|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | Web Application & Service Development with Spring Framework v5 - 2020 | |
|  | | | |
| Project author | | | |
| № | Pseudonym | | Face-to-face/ online |
| 1 | Red Devil | | Online |

|  |  |
| --- | --- |
| Project name | Movie Reviews |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| The Movie Reviews provides ability for administrators and moderators to add movies for reviews, and for users to review them. In addition to that it allows users to register, and administrators to manage them. The system will be developed using Spring 5 Application Development Framework. It will implement web-based front-end client using Thymeleaf/JSP templates and jQuery or React.js/Vue.js/Angular JavaScript/TypeScript libraries for asynchronous page data updates. Each page will have a distinct URL, and the routing between pages will be done server side using SpringMVC/ WebFlux. The backend will be implemented as a REST/JSON API using JSON data. The main user roles (actors in UML) are:  • Anonymous User – can only view the movies added by administrators  • Reviewer (extends Registered User) – can add reviews to movies.  • Moderator (extends Registered User) – can create, read, update movies and reviews.  • Administrator (extends Registered User) – can manage (create, edit user data and delete) all Registered Users, as well as reviews and movies. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| Use case name | Brief Descriptions | Actors Involved |
| * 1. Browse movie reviews | The User can browse all movie reviews | All users |
| * 1. Register | Anonymous User can register in the system by providing a valid e-mail address, username, and choosing password. By default, all new registered users have Reviewer role.  Administrator can register new user by entering User Data and choosing a Role (Reviewer, Moderator, or Administrator). | Anonymous User, Administrator |
| * 1. Change User Data | Administrator can view and edit User Data of all Users and assign them Roles: Reviewer, Moderator, or Administrator. | Registered User, Administrator |
| * 1. Manage Users | Administrator can browse and filter users based on different criteria: username, email, Role.  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 Movies | Administrator or Moderator can browse and filter Movies based on different criteria: name of Movie, release date, etc. Administrator or Moderator can create new movies for review. | Administrator, Moderator |
| * 1. Add/Edit Movie | Moderator or Administrator specifies/edits Movie meta-data such as: movie title, description. Registered User creates/edits Review using Edit Review UC. Administrator can delete reviews. Moderator or Administrator saves the edited Movie. | Registered User, Administrator, Moderator |
| 2.7 Order tickets from movie theaters. | Registered users can order tickets for movies. | Registered users |
| 2.8 Add edit movie reviews | Moderator or Administrator specifies/edits Movie Review content | Moderator, Administrator |
| 2.9 Browse profile | Registered User can browse his profile. | Registered User |
| 2.10 Browse movie theaters | All users can see in which theater a movie is played | All users |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (Frontend) | | |
| View name | Brief Descriptions | URI |
| * 1. Home | Presents the introductory information for the purpose of the system. Prominently offers ability to register and login. | / |
| * 1. Movie Reviews | Offers abilities to browse, choose, create, read, update, delete (CRUD) Movie reviews, as defined by UCs (for Administrators and Moderators only). | /reviews |
| * 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 | Provides ability to view and edit personal User Data | /personal |
| * 1. Movies | Presents ability to browse, filter and manage Movies (for Administrators and Moderators) | /movies |
| * 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 project and his owner. | /about |
| * 1. Movie details | Presents view that shows additional information for movies and where you can watch a movie | /details |
| 3.10 Payment | Presents a view allowing users to buy tickets | /payment |

|  |  |  |
| --- | --- | --- |
| 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 |
| * 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 and invalidating the issued Security Token. | /api/logout |
| * 1. Movies | GET Movies, and POST new Movie (Id is auto-filled and modified entity is returned as result from POST request), according to User's Role and identity security restrictions. | /api/movies |
| * 1. Movie | GET, PUT, DELETE for Movie with specified movieId. | /api/movies/{movieId } |
| * 1. MovieReviews | GET MovieReviews and POST new MovieReview (Id is auto-filled and modified entity is returned as result from POST request). | /api/reviews |
| * 1. MovieReview | GET, PUT, DELETE for MovieReview with specified reviewId. | /api/ reviews /{reviewId} |
| * 1. MovieTheaters | GET MovieTheaters | /api/theaters |
| 4.10. MovieTheater | GET MovieTheater | /api/theaters/{theaterId} |
| 4.11 Payment | POST Payment for buying a movie ticket. | /api/payment |