|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | Multimedia Technologies with Angular and TypeScript - 2020 | |
|  | | | |
| Project author | | | |
| № | Pseudonym | | Face-to-face/ online |
| 1 | theroommate | | online |

|  |  |
| --- | --- |
| Project name | Roommates |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| Simple progress tracker app for roommates. The main purpose is to balance the work in a residence and to eradicate laziness from certain roommates by selecting activities that the roommate has done during the day and gain different points based on the level of the activity. By the end of the week the person with the highest score receives certain amount of money from the other roommates as a reward. The system will be developed as a *Single Page Application (SPA)* using ***Angular*** as front-end, and ***Node.js*** as backend. Each view will have a distinct URL. The backend will be implemented as a ***REST/JSON API*** using JSON data serialization. The main user roles (actors in UML) are:  • *Anonymous User* – can only register  • *Registered User* – can create a room or join an existing room by providing room credentials, can only do stuff in the rooms where the user is a member. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Register** | *Anonymous User* can register in the system by providing a valid e-mail address, first and last name, and choosing password. | *Anonymous User* |
| * 1. **Create/Join a room** | The Registered User can provide a *Token Key* in order to identify the Room to be enrolled to or create one. | *Registered User* |
| * 1. **Manage Activities** | Members of a room can add their activities and assess them. | *Registered User* |
| * 1. **Select activities** | The Registered User can select from room activities and mark them as done | *Registered User* |
| * 1. **Monitor progress** | Monitor progress of all registered users in a room. | *Registered User* |

|  |  |  |
| --- | --- | --- |
| 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. **User Registration** | Presents a view allowing the *Anonymous Users* to register in *OKTS*, as well as to enroll to a *Student Group* by presenting a *Token Key* for that group. | */register* |
| * 1. **Login** | Presents a view allowing the users to login. | */login* |
| * 1. **Rooms** | Presents ability to browse and manage *Rooms*, as described in UCs. | */rooms* |
| * 1. **Dashboard** | Presents in real time the Roommates’ progress as described in UCs. | */dashboard* |

|  |  |  |
| --- | --- | --- |
| 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 *Roommates* 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 *Roommates,* and invalidating the issued *Security Token*. | */api/logout* |
| * 1. **Rooms** | GET Rooms, and POST new *Room* (Id is auto-filled by *Roommates* and modified entity is returned as result from POST request), according to *User's Role* and identity security restrictions. | */api/rooms* |
| * 1. **Room** | GET, PUT, DELETE *Room* (including assigned roommates) for *Room* with specified *roomId*. | */api/rooms/{roomId}* |
| * 1. **Activities** | GET activities for a particular room and POST new activities. | */api/activities* |