TaxiOperator Application

User Guide

Table of Contents

[1. Introduction: 3](#_Toc70899716)

[2. Requirements: 3](#_Toc70899717)

[3. Configuration: 3](#_Toc70899718)

[3.1 Database bak file: 3](#_Toc70899719)

[3.2 Edit Appsettings.json File: 3](#_Toc70899720)

[4. Running the application: 3](#_Toc70899721)

[5. Features: 3](#_Toc70899722)

[5.1 Data Controller: 3](#_Toc70899723)

[5.2 User Controller: 4](#_Toc70899724)

# Introduction:

TaxiOperator application is an application that is used to manage and view the all operations that are done on taxi model and to makes authentication for users using JWT tokens and cookies in .net core framework.

# Requirements:

* .Net core version 3.1.
* Sql server instance 16.
* Visual studio 2019 or any other IDE that supports the mentioned .net core version
* PostMan

# Configuration:

## Database bak file:

Start by restoring the bak file in the solution to your database.

## Edit Appsettings.json File:

After opening the application in your favorite IDE, go to **appsettings.json** file and make sure to update the **DefaultConnection** field to suits your database configurations.

# Running the application:

In case of using visual studio 2019 the application can run directly by pressing F5 to run in debugging mode or (ctrl + F5) to run without debugging mode. Also as a second way open a new Cmd and change directory to the location of the application **Ex: Cd D:\WorkBanch\taxiOperator**. And then run **dotnet run** command in the Cmd.

# Features:

## Data Controller:

Use those URLs to get data from DataController:

To Gat random Customer [GetRequest] : <https://localhost:44319/data>;

To setVip customer [PostRequest]: https://localhost:44319/data/SetVip?id=1

To vip Cab driver list [GetRequest]: https://localhost:44319/data/GetCabDriverShifts

## User Controller:

Use this URL to authenticate user and get required token:

Use username = Admin

Password = Admin

[PostRequest] : <https://localhost:44319/Account/login>

**NOTE**:

In case of using postman make sure to add the following script in Tests sections of the request to make sure to add the token to your current enviroment:

const user = pm.response.json();

pm.test("Has properties", function () {

pm.expect(user).to.have.property('username');

pm.expect(user).to.have.property('token');

});

if (pm.test("Has properties")) {

pm.globals.set('token', user.token);

}

Or by adding Authorization field to request Header and giving it value : Bearer Token