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

|  |  |
| --- | --- |
| Project name | Group trip planner |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| The group trip planner will help with organizing group holidays easier. You will be able to create a trip and add your friends to that trip so they can contribute to the schedule and track your group expenses more easily. The system will be developed using Spring 5 Application Development Framework. The backend will be implemented as a REST/JSON API using JSON data serialization. The main user roles are:  •Unregistered user  •Registered user (trip owner / trip participant)  •*Administrator* (extends *Registered User*) – can manage (create, edit user data and delete) all *Registered Users*, as well as *Trips*. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Register** | *Unregistered User* can register in the system by providing a valid e-mail address, first and last name, and choosing password. | *Unregistered User, Administrator* |
| * 1. **Change User Data** | *Registered User* can view and edit her personal *User Data Administrator* can view and edit *User Data* of all *Users*. | *Registered User, Administrator* |
| * 1. **Manage Users** | *Administrator* can browse and filter users based on different criteria: first and last name, email.  *Administrator* can choose a *User* to manage, and can manage the chosen User - edit (using Change User Data UC) or delete.  *Administrator* can create a new user using *Register UC*. | *Administrator* |
| * 1. **Manage Trips** | *Administrator* or *Trip owner* can browse and filter *Trips*  based on different criteria: name of *Trip* etc.  *Administrator* or *Instructor* can create a new *Trip,* and invite other registered users to join the trip as Trip Participants;*.*  *Administrator* can choose a *Trip* to manage, and can add or remove *Participants* to the *Trip,* or delete it if necessary. | *Administrator, Trip owner* |
| * 1. **Contribute to schedule** | *Trip owner and trip participant* can see the trip schedule and contribute to it. | *Trip owner, Trip participant* |
| * 1. **Manage expenses** | *Trip owner and Trip participants can see and add/edit expenses to a certain trip* | *Trip owner, Trip participant* |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (Frontend) : throw new NotImplementedExeption(); | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Home** | Presents the introductory information for the purpose of the system as well as detailed instructions on how to start using it. Prominently offers ability to register. | / |
| * 1. **Tests** | Presents tests available according to *User's Role* and identity. Offers abilities to browse, choose, create, read, update, delete (CRUD) Tests, as defined by *UCs* (for *Administrators* and *Instructors* only). | */tests* |
| * 1. **Complete Test** | Presents tests available to particular Student (according to *Student Group* participation) allowing him/her to choose a test and complete it. | */complete-test* |
| * 1. **Test Results** | Presents *Test Results* available according to *User's Role* and identity as defined by *UCs*. | */test-results* |
| * 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. **User Data** | Provides ability to view and edit personal *User Data*, enroll to more *Student Groups* by presenting a *Token Key* for particular *Student Group*, as well as deregister from *OKTS*. | */personal* |
| * 1. **Student Groups** | Presents ability to browse, filter and manage *Student Groups (*for *Administrators* and *Instructors),* and assign Students to them (for *Administrators* only), as described in UCs. | */groups* |
| * 1. **Dashboard** | Presents in real time the Students’ progress on active Tests, subject to Role restrictions described in UCs. | */dashboard* |
| * 1. **Users** | Presents ability to manage (CRUD) Users and their User Data (available for *Administrators* only, as described in UCs). | */users* |
| * 1. **About** | Presents information about the *OKTS* project 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 *GTP* 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 described 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 *OKTS,* and invalidating the issued *Security Token*. | */api/logout* |
| * 1. **Trips** | GET *Trips* , and POST new *Trip*  (Id is auto-filled by *GTP* and modified entity is returned as result from POST request), according to *User's Role* and identity security restrictions. | */api/trips* |
| * 1. **Trip** | GET, PUT, DELETE *Trip* for *Trip* with specified *tripId*. | */api/trips/{tripId}* |
| * 1. **Tests** | GET users (according to *User's Role* and identity) and POST new *Test* (Id is auto-filled by *GTP* and modified entity is returned as result from POST request). | */api/tests* |
| * 1. **Expenses** | GET *Expenses* (according to *User's Role* and identity) ) for *Trip* with specified *tripId*, and POST new *Expense* (Id is auto-filled by *GTP* and modified entity is returned as result from POST request). | */api/trips/{tripId}/expenses* |
| * 1. **Expense** | GET, PUT, DELETE *Test Result* (according to *User's Role* and identity) for *Trip* with specified *tripId* and *expense* with specified *expenseId.* | */api/trips/{tripId}/expenses/{expenseId}/* |
| * 1. **Schedule** | GET,PUT,DELETE Trip schedule for Trip with specified tripId. | */api/trips/{tripId}/schedule* |