Dokumentacja projektu “Wypożyczalnia hulajnóg”

AGH, Wydział Fizyki i Informatyki Stosowanej

Autor: Oskar Szewczyk

I. Projekt koncepcji, założenia

1.1

Internetowa wypożyczalnia hulajnóg elektrycznych. Aplikacja umożliwia założenie konta w serwisie internetowym, wprowadzenie przez użytkownika danych potrzebnych do wynajęcia hulajnogi oraz zlokalizowania na terenie miasta dostępnych hulajnóg które aktualnie nie są wynajmowane i wynajęcia wybranej przez klienta hulajnogi. Użytkownik będzie miał również możliwość przeglądania swojej historii wynajmów.

1.2

Funkcjonalności bazy danych:

- Rejestracja użytkowników;
- Logowanie użytkowników;
- Udostępnianie aktualnej lokalizacji hulajnóg dostępnych do wynajęcia;
- Możliwość edycji danych użytkownika;
- Obsługa wynajmu hulajnogi;
- Prowadzenie historii przejazdów;
- Obsługa systemu opłat za wynajem hulajnogi;

1.3

Podstawowe funkcje realizowane w bazie danych:

- Weryfikacja danych wprowadzonych do formularza podczas rejestracji użytkownika;
- Sprawdzenie czy istnieje już użytkownik o wprowadzonych danych
- Dodanie nowego użytkownika do bazy

- Weryfikacja danych danych wprowadzonych do formularza podczas logowania się użytkownika
- Sprawdzenie czy istnieje użytkownik o podanym adresie email
- Sprawdzenie czy hasło do konta wprowadzone przez użytkownika jest prawidłowe

- Weryfikacja stanu wynajmu hulajnogi;
- Udostępnienie informacji o numerze i lokalizacji hulajnogi dostępnej do wynajmu;

- Edycja atrybutów użytkownika;

- Sprawdzenie salda użytkownika;
- Zmiana stanu wynajmu hulajnogi;
- Stworzenie nowego przejazdu który wiąże użytkownika z hulajnogą;
- Uaktualnienie atrybutów należących do wcześniej utworzonego przejazdu;

- Udostępnienie informacji o przejazdach powiązanych z danym użytkownikiem;

- Weryfikacja poprawności danych karty kredytowej wprowadzonej przez użytkownika;
- Sprawdzenie czy użytkownik jest powiązany z kartą kredytową;
- Aktualizacja salda użytkownika;
- Udostępnianie informacji o saldzie powiązanym z danym użytkownikiem;

II. Projekt diagramów (konceptualny)

2.4

Diagramy w oryginalnych rozmiarach znajdują się w folderze “diagramy”, znajdującym się w tym samym katalogu co dokumentacja.

Diagram DFD - kontekst

Diagram DFD - poziom 0

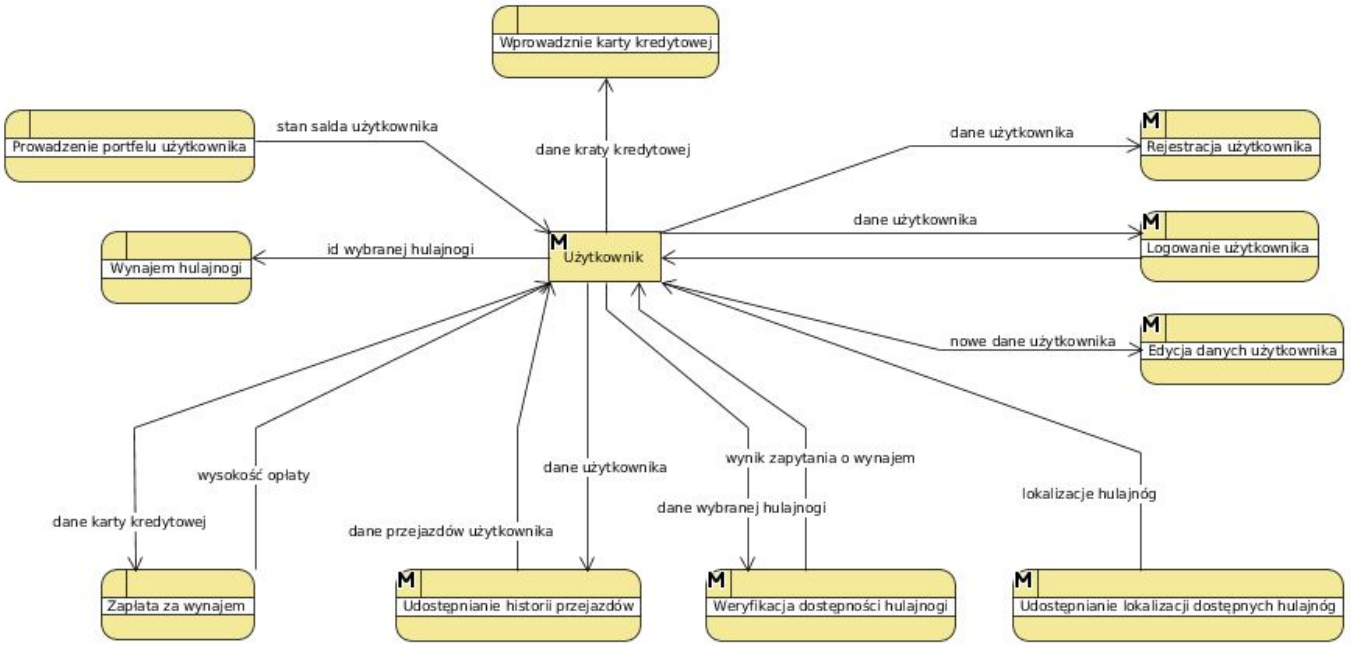
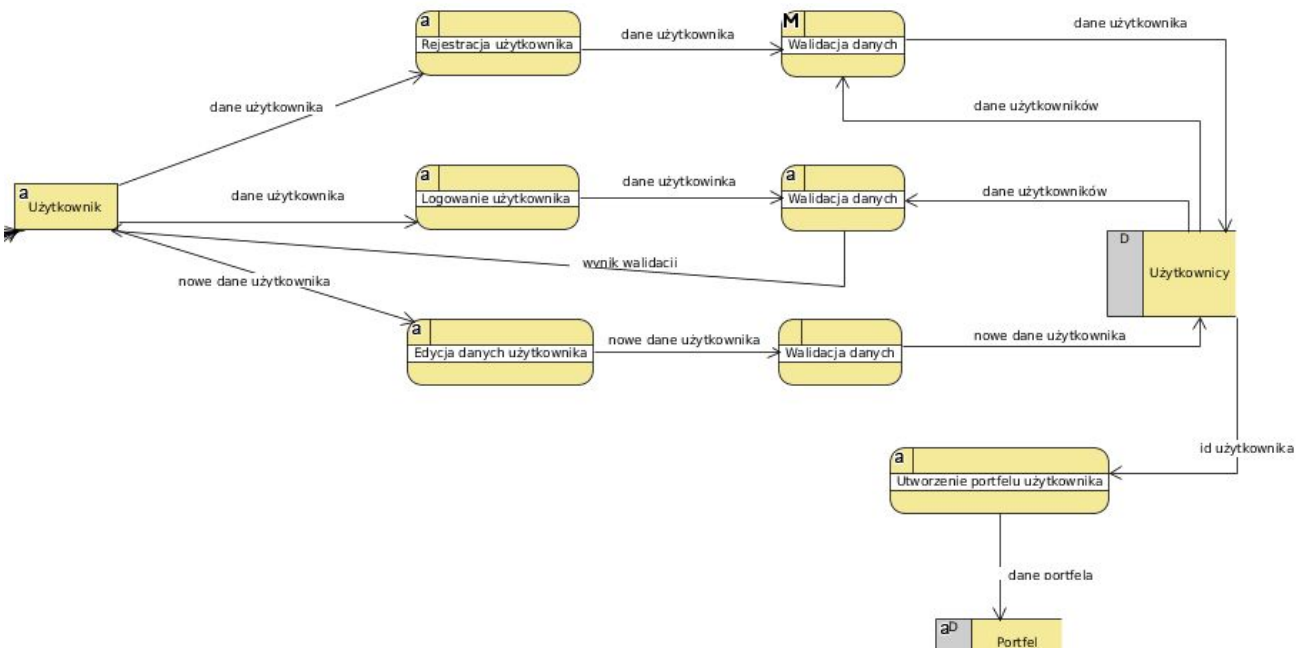
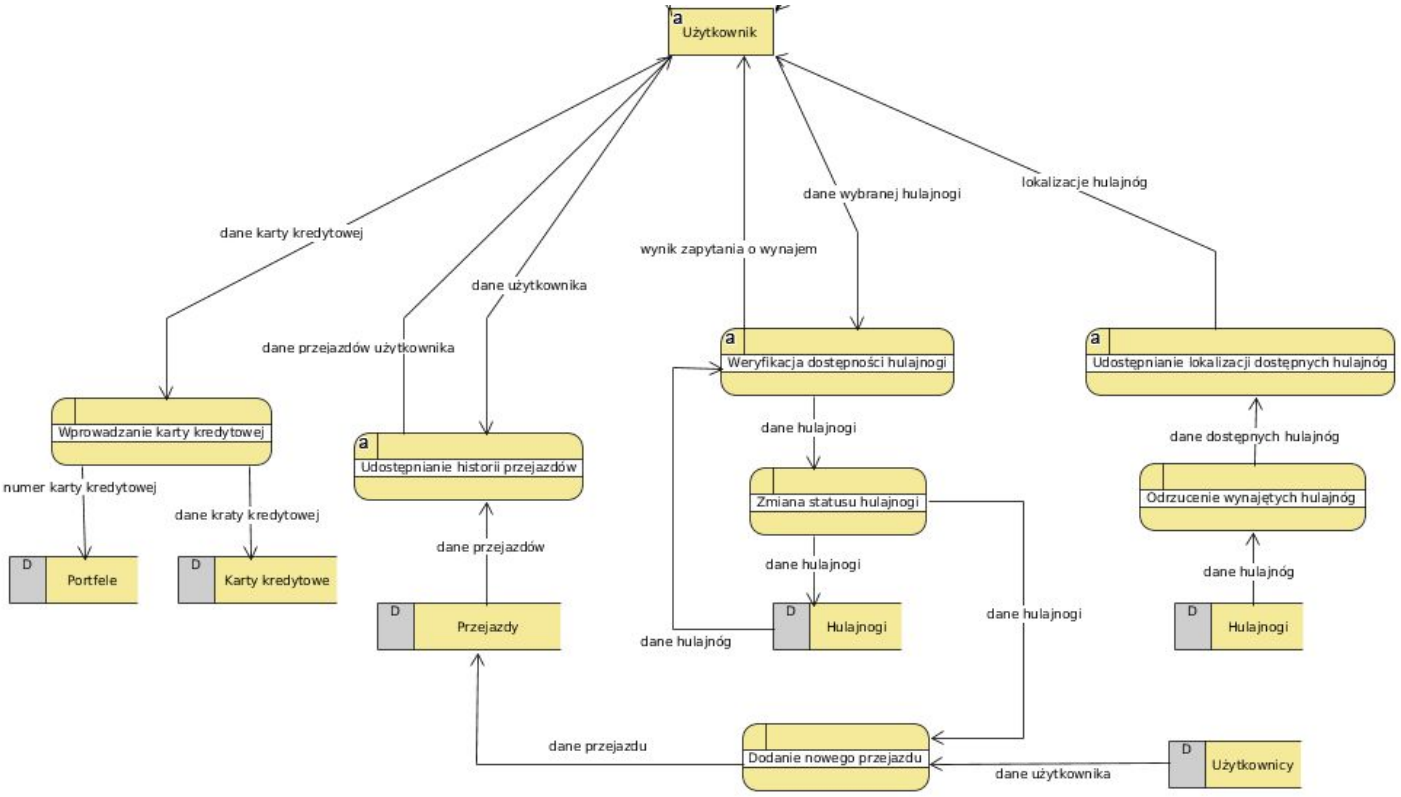


Diagram DFD - poziom 1

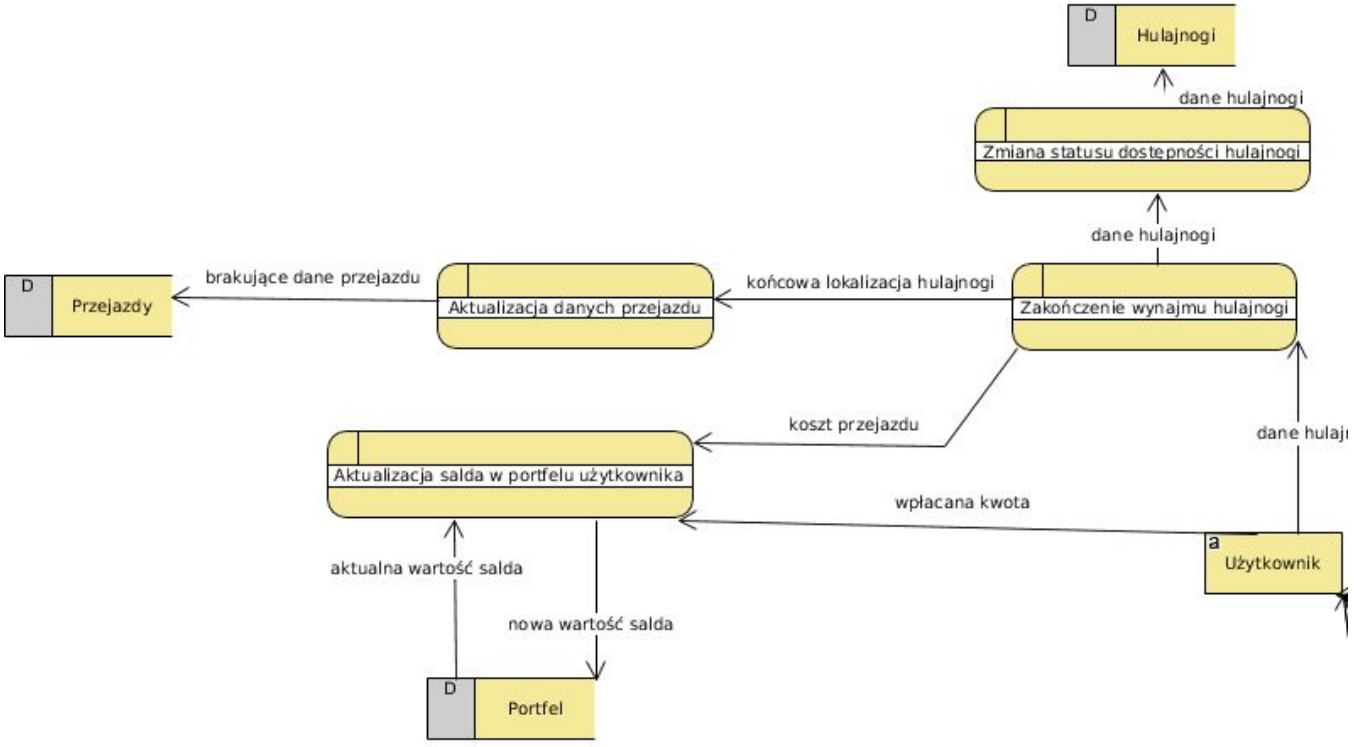
część 1



część 2

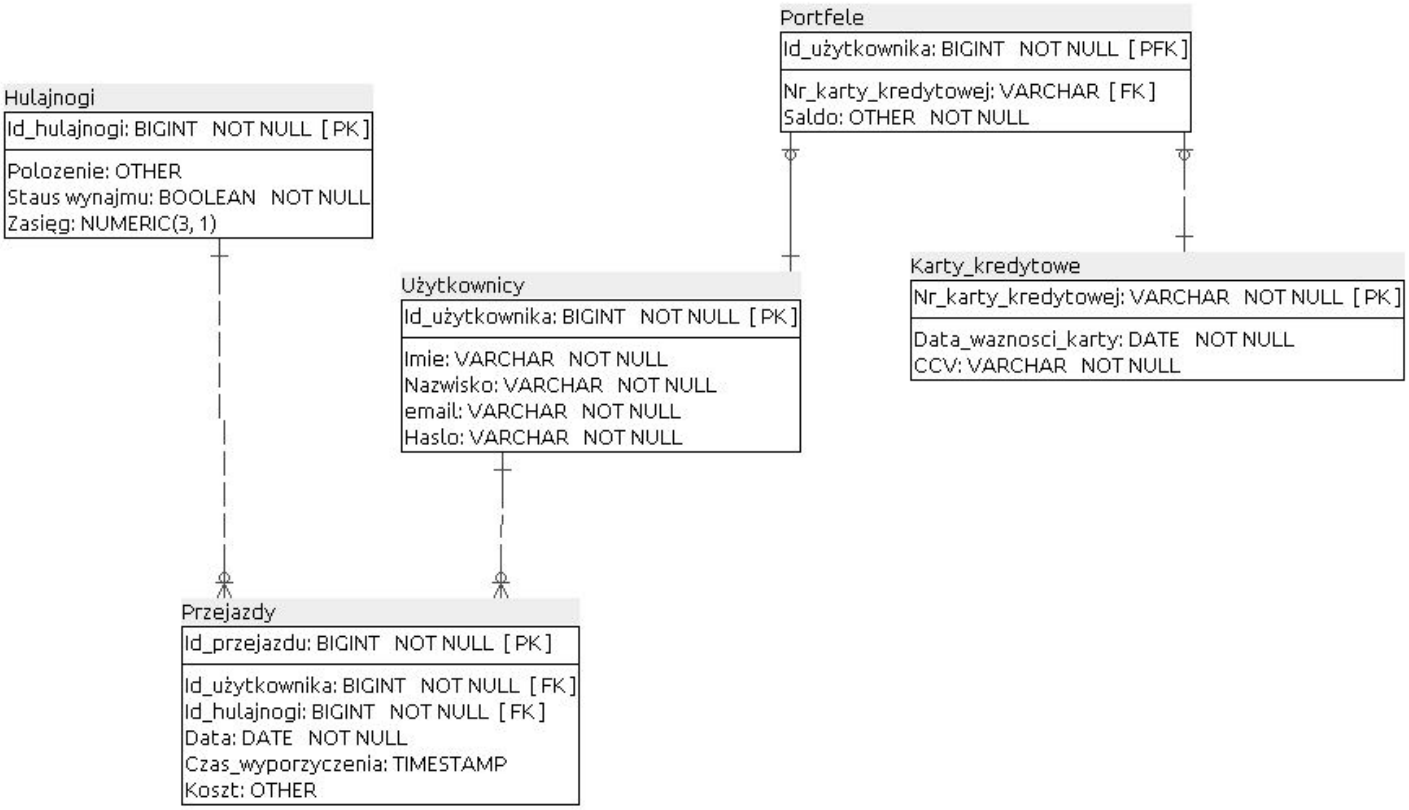


część 3



2.5 i 2.6

Diagram ERD.



III. Projekt logiczny

Uwaga:

Atrybut nr_karty_kredytowej w tabeli karty_kredytowe pierwotnie był przechowywany jako typ BYTEA. Wartości tego atrybutu było zakodowane procedurą znajdującą się w rozszerzeniu “pgcrypto”. Utworzenie tego rozszerzenie w bazie danych znajdującej się na serwerze pascal jest zablokowane prawami superużytkownika. Z tego względu musiałem zrezygnować z przechowywania numeru karty kredytowej w bezpiecznej postaci hashcode. Wszystkie procesy związane z pgcrypto zostały zakomentowane w moim skrypcie definiującym bazę danych i zastąpione opracjami na typie VARCHAR.

3.7

Elementy uzupełniające załączony diagram ERD:

- **sekwencje** dla atrybutów id_użytkownika, id_przejazdu, id_hulajnogi;
- **domeny** dla trybutów email, nr_karty_kredytowej, ccv;
- **procedury** mające na celu weryfikacje danych wprowadzanych przez użytkownika i oferujące funkcjonalność wprowadzenia odpowiednich danych;

- **wyzwalacze** służące wyzwalaniu procedur weryfikujących danych wprowadzanych przez użytkownika, oraz procedur aktualizujących bazę danych po usunięciu rekordów z odpowiednich tabel;

3.8

Słownik danych:

Tabela uzytkownicy:

- domena **typ_adres_email** - adres email użytkownika;

Tabela hulajnogi:

- atrybut **polozenie** - wskazuje aktualne położenie hulajnogi
- atrybut **status_wynajmu** - informuje o aktualnym stanie wypożyczenia hulajnogi. true = hulajnoga wypożyczona;
- atrybut **zasieg** - aktualny zasięg hulajnogi;

Tabela przejazdy:

- atrybut **koszt** - koszt przejazdu;

Tabel karty_kredytowe:

- domena **typ_nr_karty_kredytowej** - numer karty kredytowej składający się z 16 cyfr;
- domena **typ_nr_csc** - “card security code” składający się z 3 cyfr;

Tabla portfele:

- atrybut **slado** - wskazuje aktualne saldo portfelu użytkownika;

3.9

Baza odpowiada postaci 3NF. Poszczególne tabele reprezentują jeden typ obiektu, atrybuty niekluczowe są od siebie niezależne i całkowicie zależne od klucza głównego.

3.10

Definicje procedur i wyzwalaczy znajdujących się w projekcie:

- procedura **czy_uzytkownik_istnieje(adresEmail VARCHAR)** - waliduje format wprowadzonego adresu email i sprawdza czy podany adres jest już wykorzystany przez innego użytkownika;
- procedura i wyzwalacz **waliduj_rejestracje()** - waliduje rejestrację użytkownika, wykorzystując procedurę *czy_użytkownik_istnieje* oraz sprawdzając długość atrybutów imie i nazwisko;

- procedura i wyzwalacz **waliduj_edycje_uzytkownika()** - waliduje edycję danych użytkownika, wykorzystując procedurę *czy_użytkownik_istnieje* oraz sprawdzając długość atrybutów imie i nazwisko;
- procedura i wyzwalacz **utworz_portfel_uzytkownika()** - tworzenie portfela dla nowo utworzonego konta użytkownika. Użytkownik dostaje 10zł za założenie konta;
- procedura **waliduj_nr_karty_kredytowej(nr_karty_kredytowej VARCHAR)** - walidacja formatu karty kredytowej. Format to 16 cyfr 0-9;
- procedura i wyzwalacz **waliduj_dane_karty_kredytowej()** - procedura wyzwalana podczas dodawania karty kredytowej przez użytkownika. Sprawdza datę ważności karty (data ważności > aktualnej_daty), długość numeru csc i fromat karty kredytowej korzystając z procedury *waliduj_nr_karty_kredytowej()*;
- procedura **dodaj_karte_kredytowa(id_uzytkownika BIGINT, nr_karty_kredytowej VARCHAR, data_waznosci DATE, nr_csc VARCHAR)** - umożliwia dodanie karty kredytowej przez użytkownika, oraz aktualizuje portfel użytkownika poprzez dodanie do niego numeru karty kredytowej;
- procedura i wyzwalacz **usun_karte_kredytowa()** - zapewnia usunięcie karty kredytowej z tabeli karty_kredytowe użytkownika, który usunął swoje konto.
- procedura **usun_karte_kredytowa_uzytkownika(id_uzytkownika BIGINT)** - dostarcza funkcjonalność usuwania karty kredytowej poprzez podanie id_uzytkownika;
- procedura **doladowanie_salda(id_uzytkownika BIGINT, kwota MONEY)** - zapewnia możliwość doładowania portfelu użytkownika, walidując wpłaconą kwotę i sprawdzając istnienie i poprawność karty kredytowej użytkownika;
- procedura **pobranie_oplaty(id_uzytkownika BIGINT, kwota MONEY)** - pobiera opłatę za przejazd z portfelu użytkownika;
- procedura **start_wynajmu_hulajnogi(id_uzytkownika BIGINT, id_hulajnogi BIGINT)** - dostarcza funkcjonalność wynajmu hulajnogi, sprawdzając saldo i kartę użytkownika oraz weryfikując dostępność wybranej hulajnogi. Procedura aktualizuje również status wynajętej hulajnogi i tworzy nowy rekord w tabeli przejazdy odpowiednio ustawiając wymagane atrybuty.
- procedura **koniec_wynajmu_hulajnogi(id_uzytkownika BIGINT, id_hulajnogi BIGINT, pozycja JSON)** - obsługuje zakończenie wynajmu hulajnogi przez użytkownika. Uaktualnia nową pozycję i status hulajnogi, oblicza opłatę za wynajem i aktualizuje odpowiedni rekord przejazdu w tabeli przejazdy.

Projekt w języku sql znajduje się w folderze “skrypty_sql”.

IV. Projekt funkcjonalny

4.11

Interfejsy do prezentacji, edycji i obsługi danych.

Głównym sposobem wprowadzania danych do aplikacji przez użytkownika są formularze.

- **formularz rejestracji**, zapewnia możliwość utworzenia nowego użytkownika aplikacji. Formularz składa się z pól: imie, nazwisko, adres email, hasło;
- **formularz logowania**, zapewnia możliwość zalogowania się do aplikacji. Formularz składa się z pól adres email, hasło;
- **formularz edycji danych użytkownika**, zapewnia możliwość edytowania danych wprowadzonych podczas rejestracji. Formularz jest częściowo wypełniony, co pozwala użytkownikowi przypomnieć sobie wcześniej wprowadzone dane. Formularz składa się z tych samych pól co formularz rejestracji;
- **formularz dodawania karty kredytowej**, zapewnia możliwość dołączenia karty kredytowej do portfelu użytkownika. Formularz składa się z pól numer karty kredytowej (16 cyfrowy ciąg znaków, walidowany sumą kontrolną implementowaną przez algorytm Luhna), daty ważności karty (należy wprowadzić miesiąc i rok), numeru csc (3 cyfrowy kod);
- **formularz transferu określonej kwoty z karty kredytowej do portfela użytkownika**, zapewnia możliwość zwiększania salda użytkownika. Formularz składa się z pola kwota (liczba większa od zera);

Kolejnym sposobem interakcji użytkownika z aplikacją są przyciski, niosące za sobą konkretną funkcjonalność zaimplementowaną w bazie danych.

- **przycisk wynajmu hulajnogi**, wywołuje procedurę start_wynajmu_hulajnogi znajdującą się w bazie danych;
- **przycisk koniec wynajmu hulajnogi**, wywołuje procedure koniec_wynajmu_hulajnogi znajdująca się w bazie danych;

Powyższa funkcjonalność jest dostępna dla użytkownika po wprowadzeniu karty kredytowej, bądź posiadaniu stanu salda większego od zera.

- **przycisk odłączenia karty kredytowej**, zapewnia możliwość odłączenia karty kredytowej od portfela użytkownika i usunięcia jej z bazy danych;
- **przycisk usunięcia konta użytkownika**, zapewnia możliwość usunięcia konta użytkownika, co implikuje usunięcie wszystkich danych dotyczących użytkownika z bazy danych.

W aplikacji znajdują się również interfejsy pełniące funkcjonalność prezentowania danych znajdujących się w bazie danych.

- **mapa**, dostarczona przez Google Maps API zapewnia możliwość wyświetlania aktualnych lokalizacji hulajnóg, wybrania hulajnogi którą chce wynająć użytkownik, wybrania końcowej pozycji końcu wynajmu hulajnogi;
- **lista historii przejazdów**, zapewnia możliwość przeglądnięcia historii przejazdów użytkownika;
- **stan salda użytkownika**, zapewnia możliwość wyświetlania aktualnego salda użytkownika;
- **formularz edycji danych**, oprócz funkcji edycji danych, formularz zapewnia również funkcjonalność wyświetlania aktualnych danych przypisanych do użytkownika;

4.12

Panel sterowania aplikacją można podzielić na dwie części:

- **część dostępna dla niezalogowanego użytkownika**:

Niezalogowany użytkownika steruje aplikacji za pomocą paska nawigującego znajdującego się w górnej części strony, na którym znajdują się przycisk przekierowujący użytkownika do części logowania lub rejestracji. Dodatkową funkcjonalnością dostępną dla niezalogowanego użytkownika jest sprawdzenie aktualnych pozycji hulajnóg, za pomocą mapy znajdującej się w środkowej części startowej strony aplikacji.

- **część dostępna dla zalogowanego użytkownika**:

Zalogowany użytkownika zostaje przekierowywany do panelu sterowania oferujacego dodatkowe funkcjonalności. Znajduje się tu pasek nawigujący w górnej części strony oferujący możliwość wylogowania się z aplikacji. W lewej części strony umiejscowiony został pasek boczny oferujący możliwość nawigacji po panelu sterowania. Dostępne komponenty to:

- **wynajem hulajnogi**, obsługujący cały proces wynajmu hulajnogi;
- **portfel**, obsługujący portfel przypisany do danego użytkownika;
- **historia przejazdów**, obsługujący listę wyświetlającą dotychczasowe przejazdy użytkownika;
- **edycja profilu**, obsługujący edycję profilu;

V. Dokumentacja

5.13

Do bazy danych zostały wprowadzone jedynie dane hulajnóg dostępnych do wynajmu przez użytkowników aplikacji. Za pomocą skryptu w pythonie został wygenerowany skrypt w języku sql który wprowadza do bazy 50 hulajnóg znajdujących się na terenie miasta.

5.14

Aby w pełni skorzystać z funkcjonalności aplikacji należy założyć konto użytkownika. Po utworzeniu nowego konta, baza danych tworzy portfel dla nowego użytkownika ustawiając w nim saldo o wartości 10 zł. Po założeniu konta użytkownik zostaje przekierowany do panelu sterowania z wybranym komponentem mapy wynajmu hulajnogi. Na mapie znajdują się znaczniki reprezentujące hulajnogi i ich aktualne pozycje. Klikając na wybrany marker pojawia się przycisk rozpoczynający wynajem hulajnogi. Aby zakończyć wynajem użytkownik wybiera pozycję na mapie i potwierdza to kliknięciem przycisku “koniec wynajmu”.
Na pasku bocznym znajdują się pozostałe funkcjonalności panelu sterowania użytkownika.
W komponencie portfel użytkownik ma możliwość dołączenia karty kredytowej i przelania określonej kwoty z konta bankowego do portfela użytkownika (są to tylko symulacje wspomnianych operacji).
Po wybraniu opcji “historia przejazdów” z panelu bocznego użytkownika, wyświetla się lista dotychczasowych przejazdów użytkownika informująca o dacie i koszcie wynajmu.
Opcja “edycja profilu” dostarcza użytkownikowi możliwość sprawdzenia aktualnych danych (z wyjątkiem hasła), a także możliwość edycji danych przypisanych do konta. Należy zmienić wybrane pola i wpisać aktualne lub nowe hasło do konta. W tym komponencie pojawia się również przycisk usunięcie konta. Po kliknięciu konto użytkownika zostaje usunięta, a przeglądarka zostaje przekierowana do strony startowej aplikacji.

Aplikacja jest symulacją funkcjonowania rzeczywistej wypożyczalni hulajnóg. Faktyczna aplikacja musiałaby operować na urządzeniach mobilnych, gdzie przykładowo rozpoczęcie wynajmu odbywałoby się poprzez zeskanownie QR codu znajdującego się na hulajnodze.

5.15

Dostęp do bazy danych został zaimplementowany za pomocą dwóch warstw które są oddzielnymi aplikacjami:

- back-end

Aplikacja webowa w stylu RESTful napisana w środowisku Node.js. Do zaimplementowania API wykorzystano framework Express. Aplikacja łączy się z bazą danych za pomocą pakietu node-postgres.

Struktura:

Wylistowane poniżej foldery znajdują się w folderze api.

- ./index.js - główny plik aplikacji;
- ./package.json - plik z konfiguracjami;
- ./controllers/ - kontrolery aplikacji w których wykorzystywana jest komunikacja z bazą danych;
- ./db/ - warstwa abstrakcji implementująca dostęp do bazy danych;
- ./middleware/ - implementacja funkcjonalności autoryzacji i autentykacji;

- front-end

Aplikacja w stylu “Single page application” w JavaScript z użyciem biblioteki React, implementująca interfejs pomiędzy użytkownikiem, a wcześniej wspomnianą aplikacją webową.

Struktura:

Wylistowane poniżej foldery znajdują się w folderze client.

- ./public/ - dokument typu html w którym generowane są wszystkie komponenty napisane z pomocą biblioteki React;
- ./src/ - pliki źródłowe aplikacji w których znajdują się komponenty z których składa się aplikacja;

Podsumowanie

Aplikacje znajdują się na serwerze Pascal:
back-end: http:/pascal.fis.agh.edu.pl:1777/
front-end: http:/pascal.fis.agh.edu.pl:1776/

Jest możliwość skorzystania z utworzonego konta:
email: admin@admin.pl hasło: adminadmin

5.16

Źródła:

- https://www.postgresql.org/docs/
- http://www.postgresqltutorial.com/
- https://reactjs.org/docs/getting-started.html
- https://www.npmjs.com/
- https://www.w3schools.com/nodejs/