|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | **Web Application & Service Development with Spring Framework v5** **- 2019** | |
|  | | | |
| Project author | | | |
| № | Pseudonym | | Face-to-face/ online |
| 1 | MyMan | | online |

|  |  |
| --- | --- |
| Project name | Car Trading Bulgaria |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| The way we trade cars has drastically changed over the years. Car Trading Bulgariaprovides ability for sellers to post their cars, and for normal users to find their dream car without leaving the house. Administrators can manage them. The system will be developed using ***Spring 5 Application Development Framework***.It will implement web-based front-end client using ***Thymeleaf*** templates. Each page will have a distinct URL, and the routing between pages will be done server side using ***SpringMVC***. The backend will be implemented with controllers returning views. The main user roles (actors in UML) are:  • *Anonymous User* – can only view the landing page  • *Buyer* (extends *Authenticated User*) – can view car’s posted by Sellers  • *Seller* (extends *Authenticated User*) – can create *Car sales,* see all cars posted  • *Administrator* (extends *Authenticated User*) – can manage (create, edit user data and delete) all *Authenticated Users*, as well as *Car Sale Posts*. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Browse landing page** | The *Anonymous User* can browse the landing page only. | All users |
| * 1. **Change User Data** | *Authenticated User* can view and edit her personal *User Data*  *Administrator* can view and edit *User Data* of all *Users* and assign them *Roles*: *Student, Instructor*, or *Administrator*. | *Authenticated 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 (using Change User Data UC) or delete. | *Administrator* |
| * 1. **Manage Car Sale Posts** | *Authenticated User* can browse and filter *Car Sale Posts* based on different criteria: name of *Car Sale Post,* creation date, etc.  *Administrator* or *Seller* can create a new *Car Sale Post.*  *Administrator* can choose a *Car Sale Post* to manage, and can edit the Car Sale Post*,* or delete it if necessary.  *Seller* can choose *Car Sale Post* to manage from his own, and can edit the Car Sale Post*,* or delete it if necessary. | *Authenticated User , Administrator, Seller* |
| * 1. **Manage Comments** | *Car Sale Post can have comments. Authenticated user can add/edit/remove his own comments.*  *Seller can edit/remove comments if they are on his own post.*  *Administrator can edit/remove all user comments.* |  |
| * 1. **Add/Edit Car Sale Post** | *Seller* or *Administrator* specifies/edits *Car Sale Post* meta-data such as: name, brand, horsepower, description, city, images. *Seller* or *Administrator* adds/edits *Images* using *Edit Image UC,* and can delete all images*. Seller* or *Administrator* saves the edited *Car Sale Post.* | *Seller, Administrator* |
| * 1. **Upload/Edit Image** | *Seller* or *Administrator* uploads an image from his PC. | *Seller, Administrator* |
| * 1. **End Car Sale** | *When an Authenticated User ends up buying a car the Car Sale Post is removed and added to the Seller’s collection of successful car sales accessed by Browse Seller Information UC.* | *Authenticated User, Seller* |
| * 1. **Browse Seller Information** | *Authenticated User* has User Data on his profile page.  Seller has also his successful car sales added. | *Authenticated User, Seller* |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Home** | Presents the introductory information for the purpose of the system as well as a login button.  If user is authenticated just show him buttons to go to the car sale posts and personal page | / |
| * 1. **Cars** | Presents cars available according to *User's Role* and identity. Offers abilities to go to browse, choose, create, read, update, delete (CRUD) for Car Sale Posts, as defined by *UCs* (for *Administrators* and *Sellers* only).  In view you can create comments and add them to postz. | */car-sale-posts*  */car-sale-posts/new*  */car-sale-posts/edit*  */car-sale-posts/view* |
| * 1. **Login** | Presents a view allowing the users to login. | */login* |
| * 1. **User Data** | Provides ability to view and edit personal *User Data*.  For Sellers and Admins it has also successful car sales. | */personal* |
| * 1. **Users** | Presents ability to manage (CRUD) Users and their User Data (available for *Administrators* only, as described in UCs). | */users*  */users /new*  */users /edit*  */users /view* |
| * 1. **About** | Presents information about the project and his owner. | */about* |

|  |  |  |
| --- | --- | --- |
| 1. API Resources (Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Users** | GET *User Data* for all users, and POST new *User Data* (Id is auto-filled and modified entity is returned as result from POST request). Available only for *Administrators*. | */api/users*  */api/users/register(GET, POST)* |
| * 1. **User** | GET, PUT, DELETE *User Data* for *User* with specified *userId*, according to restrictions described in UCs. | */api/users/{userId}*  */api/users/{userId}/edit (GET,POST)*  */api/users/{userId}/delete* |
| * 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. **Register** | POST UserCredentials | */api/register* |
| * 1. **Logout** | POST a logout request for ending the active sessionand invalidating the issued *Security Token*. | */api/logout* |
| * 1. **Car Sale Posts** | GET Car Sale Posts (according to *User's Role* and identity) and POST new *Car Sale Post* (Id is auto-filled and modified entity is returned as result from POST request). | */api/car-sale-posts/*  */api/car-sale-posts/new(GET, POST)* |
| * 1. **Car Sale Post** | GET, PUT, DELETE *Car Sale Post Data* (including *Images*) for *Car Sale Post* with specified *carPostId*. | */api/ car-sale-posts /edit/{carPostId}(GET, PUT)*  */api/ car-sale-posts /delete/{carPostId}* |
| * 1. **Successful Car Sale Posts** | GET Successful Car Sale Posts for specific Seller with sellerId. | */api/successful-car-sales/{sellerId}* |