| Project name | | Threddit | | |
| --- | --- | --- | --- | --- |
| Project author | | |  |
| № | Names of Participants | | FN |
| 1 | Aleks Uzunov | | 7MI0600096 |
| 1. Short project description (Business needs and system features) | | | | |
| ***Threddit*** is a ***Golang***-based system similar to the already existing ***Reddit*** *–* a [social news](https://en.wikipedia.org/wiki/Social_news) aggregation, content rating, and discussion website. Registered users (commonly referred to as "***redditors***") submit content to the site such as links, text posts, images, and videos, which are then voted up or down by other members. Posts are organized by subject into user-created boards called "communities" or "***subreddits***". Submissions with more upvotes appear towards the top of their subreddit and, if they receive enough upvotes, ultimately on the site's front page. ***Reddit*** administrators moderate the communities. Moderation is also conducted by community-specific moderators, who are not ***Reddit*** employees. The system will be developed using ***Golang, Chi, gorm and MySQL DB***. It will implement a web-based front-end client using Golang’s ***html/template*** package and ***jQuery*** for asynchronous page data updates. 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 view the information and individual subreddit pages. * *Redditor (extends Registered User)* – can contribute to the site with content (such as images, text, videos, and links), as well as rate (upvote or downvote) already existing content. A *Redditor* that creates their own community (**subreddit**) automatically becomes its first **community-specific** *Moderator*. * *Administrator* (extends *Registered User*) – can manage (create, edit user data and delete) all *Registered Users*. | | | | |



| 2. Main Use Cases / Scenarios | | |
| --- | --- | --- |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| **2.1. Browse information** | The *User* can browse the information views (Home, About) and individual subreddit views. | All users |
| **2.2. Register** | An *Anonymous User* can register in the system by providing a valid e-mail address and choosing a username and a password. By default, all new registered users have the *Redditor* role.  *Administrator* can register new by entering *User Data* and choosing a Role (*Redditor* or *Administrator*). | *Anonymous User, Administrator* |
| **2.3. Change User Data** | A *Redditor* can view and edit their personal *User Data*.  An *Administrator* can view and edit *User Data* of all *Users* and assign them the *Administrator* role. | *Redditor, Administrator* |
| **2.4. 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* |

| **2.5. Manage Posts** | A *Redditor* can browse their *Posts*, add a new *Post* using  *Add/Edit Post UC*, and delete their own *Post*.  *Administrator* can browse the posts of all *Redditors*, edit and delete them. | *Redditor, Administrator* |
| --- | --- | --- |
| **2.6. Manage Comments** | A *Redditor* can browse their *Comments*, add a new *Comment* using *Add/Edit Comment UC*, and delete their own *Comment*.  *Administrator* can browse the comments of all *Redditors*, edit and delete them. | *Redditor, Administrator* |
| **2.7. Add/Edit Post** | *Redditor* or *Administrator* specifies/edits *Post* meta-data such as: title, caption, tags, etc.    *Redditor* or *Administrator* creates/edits *Comments* using | *Redditor, Administrator* |

|  | *Add/Edit Comment UC,* and can delete chosen *Comment. Redditor* or *Administrator* saves the edited *Post.* |  |
| --- | --- | --- |
| **2.8. Add/Edit Comment** | *Redditor* or *Administrator* specifies/edits a *Comment*’s content. | *Redditor, Administrator* |
| **2.9. Upvote/Downvote Post/Comment** | *Redditor* browses posts and comments and can sort and filter them using different metadata fields. *Redditor* hasthe option to upvote or downvote a post or a comment. | *Redditor* |
| **2.10. Manage Subreddits** | A *Redditor* can browse their *Subreddits (*the ones they’ve created as well as the ones they’ve joined*)*, add a new *Subreddit* using *Add/Edit Subreddit UC*, and delete their own *Subreddit*.  *Administrator* can browse the subreddits of all *Redditors*, edit and delete them. | *Redditor (Moderator), Administrator* |
| **2.11. Add/Edit Subreddit** | *Redditor (Moderator)* or *Administrator* specifies/edits a *Subreddit*’s content. A *Redditor (Moderator)* can only modify a *Subreddit* they themselves created*.* | *Redditor (Moderator), Administrator* |
| 3. Main Views (Frontend) | |  |
| **View name** | **Brief Descriptions** | **URI** |
| **3.1. Home** | Presents a user’s feed if they have an ongoing session (otherwise, the feed is randomized). Prominently offers ability to register. | / |

| **3.2. User Registration** | Presents a view allowing the *Anonymous Users* to register. | */register* |
| --- | --- | --- |
| **3.3. Login** | Presents a view allowing the users to login. | */login* |
| **3.4. User Data** | Provides ability to view and edit personal *User Data*. | */profile* |
| **3.5. Users** | Presents ability to manage (CRUD) Users and their User Data (available for *Administrators* only, as described in UCs). | */users* |
| **3.6. About** | Presents information about the***Threddit*** project and his owner. | */about* |
| **3.7. Search** | Presents information about posts, comments, communities or people based on a user’s search query. | */search* |
| **3.8. Subreddit** | Presents a view of the specified subreddit. | */r/{subredditName}* |
| **3.9. Subreddit Posts and Comments** | Presents a view of a specific post and its comments. | */r/{subredditName}/posts/{postId}* |

| 4. API Resources (Backend) | | |
| --- | --- | --- |
| **View name** | **Brief Descriptions** | **URI** |
| **4.1. Users** | GET *User Data* for all users, and POST new *User Data* (Id is auto-filled by ***Threddit***and modified entity is returned as result from POST request). Available only for *Administrators*. | */api/users* |
| **4.2. User** | GET, PUT, DELETE *User Data* for *User* with specified *username*, according to restrictions described in UCs. | */api/users/{username}* |
| **4.3. Login** | POST *User Credentials* (username and password) and receive a valid *Security Token* to use in subsequent API requests. | */api/login* |
| **4.4. Register** | POST *User Credentials* (email, username and password), create *User* and receive a valid *Security Token* to use in subsequent API requests. | */api/register* |
| **4.4. Logout** | POST a logout request for ending the active session and invalidating the issued *Security Token*. | */api/logout* |
| **4.5. Posts** | POST new *Post* (Id is auto-filled by ***Threddit***and modified entity is returned as result from POST request). | */api/posts* |
| **4.6. Post** | GET, PUT, DELETE *Post Data* for *Post* with specified *postId*. | */api/posts/{postId}* |
| **4.7. Comments** | POST new *Comment* (Id is auto-filled by ***Threddit***and modified entity is returned as result from POST request). | */api/comments* |

| **4.8.** | **Comment** | PUT, DELETE *Comment Data* (according to *User's Role* and identity) for *Comment* with specified *commentId.* | */api/comments/{commentId}* |
| --- | --- | --- | --- |
| **4.9.** | **Subreddits** | POST new *Subreddit* (Id is auto-filled by ***Threddit***and modified entity is returned as result from POST request). | */api/subreddits* |
| **4.10.** | **Subreddit** | GET, PUT, DELETE *Comment Data* (according to *User's Role* and identity) for *Subreddit* with specified *subredditName.* | */api/subreddits/{subredditName}* |