Ola-Uber-Clone

Technologies used for this Project

Backend: - Node.js

Database: - PostgreSQL

Npm modules: - axios, bcryptjs, dotenv, express, express-rate-limit, jsonwebtoken, sequelize, pg, pg-store, jest, supertest, morgan, nodemon

Environmental variables: -

POSTGRES_PASS=UJW4edE469xih4-h5UqIJY94DNl8a00T

POSTGRES_USERNAME=hillkslt

NODE_ENV=test // this is required only while testing

Note: - if using seeder and migration please ignore environmental variables

Migration & Seeder: -

Please add your postgres detail (username, password, database name,) in the **config.json** file for example: -

```
{
  "development": {
    "username": "postgres",
    "password": "light159",
    "database": "database_development",
    "host": "127.0.0.1",
    "dialect": "postgres"
  },
  "test": {
    "username": "postgres",
    "password": "light159",
    "database": "database_test",
    "host": "127.0.0.1",
    "dialect": "postgres"
  },
}
```

If you are using different port apart from default postgres port: - 5432, add “port”: ”your port number goes here”, add this inside the json

Please run following commands to successfully run the program: -

1) npm i
2) npx sequelize-cli db:create
3) npx sequelize-cli db:migrate:all
4) npx sequelize-cli db:seed:all
5) npm start or npm run dev

User Endpoints

ID and Emails are unique.

Note: - if using running locally the endpoint starts with **localhost:8080**

Register / Signup User: -

POST – (“users/signup”)

Body: name, email, password

Here provide name, email and password as json body

For example: -

```
{
    "name":"Rahul R Ghimire",
    "email":"ghimirerahul@gmail.com",
    "password":"Badlapur@123"
}
```

Login User: -

POST – (“users/login”)

Body: email, password

Here provide email and password as json body

For example: -

```
{
    "email":"ghimirerahul@gmail.com",
    "password":"Badlapur@123"
}
```

Logout User: -

POST – (“users/logout/:token”)

Here provide JWT token as a params to get logged out

For example: -

```
localhost:8080/users/logout/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjoiZ2hpbWlyZXJhaHVsQGdtYWlsLmNvbSIsImlhdCI6MTYwMDc5MzE0NywiZXhwIjoxNjAwNzk2NzQ3fQ.1RCPs_fJge-8brHVbSEBsPZM1SY1OrJAfX1HkIm9eYU
```

User Cab Booking: -

POST – (“users/booking”)

Header: - Authorization: user’s JWT Token

Body: pickup_address (street, city, state, postalcode), destination_address (street, city, state, postalcode)

Here provide JWT token as an authorization in the header, and provide pickup and destination address breaking the address into street, city, state, postal code

For example: -

```
{
    "pickup_address":{
        "street":"dp road",
        "city":"Badlapur",
        "state":"Maharashtra",
        "postalcode":"421503"
    },
    "destination_address":{
        "street":"manjarli",
        "city":"Badlapur",
        "state":"Maharashtra",
        "postalcode":"421503"
    }

}
```

User Select Driver: -

GET – (“users/cab_list/:booking id”)

Header: - Authorization: user’s JWT Token

Body: - proximity (in kilometers)

Here provide JWT token as an authorization in the header and booking id generated by the server from previous / above request as a params and proximity as a body, proximity is optional default proximity range is 2 kms hence if you want to set the range lower or higher you can do it so that server will provide you cabs within this proximity range

For example: -

Request: - GET localhost:8080/users/cab_list/5

Body: -

```
{
    "proximity":1
}
```

User Checking Out: -

POST – (“users/checkout/:booking id”)

Header: - Authorization: user’s JWT Token

Body: - driver’s id

Here again provide JWT token as an authorization in the header, and the booking id and your preferred driver’s id from the list of drivers provided by the server

For example: -

```
{
    "driver_id":4
}
```

User's Booking History: -

GET – ("users/get_booking_history")

Header: - Authorization: user's JWT Token

Query: - pageNo = 'enter page no.' for example: - 0,1,2,3

Here provide JWT token as an authorization in the header and pageNo as a query to get paginated booking history

For example: -

```
localhost:8080/users/get_booking_history?pageNo=1
```

Drivers Endpoints

Register / Signup Driver: -

POST – ("drivers/signup")

Body: name, email, password, address (street, city, state, postal code)

Here provide name, email, password and address as json body

For example: -

```
{
    "name":"rahul ghimire",
    "email":"rahulghimire@gmail.com",
    "password":"qwerty",
    "address":{
        "street":"rameshwadi",
        "city":"badlapur",
        "state":"Maharashtra",
        "postalcode":421503
    }
}
```

Login Driver: -

POST – (“drivers/login”)

Body: email, password, address (street, city, state, postal code)

Here provide email, password and address as json body

For example: -

```
{
    "email":"rahulghimire@gmail.com",
    "password":"qwerty",
    "address":{
        "street":"dp road",
        "city":"Badlapur",
        "state":"Maharashtra",
        "postalcode":421503
    }
}
```

Logout Driver: -

POST – (“drivers/logout/:token”)

Here provide JWT token as a params to get logged out

For example: -

```
localhost:8080/drivers/logout/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjoiZ2hpb_WlyZXJhaHVsQGdtYWlsLmNvbSIsImlhdCI6MTYwMDg4MzU0MCwiZXhwIjoxNjAwODg3MTQwfQ.Bed9j__ZlPdHOzFr-ZafiMUXVHvA2T4CJFwyZb31LMs
```