|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | Full stack Application Development with Node.js + Express.js + React.js - 2022 | |
|  | | | |
| Project author | | | |
|  | Name | |  |
|  | Mehmet I. | |  |

|  |  |
| --- | --- |
| Project name | Carly (Car App with Car Data Base, Classifieds and Blog) |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| Webb app for Car classifieds combined with Car Data Base and Blog with Car Reviews. The app will give opportunity for people to list their cars and browse other cars which are for sale.  The system will be developed as follows:  Front-end: **React.js**  Back-end: **Node.js** and **Express.js**  Communication: **socket.io or Mail Functionality (mailJet)**  DB: **Mongo DB**  Each view will have his unique URL, and the routing between pages will be done with **React Router v6.** The back end will be implemented as **REST/JSON API.**  The main user roles will be as follows:  • *Anonymous User* – can only view the information pages such as News, Blog and classifieds.  • *Registered User* – can CRUD his own publications, write comments to his and other user’s listings and have access to their own profile page.  • *Reviewer (extends Registered User)* – can add post to Blog as well full CRUD to his own posts in Blog page.  • *Administrator* (extends *Reviewer*) – can manage (CRUD) all Registered Users, as well as Their comments. Can manage all the Information pages and their posts. |
|  |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Browse information** | The *User* can browse the information views (Home, Cars, Classifieds, Blog, About) | All users |
| * 1. **Register** | *Anonymous User* can register in the system by providing a valid e-mail address, first and last name, and choosing password. By default, all new registered users have *Registered* role.  *Administrator* can register new by entering *User Data* and choosing a Role (*Registered, Reviewer*, or *Administrator*). | *Anonymous User, Administrator* |
| * 1. **Change User Data** | *Registered User* can view and edit their personal *User Data.*  *Administrator* can view and edit *User Data* of all *Users* and assign them *Roles*: *Registered, Reviewer*, or *Administrator*. | *Registered User, Administrator* |
| * 1. **Manage Users** | *Administrator* can browse and filter users based on different criteria: first and last name, email, Role.  *Administrator* can choose a *User* to manage, and can manage the chosen User - edit view or delete.  *Administrator* can create a new user using *Register view*. | *Administrator* |
| * 1. **Manage Blog** | *Administrator* can browse and filter *Blog Page* based on different criteria: name of *Reviewer,* creation date, etc.  *Administrator* can create, edit and delete a new *Blog Post.*  *Reviewer can Create a new post in Blog page and update and delete his own posts.* | *Administrator, Instructor* |
| * 1. **Manage Car Data Base** | *Administrator* can browse car data base Page as well as create, edit and delete blog post. | *Administrator* |
| * 1. **Manage Classifieds** | *Administrator* can browse classifieds, add new *classified* using *Add/Edit Classified View*, and delete a *Classified* | *Administrator* |
| * 1. **Add/Edit Classified** | *Registered User and Reviewer can* create, edit and delete his own classifieds.  Administrator can create, edit and delete all users classifieds | *Registered User, Reviewer, Administrator* |
| * 1. **Add/Edit comment to classified** | *Registered User, Reviewer* can add comment to his Classifieds as well as update and delete them.  Administrator can add comment as well as update and delete all users’ comments. | *Registered user, Reviewer, Administrator* |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (SPA Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Home** | Presents the introductory information for the purpose of the system as well as detailed instructions how to start using it. Prominently offers ability to register. | / |
| * 1. **Cars** | Presents cars characteristics as per manufacturer. Basic picture and information on Manufacturer, models, Price Range and etc. | */cars* |
| * 1. **Blog** | Presents reviewer posts with review of car of their choosing. Testing data and more in depth characteristic. | */blog* |
| * 1. **Classifieds** | Presents *classifieds for sale of a car, with pictures and information, such as Manufacturer and model of the car, year of production, mileage, pictures and etc.* | */classifieds* |
| * 1. **User Registration** | Presents a view allowing the *Anonymous Users* to register. | */register* |
| * 1. **Login** | Presents a view allowing the users to login. | */login* |
| * 1. **User Data** | Presents ability to view and edit personal *User Data*, | */myProfile* |
| * 1. **Users** | Presents ability to manage (CRUD) Users and their User Data (available for *Administrators* only). | */users* |
| * 1. **About** | Presents information about the *Car App and his owner* | */about* |

|  |  |  |
| --- | --- | --- |
| 1. API Resources (Node.js Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Users** | GET *User Data* for all users, and POST new *User Data* (Id is auto-filled by *Carly* and modified entity is returned as result from POST request). Available only for *Administrators*. | */api/users* |
| * 1. **User** | GET, PUT, DELETE *User Data* for *User* with specified *userId*, according to restrictions decribed in UCs. | */api/users/:userId* |
| * 1. **Login** | POST *User Credentials* (e-mail address and password) and receive a valid *Security Token* to use in subsequent API requests. | */api/login* |
| * 1. **Logout** | POST a logout request for ending the active session with *Carly,* and invalidating the issued *Security Token*. | */api/logout* |
| * 1. **Blog** | GET *All blog Posts*, and POST new *blog post*(Id is auto-filled by *Carly* and modified entity is returned as result from POST request), according to *User's Role* and identity security restrictions. | */api/blog* |
| * 1. **Blog Post** | GET, PUT, DELETE *Blog Post* (including Comments) with specified *postId*. | */api/blog/:postId* |
| * 1. **Classifieds** | GET *All classifieds*, and POST new *classified*(Id is auto-filled by *Carly* and modified entity is returned as result from POST request), according to *User's Role* and identity security restrictions. | */api/classifieds* |
| * 1. **Classified** | GET, PUT, DELETE *Classified* (including Comments) with specified *classifiedId*. | */api/classifieds/:classifiedId* |
| * 1. **Cars** | GET *All cars*, and POST new *car*(Id is auto-filled by *Carly* and modified entity is returned as result from POST request). Available only for Administrators. | */api/cars* |
| * 1. **Car** | GET, PUT, DELETE *car* with specified *classifiedId*. Available only for Administrators. | */api/cars/:carId* |