|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Assignment 1 | | Project Summary | | |
| Course | | Fullstack Application Development with Node.js + Express.js + React.js - 2017 | | |
|  | | | | |
| Project author | | | | |
| № | First name, last name | | E-mail | Face-to-face/ online |
| 1 | Milena Monova | | Mimonova13@gmail.com | face-to-face |

|  |  |
| --- | --- |
| Project name | PinterestLike |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| The application will be web application, developed to discover and save posts and pictures from the users. The system will be developed as a *Single Page Application (SPA)* using ***React.js*** as front-end, and ***Node.js + express*** as backend technologies.  The main user roles (actors in UML) are:  • *Anonymous User* – can only view the posts and pictures.  • *User* – can view, save and create posts and relate them to existing category. The user also will be able to offer new categories.  • *Administrator* – can do everything that the regular user can, and also can create new cateogires. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Browse information and search by category name** | The *User* can browse and search for posts and pictures by category. | 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 *User* role. | *Anonymous User* |
| * 1. **Change User Data** | *Registered User* can view and edit her personal *User Data*. | *Registered User, Administrator* |
| * 1. **Manage Users** | *Administrator* can browse, and filter users based on different criteria: first and last name, email, Role.  *Administrator* can Delete users. | *Administrator* |
| * 1. **Manage Categories and Posts** | *Administrator* can create new category, browse and accept offered by user’s categories.  *Administrator* can change posts categories.  *Administrator* can delete posts. | *Administrator* |
| * 1. **Add Post** | *User* can add post and relate it to many categories | *User, Administrator* |
| * 1. **Search and save posts to albums** | *User* or *Administrator* can search for posts by categories and save them in his album/s. | *User, Administrator* |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (SPA Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Home** | Page with popular posts and pictures. The page will have search bar. | / |
| * 1. **Login or Registration** | Login or Registration page | */login* |
| * 1. **User Data** | Details about the user. | */user* |
| * 1. **Albums** | User albums | */users/id/albums* |
| * 1. **Single album** | User album with all the posts inside. | */users/id/albums/id* |

|  |  |  |
| --- | --- | --- |
| 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 *OKTS* 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*. | */api/users/{userId}* |
| * 1. **Categories** | GET categories (according to *User's Role* and identity) and POST new *Category* (Id is auto-filled by *OKTS* and modified entity is returned as result from POST request). | */api/tests* |
| * 1. **Category** | GET, PUT, DELETE *Category Data* for *Category* with specified *categoryId*.  GET category will also return all pins associated with this cateogry | */api/categories/{categoryId}* |
| * 1. **Albums** | GET *User Albums Results* (according to *User's Role* and identity)) | */api/albums* |
| * 1. **Pin** | GET, POST, DELETE *Pin Result* (according to *User's Role* and identity) for *Pin* with specified *pinId* and *Test Result* with specified *albumId.* | */api/albums/{albumId }/pins/{pinId}/* |