CMPE 451 FALL 2021

Group Milestone 1 Report

GROUP 3 - BOXY

Executive Summary

Project Description

This project is a social media platform which lets people benefit from cumulation of content while they are sharing experiences and knowledge. The main building block of the platform is data types (post types). Data types organize the posts on the platform. It makes communities more organized and allows people to find the content they are looking for faster. The communities allow people with similar interests and purposes to contact each other. Users can share and reach contents of common interests in the communities. There are different types of users with different privileges. Community moderators are the ones who created the community or are assigned by another community moderator. They can create data types so that the users post community specific contents. They are able to remove posts, comments or users from their communities if needed. They can also restrict access to the community via making it private. Private communities are only accessible to the community members and common users can join them only if the community moderators accept their requests. However, common users are able to join the public communities without a request. Users can post contents, comment and like the existing posts and report the inappropriate ones to the moderators. The things that are offered to the users are not limited to these. The platform provides search and filter mechanisms to the users so that they do not have difficulties with finding the communities and the users they want to interact with. They are able to follow other users, visit their profiles and message them privately. The platform, of course, does not disregard the privacy of its users. It allows users to change their privacy settings and let them decide who can follow, message and view their profiles. All in all, this application aims to reach the next level of what we know as social media by letting self-structured communities be formed by themselves.

Project Status

For the first two weeks, we welcomed our new team members and reviewed the requirements together before proceeding to the implementation. After the review, we updated our scenarios and mockups accordingly. We refreshed our knowledge of using GitHub effectively and started discussing the distribution of the team members among the frontend, backend and mobile areas.

After deciding on the sub-team members, each team started discussing the technologies and tools that will be used in the development phase. For frontend React and Ant Design, for backend node.js and MongoDB, for mobile React Native was decided.

For the first milestone, we decided to implement the basic functionalities of a community based application. We decided to proceed with the mobile and frontend implementations in parallel. First of all, we implemented the login page, the landing page of our application. If not logged in, the user can register and create a new account in our application. While logged in, the user can log out of the application. Then, we decided to implement a navigation bar which the users can use to navigate between different pages of the application. Our first community based functionality is creating new communities. The user can fill out the community form to create a new community. After that, we created the community page, where the user can examine the details of a community. Within the community page, the user can create new post types in that community.

Road Ahead

Since the start of this course each and every one of our team members have gained great experience while developing the features assigned to them, so this will surely help us gain speed towards creating the next features of our projects. Our main goal is to complete most of our project requirements until the next milestone. Until this milestone as backend, web, and mobile teams, we not only implemented the features that were planned to be finished, but we also designed some of the future aspects of the project that are “View Posts component on Web and Mobile”, “View Profile component on Web”, “Home Page on Mobile”, “Settings component on Mobile”, “Profile Settings component on Web”, “Email

Confirmation page on Web", "Notification page on Mobile", and "Direct message page on Mobile". You can see those future work designs here.

According to our project plan, our Mobile and Web teams will complete the implementation of "Confirmation", "Home", "Community Posts", "Search Results", "Chat", "Notifications", "Profile", and "Settings" pages by the end of the next milestone. Also, according to our project plan, our Backend team will complete the implementation of "Socket", "Notification", "Search", "Email Configuration", "Mapper Interfaces", "Exception Handling", "Configuration", and the "Service, Controller, and Data Transfer Classes" that frontend pages require.

We will keep the communication within and between teams strong and continue having meetings to make sure that tasks are distributed evenly among all members, and make sure that everyone is on the same page. We are looking forward to implementing the next features that we have planned, and keeping up the good work.

List and status of deliverables

Group

Table 1: List and status of deliverables in team

Deliverable	Status
Communication Plan	**COMPLETED**
GitHub Wiki	**IN PROGRESS**
GitHub Issues	**IN PROGRESS**
Meeting Notes	**IN PROGRESS**
Project Requirements	**COMPLETED**
Project Plan & RAM	**IN PROGRESS**
User Scenarios & Mockups	**COMPLETED**
Use Case Diagram	**COMPLETED**

Class Diagram	**COMPLETED**
Sequence Diagrams	**COMPLETED**

Backend

Deliverable	Status
User Authentication	**COMPLETED**
Register/Login/Logout	**COMPLETED**
Create Post Type	**COMPLETED**
Create Community	**COMPLETED**
Create Post	**COMPLETED**
Get Post Types	**COMPLETED**
Get Posts	**COMPLETED**
Get Community	**COMPLETED**
View Post Detail	**COMPLETED**
View Post Type Detail	**COMPLETED**

Mobile

Table 3: List and status of deliverables in Mobile team

Deliverable	Status
User Authentication	**COMPLETED**
Register/Login/Logout	**COMPLETED**
Create Post Type	**COMPLETED**
Create Community	**COMPLETED**
Create Post	**COMPLETED**

View Post Detail	**PARTIALLY COMPLETED**

Frontend

Table 4: List and status of deliverables in Frontend team

Deliverable	Status
User Authentication	**COMPLETED**
Register/Login/Logout	**COMPLETED**
Create Post Type	**COMPLETED**
Create Community	**COMPLETED**
Create Post	**PARTIALLY COMPLETED**
View Post Detail	**PARTIALLY COMPLETED**

Evaluation of the status of deliverables

Group

Communication Plan

We think having a good communication plan is important to use time efficiently and have healthy communication among team members. We also observed that using different types of communication channels such as video calls, voice calls and chat is efficient in terms of time and organization. So we use several channels to communicate such as Google Meet, Slack and Discord. Everyone attends weekly meetings during lab hours. Besides, we hold online meetings as subgroups (backend, mobile, web) to work together. Slack is our discussion environment for tasks and each group has its own Alack channel. We use WhatsApp for daily communication.

Project Requirements

Details of the project are clarified as we prepare the Requirements. As we get feedback from our customer or we think improvements were needed, modifications were made on the requirements. We finalized our requirements in the first two weeks of the term.

User Scenarios & Mockups

We prepared the user stories and scenarios for different users. Corresponding mockups are designed for mobile or web interface. This part pictures the target group, features of the application and use of the application. Although designs can be improved during the implementation phase, we think our mockups are sufficiently descriptive to get a quick idea about the project. We updated our scenarios and mockups after we finalized the requirements.

UML Software Designs

Use case diagram illustrates the paths that each user type can take for a specific action. It describes the user roles that are explained in the Requirements in a visual way. One can find which steps to take in order to achieve a goal in the application. Coloring of the diagram helps to distinguish the separate actions and track the steps easily.

Class diagram specifies the objects, their attributes, methods and relations. It describes the project in a lower level in comparison to other diagrams. We use it when looking at required functions & fields and relationships between objects.

Sequence diagrams show the object interaction for a specific use case. Although sequence diagrams do not include all use cases we believe that most essential cases are provided. These diagrams are consistent with the class diagram and the use case diagram.

RAM (Responsibility Assignment Matrix)

RAM illustrates the work done by our team members in a detailed way. One can easily track the responsibility taken by each person. Tasks are classified to main categories and responsibilities classified as lead, contributor, reviewer and none. It can be seen that we tried to distribute tasks among team members in a balanced way. Since each sub-teams

have their own tasks and plans, there are separate RAMs for mobile, web and backend sub-teams.

Project Plan

Project plan visualizes the tasks, its date and duration. Our approximate progress in time can be tracked easily from there. We think planning is a key part of a project and we do our best to stick to the timing. Although we assign someone to each task in the plan, the owner of the task can be changed at our weekly meetings a week beforehand.

Backend

In this section, status of the deliverables seen in *table 3* will be evaluated. All deliverables listed here are connected to the backend.

USER AUTHENTICATION

User authentication system was implemented using a token. The token taken from the login endpoint was stored in the table of UserTokens in the database. Whenever a request is sent by the user except registration, this token also should be sent in the header of the request. This token is deleted when the user logs out. In this way, as stated in the requirements, we do not have a guest user in the system.

LOGIN, REGISTER AND LOGOUT

Since our application does not accept guest users, the user who is not logged in will not be able to benefit from any functionality. As Requirements 1.1.2. states, the user can log in to the application by entering username or email and password. As stated in the requirements, the user can register with username, e-mail address, and password. Email verification process has not been implemented in M1, but email verification and GDP & KVKK acceptance functionality will be implemented until M2.

CREATE POST TYPE

The most important feature of our project is that it is an organized social media platform. We provide this with post types.

Create Post Type endpoints takes some arguments:

- Token
- communityId: the community id that the post type is created
- Name: the name of the post type
- textFieldNames (list): the names of the text fields
- dateFieldNames (list): the names of the date fields
- linkFieldNames (list): the names of the link fields
- locationFieldNames (list): the names of the location fields

According to those arguments, post type is created and stored in our Post Type table in the database.

CREATE COMMUNITY

Create Community arguments:

- Token
- Name: the name of the community
- iconUrl: the icon url of the community

Creator is found from the given token and it automatically appointed as a moderator.

CREATE POST

Create Post Type endpoints takes some arguments:

- Token
- communityId
- postTypeId
- textFields (list): the values of the corresponding text fields
- dateFields (list): the values of the corresponding date fields
- linkFields (list): the values of the corresponding link fields
- locationFields (list): the values of the corresponding location fields

With given arguments, the post is created and stored in the database. Before storing, all data fields are checked if they are suitable for their corresponding field.

GET COMMUNITIES

Returns the communities that the user is the moderator or the member of the community.

GET POSTS

It takes a token as a header input and the community id. Then, it returns the posts of this community.

GET POST TYPES

It returns the post types of a community.

GET POST DETAIL

It returns the details of a given post id including date of the post creation and all the other field values.

GET POST TYPE DETAIL

It returns the details of a given post type id including the name of the post type and all the other field names.

Mobile

In this section, status of the deliverables seen in *table 3* will be evaluated. All deliverables listed here are connected to the backend.

USER AUTHENTICATION

User authentication system was implemented using a token. The token taken from the login endpoint was stored in async storage in the mobile application. We send this token in the header to verify that the user is logged in when sending a request to the backend. This token is deleted when the user logs out. In this way, as stated in the requirements, we do not have a guest user in the system.

LOGIN, REGISTER AND LOGOUT

Since our application does not accept guest users, the user who is not logged in will not be able to benefit from any functionality. For this reason, first of all, the registration and login pages were implemented. As Requirements 1.1.2. states, the user can log in to the application by entering username or email and password. A logged in user can log out with the logout button in the drawer. As stated in the requirements, the user can register with username, e-mail address, and password. Email verification process has not been implemented in M1, but email verification and GDP & KVKK acceptance functionality will be implemented until M2.

CREATE POST TYPE

The most important feature of our project is that it is an organized social media platform. We provide this with post types. Hence we wanted to include it in the M1. Since it is also the most difficult part of the project, we wanted to see and solve the challenges early.

We implemented two pages for this functionality. On the first page, the user is shown the communities he/she moderates and he/she is asked to choose one. On the second page, there is a form to create a custom post type for that community. On the last page, the user designs the post type by adding and naming the desired field types. These fields can be text, number, link, date and location as requirements suggest.

In the M2 we may improve this feature so that it can be accessed from the community page.

CREATE COMMUNITY

The user must be a creator or moderator of a community to create a post type. Therefore, we completed a simple community creation functionality in M1.

CREATE POST

We implemented three pages for creating post functionality. On the first page, the user selects a community among the communities he has joined. In the second

functionality, the user chooses a post type. On the last page, the user fills a post form that changes according to the post type. But we have a bug in this page for the date field type which will be fixed till M2.

Frontend

In this section, status of the deliverables seen in *table 4* will be evaluated. All deliverables listed here are connected to the backend.

USER AUTHENTICATION

In our application, authentication is handled using a token received from the backend upon logging in. When the frontend receives that token, it stores that token using the Redux library. This token is used to access the endpoints provided by the backend.

LOGIN, REGISTER AND LOGOUT

The user is greeted with a login page, which takes username and password as an input. Below the login field there are two buttons, one for resetting password and one for registering to the website. While the user is logged in the website, the user can log out of the website using the profile icon in the top navigation bar.

CREATE POST TYPE

Within the community page, the user can click a button which says “Create Post Type” and will be greeted with a post type form page. In the form page, the user is asked to enter a post type name and select the amount, type and names of post type fields. Finally, the user clicks the “Create” button and the post type becomes created belonging to that community.

CREATE COMMUNITY

In the top navigation bar, there is a “Create Community” button. When the user clicks that button, he/she is greeted with a community form page. In the form page, the user is asked to enter a community name, a list of topics, community description and the privacy of the community. Finally, the user clicks the “Create” button and the community becomes created.

Evaluation of tools and processes

Backend

We used Zoom for communication and planning. Our programming language is based on JavaScript - NodeJs. With the Mongoose tool, it was easier to implement some endpoints because Mongoose has built-in type casting, validation, query building etc. For our database, we used MongoDB (NoSQL). To display our database, we used the NoSQLBooster program which made it easy to change the data in the database for testing. To send requests, we used Postman which has a user friendly interface where you can easily add parameters, headers and body inputs.

We used Visual Studio Code as our IDE because it has a user-friendly interface where you can see errors and your changes in the branch. You can also easily pull and push your changes to your branch.

Mobile

We have used Zoom, Discord for mobile team meetings and Slack for instant communication. We are using React Native Framework for mobile development. In this framework the coding style is so similar to the React library which is the most popular library for web development in the world. We can easily create View components in just a few lines of code and give a style in css format just like in web development. This framework is the most popular framework in the world, so we could easily find lots of packages, code examples, projects, etc for a few minutes. In case the code we write throws an exception, we could find lots of related questions in the StackOverflow site, Github issues, or other platforms with that exception. This framework is working with the android emulator in harmony, we did not have any issues for the emulator connections, etc. We are using VsCode for mobile development, it has lots of extensions related to React Native, Javascript etc.

Frontend

We used Discord, Zoom and Slack for frontend team meetings, communication and planning. We are using React framework for frontend development, Axios and Redux for API and routing and antd library for UI and components. React is a popular framework therefore we can easily handle any problems. Axios is created just for API usage so the library is useful and suitable for our project. Redux open source library is for managing application state. It generally is using for building user interfaces in React. We use antd library for UI and components. Antd is the second biggest UI library for React and it gives component APIs for components' functionality and design. We use Visual Studio Code for source-code editor because sometimes we use live share plugin for simultaneous pair coding. Lastly, the most effective way for solving our problems is using Github repos, StackOverFlow and other libraries documentation. We can easily fix our problems by using those websites.

The requirements addressed

1.1.1. Registration

1.1.1.1. Users shall be able to register by providing a valid username, e-mail address, and password and accepting the terms of GDPR and KVKK.[1]

1.1.2. Login/Logout

1.1.2.1. Users shall be able to log in via their registered username/e-mail address and password.

1.1.2.2. Users shall be able to log out after logging in.

1.2.3 Posting Content

1.2.3.1. Users shall be able to share posts on communities.

1.2.3.1.1. Users shall be able to select a data type for their posts.

[1] Accepting GPDR and KVKK will be added in M2.

1.2.3.1.2. Users shall be able to include images, numbers, date, location and text as data type fields in posts.

1.2.3.3. Users shall be able to create new data types.

1.2.3.3.1. Users shall be able to define multiple attributes for data types. Attributes shall be images, numbers, date, location or text.

1.2.3.4. Users shall be able to see data types that exist in a community as a list.

1.2.6 Creating Community

1.2.6.2. Users shall be able to add a picture for the community.

1.4.1. Common Users

1.4.1.1 Common users shall complete the registration process as 1.1.1 implies.

1.4.1.4 Common users shall be able to log in/log out as explained in 1.1.2.

1.4.2. Community Members

1.4.2.2. Community members shall be able to post content as explained in the 1.2.3 section except 1.2.3.3.

1.4.3. Community Moderators

1.4.3.2. Community moderators shall be able to create data types as explained in 1.2.3.3.

Table of Work Done by Each Member

Member Name	Contributions
Kıymet Akdemir	• Participating in the weekly meetings • Participating in the Frontend & Mobile meetings • Participating in the Mobile meetings • Reviewing and updating the requirements • Reviewing and updating the class diagram • Reviewing and updating Scenario 4 • Reviewing and updating sequence diagram: Search Community • Learning React Native, watching tutorial • Designing and implementing Logout for mobile • Designing and implementing Post Details page for mobile • Modifying Swagger documentation

	● Preparing presentation for the First Customer Presentation with Zeynep ● Documenting First Customer Presentation notes
İrem Zeynep Alagöz	● Participating in the weekly meetings ● Attending the Frontend & Mobile Meetings ● Attending the Mobile Meetings ● Documenting the meeting note 10 ● Reviewing and updating the requirements ● Reviewing and updating the class diagram ● Updating the Scenario 1 ● Reviewing the two sequence diagrams: Login and Registration Issue ● Reviewing and redesigning mockup 4 Issue #193 & mockup 5 ● Updating communication plan ● Creating project plan for mobile development ● Creating project board for mobile, backend and frontend ● Creating issue templates for backend, frontend and mobile bug reports ● Organizing Meetings Notes for group, mobile, frontend and backend ● Learning about React Native by watching tutorial ● Designing and Documenting Backend Spec on Swagger. ● Designing and Implementing the Login Page ● Designing and Implementing the Drawer ● Designing and Implementing the Choose Community Page for Posting ● Designing and Implementing the Choose Post Type Page for Posting ● Designing and Implementing the Create Post Type Page ● Designing and Implementing the Choose Moderator Community Page for Post Type ● Designing and Implementing the Create Community Page ● Designing and Implementing the Under construction Page ● Preparing presentation for the First Customer Presentation ● Presenting First Customer Presentation: Introduction
Ufuk Arslan	● Participating in weekly general meetings ● Participating in frontend and mobile meeting ● Participating in the frontend meeting ● Taking notes in frontend meeting, and documenting them

	• Revising the project requirements • Revising and updating use case diagram • Reviewing and updating Mockup #1 • Learning to use React • Designing and implementing login page for Web • Reviewing Swagger documentation. • Reviewing axios and routing features • Reviewing create community component for web • Reviewing about community component for web • Reviewing navigation bar component for web • Reviewing confirmation page for web • Designing the post view component for web • Designing the profile view page for web • Designing the profile edit/settings page for web • Designing and implementing create post type components for Web
Halil Baydar	• Participating in the weekly meetings • Reviewing the mockups and scenarios • Attending in frontend and mobile meeting • Writing swagger documentation • Designing and implementing the Bottom navigation • Designing and implementing the create post screeen • Designing and implementing the dummy profile screen • Designing and implementing the dummy main screeen • Designing and implementing the dummy chat screeen • Designing and implementing the dummy notifications screeen • Building the apk file in release mode • Reviewing the future plan • Research React Native, Kotlin technologies • Opening issues and pull requests • Reviewing pull requests • Documenting meeting note
Burak Onur Duman	• Reviewing and updating the requirements • Updating the wiki homepage & sidebar • Reviewing the class diagram • Reviewing and updating the sequence diagram: Report Comment • Reviewing and updating the sequence diagram: Edit Profile Page • Reviewing and updating the scenario: Posting Content • Reviewing the Swagger YAML • Attending the Frontend & Mobile Meeting • Attending the Frontend Meetings • React Research & Study • Implemented Confirmation Page for Web

	• Implemented Navigation Bar for Web • Implemented About Community Component for Web • Implemented Create Community Component for Web • Deployed the React App to GitHub Pages at https://jcbod.github.io/my_app
Kaan Dura	• Participating in weekly general meetings • Participating in backend meetings • Revising the project requirements • Updating Swagger documentation • Nodejs & MongoDB Research & Study • Implementing general base & boilerplate of the backend • Implementing Auth Controller • Implementing Community Controller • Implementing Post Type Controller • Implementing Post Controller • Reviewing pull requests • Configuring & deploying MongoDB database • Implementing a Github Action for auto deployment of backend
Ufuk Karagöz	• Participating in weekly general meetings • Participating in frontend and mobile meeting • Participating in the frontend meeting • Revising the project requirements • Revising and updating use case diagram • Reviewing and updating Mockup Scenario #3 • Reviewing and updating sequence diagram: Like Post • Learning to use React • Designing and implementing registration page for Web • Reviewing Swagger documentation. • Designing the post view component for Web • Designing the create post component for Web • Designing the create post form component for Web • Designing the create post page for Web • Implementing Redux statements for Web • Implementing Axios API handling for Web • Implementing Routing for pages • Implementing Private and Public Routing for pages • Designing and implementing register API handling • Designing and implementing login API handling • Designing and implementing logout API handling • Designing and implementing create community API handling • Designing and implementing create post type API handling • Designing and implementing get communities component

Sevde Sarıkaya	● Participating in the weekly meetings ● Attending the Backend Meetings ● Reviewing and updating the requirements ● Reviewing and updating the use case diagram ● Reviewing the class diagrams: Change Notifications ● Reviewing mockup 3 ● Creating project plan for backend** ● Research & Study NodeJs with tutorials and coding ● Arrangements on Backend Spec on Swagger. ● Implementing Get Post Detail API ● Implementing Get Post Type Detail API ● Implementing formatters for viewing post and post type details. ● Implementing validations for these APIs. ● Testing the Backend endpoints by sending requests from Postman ● Testing the Swagger doc by sending requests from Swagger doc ● Creating samples for the demo ● Doing demo rehearsal with Ufuk and Halil ● Preparing presentation for the First Customer Presentation: Future Work

Deliverables

Requirements

Glossary

- **Account:** The whole structure that keeps the necessary data of users including username, password, communities, and posts.
- **Admin:** A person who has special privileges and is responsible for sustaining the system.
- **Alphanumeric:** A combination of alphabetical and numerical characters.
- **Attributes:** The features stored in data types.
- **Basics:** The collection of topics and features that are shared commonly in many platforms in the domain of our application.
- **Comment:** Text written by a user under a post.
- **Common User:** A registered user who has access to basic functionalities of the system without even joining a community.
- **Community:** A hub for sharing information among a group of members that have a common interest.
- **Community Member:** A person who is a member of a community.
- **Community Moderator:** A person who is the creator of the community or assigned by the moderator of the community and responsible for moderating the community.
- **Copyright content:** The content that needs to be protected from replication of its original properties.
- **Database:** The structured set of data-keeping everything about our application.
- **Data type:** The format and structure of posts about various topics in communities.
- **Discussion:** A post that enables the exchange of information between users.
- **Downtime:** The time during which the IT system is unavailable for service.
- **Event:** An activity that will take place at a specific time and place.
- **Follow:** The action of subscribing to see others' posts and getting notifications about them.
- **Home page:** An introductory page that contains trending posts, posts of followed users/communities.

- **Join:** The action of subscribing to see community's posts, get notifications about them, and post new content.
- **Notification:** The updates that are signaled to the user in some special cases like user interactions, upcoming events.
- **Offensive:** The unacceptable acts of users in terms of violence or discrimination.
- **Password:** A string of characters to log into the application.
- Post: A content that is uploaded to the platform and corresponds to a chosen data type.
- **Privacy of community:** Accessibility of community content to users, can be private or public.
- **Privacy of user:** Users' preferences on whether their profile is private or public.
- **Private community:** Community that is available to access for invited members.
- Profile: A page that is associated with a specific user and provides personal information.
- **Public community:** Community that is available to access for any user.
- **Read-only:** A data that is available for viewing but not modification.
- **Recommendation:** System that suggests posts and communities based on the preferences of the user.
- **Registered User:** A person who has completed the signup process.
- **Registration:** The official procedure to create an account in the application.
- **Report:** Filing a complaint against a user or post.
- **Response time:** The duration until the system responds to a user request.
- **System:** The entire application with all features and implementation, for both web and Android.
- **Search bar:** The structure that users make use of when they want to look for other users, communities, or keywords.
- **Privacy Policy and User Agreement:** The collection of ground rules and commitments that users need to agree on before registering into the application.
- **Tag:** A descriptive phrase that can be attached to any post.
- **User:** Anyone who benefits from the platform as a registered or admin.
- **Username:** The unique string for user identification.
- **User interaction:** Any means of communication that occurs between users.

- **User roles:** A title that defines the user's status and accessible actions on the platform.
- **Wikidata:** A free knowledge base that can be accessed and edited by everyone.
- **Visible:** Eligible to view, not concealed.

Functional Requirements

- **1. User Requirements**
- 1.1. Basics
    - 1.1.1. Registration
        - 1.1.1.1. Users shall be able to register by providing a valid username, e-mail address, and password and accepting the terms of GPDR and KVKK.

        - 1.1.1.2. After completing step 1.1.1.1, users shall validate their e-mail addresses to complete the registration process.
    - 1.1.2. Login/Logout
        - 1.1.2.1. Users shall be able to log in via their registered username/e-mail address and password.
        - 1.1.2.2. Users shall be able to log out after logging in.
    - 1.1.3. Account Management
        - 1.1.3.1. Users shall be able to reset their passwords by using the authentication link in the verified email.
        - 1.1.3.2. Users should be able to delete their accounts.
    - 1.1.4. Profile
        - 1.1.4.1. Users shall have their usernames and default grey picture on their profile pages.
        - 1.1.4.2. Users shall be able to add profile pictures to their profile pages.
        - 1.1.4.3. Users shall be able to add/edit their personal information to their profile pages. Personal information shall include name, bio, birthday, location.
        - 1.1.4.4. Users shall have public profiles as default.
        - 1.1.4.5. Users shall be able to change the privacy setting (public or private) of their birthdays, profile pictures, location, and bio.
- 1.2. Community
    - 1.2.1 Search & Filter
        - 1.2.1.1. Users shall be able to search for communities using a search bar.
        - 1.2.1.2. Users shall be able to filter communities as implied in 2.4.3.
        - 1.2.1.3. Users shall be able to search for posts using a search bar.
        - 1.2.1.4. Users shall be able to filter posts as implied in 2.4.4.
        - 1.2.1.5. Users shall be able to search for a data type using a search bar.
        - 1.2.1.6. Users shall be able to filter data types as implied in 2.4.4.
    - 1.2.2 Joining
        - 1.2.2.1. Users shall be able to join communities to be able to post content and receive notifications about updates.
        - 1.2.2.2. Users shall be able to turn on or off notifications.

        - 1.2.2.3. Users shall be able to invite their friends to the community.
    - 1.2.3 Posting Content
        - 1.2.3.1. Users shall be able to share posts on communities.
            - 1.2.3.1.1. Users shall be able to select a data type for their posts.
            - 1.2.3.1.2. Users shall be able to include images, numbers, date, location and text as data type fields in posts.
        - 1.2.3.2. Users shall be able to add tags to their posts.
        - 1.2.3.3. Users shall be able to create new data types.
            - 1.2.3.3.1. Users shall be able to define multiple attributes for data types. Attributes shall be images, numbers, date, location or text.
        - 1.2.3.4. Users shall be able to see data types that exist in a community as a list.
    - 1.2.4 Liking & Commenting
        - 1.2.4.1. Users shall be able to comment on posts.
        - 1.2.4.2. Users shall be able to filter comments as implied in 2.4.5.
        - 1.2.4.3. Users shall be able to reply to comments on posts.
        - 1.2.4.4. Users shall be able to like posts on communities.
        - 1.2.4.5. Users shall be able to like comments on posts.
    - 1.2.5 Reporting
        - 1.2.5.1. Users shall be able to report offensive & inappropriate posts.
        - 1.2.5.2. Users shall be able to report offensive & inappropriate comments.
    - 1.2.6 Creating
        - 1.2.6.1. Users shall be able to create a community with a unique community name.
        - 1.2.6.2. Users shall be able to add a picture for the community.
        - 1.2.6.3. Users shall be able to add a description to the community.
        - 1.2.6.4. Users shall be able to set the privacy of the community as public or private.
-

  1.3. User Interaction
    - 1.3.1. Search & Filter
        - 1.3.1.1. Users shall be able to search for other users' profiles as 2.4.1 implies.

        - 1.3.1.2. Users shall be able to sort/filter their search results on users' profiles as 2.4.2 implies.
        - 1.3.1.3. Users shall be able to choose to view other users' profiles which are listed as search results.
    - 1.3.2. Profile View & Follow
        - 1.3.2.1. Users shall be able to view other users' private profiles which they follow and other users' public profiles.
        - 1.3.2.2. Users shall be able to choose to follow other users' public profiles.
            - 1.3.2.2.1 Users who have public profiles should get notifications when they are followed by other users.
        - 1.3.2.3. Users shall be able to send a request to other users who have private profiles, to follow their profiles.
            - 1.3.2.3.1 Users who have private profiles should get notifications when they are requested to get followed by other users.
        - 1.3.2.4. Users shall be able to unfollow users which they follow already.
    - 1.3.3.Private Messaging
        - 1.3.3.1 Users shall be able to message other users.
            - 1.3.3.1.1 Users should get notifications when they got messages from other users as 2.2.1 implies.
    - 1.3.4. Reporting
        - 1.3.4.1 Users shall be able to report other users who are behaving inappropriately.
        - 1.3.4.2 Users shall be able to report community moderators.
-

1.4. User Roles

    - 1.4.1. Common Users
        - 1.4.1.1 Common users shall complete the registration process as 1.1.1 implies.
        - 1.4.1.2 Common users shall be able to view public user-profiles and community content read-only.
        - 1.4.1.3 Common users shall be able to use searching functionality.

        - 1.4.1.4 Common users shall be able to log in/log out as explained in 1.1.2.
        - 1.4.1.5 Common users shall be able to manage their accounts as 1.1.3 implies.
        - 1.4.1.6 Common users shall have a profile page as explained in 1.1.4.
        - 1.4.1.7 Common users shall be able to join public communities directly.
        - 1.4.1.8 Common users shall be able to send a request to join private communities.
    - 1.4.2. Community Members
        - 1.4.2.1. Community members shall be able to have all the privileges of a common user.
        - 1.4.2.2. Community members shall be able to post content as explained in the 1.2.3 section except 1.2.3.3.
    - 1.4.3. Community Moderators
        - 1.4.3.1. Community moderators shall be able to have all privileges of community members.
        - 1.4.3.2. Community moderators shall be able to create data types as explained in 1.2.3.3.
        - 1.4.3.3. Community moderators shall be able to review reports.
        - 1.4.3.4. Community moderators shall be able to remove posts, comments, users from their communities.
        - 1.4.4.2. Community moderators shall be able to assign a new moderator.
        - 1.4.4.3. Community moderators shall be able to change the community between public & private.
        - 1.4.4.4. Community moderators shall be able to close their communities.
    - 1.4.4 Admin Users
        - 1.4.5.1. Admin shall be able to have all privileges of community moderators.
        - 1.4.5.2. Admin shall be able to delete communities and also delete users

- **2. System Requirements**
- 2.1. Recommendation
    - 2.1.1. The system shall recommend communities and posts to users based on the communities they are member of and the posts they like.

    - 2.1.2. The system shall provide trending posts and communities to users.
    - 2.1.3. The system shall recommend the posts of followed users.
    - 2.1.4. The system shall recommend the communities of followed users.
- 2.2. Notification
    - 2.2.1. The system shall notify users when they get private messages.
    - 2.2.2. The system shall notify users when their posts get a like/dislike.
    - 2.2.3. The system shall notify users when someone comments on their posts.
    - 2.2.4. The system shall notify community moderators when a user requests to join their private communities.
    - 2.2.5. The system shall notify users when someone posts in a community they are a member of.
- 2.3. Community
    - 2.3.1. The system shall disable users if they try to perform unauthorized actions.
    - 2.3.2. The system shall allow community moderators to create new data types.
    - 2.3.3. The system shall not allow duplicate community names.
    - 2.3.4. Users' accounts shall be deleted if they behave offensively.
- 2.4. Searching / Filtering / Sorting
    - 2.4.1. The system shall have a semantic search mechanism.
    - 2.4.2. The system shall allow users to sort other users by a number of followers and filter by their communities.
    - 2.4.3. The system shall allow users to sort communities by number of members, number of posts, and filter by tags.
    - 2.4.4. The system shall allow users to sort posts by the number of likes, dates, and filter by data types and tags.
    - 2.4.5. The system shall allow users to sort comments by a number of likes and dates.
    - 2.4.6. The system shall allow users to filter results as communities, users, and posts.
    - 2.4.7. The semantic taggings should be supported with Wikidata.

- 2.5. Security
    - 2.5.1. In the case of a password change attempt, the system should send a verification code to the e-mail address of the account owner.

Non-Functional Requirements

- 3.1. Security
    - 3.1.1. Usernames and emails for registrations shall be valid and unique for each user.
    - 3.1.2. Passwords shall be 8-16 characters long and must not contain special characters.
    - 3.1.3. Passwords shall be encrypted with SHA-256 before being stored in a database.
- 3.2. Privacy
    - 3.2.1. System shall obey the rules defined by GPDR and KVKK.
    - 3.2.2. System should provide the mechanisms to preserve the uniqueness of Copyright contents.
    - 3.2.3. Users must agree to the Privacy Policy and User Agreement before registration.
- 3.3. Performance and Reliability
    - 3.3.1. Respond times shall not exceed 3 seconds.
    - 3.3.2. Downtime occurrence shall not be more than once in a month.
    - 3.3.3. The platform shall support up to 10000 simultaneous users. This number should be scalable when necessary.
- 3.4. Accessibility and Availability
    - 3.4.1. System shall be accessed via Android based mobile application and web application compatible with modern browsers.
    - 3.4.2. Platform should support the English language.
    - 3.4.3. Platform should support all ASCII symbols and Turkish symbols.

- 3.5. Standards
    - 3.5.1. The implementation should obey the guidelines of W3C Activity Stream Standard 2.0.

UML Software Designs:

Class Diagram [GitHub](#) • [LucidChart](#)

Use Case Diagram

[GitHub ● LucidChart]

Sequence Diagrams

GitHub

1.Login

2.Registration

3.Change Notifications

5. Like Post

6. Message Channel

7. Create Data Type

8. Search Community

9. Report Comment

10. Post Content

11. Edit Profile Page

Scenarios and Mockups

Scenario & Mockup #1: CREATING COMMUNITY

Persona

- **Satoshi Yılmaz**
- **25 years old**
- **Computer Engineer**
- **Interested in Cryptocurrencies**
- **Sushi lover**
- **User**

Story

- **Satoshi is born to a Turkish father and Japanese mother and living in Turkey.**
- **He is working as a software developer and looking to buy a new computer.**
- **Seeing that GPU prices are very high, he thinks that he needs to find another way to pay for his GPU.**
- **Being a long-time lurker of cryptocurrencies, he finally decides that it is time to invest.**
- **While he is searching for a coin to invest in, he sees his favorite food as a coin, SUSHI.**
- **However, he does not know if this is a smart investment option so he decides to join a community for some opinion.**
- **After searching a community for SUSHI, he realizes that there is not any yet.**
- **At this point, he re-thinks it if SUSHI is the right investment. Nevertheless, he decides to create the community himself.**

Preconditions

- **Satoshi is a registered user of the application.**
- **Satoshi logged in to the application.**

Goals

- **Satoshi wants to connect to other people who are interested in the SUSHI coin.**
- **Satoshi wants to create a community where he can discuss the future of the SUSHI coin and hear the news about it.**

Acceptance Criteria

- **(1.2.1.1.) Users shall be able to search for communities using a search bar.**
- **(1.2.6.1.) Users shall be able to create a community with a unique community name.**
- **(1.2.6.2.) Users shall be able to add a picture for the community.**
- **(1.2.6.3.) Users shall be able to add a description to the community.**
- **(1.2.6.4.) Users shall be able to set the privacy of the community as public or private.**
- **(1.2.2.3.) Users shall be able to invite their friends to the community.**

Scenario

1. **Satoshi searched for the SUSHI coin community using the search bar.**
2. **In the search results, Satoshi could not find any community related to the SUSHI coin.**
3. **He decides to create the community himself to discuss the future of the SUSHI coin and hear the news about it.**
4. **He presses the "Create a new community" button.**
5. **He enters necessary information such as community name, description.**
6. **He selects the community as public from the community types section.**
7. **He selects a picture for the community.**
8. **Finally, he pressed the "Create" button to create the community.**
9. **On the community home page, he presses the "Invite" button to get a link for inviting his friends.**

Mockup

1**-Satoshi searched for the SUSHI coin community using the search bar. In the search results, Satoshi could not find any community related to the SUSHI coin.**[2]

[2] Recommended communities and users tabs on mockups 1 to 4 were removed from the project requirements, please ignore them.

2-He decides to create the community himself. He presses the "Create community" button.

3-He enters necessary information such as community name, description. He selects the community as public from the community types section.

4-He selects a picture for the community. Finally, he pressed the "Create Community" button to create the community.

5-On the community home page, he presses the "Invite" button to get a link for inviting his friends.

Scenario & Mockup #2: POSTING CONTENT

Persona

- **Lenny Wosniak**
- **55 years old**
- **A fellow high school teen**
- **Interested in skateboarding**

Story

- **Lenny recently got into skateboarding and he loves it.**

- **He wants to interact with other people who are also interested in skateboarding.**
- **He looks for a skateboarding community in our website.**
- **He finds a popular skateboarding community and joins it.**
- **He has a skateboarding event idea that he wants to share with others.**
- **He creates a skateboarding event post based on a already created data type and shares it in the community.**

Preconditions

- **Lenny is already a registered user of our website.**
- **Lenny is already signed in to the website.**

Goals

- **Lenny wants to join a community in which he can interact with other skateboarders.**
- **Lenny wants to create and share a skateboarding event.**

Acceptance Criteria

- **(1.2.1.1.) Users shall be able to search for communities using a search bar.**
- **(1.2.2.1.) Users shall be able to join communities to be able to post content and receive notifications about updates.**
- **(1.2.3.1.) Users shall be able to share posts on communities.**
- **(1.2.3.1.1.) Users shall be able to select a data type for their posts.**

Scenario & Mockup

- **Lenny opens up the website and types "skateboarding" in the search bar.**
- **In the search results, he sees a popular skateboarding community and clicks it.**
- **In the community page, he clicks the "Join" button and joins the community.**
- **After joining, he clicks "Create New Post" button to create and share his event idea.**
- **In the post creation screen, he selects the "Event" data type.**
- **He fills out necessary information such as event name, time, location, description.**
- **He clicks the "Share" button and completes creating his event post.**

Search Community

Join Community

Create Post

Scenario & Mockup #3: CREATING POST TYPE

Persona

- **Berkecan Bilgin**
- **22 years old**
- **Student at Boğaziçi University Sociology department**
- **Social leader**
- **Gemini**
- **Animal lover**

Story

- **Berkecan is a very interactive student in university. He participates in many events both social and academical.**
- **He is also a member of Boğaziçi University Community in our application. He thinks that the community is very helpful as it brings a continous data flow about various topic considering the university.**
- **However, he also thinks that this community could be better if the posts were more structured and classified.**
- **He decides that it is time to be a moderator in community and make some serious contribution.**
- **At the end of the elections, he is proudly selected as one of the moderators.**
- **What would be the first thing to do? Of course, creating new data types for frequently mentioned topics.**
- **The first example of those topics is pet adoption. He sees lots of people are trying to give or adopt pets temporally or permanently.**
- **He decides to create a data type for adopting animals.**

Preconditions

- **Berkecan is a registered user of the application.**
- **Berkecan logged in to the application.**
- **Berkecan is a member of the corresponding community.**
- **Berkecan is a moderator in the community.**

Goals

- **Berkecan wants the community to keep more structured data.**
- **Berkecan wants to fit pet adoption posts into same format.**
- **Berkecan wants pet adoption posts to be found easily.**

Acceptance Criteria

- **(1.2.3.1.1.) Users shall be able to select a data type for their posts.**
- **(1.2.3.3.) Users shall be able to create new data types.**
- **(1.2.3.3.1.) Users shall be able to define multiple attributes for data types.**
- **(1.2.3.4.) Users shall be able to see data types that exist in a community as a list.**

Scenario

1. **Berkecan opens the page of Boğaziçi Community.**
2. **He encounters many random pet adoption posts.**
3. **He looks at the data types of the community and sees that there is no pet adoption category.**
4. **He clicks to "Create data type" button.**
5. **He enters the name of new data type.**
6. **He enters attribute names for the data type which are type, age, and location of the pet and adoption status (permanent/temporary/etc.) for example.**
7. **He also selects number of images in pet adoption posts.**
8. **Then he clicks to "Create" button.**

9. **After this, users are going to be able to post pet adoption posts with the structure determined by Berkecan.**

Mockup

1-He sees the posts on Boğaziçi Community.

2-He looks at the data types of the community.

3-He creates "Pet Adoption" data type.

4-Now, all members are able to post pet adoption posts with the structure created by Berkecan.

Scenario & Mockup #4: SEARCHING AND FILTERING

Persona

- **Enes Durak**
- **13 years old**
- **Student at Atatürk Secondary School**
- **Chess prodigy**

Story

- **Enes is a 6th grade student at Atatürk Secondary School.**
- **He is a member of Chess Student Club of his school.**
- **Enes is ambitious and likes challenges.**
- **He is bored of winning all matches in the school club.**
- **He wants to attend to global tournaments and win a championship.**
- **His role model is Fabiano Caruana.**
- **He decides to find a tournament from our platform by looking posts of his role model.**

Preconditions

- **Enes is a registered user of our platform.**
- **Enes has already logged in to the platform from the mobile application.**

Goals

- **Enes wants to attend an online chess tournament.**
- **Enes wants to find a tournament event posted by a chess grandmaster from his mobile phone.**

Acceptance Criteria

- **(1.4.1.2.) Common users shall be able to view public user-profiles and community content read-only.**
- **(1.4.1.3.) Common users shall be able to use searching functionality.**
- **(2.4.4.) The system shall allow users to sort posts by the number of likes, dates, and filter by data types and tags.**

- **(2.4.6.) The system shall allow users to filter results as communities, users, and posts.**

Scenario

1. **Enes opens the home page from the mobile application.**
2. **Enes types "Fabiano Caruana" to search bar.**
3. **He sees the related users, public posts and communities as result.**
4. **Enes filters posts by "tournament" data type.**
5. **Enes sorts filtered posts by date.**
6. **Enes sees the filtered search results.**
7. **He clicks on a tournament and sees its content and comments.**

Mockup

1-Enes opens the home page from the mobile application.

2-Enes types "Fabiano Caruana" to search bar.

3-He sees the related users, public posts and communities as result.

4-Enes filters posts by "tournament" data type. Enes sorts filtered posts by date.

5-Enes sees the filtered search results.

6-He clicks on a tournament and sees its content and comments.

Scenario & Mockup #5: PROFILE UPDATE

Persona

- **Afet Güçverir**
- **49 years old**
- **History Teacher**
- **Idealist**

Story

- **Afet is a history teacher in Rıdvan Kanat High School.**
- **She loves to teach and inspire her students.**
- **Due to the pandemic, she gives her classes online. She notices that some students need extra support during this period.**
- **She decides to give online private lessons and help students overcome their problems.**
- **She recently joined a community to post about her private lessons.**
- **She realizes that her profile is incomplete and needs to be updated.**

Preconditions

- **Afet is a recently registered user.**
- **She has already logged in to the platform from the mobile application.**

Goals

- **Afet wants to update her profile and add information such as date of birth, profile photo etc..**
- **Afet wants to make sure that everyone can see her profile.**

Acceptance Criteria

- **1.1.4.1. Users shall have a profile page.**
- **1.1.4.2. Users shall be able to add profile pictures to their profile pages.**
- **1.1.4.3. Users shall be able to add their personal information to their profiles pages.**
- **1.1.4.4. Users shall be able to edit the content of their profile pages.**
- **1.1.4.5. Users shall be able to set their profiles as public or private to other users.**

Scenario

1. **Afet opens her profile page from the mobile application.**
2. **She clicks on edit profile.**
3. **She clicks on "Add a Photo" and adds her profile photo in PNG format.**
4. **She enters her date of birth and personal information including her teaching experience.**
5. **She checks her privacy settings. She notices that most of her profile is private and users do not follow her cannot see her information. She makes all options public and clicks on confirm.**
6. **She clicks on save changes and sees that she has successfully updated her profile.**

Mockup

1-Afet opens her profile page from the mobile application.

2-She clicks on edit profile.

3-She clicks on "Add a Photo" and adds her profile photo in PNG format.

4-She enters her date of birth and personal information including her teaching experience.

5-She checks her privacy settings. She notices that most of her profile is private and users do not follow her cannot see her information. She makes all options public and clicks on confirm.

6-She clicks on save changes and sees that she has successfully updated her profile.

Communication Plan

Github

Tool	Attandence	Purpose	When	Where
Slack	Team Members	In order to discuss the project details outside face-to-face meeting time	Whenever needed	Mobile App
Discord	Team Members	Evaluation of the process, distributing tasks, making decisions weekly on new plans as sub-teams	Every Thursday at 9.30 pm	Web
Zoom	Team Members	Evaluation of the process, distributing tasks, making decisions weekly on new plans as teams	Every Tuesday at lab hours	Web
Whatsapp	Team Members	Not an official channel, just for daily communication	All-time	Mobile App
Slack	Subteam Members	To discuss sub-tasks	Whenever needed	Mobile App
GitHub	Team Members	To track ıssues/tasks and submit materials	Whenever needed	Web

Project plan

Github

		Ad	Süre	Başlat	Bitirme	Ö...	Kaynak Adları
1		**Meetings and Communication**	**24 günler?**	**25.03.2021 08:00**	**27.04.2021 17:00**		
2		Poll for the meeting dates	0,375 günl...	25.03.2021 14:00	25.03.2021 17:00		Sevde Sarıkaya
3		Creating recurring Google Meet session	1 gün?	26.03.2021 08:00	26.03.2021 17:00		İrem Zeynep Alagöz
4		Documenting meeting Notes #1	1 gün?	25.03.2021 08:00	25.03.2021 17:00		Kutay Saran
5		Documenting meeting Note #2	1 gün?	03.04.2021 08:00	05.04.2021 17:00		Kıymet Akdemir
6		Documenting meeting Notes #3	1 gün?	08.04.2021 08:00	08.04.2021 17:00		Burak Onur Duman
7		Documenting meeting Notes #4	1 gün?	17.04.2021 08:00	19.04.2021 17:00		Halil Baydar
8		Documenting meeting Notes #5	1 gün?	21.04.2021 08:00	21.04.2021 17:00		Abdurrahman Emre Yılmaz
9		Documenting meeting Notes #6	1 gün?	27.04.2021 08:00	27.04.2021 17:00		Sevde Sarıkaya
10		Initializing the Slack Workspace	1 gün?	25.03.2021 08:00	25.03.2021 17:00		Ufuk Arslan
11		Creating Slack channels for decided tasks	1 gün?	03.04.2021 08:00	05.04.2021 17:00		Ufuk Arslan
12		**Project Repository & Issues**	**13 günler?**	**25.03.2021 08:00**	**12.04.2021 17:00**		
13		Deciding on the issue labels and creating them	5 günler?	29.03.2021 08:00	02.04.2021 17:00		Ufuk Arslan;İrem Zeynep Alagöz
14		Creating a template for future issues	6 günler?	05.04.2021 08:00	12.04.2021 17:00		Kutay Saran
15		Initializing Wiki home page	2 günler?	26.03.2021 08:00	29.03.2021 17:00		Sevde Sarıkaya
16		Initializing README	2 günler?	26.03.2021 08:00	29.03.2021 17:00		Abdurrahman Emre Yılmaz
17		Updating Wiki home page and sidebar	6 günler?	02.04.2021 08:00	09.04.2021 17:00		Orhun Görkem;İrem Zeynep Alagöz
18		Updating README	3 günler?	02.04.2021 08:00	06.04.2021 17:00		Orhun Görkem;İrem Zeynep Alagöz
19		Domain Analysis	3 günler?	02.04.2021 08:00	06.04.2021 17:00		Bekir Keldal
20		Exploring popular GitHub repositories	6 günler?	26.03.2021 08:00	02.04.2021 17:00		Everyone
21		Git Research	3 günler	25.03.2021 08:00	29.03.2021 17:00		Kıymet Akdemir;Orhun Görkem
22		Communication plan	2 günler?	26.03.2021 08:00	29.03.2021 17:00		Halil Baydar
23		Filling Out Personal Information Wiki Page	2 günler?	26.03.2021 08:00	29.03.2021 17:00		Everyone
24		Creating template for member wiki pages	4 günler?	02.04.2021 08:00	07.04.2021 17:00		Bekir Keldal
25		Creating an effort tracking table template	4 günler?	02.04.2021 08:00	07.04.2021 17:00		Bekir Keldal
26		Creating meeting notes template	2 günler?	25.03.2021 08:00	26.03.2021 17:00		Bekir Keldal
27		**Requirements**	**19 günler?**	**02.04.2021 08:00**	**28.04.2021 17:00**		
28		Identifying and documenting user requirements	16 günler?	02.04.2021 08:00	23.04.2021 17:00		Abdurrahman Emre Yılmaz;Burak Onur Duman;Halil Bay...
29		Identifying and documenting system requirements	16 günler?	02.04.2021 08:00	23.04.2021 17:00		Bekir Keldal;Kıymet Akdemir;Sevde Sarıkaya;Ufuk Arslan
30		Identifying and documenting non-functional requirements	16 günler?	02.04.2021 08:00	23.04.2021 17:00		Kutay Saran;Orhun Görkem
31		Adding glossary to requirements page	19 günler?	02.04.2021 08:00	28.04.2021 17:00		Orhun Görkem;İrem Zeynep Alagöz
32		**Scenarios & Mockups**	**4 günler?**	**14.04.2021 08:00**	**19.04.2021 17:00**	**27**	
33		Scenario: Creating Community	4,125 günler	14.04.2021 08:00	15.04.2021 17:00		İrem Zeynep Alagöz
34		Scenario: Posting Content	2 günler?	14.04.2021 08:00	15.04.2021 17:00		Burak Onur Duman
35		Scenario: Creating Data Type	2 günler?	14.04.2021 08:00	15.04.2021 17:00		Orhun Görkem
36		Scenario: Searching & Filtering	2 günler?	14.04.2021 08:00	15.04.2021 17:00		Kıymet Akdemir
37		Scenario: Updating Profile	2 günler?	14.04.2021 08:00	15.04.2021 17:00		Bekir Keldal
38		Mockup: Creating Community	2 günler?	16.04.2021 08:00	19.04.2021 17:00	33	Ufuk Arslan
39		Mockup: Posting Content	2 günler?	16.04.2021 08:00	19.04.2021 17:00	34	Halil Baydar
40		Mockup: Creating data type	2 günler?	16.04.2021 08:00	19.04.2021 17:00	35	Sevde Sarıkaya
41		Mockup: Searching & Filtering	2 günler?	16.04.2021 08:00	19.04.2021 17:00	36	Abdurrahman Emre Yılmaz
42		Mockup: Updating Profile	2 günler?	16.04.2021 08:00	19.04.2021 17:00	37	Kutay Saran
43		**Design Diagrams**	**8 günler?**	**21.04.2021 08:00**	**30.04.2021 17:00**	**27**	
44		Class diagram	8 günler?	21.04.2021 08:00	30.04.2021 17:00		Abdurrahman Emre Yılmaz;Burak Onur Duman;Kıymet A...
45		Use case diagram	8 günler?	21.04.2021 08:00	30.04.2021 17:00		Bekir Keldal;Halil Baydar;Kutay Saran;Orhun Görkem;Se...
46		Sequence diagrams	5 günler?	26.04.2021 08:00	30.04.2021 17:00		Everyone

47		**Milestone Report 1**	**4 günler**	**05.05.2021 08:00**	**10.05.2021 17:00**	**1...**	
48		Evaluation of tools and processes	4 günler	05.05.2021 08:00	10.05.2021 17:00		Burak Onur Duman
49		Status of Deliverables	4 günler	05.05.2021 08:00	10.05.2021 17:00		Kıymet Akdemir
50		Project Description	4 günler	05.05.2021 08:00	10.05.2021 17:00		Abdurrahman Emre Yılmaz
51		Moving forward	4 günler	05.05.2021 08:00	10.05.2021 17:00		Ufuk Arslan
52		Responsibility assignment matrix (RAM)	4 günler	05.05.2021 08:00	10.05.2021 17:00		Bekir Keldal;Halil Baydar;İrem Zeynep Alagöz
53		Documenting project plan for past work	4 günler	05.05.2021 08:00	10.05.2021 17:00		Orhun Görkem;Sevde Sarıkaya;İrem Zeynep Alagöz
54		Documenting project plan for future work	4 günler	05.05.2021 08:00	10.05.2021 17:00		Abdurrahman Emre Yılmaz;Halil Baydar
55		Gathering deliverables	4 günler	05.05.2021 08:00	10.05.2021 17:00		Kutay Saran
56		**Practice App Backend**	**21 günler?**	**13.05.2021 08:00**	**10.06.2021 17:00**		
57		Implementing joining a community API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Burak Onur Duman
58		Implementing creating a community API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		İrem Zeynep Alagöz
59		Implementing getting community by name API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Halil Baydar
60		Implementing following a user API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Abdurrahman Emre Yılmaz
61		Implementing get communities by name API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Bekir Keldal
62		Implementing updating profile API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Sevde Sarıkaya
63		Implementing getting a user API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Orhun Görkem
64		Implementing getting posts by a community API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Kıymet Akdemir
65		Implementing updating a post API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Ufuk Arslan
66		Implement getCommunitiesByPublicity API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		İrem Zeynep Alagöz
67		Implementing creating a post API	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Orhun Görkem
68		Implementing model and repository for User	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Halil Baydar
69		Implementing model and repository for Community	21 günler?	13.05.2021 08:00	10.06.2021 17:00		İrem Zeynep Alagöz
70		Implement profile model	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Sevde Sarıkaya
71		Implementing model and repository for Post	21 günler?	13.05.2021 08:00	10.06.2021 17:00		Orhun Görkem
72		**Practice App Configuration**	**6 günler?**	**03.06.2021 08:00**	**10.06.2021 17:00**		
73		Implement Swagger Configuration	6 günler?	03.06.2021 08:00	10.06.2021 17:00		Halil Baydar
74		Creating Docker file and image	6 günler?	03.06.2021 08:00	10.06.2021 17:00		Halil Baydar;Sevde Sarıkaya
75		Storing third-party API keys	6 günler?	03.06.2021 08:00	10.06.2021 17:00		Orhun Görkem;İrem Zeynep Alagöz
76		AWS Deployment	6 günler?	03.06.2021 08:00	10.06.2021 17:00		Everyone
77		**Practice App Unit Tests**	**8 günler**	**01.06.2021 08:00**	**10.06.2021 17:00**		
78		Implement Test: CreateCommunityController	8 günler	01.06.2021 08:00	10.06.2021 17:00		İrem Zeynep Alagöz
79		Implement Test: PostService	8 günler	01.06.2021 08:00	10.06.2021 17:00		Orhun Görkem
80		Implement Test: CommunityService	8 günler	01.06.2021 08:00	10.06.2021 17:00		İrem Zeynep Alagöz
81		Implement Test: CreatePostController	8 günler	01.06.2021 08:00	10.06.2021 17:00		Orhun Görkem
82		Implement Test: UpdatePostController	8 günler	01.06.2021 08:00	10.06.2021 17:00		Ufuk Arslan
83		Implement Test: GetPostByCommnunityController	8 günler	01.06.2021 08:00	10.06.2021 17:00		Kıymet Akdemir
84		Implement Test: GetCommnunityController	8 günler	01.06.2021 08:00	10.06.2021 17:00		Bekir Keldal
85		Implement Test: GetCommunitiesByPublicityController	8 günler	01.06.2021 08:00	10.06.2021 17:00		İrem Zeynep Alagöz
86		Implement Test: UserService	8 günler	01.06.2021 08:00	10.06.2021 17:00		Halil Baydar
87		Implement Test: UserController	8 günler	01.06.2021 08:00	10.06.2021 17:00		Halil Baydar
88		Implement Test: GetProfile	8 günler	01.06.2021 08:00	10.06.2021 17:00		Sevde Sarıkaya
89		Implement Test: UpdateProfile	8 günler	01.06.2021 08:00	10.06.2021 17:00		Sevde Sarıkaya
90		**Practice App Frontend**	**10 günler**	**28.05.2021 08:00**	**10.06.2021 17:00**		
91		Implementation: Frontend for Profile & Create user	10 günler	28.05.2021 08:00	10.06.2021 17:00		Burak Onur Duman;Kıymet Akdemir
92		Implementation: Frontend for Creating & Getting Community	10 günler	28.05.2021 08:00	10.06.2021 17:00		Orhun Görkem;İrem Zeynep Alagöz
93		Implementation: Frontend for Creating & Updating Posts	10 günler	28.05.2021 08:00	10.06.2021 17:00		Abdurrahman Emre Yılmaz;Ufuk Arslan
94		Implementation: Frontend for Third Party APIs	10 günler	28.05.2021 08:00	10.06.2021 17:00		Bekir Keldal;Halil Baydar
95		Implementation: Frontend for Getting Posts & Joining Commu...	10 günler	28.05.2021 08:00	10.06.2021 17:00		Kutay Saran;Sevde Sarıkaya

96		**⊟Milestone Report 2**	**6 günler?**	**07.06.2021 08:00**	**14.06.2021 17:00**	
97		Creating the Table of Contents	6 günler?	07.06.2021 08:00	14.06.2021 17:00	
98		Status of Deliverables	6 günler?	07.06.2021 08:00	14.06.2021 17:00	İrem Zeynep Alagöz
99		Overall Status	6 günler?	07.06.2021 08:00	14.06.2021 17:00	Kıymet Akdemir
100		Challenges	6 günler?	07.06.2021 08:00	14.06.2021 17:00	Bekir Keldal
101		Introduction	6 günler?	07.06.2021 08:00	14.06.2021 17:00	Bekir Keldal
102		Look ahead	6 günler?	07.06.2021 08:00	14.06.2021 17:00	Ufuk Arslan
103		Responsibility assignment matrix (RAM)	3 günler?	07.06.2021 08:00	09.06.2021 17:00	Kıymet Akdemir;İrem Zeynep Alagöz
104		Documenting project plan for the practice-app	3 günler?	07.06.2021 08:00	09.06.2021 17:00	Kıymet Akdemir;İrem Zeynep Alagöz
105		Basic functionality of the practice-app	6 günler?	07.06.2021 08:00	14.06.2021 17:00	Ufuk Arslan
106		**⊟BACKEND**	**68 günler?**	**01.10.2021 08:00**	**04.01.2022 17:00**	**Kaan Dura;Sevde Sarıkaya**
107		Service Classes	3,5 günler?	01.11.2021 08:00	04.11.2021 13:00	Kaan Dura;Sevde Sarıkaya
108		Controller classes	2,5 günler?	01.11.2021 08:00	03.11.2021 13:00	Kaan Dura;Sevde Sarıkaya
109		Abstract Service classes	4 günler?	12.11.2021 08:00	17.11.2021 17:00	Kaan Dura;Sevde Sarıkaya
110		Abstract Controller classes	4 günler?	12.11.2021 08:00	17.11.2021 17:00	Kaan Dura;Sevde Sarıkaya
111		Resource classes	5 günler?	28.11.2021 08:00	03.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
112		Security configuration	8 günler?	03.12.2021 08:00	14.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
113		Aws configuration	1,5 günler?	12.12.2021 08:00	14.12.2021 13:00	Kaan Dura;Sevde Sarıkaya
114		Data transfer classes	3,25 günler?	01.12.2021 08:00	06.12.2021 10:00	Kaan Dura;Sevde Sarıkaya
115		Entity Classes	6,8 günler?	01.10.2021 08:00	11.10.2021 15:24	Kaan Dura;Sevde Sarıkaya
116		Database configuration	1,5 günler?	27.11.2021 08:00	30.11.2021 13:00	Kaan Dura;Sevde Sarıkaya
117		Mapper interfaces	2,5 günler?	01.12.2021 08:00	03.12.2021 13:00	Kaan Dura;Sevde Sarıkaya
118		Exception handling	10 günler?	01.12.2021 08:00	14.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
119		Socket implementation	0 günler?	01.11.2021 08:00	01.11.2021 08:00	Kaan Dura;Sevde Sarıkaya
120		Notification implementation	4 günler?	18.12.2021 08:00	23.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
121		Search implementation	5,5 günler?	10.12.2021 08:00	17.12.2021 13:00	Kaan Dura;Sevde Sarıkaya
122		Repository interfaces	5,5 günler?	01.12.2021 08:00	08.12.2021 13:00	Kaan Dura;Sevde Sarıkaya
123		Util classes	9,5 günler?	20.12.2021 08:00	31.12.2021 13:00	Kaan Dura;Sevde Sarıkaya
124		Swagger configuration	2 günler?	01.11.2021 08:00	02.11.2021 17:00	Kaan Dura;Sevde Sarıkaya
125		Email configuration	3 günler?	01.12.2021 08:00	03.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
126		Authentication configuration	5 günler?	26.12.2021 08:00	31.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
127		Authorization configuration	1 gün?	26.12.2021 08:00	27.12.2021 17:00	Kaan Dura;Sevde Sarıkaya
128		**⊟Mobile Development / Frontend**	**68 günler?**	**01.10.2021 08:00**	**04.01.2022 17:00**	**Burak Onur Duman;Halil Baydar;Kıymet Akdemir;Ufu...**
129		**⊟Application – Websites**	**66 günler?**	**01.10.2021 08:00**	**31.12.2021 17:00**	**Burak Onur Duman;Halil Baydar;Kıymet Akdemir;Ufu...**
130		Login page	6 günler?	01.10.2021 08:00	08.10.2021 17:00	Ufuk Arslan;İrem Zeynep Alagöz
131		Registration page	6 günler?	12.10.2021 08:00	19.10.2021 17:00	Halil Baydar;Ufuk Karagöz
132		Confirmation page	6 günler?	14.10.2021 08:00	21.10.2021 17:00	Burak Onur Duman;Kıymet Akdemir
133		Home page	4,5 günler?	27.10.2021 08:00	02.11.2021 13:00	Burak Onur Duman;Kıymet Akdemir
134		**⊟Community page**	**8 günler?**	**01.10.2021 08:00**	**12.10.2021 17:00**	**Ufuk Arslan;İrem Zeynep Alagöz**
135		Community posts page	4,5 günler?	01.10.2021 08:00	07.10.2021 13:00	Ufuk Arslan;İrem Zeynep Alagöz
136		About community page	4 günler?	07.10.2021 08:00	12.10.2021 17:00	Ufuk Arslan;İrem Zeynep Alagöz
137		**⊟Search page**	**7,5 günler?**	**03.11.2021 08:00**	**12.11.2021 13:00**	**Burak Onur Duman;Kıymet Akdemir**
138		General results page	4,5 günler?	03.11.2021 08:00	09.11.2021 13:00	Burak Onur Duman;Kıymet Akdemir
139		User results page	7,5 günler?	03.11.2021 08:00	12.11.2021 13:00	Burak Onur Duman;Kıymet Akdemir
140		Post results page	7,5 günler?	03.11.2021 08:00	12.11.2021 13:00	Burak Onur Duman;Kıymet Akdemir
141		Community results page	7,5 günler?	03.11.2021 08:00	12.11.2021 13:00	Burak Onur Duman;Kıymet Akdemir
142		**⊟Left side navigation menu**	**12 günler?**	**10.11.2021 08:00**	**25.11.2021 17:00**	**Burak Onur Duman;Kıymet Akdemir;Ufuk Arslan;İre...**
143		Profile page	4 günler?	20.11.2021 08:00	25.11.2021 17:00	Burak Onur Duman;Kıymet Akdemir
144		Create community page	5 günler?	15.11.2021 08:00	19.11.2021 17:00	Ufuk Arslan;İrem Zeynep Alagöz
145		Settings pages	4,5 günler?	10.11.2021 08:00	16.11.2021 13:00	Ufuk Arslan;İrem Zeynep Alagöz
146		Chats page	5 günler?	20.11.2021 08:00	26.11.2021 17:00	Halil Baydar;Ufuk Karagöz
147		Notifications page	3 günler?	26.11.2021 08:00	30.11.2021 17:00	Halil Baydar;Ufuk Karagöz
148		Message page	3 günler?	26.11.2021 08:00	30.11.2021 17:00	Halil Baydar;Ufuk Karagöz
149		**⊓Create post page**	**16,5 günler?**	**01.10.2021 08:00**	**25.10.2021 13:00**	**Burak Onur Duman;Kıymet Akdemir;Ufuk Arslan;İre...**
150		Select community page	4,5 günler?	01.10.2021 08:00	07.10.2021 13:00	Ufuk Arslan;İrem Zeynep Alagöz
151		Select data type page	7,5 günler?	14.10.2021 08:00	25.10.2021 13:00	Ufuk Arslan;İrem Zeynep Alagöz
152		Set content page	2 günler?	01.10.2021 08:00	04.10.2021 17:00	Burak Onur Duman;Kıymet Akdemir
153		Add tag page	2 günler?	01.10.2021 08:00	04.10.2021 17:00	Burak Onur Duman;Kıymet Akdemir
154		**⊔Community moderator panel**	**16 günler?**	**01.12.2021 08:00**	**22.12.2021 17:00**	**Burak Onur Duman;Halil Baydar;Kıymet Akdemir;Ufu...**
155		Community description page	4,5 günler?	01.12.2021 08:00	07.12.2021 13:00	Burak Onur Duman;Kıymet Akdemir
156		Banned users	6 günler?	08.12.2021 08:00	15.12.2021 17:00	Burak Onur Duman;Kıymet Akdemir
157		List data types	4 günler?	08.12.2021 08:00	13.12.2021 17:00	Ufuk Arslan;İrem Zeynep Alagöz
158		Create data type	7,5 günler?	01.12.2021 08:00	10.12.2021 13:00	Ufuk Arslan;İrem Zeynep Alagöz
159		Change community image	10,5 günler?	01.12.2021 08:00	15.12.2021 13:00	Halil Baydar;Ufuk Karagöz
160		Delete community	6 günler?	15.12.2021 08:00	22.12.2021 17:00	Halil Baydar;Ufuk Karagöz
161		Community moderators	6 günler?	15.12.2021 08:00	22.12.2021 17:00	Halil Baydar;Ufuk Karagöz
162		Admin panel	4 günler?	22.12.2021 08:00	27.12.2021 17:00	Ufuk Arslan;İrem Zeynep Alagöz
163		Permission configuration	5 günler?	27.12.2021 08:00	31.12.2021 17:00	Ufuk Arslan;İrem Zeynep Alagöz
164		Delivery of Project	4 günler?	30.12.2021 08:00	04.01.2022 17:00	Everyone

RAM

Github

L: Lead C: Contributor R: Reviewer N: None	Kıymet Akdemir	İrem Zeynep Alagöz	Halil Baydar	Ufuk Arslan	Burak Onur Duman	Ufuk Karagöz	Sevde Sarıkaya	Kaan Dura
MOBILE								
Design for Mobile: Login Page	R	L	R	N	N	N	N	N
Design for Mobile: Information Bar Component	R	L	R	N	N	N	N	N
Configuration for Mobile: General Axios Instance	R	L	R	N	N	N	N	N
Mobile: Choose Community Page	N	L	R	N	N	N	N	N
Main and Notifcation pages	N	R	L	N	N	N	N	N
Mobile: Choose Data Type Page	N	L	R	N	N	N	N	N
Mobile: Implementing Creating Data Type Functionality	N	L	R	N	N	N	N	N
Mobile: Updating Requests Based on the Swagger Changes	R	L	N	N	N	N	N	N
Mobile/feature/269 profile screen	R	R	L	N	N	N	N	N
Mobile: Logout	L	R	N	N	N	N	N	N
Mobile: Post Detail	L	R	N	N	N	N	N	N
Mobile: Create Post	R	R	L	N	N	N	N	N
Mobile: Create Community	R	L	R	N	N	N	N	N
[Bug]: Mobile Client is getting Network Error	R	L	R	N	N	N	N	N
Mobile: Using AsyncStorage for token	N	L	R	N	N	N	N	N
Mobile: Design for pages under construction	R	L	N	N	N	N	N	N
FRONTEND								
Design for Web: Create Community	N	N	R	N	N	L	N	N
Design for Web: Login Page	N	R	N	L	N	N	N	N
Design for Web: Confirmation Page	N	N	N	R	L	N	N	N
Design for Web: Post Component	N	N	N	L	R	N	N	N
Design for Web: Profile View and Edit Components	N	N	N	L	N	R	N	N
Design for Web: Navigation Bar Component	N	N	N	R	L	N	N	N
Design for Web: About Community Component	N	N	N	R	L	N	N	N
Design for Web: Create Community Component	N	N	N	R	L	N	N	N
Design for Web: Implement Create Communtity, Get Communities and C	N	N	N	C/R	C	L	N	N
Web: Create Deployment	N	N	N	R	L	N	N	N
BACKEND								
[Bug]: Swagger UI Authorization header problem	N	R	N	N	N	N	N	L
Create details endpoint for post and post-type	N	N	N	N	N	N	L	R
Backend Correction: Returning Posts instead of PostTypes	N	N	N	N	N	N	R	L
Create a Github Action for automatic deployment	N	N	N	N	N	N	R	L
Implement get and post posts endpoints	N	R	N	N	N	N	R	L
Implement get and post post-types endpoints	N	N	N	N	N	N	R	L
Implement get and post communities endpoints	N	N	N	N	N	N	R	L
Create Register, Login, Logout endpoints	N	N	N	N	N	N	R	L
Redesigning Swagger YAML for backend	R	L	N	N	N	N	N	R
Serve Swagger documentation on dev server	N	N	N	N	N	N	R	L
Configure MongoDB database	N	N	N	N	N	N	R	L
Create a MongoDB database on cloud	N	N	N	N	N	N	R	L
Preparing Swagger YAML for backend	C	L	C	C	C	C	R	R

GENERAL								
Creating Personal WikiPage	N	R	N	N	N	L	R	L
Documenting Meeting Notes #10	N	L	N	N	N	N	R	N
Updating Communication Plan	N	L	N	N	N	N	N	N
Reviewing and updating Scenario 1: Creating Community	R	L	N	N	N	N	N	N
Revision of the Class Diagram	L	C	N	N	C	N	R	C
Reviewing and updating scenario 2	N	N	N	R	L	N	N	N
Reviewing and updating sequence diagram: Report Comment	N	N	N	R	L	N	N	N
Reviewing and updating sequence diagram: Edit Profile Page	N	N	N	R	L	N	N	N
Reviewing and updating Scenario 3: Creating Data Type	N	R	N	N	N	L	N	N
Reviewing and updating sequence diagram: Like Post	N	R	N	N	N	L	N	N
Reviewing and updating Mockup 5: Profile Update	R	L	N	N	N	N	N	N
Reviewing and updating sequence diagrams: Login, Registration	R	L	N	N	N	N	N	N
Revision of the requirements	R	L	C	C	L	C	C	C
Reviewing and updating Mockup 1: Creating Community	R	N	N	L	N	N	N	N
Reviewing and updating sequence diagram: Search Community	L	R	N	N	N	N	N	N
Reviewing and updating Scenario 4: Searching and Filtering	L	R	N	N	N	N	N	N
Reviewing and updating Mockup 4: Searching and Filtering	N	L	N	N	R	N	N	N
Updating Wiki Home Page & Sidebar	N	R	N	N	L	N	N	N
Use Case Diagram revise and update	N	R	N	L	N	L	L	N
Creating issue template: backend, frontend and mobile bug reports	N	L	N	R	N	N	N	N
Organizing Meetings Notes	N	L	N	R	N	N	N	N
Creating Project Board for Backend, Frontend and Mobile	R	N	N	N	N	N	N	N
Documenting Frontend Meeting at 11.11.2021	N	N	N	L	R	N	N	N
Project Plan for Mobile Development	L	C	N	R	R	N	R	N
Project Plan for Web Development	R	R	N	L	C	C	R	N
Project Plan for Backend Development	R	R	N	R	R	N	L	C

Swagger Documentation for the current Backend:

https://api.cmpegroupthree.store/docs/

Swagger Json file:
https://github.com/bounswe/2021SpringGroup3/blob/backend/deploy/backend/resource/swagger.json

Mobile APK:

https://drive.google.com/file/d/1FgFV2Vu06iIMN1HM2tJFZSZCGSP_9Whd/view?usp=sharing

Mobile Deliverables

Register

Login

Logout

Create Post

11:03 PM
Choose Post Type
discussion
discussion2
Event
Discussion

11:03 PM
Post to
fas
cmpe
mile
Refreshing...
Main
Search
Create Post
Chat
Notification

11:06 PM
Next
fas
RULES
Topic Gala
Date: 01.12.2021
People Limit 15

Create Community

Create Post Type