” użytkownik dokona wypożyczenia.

Środowisko

Środowisko developerskie składa się z wirtualnej maszyny, na której zostało zainstalowane Ubuntu a w nim Docker oraz Jenkins.

W celu zbudowania aplikacji należy w Jenkins’ie przejść zadanie CarRental i kliknąć “Build now”.

Następnie Jenkins:

- Zainstaluje wszystkie wymagane zależności,
- Uruchomi skrypt tworzący migracje
- Uruchomi skrypt uruchamiający stworzone migracje
- Uruchomi testy jednostkowe

Uruchomienie aplikacji

Po zbudowaniu aplikacji jeszcze przed jej uruchomieniem należy uruchomić seeder bazy danych, w tym celu należy przejść do folderu z projektem i wpisać w terminalu komendę:

"**sudo docker-compose run web python db_create.py**"

Aby uruchomić aplikacje należy wpisać w terminalu komendę:

"**sudo docker-compose up**"

Podczas uruchamiania aplikacji tworzony jest tymczasowy kontener dockera. Po czym aplikacja dostępna jest pod adresem:
http:/localhost:8000.

Seeder utworzy konto administratora, na które możemy się zalogować następującymi danymi:
Login: admin2
Hasło: Hasło123

W przypadku uruchamiania w innym środowisku instrukcja uruchomienia znajduje się w pliku README.md w repozytorium GitHub'a na gałęzi master

Link do repozytorium: https://github.com/Rejman/carRental/tree/docker