# UITMBER (WSIiZ/UITM UBER)

### DDD – Data Driven Development

### Założenia funkcjonalne

* Logowanie/rejestracja
* Możliwość zamówienia kursu/taxi
* Zestawienie kursów

//Kierowcy

* Możliwość zgłoszenia się jako kierowca
* Możliwość przejęcia kursu (na podstawie aktualnych zapotrzebowania użytkowników)
* Wyświetliła się ulubiona piosenka ze spotify klienta

## Aplikacja mobilna

Android/IOS

Xamarin.Forms

### Spis Funkcjonalności

1. Rejestracja konta,
2. Logowanie:
   1. SSO (tu zrobimy jak nam starczy czasu)
   2. AppleId
   3. Google
   4. Microsoft?
3. Wyświetlana mapa z kierwcami w okolicy
4. Zamawianie kursu:
   1. Wybranie punktu początkowego
   2. Wybranie punktu końcowego
   3. Typ auta (7osobowe, standard, uberx)
   4. Wybór typu płatności
   5. Określenie bagażu,
   6. Zaakceptowanie kosztów (Mnożniki, płatność),
5. Podgląd na żywo kierowcy/kursu
   1. Informacje na temat kierowcy (auto, zdjęcie, nr rejestracyjny)
6. Definiowanie/ponowienie kursu z historii
7. Ocena kierowcy/klienta
8. Ulubione lokalizacje użytkownika
9. Połączenie ze spotify
10. Kontakt pomiędzy klientem/kierowcą
11. Kupony zniżkowe
12. Wsparcie
13. \* Możliwość dzielenia się kosztami na tym samym kursie

Kierowcy

1. Mechanizm zgłaszania chęci współpracy:
   1. Zdjęcie prawo jazdy,
   2. Informacje na temat pojazdu,
   3. Zdjęcie pojazdu,
2. Nawigowanie do punktu odbioru/docelowego
3. Saldo/dzienny przychód
4. Mechanizm zarządzania kursem (rozpoczęcie, kontakt, zakończenie)
5. Wyświetlenie dostępnych kursów
6. Wybór piosenki klienta,

## Założenia

-TabbedPage

-colorPrimary #bb171e

-colorPrimaryDark #961218

-colorText #232323

-colorTextDescription #212121

-colorAccent #ffbf00

## Services

RequestService - jjonca

#### Authentication

**POST​/Authentication​/Authenticate** - jjonca

**POST​/Authentication​/Register (w60093)**

#### Application (w60080)

**POST​/Application​/SendApplication**

**GET​/Application​/GetMyApplications**

#### Car (w60065, w60084)

**GET​/Car​/GetMyCars**

**POST​/Car​/Add**

**PUT​/Car​/Update**

**DELETE​/Car​/Delete​/{id}**

#### Clients (w60089)

**GET​/Clients​/GetMyProfile**

**PUT​/Clients​/UpdatePhoto**

#### Discounts (w60092)

**GET​/Discounts​/Verify**

**POST​/Discounts​/AddToOrder**

#### Driver (w60073)

**GET​/Driver​/GetNerbyDriveres**

**GET​/Driver​/GetProfile**

#### Location **(w60093)**

**POST​/Location​/SaveMyLocation**

#### Orders (w60087, w60086, w60105, w60099)

**GET​/Orders​/GetMyOrders**

**GET​/Orders​/GetCarTypes**

**GET​/Orders​/GetClientOrderDetails**

**GET​/Orders​/GetLuggageTypes**

**GET​/Orders​/GetCost**

**POST​/Orders​/OrderPayment**

**POST​/Orders​/OrderAccept**

**PUT​/Orders​/ClientRate**

#### UFLocations (w60047, w60083)

**POST​/UFLocations​/AddLocation**

**DELETE​/UFLocations​/DeleteLocation​/{id}**

**GET​/UFLocations​/GetMyLocations**

## Serwer danych/API

.Net Core Api

### Spis Funkcjonalności

1. Mechanizm rejestracji
2. Mechanizm logowania
3. Pobranie listy dostępnych kierowców na naszej okolicy
4. Mechanizm zamawiania kursu,
5. Mechanizm zgłaszania chęci współpracy
6. Pobieranie historii przejazdów
7. Mechanizm oceny kursu,
8. Statystyki ocen
9. Mechanizm obsługi ulubionych lokalizacji
10. Integracja ze spotify

Endpoint:

**/Authentication** – jjonca

**/Cash (w60047)**

- GetMySaldo(dateStart,dateEnd) – moje saldo wydatków dla klienta

-GetMyEarnings(dateStart,dateEnd) – zarobki dla kierowcy

**/Cars (w60093)**

-GetMyCars() - HttpGet

-Add(model) - Post

-Update(model) -Put

-Delete(id) - Del

**/Clients (w60073)**

-GetMyProfile() (mój profil wraz z oceną)

-UpdatePhoto(photo) (aktualizacja zdjęcia na profilu)

**/Drivers (w60080)**

-GetNerbyDriveres(lat,long) – (Pobieramy kierowcow z jednego miasta)

-GetProfile(id)

**/Locations (w60080)**

-SaveMyLocation(id, lat,lang) – Post (zapis mojej aktualnej pozycji)

**/Favourites (w60084)**

-GetMyLocations() GET

-AddLocation() POST

-DeleteLocation() DEL

**/Orders**

-GetMyOrders() (szczegóły z informacją o kliencie, kierowcy) **(w60089)**

-GetCarTypes() – typy samochodów/przejazdów GET **(w60089)**

-GetLuggageTypes() – typy bagaży GET **(w60089)**

-GetClientOrderDetails(id) – info o kursie + info o kierowcy GET **(w60089)**

-GetCost(date, distance) GET **(w60087)**

-OrderAccept(id,date) POST -Składamy zamówienie **(w60087)**

-OrderPayment(id) Post **(w60087)**

-ClientRate(id, date, info, rating) PUT (ocena kierowcę przez klienta) **(w60087)**

**/Discounts (w60065)**

-Verify(code) – sprawdzamy czy można użyć kod

-AddToOrder(code, idorder) – wykorzystanie kodu

**/DriverOrders**

-GetAvalilableOrders(lat,long) GET (zamówienia ze statusem nowy) **(w60105)**

-ProcessOrder(id) – przejęcie zamówienia PUT **(w60105)**

-GetClientInfo(id) GET **(w60105)**

-DriverRate(id, date, info, rating) PUT (ocena klienta przez kierowcę) **(w60099)**

-GetDriverOrderDetails(id) – info o kursie + info o pasażerze GET ) **(w60099)**

-ChangeOrderStatus(id, status, data, info) PUT ) **(w60099)**

**/Applications (w60083)**

-SendApplication(Model application) POST (wysłanie zgłoszenia, id auta)

-GetMyApplications() GET

**/Music (w60097)**

-GetClientMusic(id) GET

-RegisterAccount() – POST – do przeczytania api Spotify