|  |  |  |  |
| --- | --- | --- | --- |
| Assignment 1 | | Project Summary | |
| Course | | Fullstack Application Development with Node.js + Express.js + React.js - 2020 | |
|  | | | |
| Project author | | | |
| № | Pseudonym | | Face-to-face/ online |
| 1 | Zergy | | online |

|  |  |
| --- | --- |
| Project name | *Otaku-FanArt* |

|  |
| --- |
| 1. Short project description (Business needs and system features) |
| An easy and fun way to bowse and socialize with others by the one and only true passion, the anime world.***Otaku-FanArt*** provides you with plenty of fanart content, you can also have a conversation over the same topics with other users, and most importantly save and browse your favorite liked fanart. The system will be developed as a *Single Page Application (SPA)* using ***React.js*** as front-end, and ***Node.js + express*** as backend technologies. Each view will have a distinct URL, and the routing between pages will be done client side using ***React Router***. The backend will be implemented as a ***REST/JSON API*** using JSON data serialization. The main user roles (actors in UML) are:   * ***Anonymos User*** – can view the fanart but cannot comment, like and can’t lookup profiles. * ***Registered User*** – can comment on fanart, like the pictures, and save fanart to his profile. * ***Moderator*** *(extends Resgistered User)* – checks for unsafe content and removes it and moderates the comment sections, can also ban profiles that have violated the terms. |

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Browse fanart content** | The *User* can browse and view recent fanart and regular feed of randomly selected pictures. | All users |
| * 1. **Register** | *Anonymous Users* can register. After the registration they must have a unique username, strong password. After the registration they become a *Registered User*. | *Anonymous User* |
| * 1. **View User Profile** | *Registered users* can view their own posts and the ones that they saved. *Registered users* can also access other useres profiles. | *Registered User* |
| * 1. **Personalize User Profile** | *Registered users* can edit their profile, by the abbility to change the password or edit their profile picture. | *Registered User* |
| * 1. **Delete User Profile** | *Registered users* can delete their profiles. In case of violating the terms and conditions the *moderator* can terminate an account. | *Registered User, Moderator* |
| * 1. **Make a new publication** | *Registered users* can make a publication that consists of a title tags that describe the picture and the desired picture. | *Registered User* |
| * 1. **Comment & Like on publications** | *Registered users* can comment and/or like onder publications including their own. | *Registered user* |

|  |  |  |
| --- | --- | --- |
| 1. Main Views (SPA Frontend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Home** | Shows the top 20 recent publications as well as other. There are also a sort option that the allows the user to sort the content by date and by the number of likes on a fanart post. On hovering over the publication, it shows the number of likes and comments. On clicking it leads you to the full post. There is a search bar on the navigation bar that allows the *User* search by publication’s title or related tags.On the right side of the navigation bar is the Login button as well as the *User’s* profile picture (If logged in). | / |
| * 1. **Login / Register** | Upon clicking on the Login / Sign In button on the right side of the navigation bar, if the *User* is not yet registered then it opens a form that allows the *Anonymous User* to fill the form and registered. In case of an already *Registered User* then a Login pops up enabling the *Registered user* to Log in. | */* |
| * 1. **Profile** | Presents the *Registered User* a page that consists of the user’s username, profile picture and published fanart by the user, | */profile/{username}* |
| * 1. **Current User Profile** | Same information as the Profile page but the user can see the posts that they have liked. | */profile* |
| * 1. **Create / Edit Fanart Publication** | Page that the *Registered User* can use to create a new publication. It consists of the Title that can be set to the post and tags that are related to the theme of the fanart. The Registered User can also edit their own publication. | */newFanArtPost /post/edit/{id}* |
| * 1. **View Fanart Publication** | Allows the *Anonymus User* and the *Registered User* to view publicated content. | */post/{id}* |
| * 1. **About** | Presents information about the *Otaku-FanArt* project and its owner. | */about* |
| * 1. **Comment** | Allows the *Registered User* to comment on posts. |  |

|  |  |  |
| --- | --- | --- |
| 1. API Resources (Node.js Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **User** | GET the information about a wanted User. DELETE the User with selected name. | */api/users/{userId}* |
| * 1. **Login** | POST the username and hashed password. (The server sends a vaid security token if the login is successful) | */api/login* |
| * 1. **Logout** | POST the security token. | */api/logout* |
| * 1. **Register** | POST creates a new User (Id is auto filled). | */api/register* |
| * 1. **Like** | POST sends the id of the post.  DELETE sends the id of the post. | */api/post/like* |
| * 1. **Posts** | GET returns all the posts in chronological order. They can also be sorted by the number of likes if the sortBy parameter is equal to “likes”. There can be additional parameter tags, if it exists then it will show the posts that have those tags. | */api/posts* |
| * 1. **Post** | GET the path to the image, title, and all the comments. POST Title, Base64 encoded image, tags. DELETE id of the post | */api/post /api/post/{postId}* |
| * 1. **Post Comment** | POST the id of the post and the text for the comment. DELETE id of the comment. | */api/post/{postId}/comment* |
| * 1. **User Posts** | GET all the posts of the user ordered by publish date. | */api/user/{userId}/posts* |
| * 1. **Liked posts by the User** | GET posts that are liked by the user. | */api/user/likedPosts* |