CMPE 451 Fall
Final Report

Group 9 - Postory

Mert Alkan

Ahmet Melih Aydoğdu

Zehranaz Canfes

Mustafa Emir Çolak

Ömer Barış Erkek

Melih Özcan

Ahmet İbrahim Şentürk

Niyazi Ülke

Tag

Project Website

Table of Contents

Executive Summary

Introduction

As a group in CmpE 451 course, we aim to design a platform for people who want to share their stories based on locations they have been to and experienced. We are to implement a web and an Android application for our users to share stories with others and view them.

With this final milestone, we have completed our work. The functionalities presented in requirements are implemented and the final customer presentation is done. In the following sections, you can find the related information about different measures of our work.

Project Status and Work Done So Far

In this milestone, the project plan was followed as we did in the previous milestones. The development process was successful in those milestones so it was rather easy to make an addition to this final milestone. The new features added on top of previous implementations and application is updated.
The impact on the previous milestones was positive for this part of the project because the core functionality of the application is implemented and the users were able to start using the program. In this milestone, new features were added to improve the experience.
Searching user functionality added. By using this feature, users are now able to search for other users by their username. They can view their profile page and follow them.
Searching and filtering post functionality added. By using this feature, users are now able to search and filter posts by a user, tag, location, keyword, and time. For tag filtering, there are two options. Users can choose only to search for the tag they entered or they can choose to search for related tags. Related tags are obtained from Wikidata.
Activity Stream functionality added. Users can see the general activity stream on their Activities page.
Admin is added to the application. Users can report stories and the admins of the application will receive notification by email.
In conclusion, there is still room for improvement. However, we accomplished to implement the functionalities in our project plan. Also, the final presentation was successful.

Final Release Notes

We are happy to present the new and final v0.9 release of Postory. This release is the first release that has all the core functionalities.

Features

- Users can sign up and sign in.
- Users can edit and create posts.
- Users can view posts on home page.
- Users can like and comment on posts.
- Users can report posts.
- Users can view profile pages of other users and their own page.
- Users can search, filter and view posts on the discovery page.
- Users can search for users.
- Users can view the general activities in the application on their Activites page.

- Users can follow/unfollow other users and send follow request to private accounts.
- Admins receives notifiction for the posts that are reported.
- Admins can sign in via admin interface.

Final Project Evaluations and Lessons Learned

On our road to the Final Milestone, we continued with the implementations. As we did our planning during the first milestone, we paid attention to the project plan. Each team tried to follow the plan which helped us prepare for the final product. We realized the importance of planning and following a plan in a group project. Therefore, during this milestone, each group was more in harmony.

Moreover, the backend, android, and frontend groups were in contact throughout the semester. We planned each step we took keeping each team in mind. We made our features parallel to each other, so that, android, backend, and frontend worked simultaneously.

In this milestone, we implemented the core functionalities of our project: location and time. The search and filter functionality presents the user a timeline-like structure where each user has access to stories near a location they choose and view the posts in order similar to a memory book. With this feature, we can say that we completed the main goal of the 'location-based stories app'.

During CmpE451 and CmpE 352, we have learned a lot from the course and from each other. It is a known fact that implementing a project and maintaining it is a challenge. We faced several bugs and problems throughout this semester and the most important part while facing the problems is to stay as a group and work together. For instance, we had a problem with merging files, deploying the app, and implementing functionalities. Also, we had a problem with our customer presentation due to external applications we used during the demo. We saw that no matter how hard we work, there may be problems during presentations. The important thing is to inform the customers and each other to minimize the effects.

During each problem, we were in contact and we informed each other. Everyone, although it was not their responsibility, helped everyone when needed. We learned the importance of planning and regular meetings to move according to plan. Furthermore, we learned that documenting our project is as much important as implementing. They complete each other and prepare a foundation for a continuous project environment. No matter how hard it was to move according to plan and get prepared for the milestones, we acted as a group and faced the challenges together.

List and Status of deliverables

This section lists all deliverables for postory-app; with their delivery status, due date, and delivery date.

Deliverable Name	Delivery Status
Deliverables Report	Delivered
Group Report	Delivered
Individual Reports	Delivered (for every member)
Requirements	Delivered
UML Software Designs	Delivered
Scenarios and Mockups	Delivered
Project plan	Delivered
Communication Plan	Delivered
RAM	Delivered

Evaluation of the status of deliverables and its impact on our project plan

This section of the report includes our evaluation and comments about what we have done so far, compares our current status to the project plan at large, and briefly explains how we are going to improve the project.

Status of all requirements

This part of the report aims to evaluate the satisfaction of project requirements as the final milestone has passed. The satisfaction status of project requirements will be classified into 3 classes:

1. Completed
2. Partially completed
3. Not completed

We thought not to include the "partially completed" class but the regarding requirements were not totally clear.

Summary

As Group 9, we have completed the implementation of the Postory Project. Below, the status of our requirements is evaluated. In general, all the core functionalities that are proposed in the requirements are completed.

We decided not to implement the "Guest" type as it was not core functionality. It might even be better if everybody has to sign in before using the application. Notifications are also implemented differently. The user gets informed if she receives a follow request, but it's not in form of a

notification. Admin users can access the administration page provided by Django and take the required actions, however, we did not implement UI for this. In addition, features such as "liking comments" are not implemented because we decided to focus more on "Activity Streams" and "Wikidata Connection" features.

List of addressed requirements

This part contains the project requirements that the team tried to address until the Final Milestone. Explanations to some of the requirements will be numbered in square brackets. At the end of partially Completed requirements list, you can check the reasons why the requirements were not completely satisfied. Below are the lists of "requirements partly completed" and "completed" requirements:

Requirements Partially Completed

- **1.1.2.5.1** Registered users shall be able to view the posts of the registered users, places and tags that they follow.[1]
- **1.1.2.5.4** Registered users shall be able to view the posts from registered users they follow or posts that contain the places/tags they follow on the registered user's main page.[1]
- **1.1.2.10.1** Registered users shall be able to edit their bios, names, surnames.[2]

Explanations

1. Following places or tags is not possible, users can be followed.
2. Users can only change their profile photos at the moment, they can enter the listed information when registering but they cannot edit it.

Completed Requirements

- **1.1.1.1.1** Registered Users shall be able to fill their profile page with their information. This information includes their names, surnames, profile photos.
- **1.1.1.1.2** Registered Users shall be able to view posts.
- **1.1.1.1.3** Registered Users shall be able to sign in by using their username or email, and password.
- **1.1.1.1.4** Registered Users shall be able to sign out.
- **1.1.1.1.5** Registered Users shall be able to search posts.
- **1.1.1.1.6** Registered Users shall be able to filter posts.
- **1.1.1.1.7** Registered Users shall be able to edit their own posts.
- **1.1.1.1.8** Registered Users shall be able to save posts.
- **1.1.1.2.1** Admins shall be able to view posts.
- **1.1.1.2.2** Admins shall be able to revise reports of inappropriate post.
- **1.1.1.2.3** Admins shall be able to ban registered users.
- **1.1.1.2.4** Admins shall be able to remove a post.
- **1.1.1.2.5** Admins shall be able to remove a comment under a post.
- **1.1.1.2.6** Admins shall be able to send mail to registered users.
- **1.1.1.3.5** Guests shall be asked to sign up for the rest of the registered user functionalities listed under 1.1.1.1.
- **1.1.1.3.6** Guests shall be able to sign up.
- **1.1.2.1.1** Guests shall be able to sign up via providing their name, surname and e-mail address, and choosing a username and password.
- **1.1.2.2.1** Registered users shall be able to sign in with their username/e-mail address and password.

- **1.1.2.4.1** Registered users shall be able to create posts by selecting a location or multiple locations.
- **1.1.2.4.3** Registered users shall be able to add pictures to their posts.
- **1.1.2.4.4** Registered users should add date and time to their posts at the desired specific level as they remember their memories.
- **1.1.2.4.6** Registered users shall be able to tag their posts with words related to their posts.
- **1.1.2.5.2** Registered users shall be able to view posts on the locations they choose from the map.
- **1.1.2.5.3** Registered users shall be able to view the number of views of a post.
- **1.1.2.6.1** Registered users shall be able to search for posts posted at a specific time and date.
- **1.1.2.6.2** Registered users shall be able to search for posts posted at a specific time and date.
- **1.1.2.7.1** Registered users shall be able to leave comments under the posts.
- **1.1.2.7.2** Registered users shall be able to like posts of other registered users that they follow or that have public profiles.
- **1.1.2.8.1** Registered users shall be able to save the posts in their own profiles to access later.
- **1.1.2.9.1** Registered users shall be able to follow the registered users with public profiles without sending request.
- **1.1.2.9.2** Registered users shall be able to send follow requests to private profiles.
- **1.1.2.9.4** Registered users will be able to follow a private profile only if the owner of the private profile approves the follow request.
- **1.1.2.10.2** Registered users shall have a feed to display their activities.
- **1.1.2.12.1** Registered users shall be able to report inappropriate posts.
- **1.1.2.13.1** Registered users shall be able to edit the main part of their posts.
- **1.1.2.13.2** Registered users shall be able to edit the tags of their posts.
- **1.1.2.13.3** Registered users shall be able to edit the locations on their posts.
- **1.2.1.1** System shall ask the user to verify their address.
- **1.2.2.1** System shall have a "Forgotten Password" option in case registered users forget their password. Registered users will enter their emails and a link where they can change their password will be sent by the system via their emails.
- **1.2.2.2** System shall provide the user a "Remember Me" option to save username and password in local.[1]
- **1.2.4.1** System shall notify admins in case of a report of an inappropriate post.
- **1.2.4.2** System shall notify registered users with private accounts when they receive a follow request.
- **1.2.4.3** System shall notify registered users when someone starts to follow them.
- **1.2.4.4** System shall notify registered users when someone likes their posts.
- **1.2.4.5** System shall notify registered users when someone comments on their posts.
- **1.2.3.1** System should request the guest users to sign up for them to use the Android application.
- **2.2.1.1** Registered user passwords should be at least 8 characters long and should contain at least 1 uppercase letter, 1 number, 1 symbol.
- **2.2.1.2** Registered user passwords should not contain the registered user's username.
- **2.2.2** A verification email should be sent to the registered user if password change is requested. Until the registered user verifies the change, the password should not be changed.
- **2.3.1** Registered users' sign up data shall be stored in database.
- **2.3.2** Registered users' passwords shall be stored encrypted in database.
- **2.3.3** Registered users' profile activities shall be stored in database.
- **2.3.4** Registered users' posts shall be stored in database.
- **2.4.1** There should be a web interface for the project.
- **2.4.1.1** Known web browsers(such as Chrome, Opera, Firefox, Microsoft Edge, Safari) should support the web site.

- **2.4.2** There should be an android-based mobile app for the project.
- **2.5.1.1** Response time shall be short enough not to harm interactivity.
- **2.5.1.2** Reducing the response time for users should be prioritized.
- **2.6.1** W3C Activity Streams 2.0 Protocol will be followed throughout the project implementation.
- **2.6.2** W3C Standards will be followed.

Explanations

1. System automatically remembers the Sign In information. The user can Log Out if they want to.

List of unaddressed requirements

This part includes requirements that were not addressed by the team. Clearly, all of these requirements are not completed yet. Below are the lists of uncompleted functional and non-functional requirements:

Uncompleted Functional Requirements

- **1.1.1.3.1** Guests shall be able to view the main page.[1]
- **1.1.1.3.2** Guests shall be able to view public posts.[1]
- **1.1.1.3.3** Guests shall be able to search posts.[1]
- **1.1.1.3.4** Guests shall be able to filter posts.[1]
- **1.1.2.3.1** Guests shall be able to enter the web application without signing up.[1]
- **1.1.2.4.2** Registered users shall be able to allow location services to find their current location and add a story to it.
- **1.1.2.4.5** Registered users shall be able to tag other people to their posts.
- **1.1.2.7.3** Registered users shall be able to like comments under a post.
- **1.1.2.9.3** Registered users shall be able to follow locations.
- **1.1.2.10.3** Registered users shall be able to choose whether the profile page will be accessible to guests or registered users.
- **1.1.2.11.1** Registered users shall be able to choose to receive notifications or not.

Explanations

1. Guests can only sign up. These functionalities are implemented for Registered Users.

Uncompleted Non-Functional Requirements

- **2.2.3** Admins should change their passwords every 180 days. This should be reminded to the admin via email.
- **2.2.4** After 4 unsuccessful password entries, the registered users should verify that it was them by clicking the verification link sent to them via email.
- **2.2.4.1** Until the verification is done, the registered user can't enter the app.
- **2.2.4.2** After the verification, the registered users should reset their passwords.
- **2.5.2.1** Measures shall be taken to shorten the downtime as much as possible.

Scenario Analysis

Scenario: Serhan is a well-liked active young man who grew up in Germany. He visits Gulek - a small town in a plateau of the Toros mountains in Turkey. He is excited as this is his first visit to the land of his ancestors. He wants to discover the area from an experiential perspective and to share his experiences using your application. He will discover beautiful places, food (obviously, like sıkma, sucuk, ...), and most importantly stories about the lives of people over time (i.e. how the women gather to prepare hot pepper paste and lava/kuru ekmek while kids run around playing). Serhan is interested in stories in the area of 30km. He is on a motorcycle and will be traveling and documenting his travels via your application. His friends from all over the world will be following his adventures with great interest.

The latest version of the Postory proudly covers all the functionalities mentioned in the above scenario. Below you can find the implementation details of the functionalities (i.e., issues and pull requests), by each team (i.e., frontend, android, and backend).

Frontend

Functionality	Related Issues	Related PRs
Activity Stream page	Issue #492	PR #528
Filtering functionality	Issue #490 - Issue #532	PR #509 - PR #560 - PR #535
Wiki Data Functionalities	Issue #508	PR #523
Follow a user with private account	Issue #494 Issue #507	PR #497 PR #526

Android

Functionality	Related Issues	Related PRs
Activity Streams Page	#554, #561	PR 555, PR 562
Following Functionality	#446,#466, #576	PR 573
Filtering Functionality	#529, #557,#574	PR 530, PR 575
User Search Functionality	#489	PR 522

Serhan can filter stories according to location, time(with different precision levels), tags and title. So, he can focus on stories around his region with the topics that he's interested in. Also, he can use the Wikidata functionality of the application to see semantically relevant tags.
Friends of Serhan can use the search functionality to find his profile and follow him. The search functionality allows them to search with username. If they follow Serhan, they can see his posts at their homepages so it will be easy for them to follow. In addition to that, they will see the activies of Serhan at their activity streams pages.

Backend

Functionality

- Nearby stories (stories in the area of 30 km): By using the longitude and latitude variables in the locations, the distance between the locations are found in kilometer. If a post has a location whose distance is less than 30 kilometers to given location, the post is returned by the filter API.
    - **Related Issues:** #502
    - **Related PRs:** #512
- Follow user: Users can follow public users or sent request to private users. If the follow request is accepted, they will be able to see private users' posts.
    - **Related Issues:** #547, #544, #515, #484, #483, #368
    - **Related PRs:** #386, #513, #541
- Create post: The posts are created by given title, story, location, time tags, and images. Images are stored in AWS S3 bucket.
    - **Related Issues:** #261
    - **Related PRs:** #259
- Get posts of followed users: At the home page, users can see the posts of users that they follow.
    - **Related Issues:** #382, #380
    - **Related PRs:** #383
- Search/Filter: Users can be searched by their username. If the user contains the given keyword in his/her username it will be returned. Posts can be searched and filtered by user, tag, location, keyword and time. For user filtering, if the owner of the post is the given user, it will be returned. For tag filtering, there are two options. Users can chose only searching for the tag they entered or they can chose searching for related tags. Related tags are obtained from Wikidata. If the post has any tag of given tags, it will be returned. If a post has a location whose distance is less than given kilometer to given location, it will be returned. For keyword filtering, if the post has the given keyword in its title or story, it will be returned. For time filtering, if the post's time intersects with the given time, it will be returned. If multiple filter is used, the instersection of the filters will be returned.
    - **Related Issues:** #498
    - **Related PRs:** #512
- Like and comment: When a user likes or comment, the post object will be updated in the database. Its like count will increase and the comment will be appended to the post.
    - **Related Issues:** #370
    - **Related PRs:** #385

W3C Activity Streams 2.0 and Wikidata

W3C Activity Streams 2.0

W3C activity stream model is being used for our activity stream model. We used minimal activity containing context, actor, summary, object, type and two additions date, url and success. Context contains https://www.w3.org/ns/activitystreams, actor is the user's id that made the activity, summary is the summary of the activity, object is the thing that was effected during the activity (for example post id when a user likes a post), type is the type of the action (such as PostLike), date(published) is the time that activity was done, url is the relative url of the request, success is the success status of the request (if user was not able to like that post etc.). With each request to our endpoints, activites are stored with this model.

Wikidata

Wikidata API is used for finding related tags in the filter function.If a user wants to filter posts with tags related to given tags, Wikidata API is used.
Two actions used for finding related tags. The first one is “wbsearchentities”. The description of the entity that is most related to the given tag is found by using this action. Then, by using the ID of this entity, “wbgetentities” action is used. Claims and aliases are retrieved from the response of this request. To retrieve the most related tags, only a limited number of words are returned.

API Calls (JSON-LD)

- There are three endpoints for activity stream. Those are:
  - /api/activitystream/own
    - This endpoint returns user's own activities.
  - /api/activitystream/followed
    - This endpoint returns user's followed users' activities.
  - /api/activitystream/all
    - This endpoint returns every public users' activities.
- Below you can find an example of an activity. For each endpoint, activities such as the one below, will be returned in an array.
- Normally, in our database we store actor's ID only. But for returning activities to frontend and android and ease their work, we used serializers which gets id of user and returns its object containing its name, surname, images etc.


```
{
    "actor": {
        "id": 5,
        "username": "mertlkn",
        "password":
"pbkdf2_sha256$260000$ElLsCxN3Yj2PL1MzcY5LOl$IZRcUsqaYigMIFwvqk/cHAt3LbEKXTlNbnx3S9A2tUO=",
        "name": "Mert",
        "surname": "Alkan",
        "email": "m558mert@gmail.com",
        "followedUsers": [],
        "followerUsers": [
            {
                "id": 6,
                "username": "mstfec",
```

```
                "name": "mustafa",
                "surname": "mustafa",
                "email": "mstfec@gmail.com",
                "isBanned": false,
                "isAdmin": false,
                "isPrivate": false,
                "is_active": true,
                "userPhoto": "https://group9-451-
project.s3.amazonaws.com/image/Ekran_Resmi_2021-12-21_16.23.09_HhjSW44.png"
            },
            {
                "id": 14,
                "username": "mozcan",
                "name": "Melih",
                "surname": "Özcan",
                "email": "melihozcan98@hotmail.com",
                "isBanned": false,
                "isAdmin": false,
                "isPrivate": false,
                "is_active": true,
                "userPhoto": "https://group9-451-
project.s3.amazonaws.com/image/Ekran_Resmi_2021-12-21_16.23.09_HhjSW44.png"
            }
        ],
        "posts": [
            39,
            40,
            41,
            42,
            43,
            44,
            45,
            46,
            63,
            65,
            67,
            68
        ],
        "savedPosts": [],
        "likedPosts": [],
        "comments": [],
        "isBanned": false,
        "isAdmin": false,
        "isPrivate": false,
        "is_active": true,
        "userPhoto": "https://group9-451-
project.s3.amazonaws.com/image/Ekran_Resmi_2021-12-21_16.23.09_HhjSW44.png"
    },
    "object": "https://group9-451-
project.s3.amazonaws.com/image/Ekran_Resmi_2021-12-21_16.23.09_HhjSW44.png",
    "type": "UserAddPhoto",
    "date": "2022-01-04"
}
```

API

- **Link to API**: http://3.67.83.253:8000
- **API documentation**: The API documentation in *pdf format* can be found here.
- **Endpoints** are explained below. Example requests can be found in the following links:
    - Follow users, get user information, get post by id, report users/story, accept/decline follow requests, get pending follow requests examples can be found in this postman collection link.
    - Get main page posts, get discover page posts, get a specific user's posts, add profile photo to current user examples can be found in this postman collection link.
    - Sign up, activation, sign in, post create, filter, post update, comment and like examples can be found in this postman collection link.
    - Get own activities, get followed users' activities, get every public users' activities examples can be found in this postman collection link.

User Interface and User Experience Designs

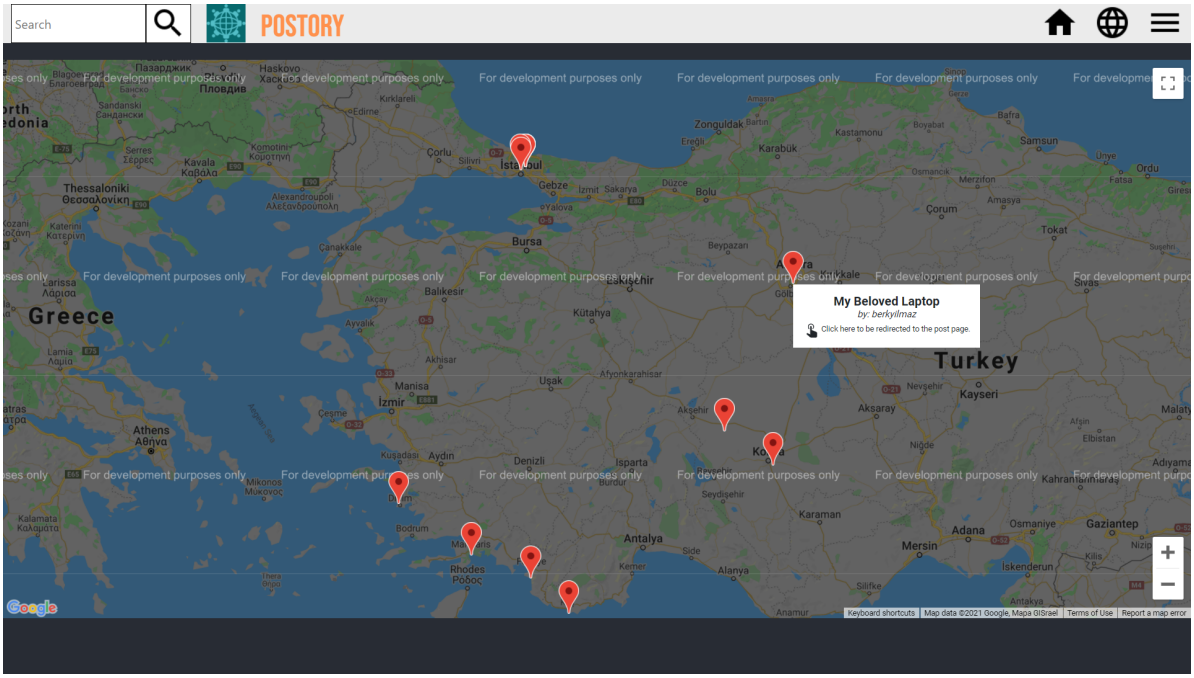
Frontend

Discover Page

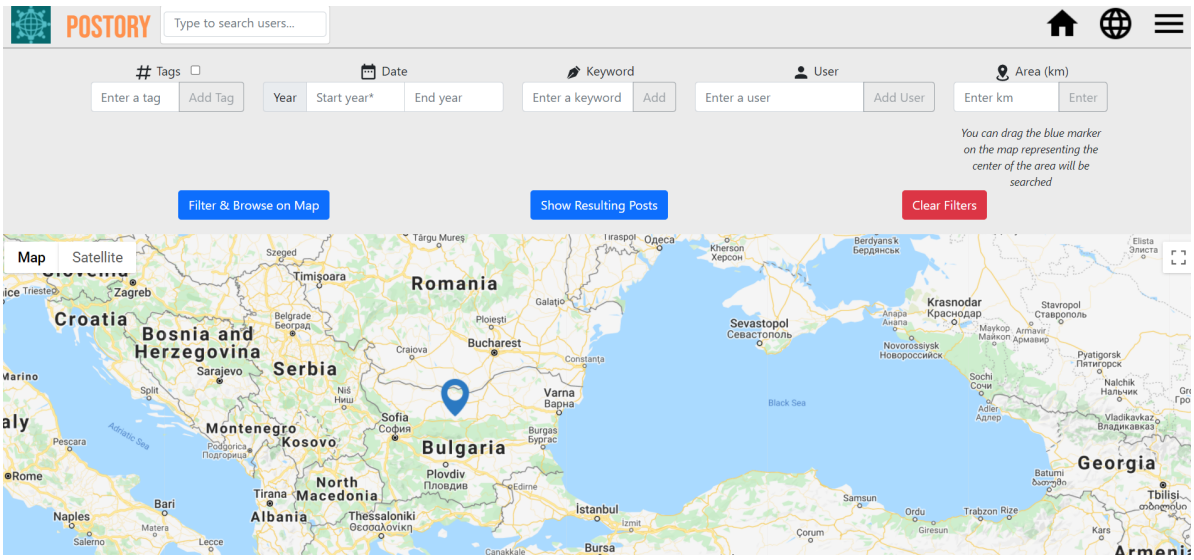
Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/DiscoverFilter.js

Screenshot(s)



Updates since Milestone 2

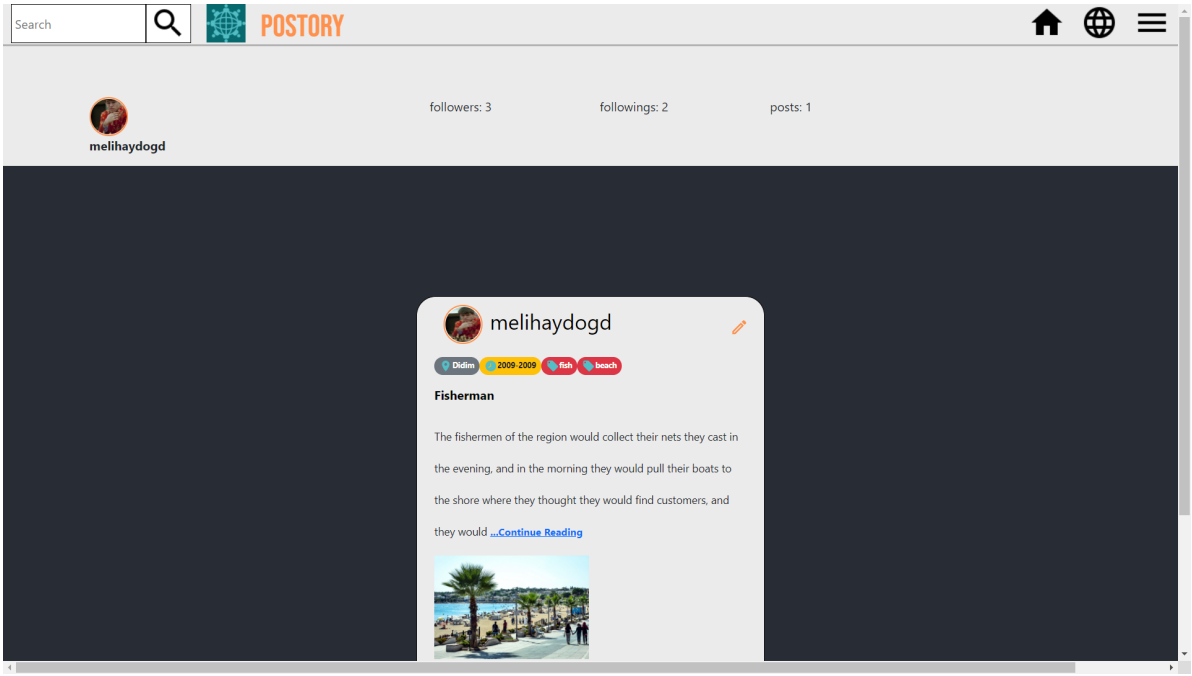


Profile Page

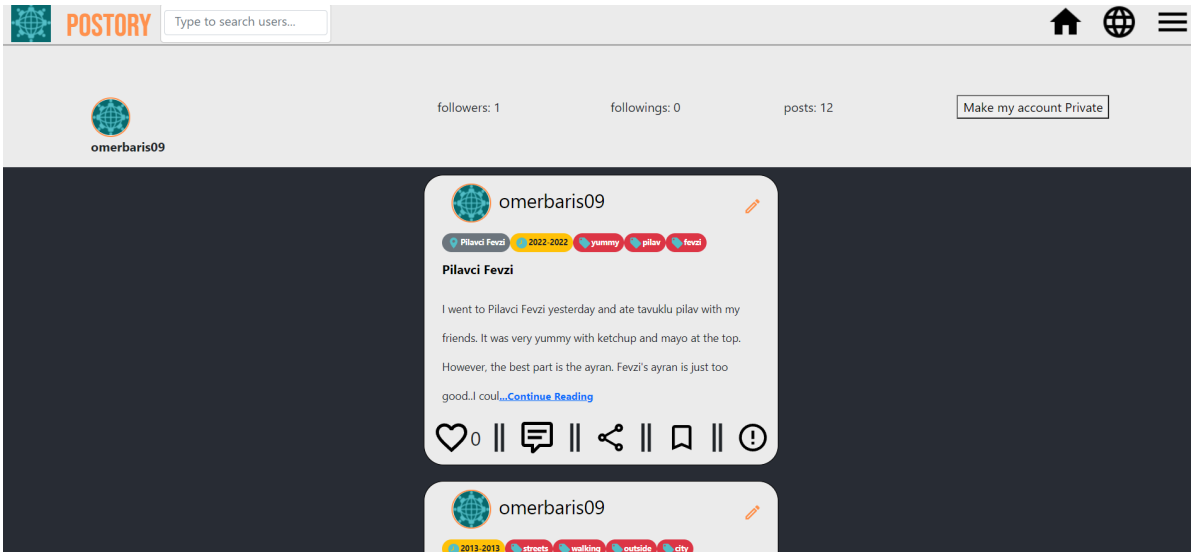
Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/ProfilePage.js

Screenshot(s)



Updates since Milestone 2

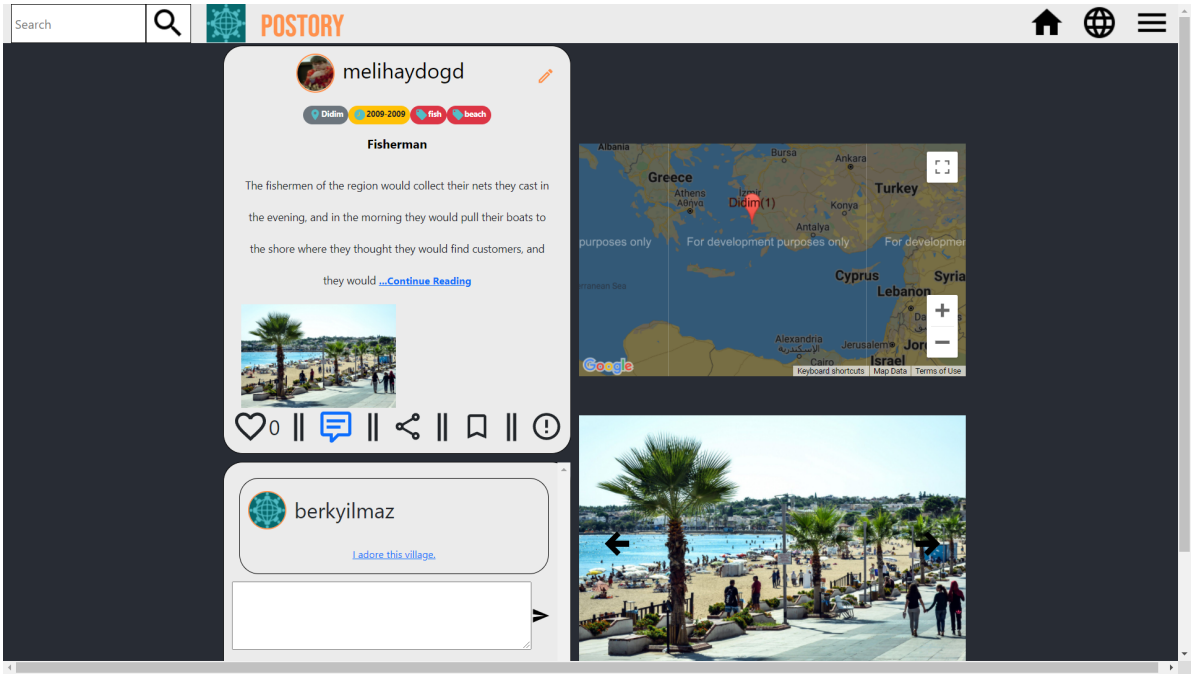


View Post Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/ViewPost.js

Screenshot(s)

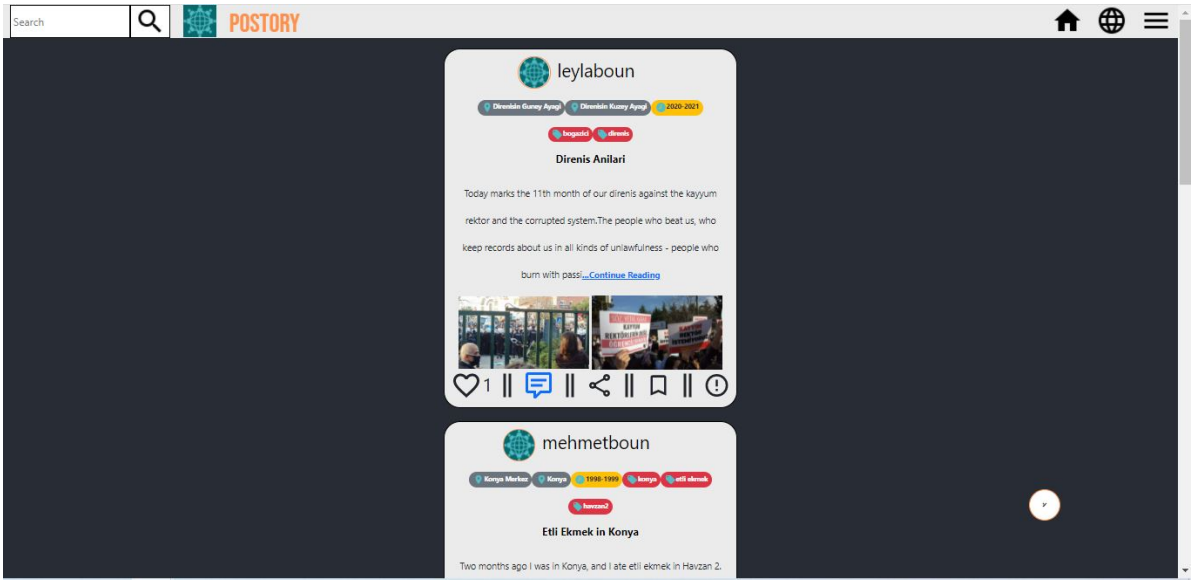


Home Page

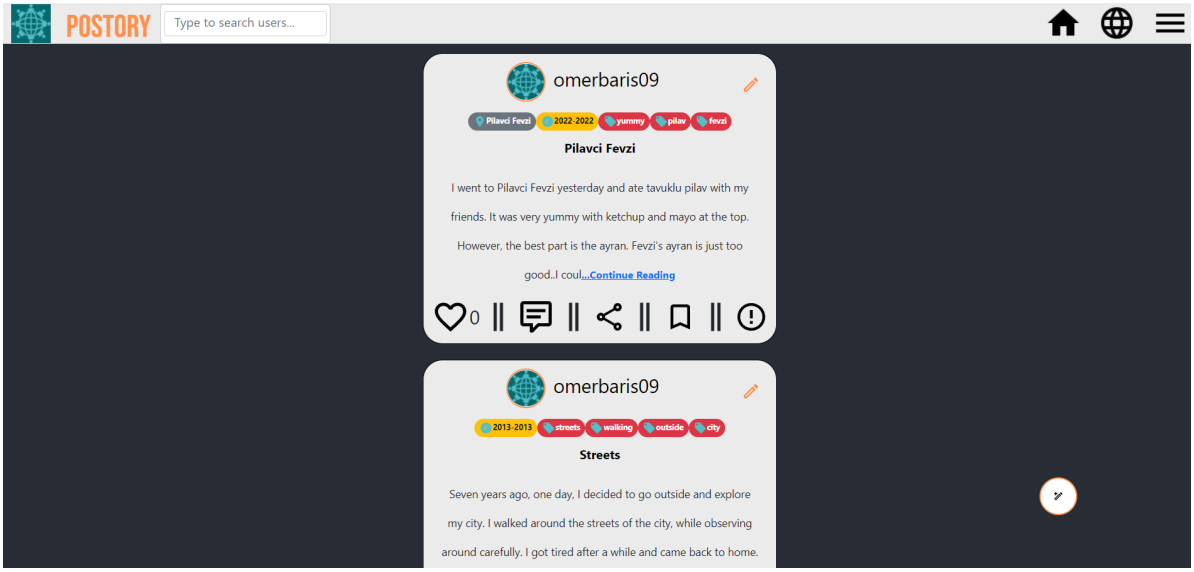
Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/App.js

Screenshot(s)



Updates since Milestone 2

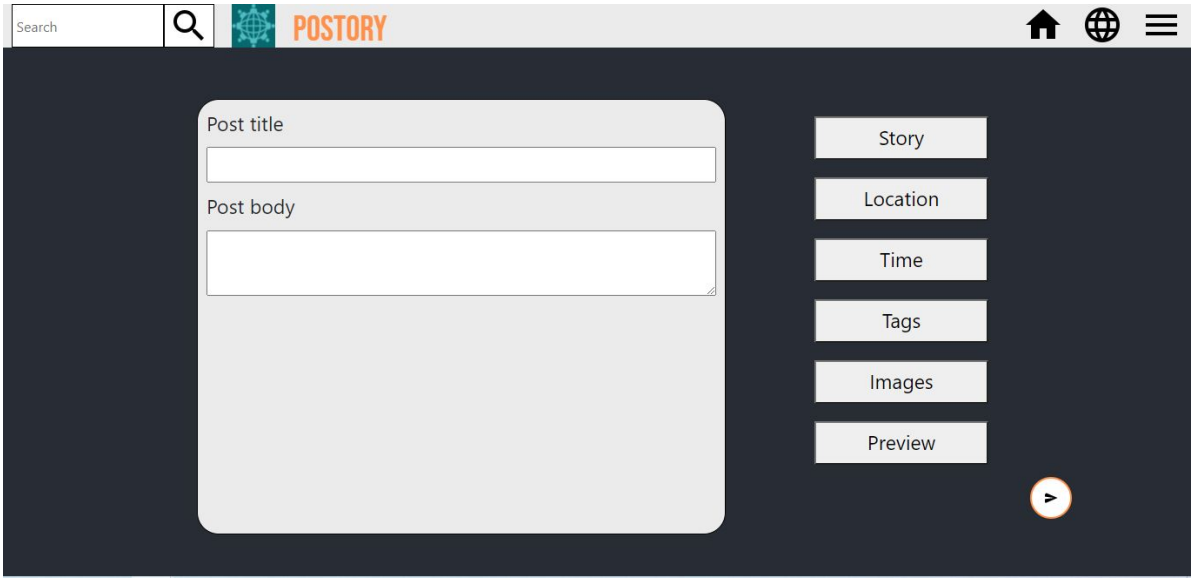


Create/Edit Post Page

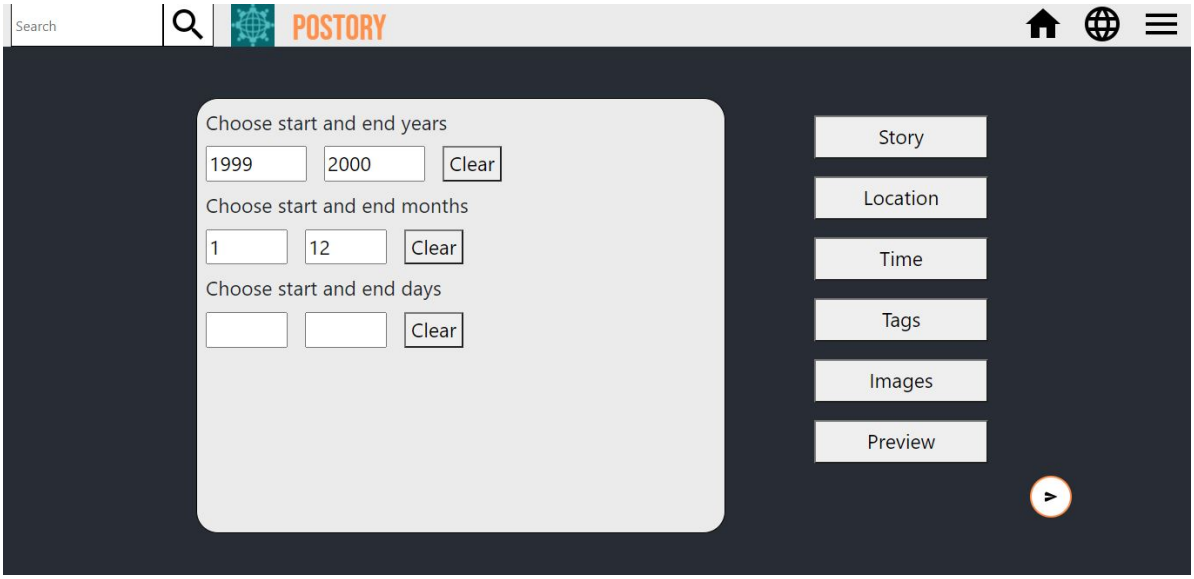
Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/CreatePost.js

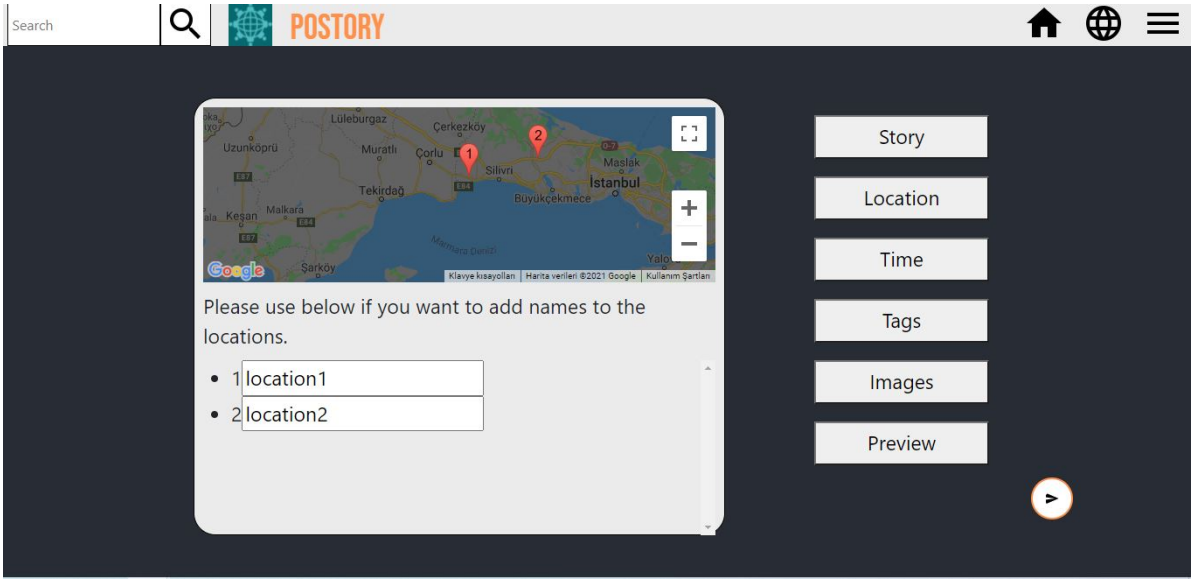
Adding Story



Adding Time

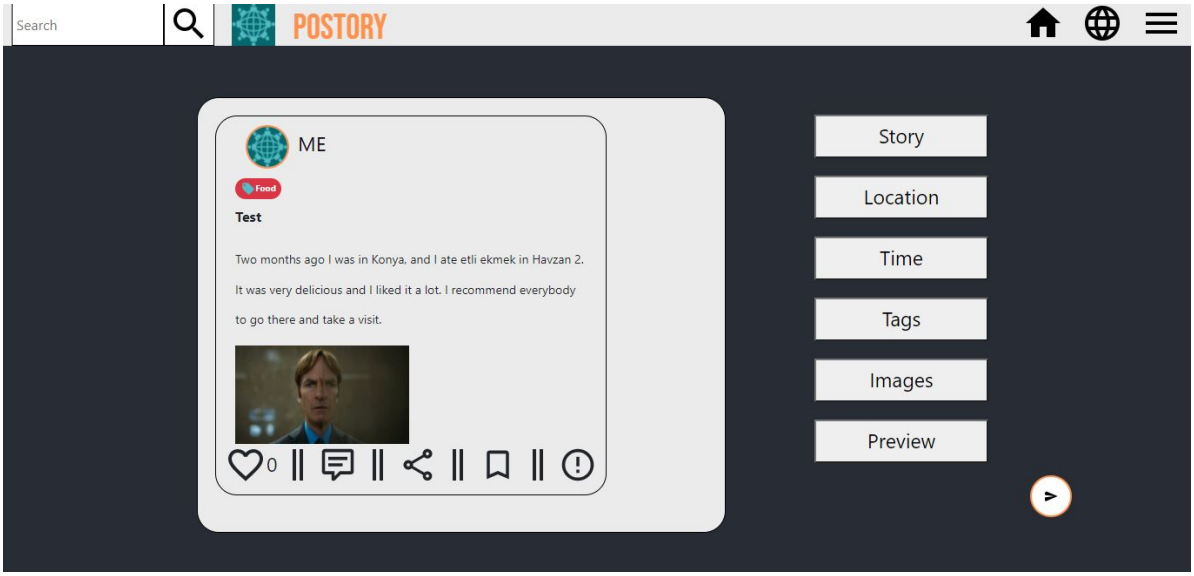


Adding Locations



Adding Tags

Seeing the preview

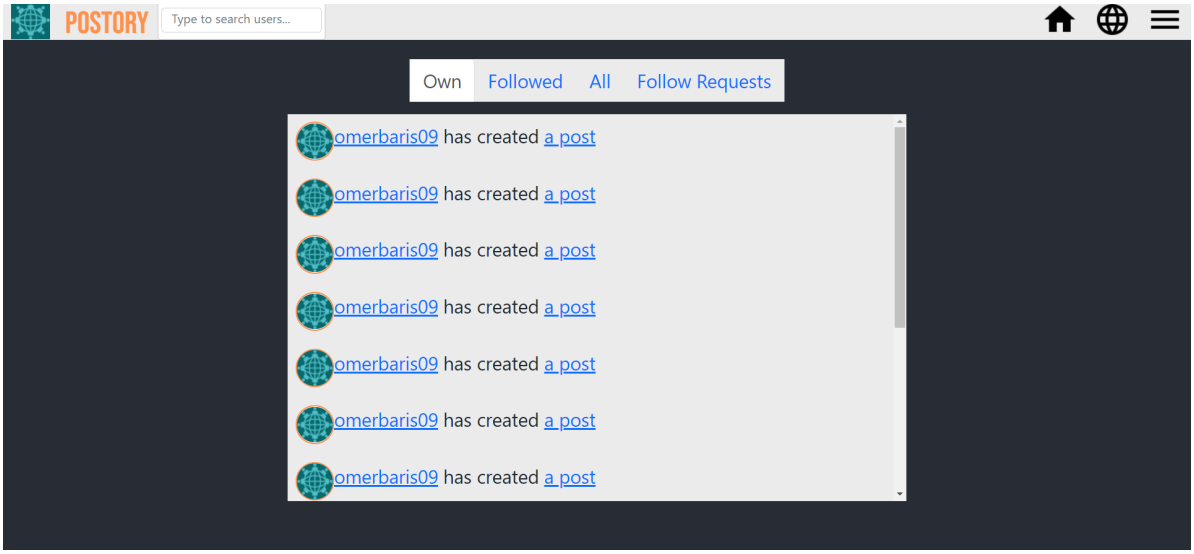


Activity Stream Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/ActivityStream.js

Screenshot(s)

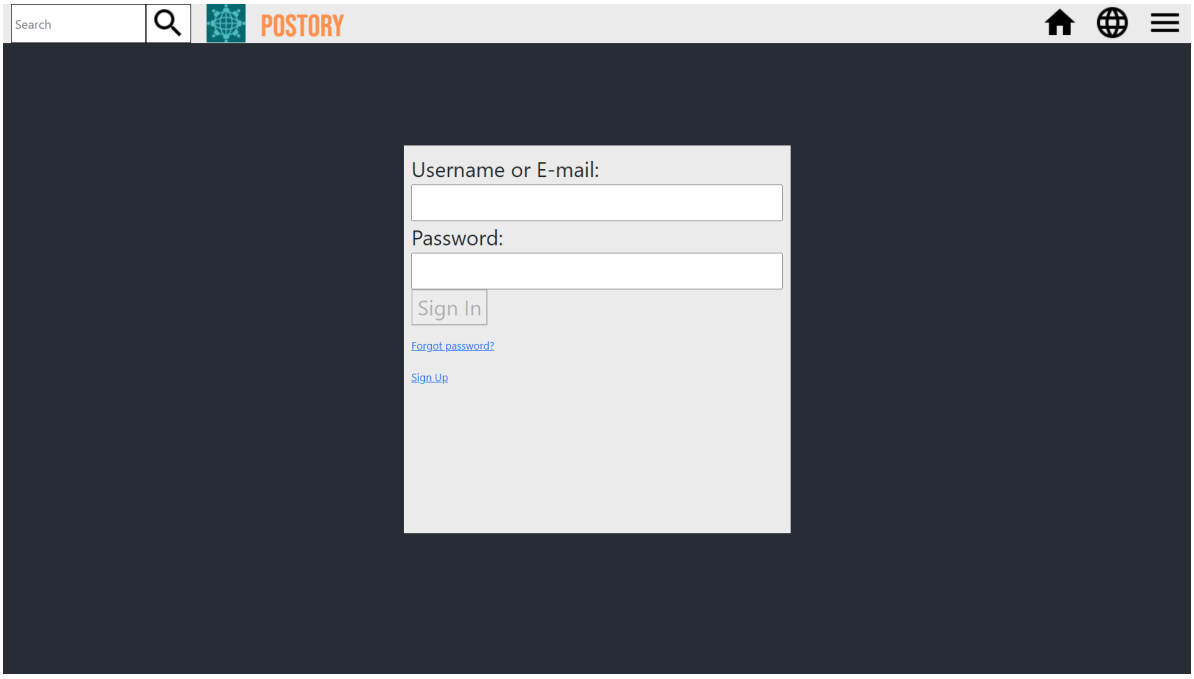


Sign In Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/SignIn.js

Screenshot(s)

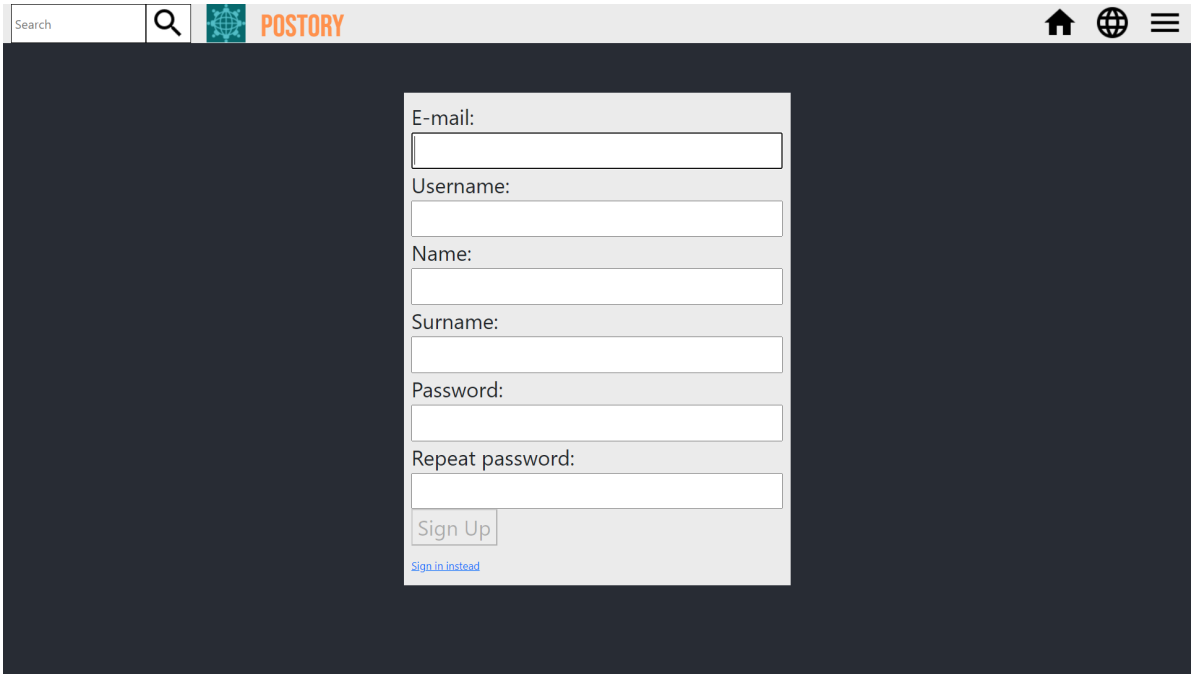


Sign Up Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/SignUp.js

Screenshot(s)

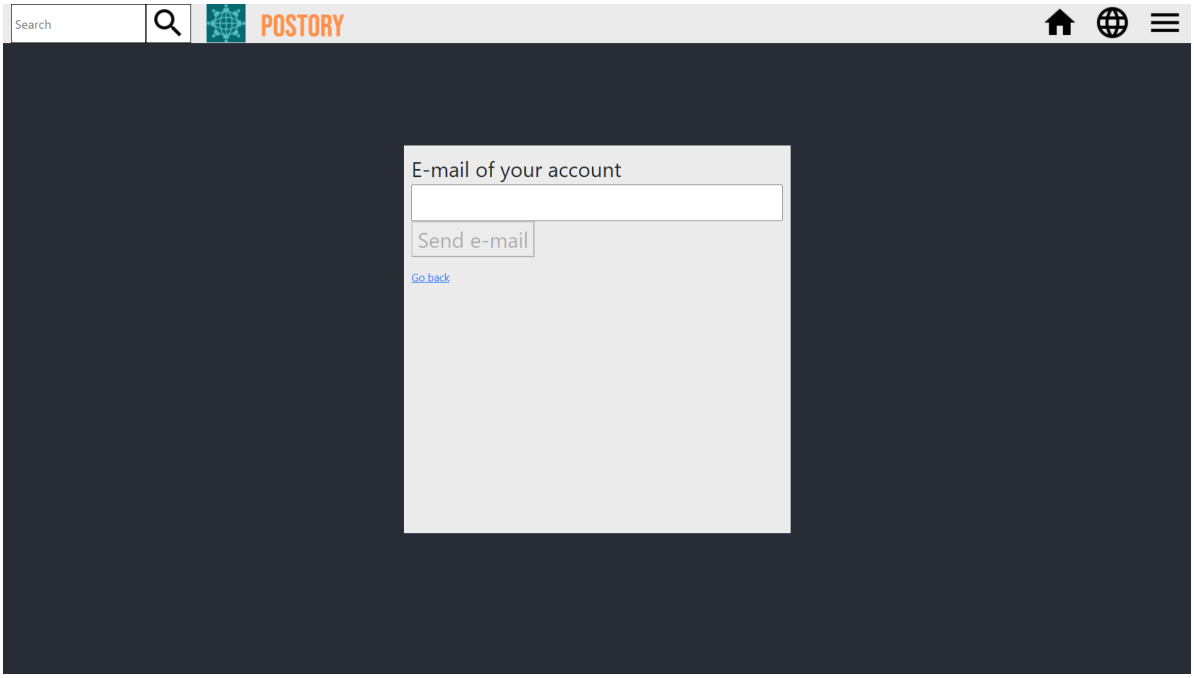


Forgot Password page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/frontend/src/ForgotPassword.js

Screenshot(s)



Android

Sign In Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/android/postory/android/Postory/app/src/main/java/com/example/postory/activities/LoginActivity.java

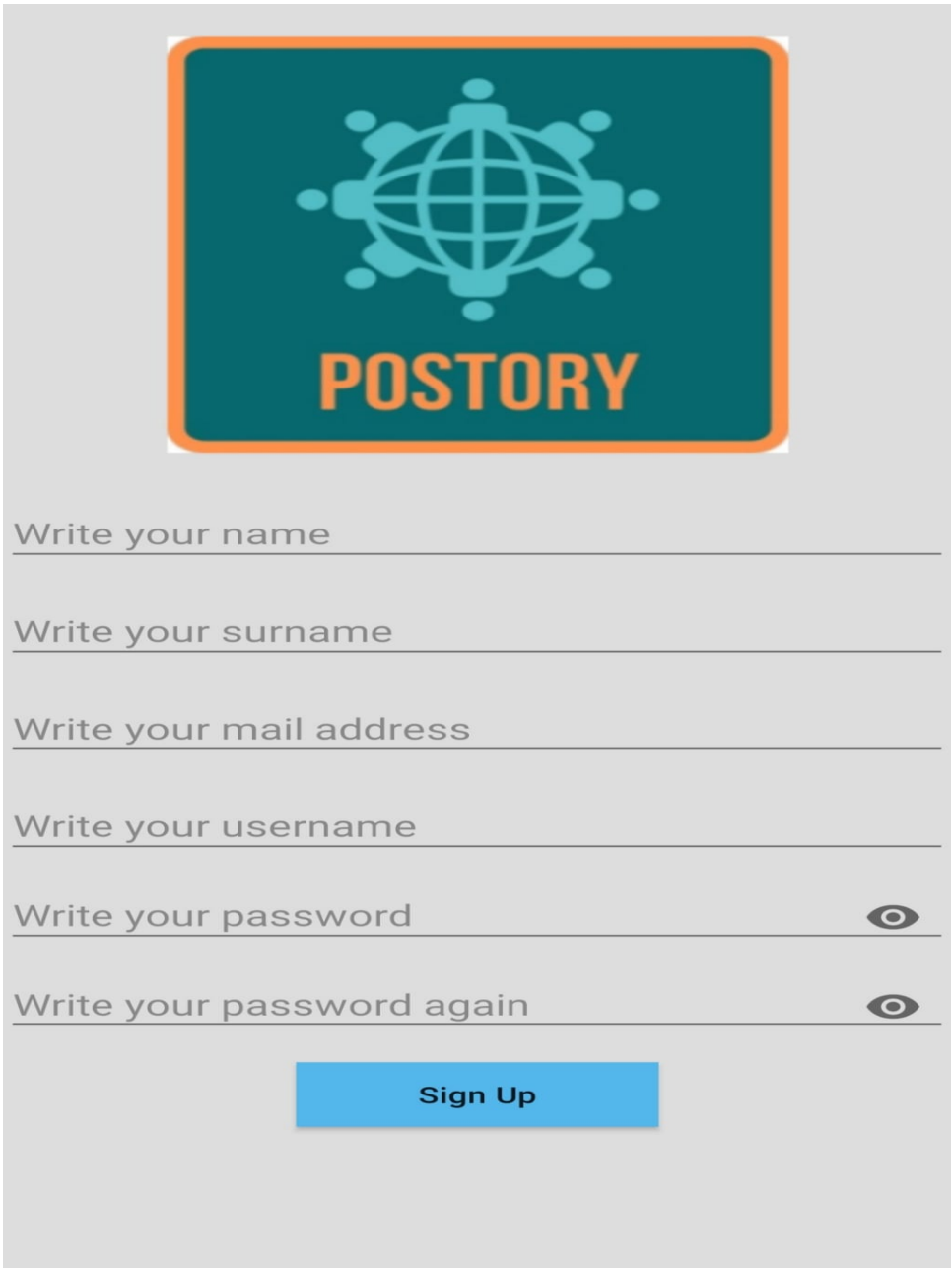
Screenshot(s)

Sign Up Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/android/postory/android/Postory/app/src/main/java/com/example/postory/activities/RegisterActivity.java

Screenshot(s)

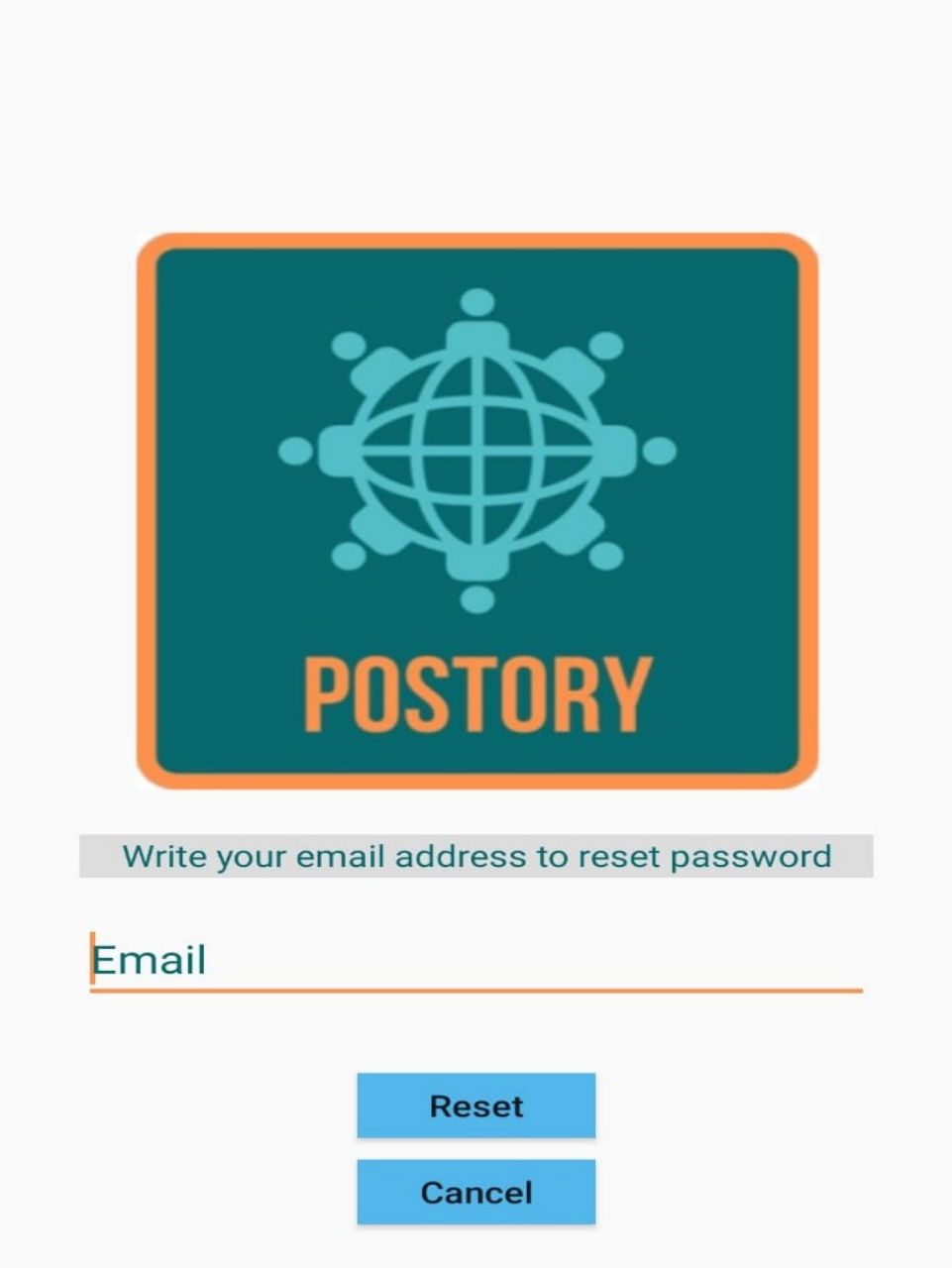


Forgot Password Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/android/postory/android/Postory/app/src/main/java/com/example/postory/activities/ForgotPasswordActivity.java

Screenshot(s)

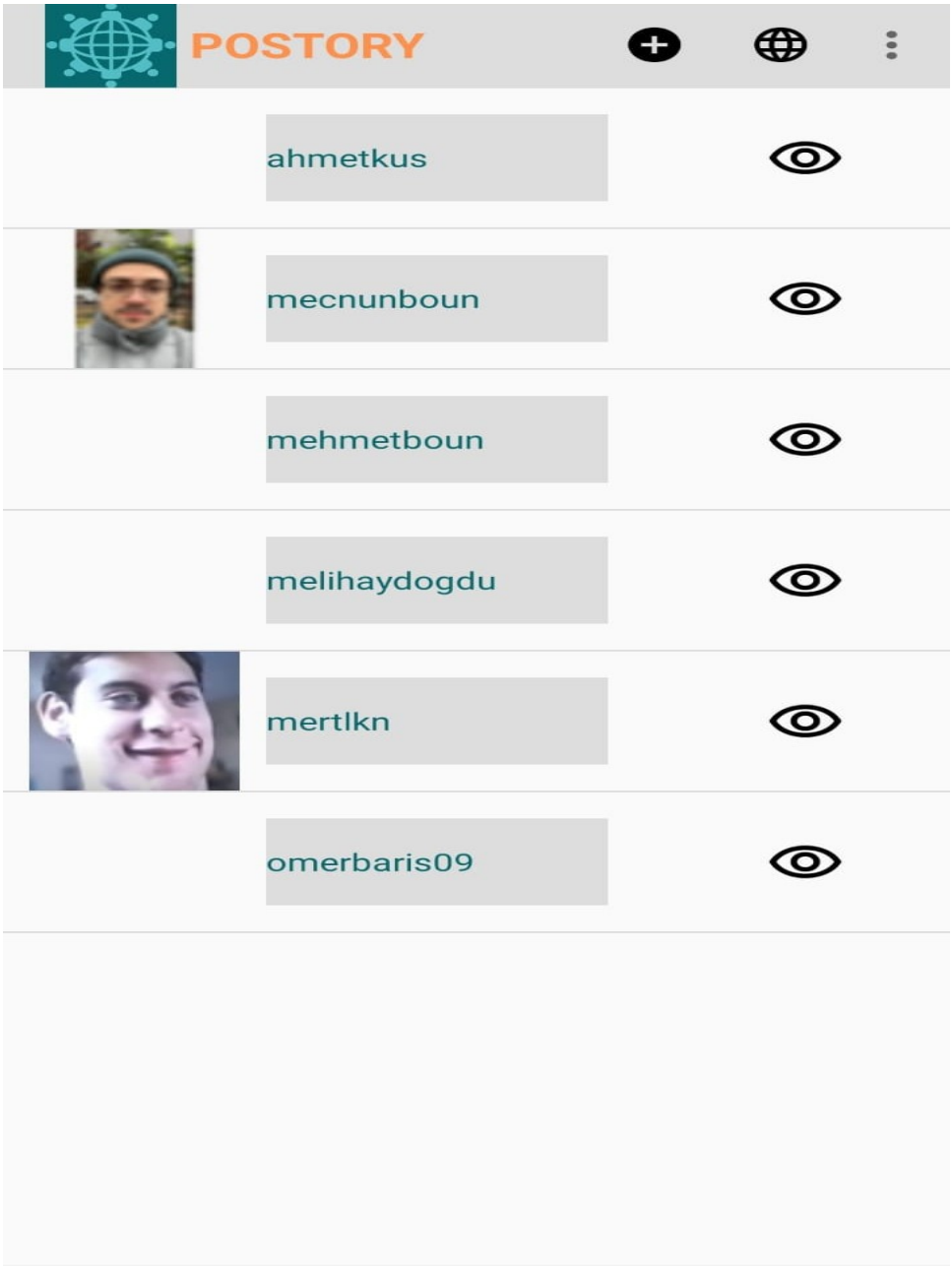


User Search Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/UserSearchActivity.java

Screenshot(s)

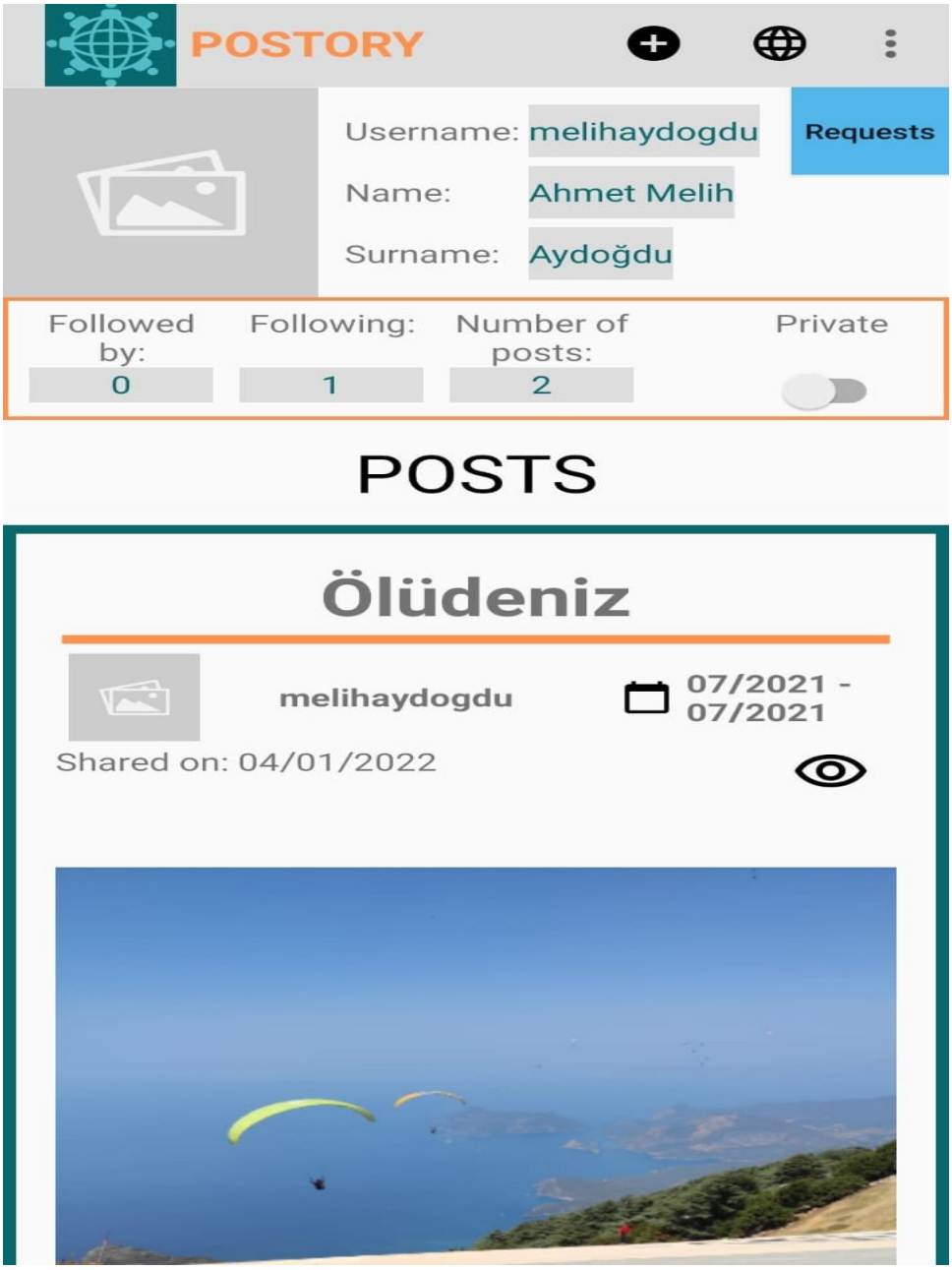


Profile Page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/SelfProfilePageActivity.java and https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/OtherProfilePageActivity.java

Screenshot(s)

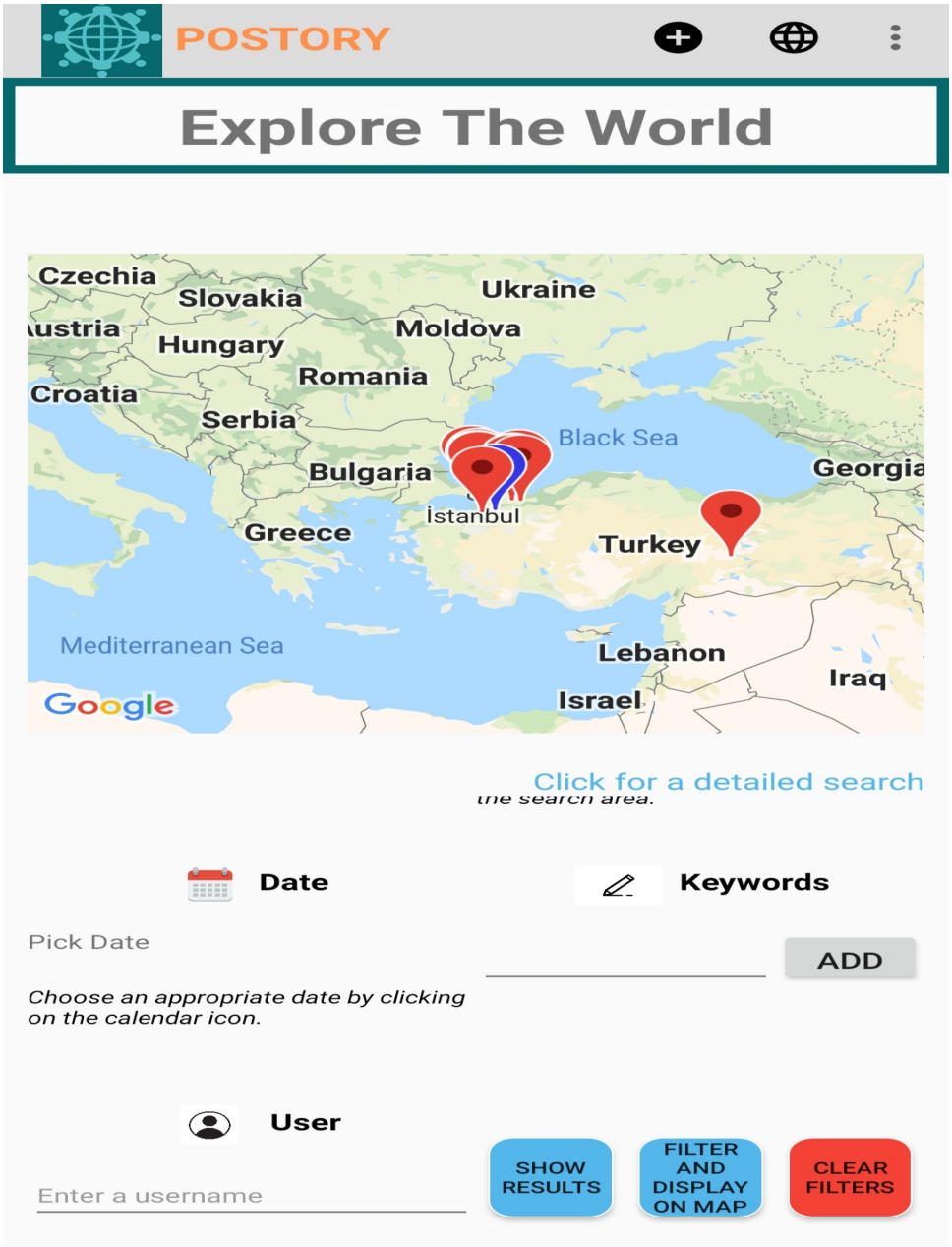


Discover page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/ExploreActivity.java

Screenshot(s)

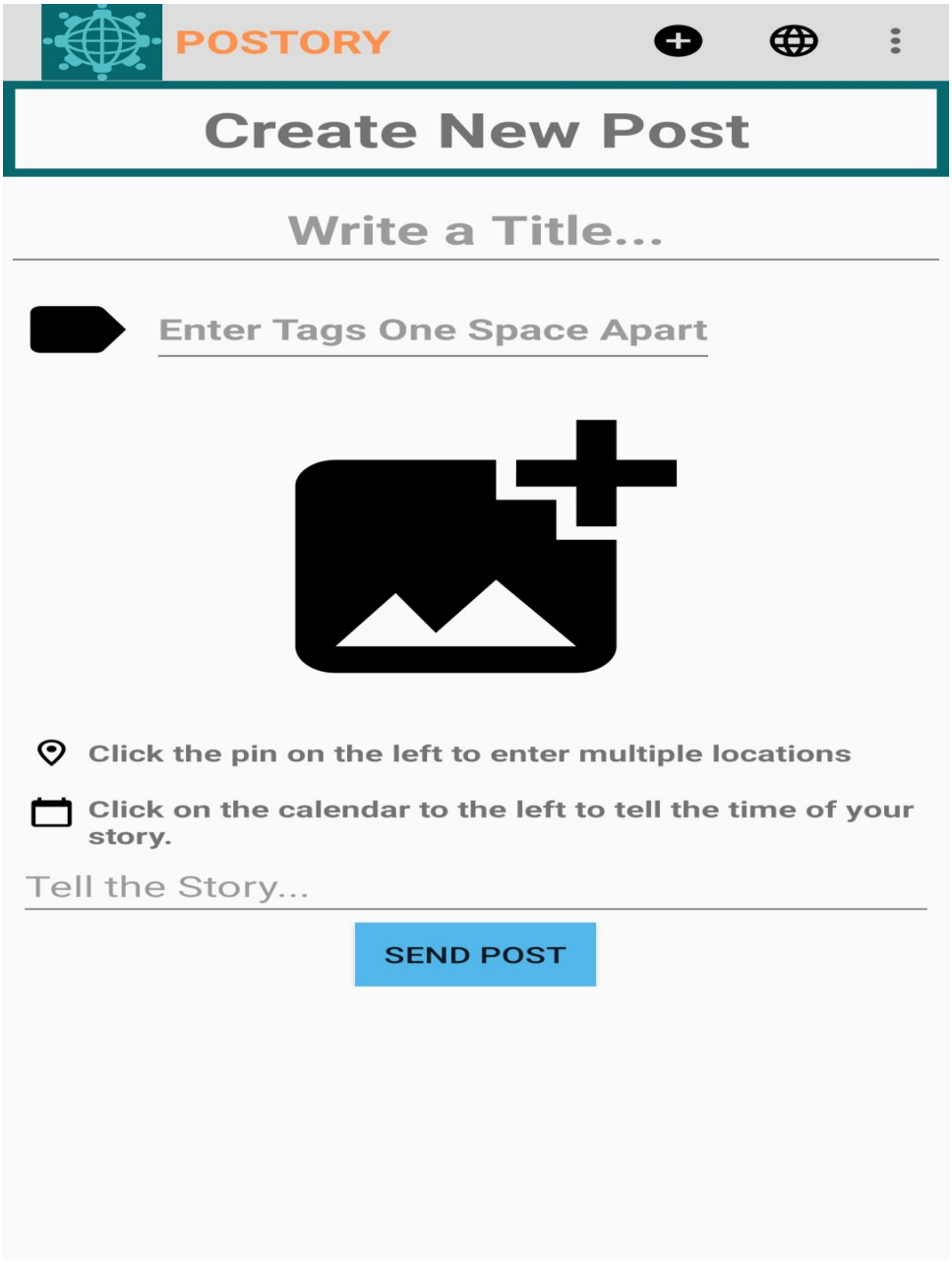


Create/Edit Post page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/CreatePostActivity.java

Screenshot(s)



POSTORY

Edit The Post

Engels and Marx

marx engels ideology

Parks of Berlin

15/06/2021 - 19/06/2021

This is a statue depicting Marx and Engels, two of the most important figures in forming the communist ideology. Many tourists and locals come to see their statues everyday.

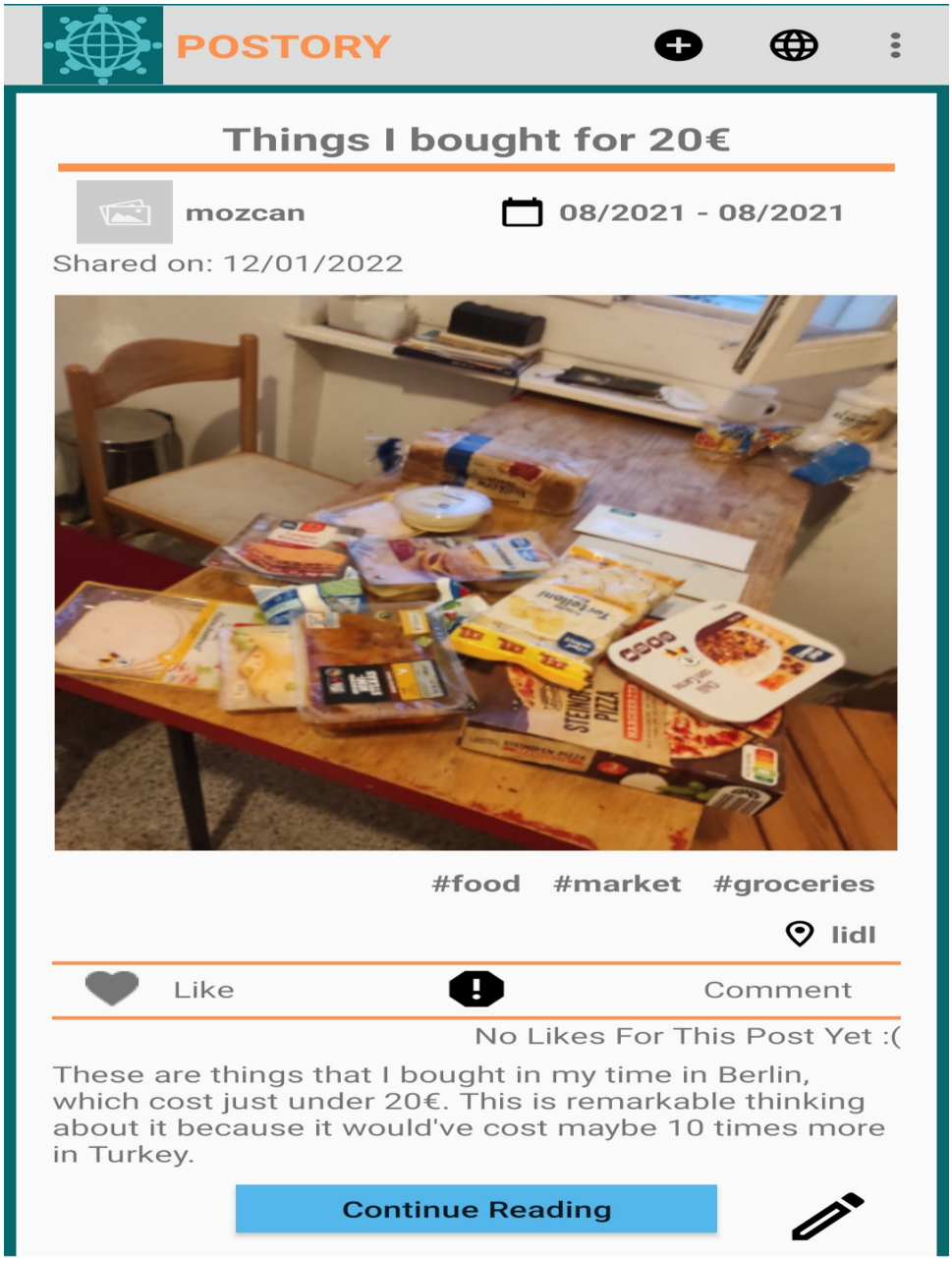
SEND POST

View Post page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/SinglePostActivity.java

Screenshot(s)

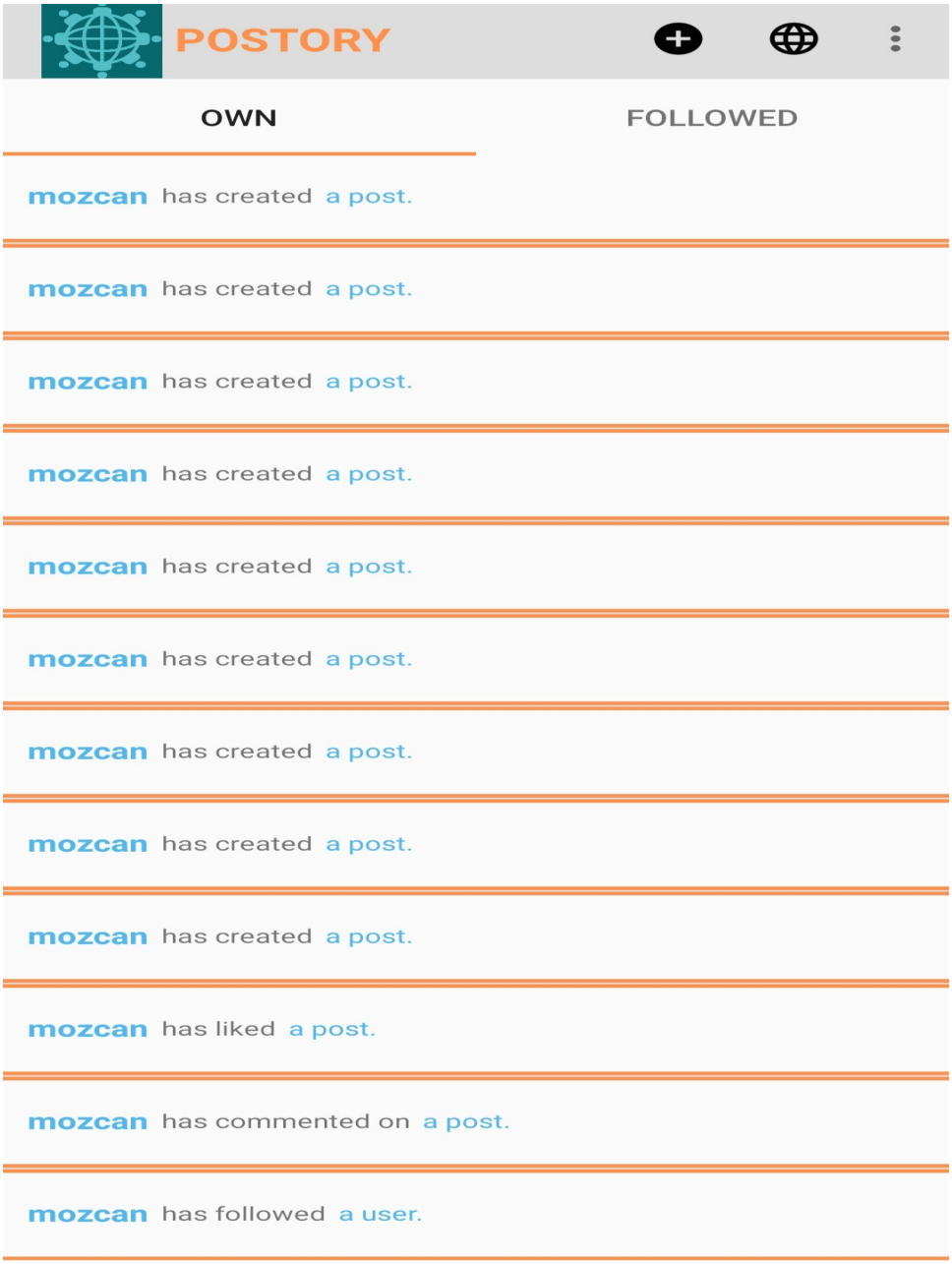


Activity Stream page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/ActivityStreamActivity.java

Screenshot(s)



Home page

Link to code

https://github.com/bounswe/2021SpringGroup9/blob/master/postory/android/Postory/app/src/main/java/com/example/postory/activities/MainActivity.java

Screenshot(s)

User manual

Web

Discover Page

Filter and Explore

Discover page is where users can see their posts, posts of the users that they follow, and the posts of all public profiles - on the map as red markers (*see* the black dashed marker on *Fig1*). Users can filter those posts using tag, date, keyword, user, and area.

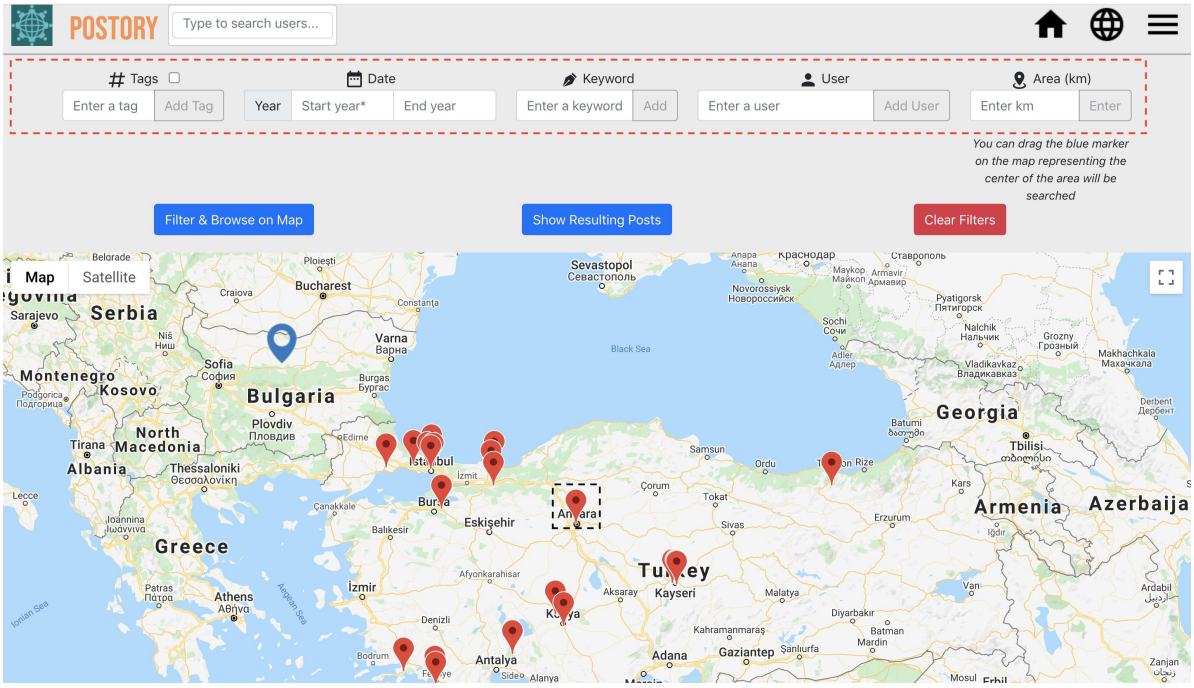


Fig1: Discover Page - 1

An example discovery page can be seen below (see *Fig1*). The dashed area is where users can specify their filters. Here, users can enter tag, date, keyword, user, and area to filter the posts. After specifying the filters of choice, users can see the resulting posts either on the map (i.e., by clicking the `Filter & Browse on Map` button) or on a different page (i.e., by clicking the `Show Resulting Posts` button).

An **example sequence** of filtering is explained below step by step:

1. Here, the user first enters a tag `food` into `Enter a tag` input space, clicks on `Add Tag`, and clicks on the `Filter & Browse on Map` button (see red dashed button on *Fig2*). The resulting markers represent the posts containing the tag `food`.

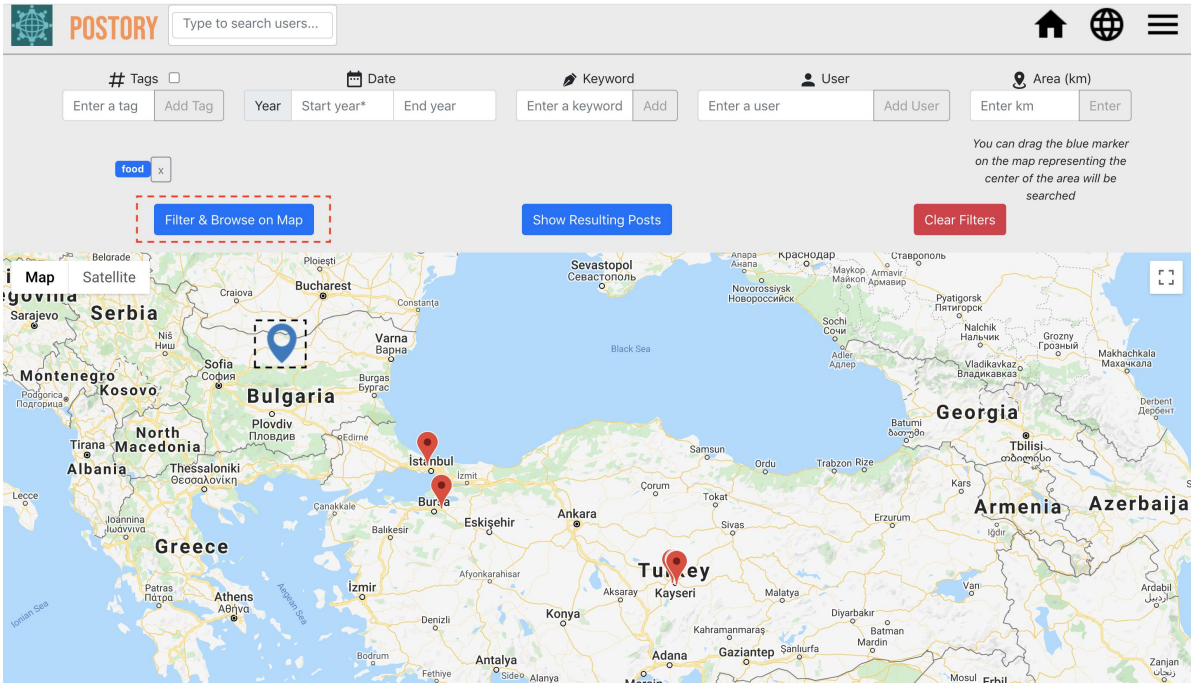


Fig2: Discover Page - 2

2. Later the user filters the results by specifying an area of choice. Here, the user first drags the blue marker (*see* the black dashed marker on *Fig2*) which represents the center of the area that is specified. Then the user enters `30` into `Enter km` input space and clicks on `Enter`.
3. Finally, the user clicks on the `Filter & Browse on Map` button again and sees the resulting posts on the map. (*see Fig3*)

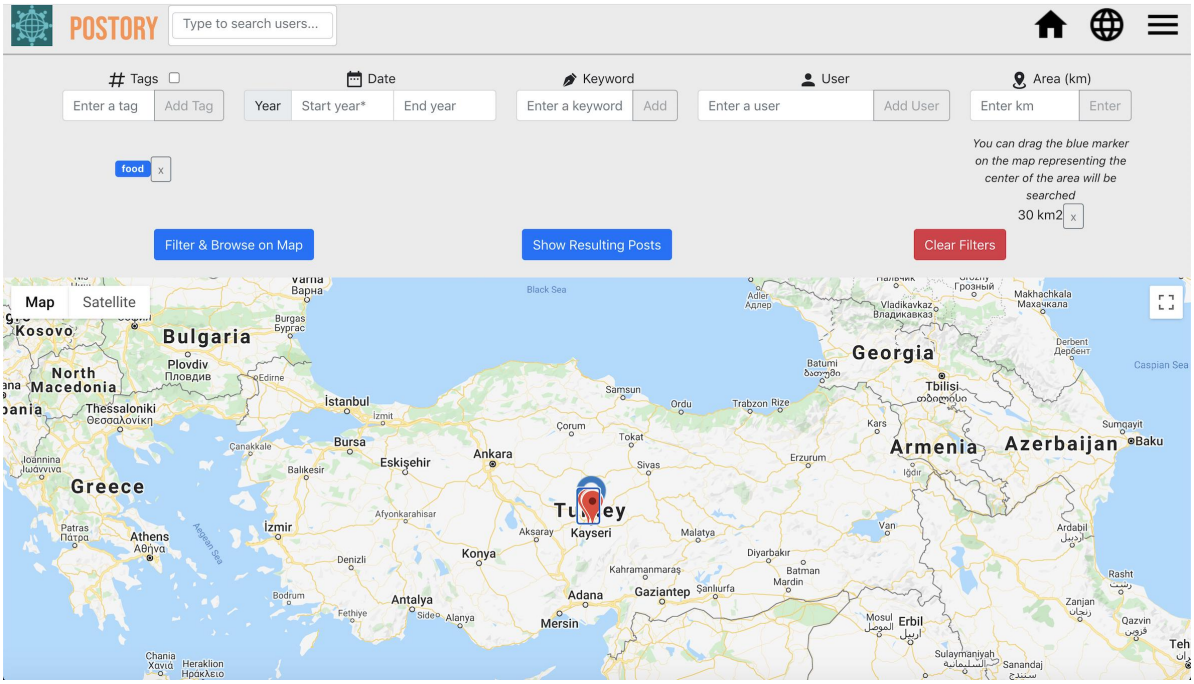


Fig3: Discover Page - 3

Users can click on the red markers, representing the posts, and see the brief information about the posts (i.e., its title and author) in a pop-up (*see Fig4*). They can click on the pop-up to see the full posts on a View Page (*see* View Post Page)

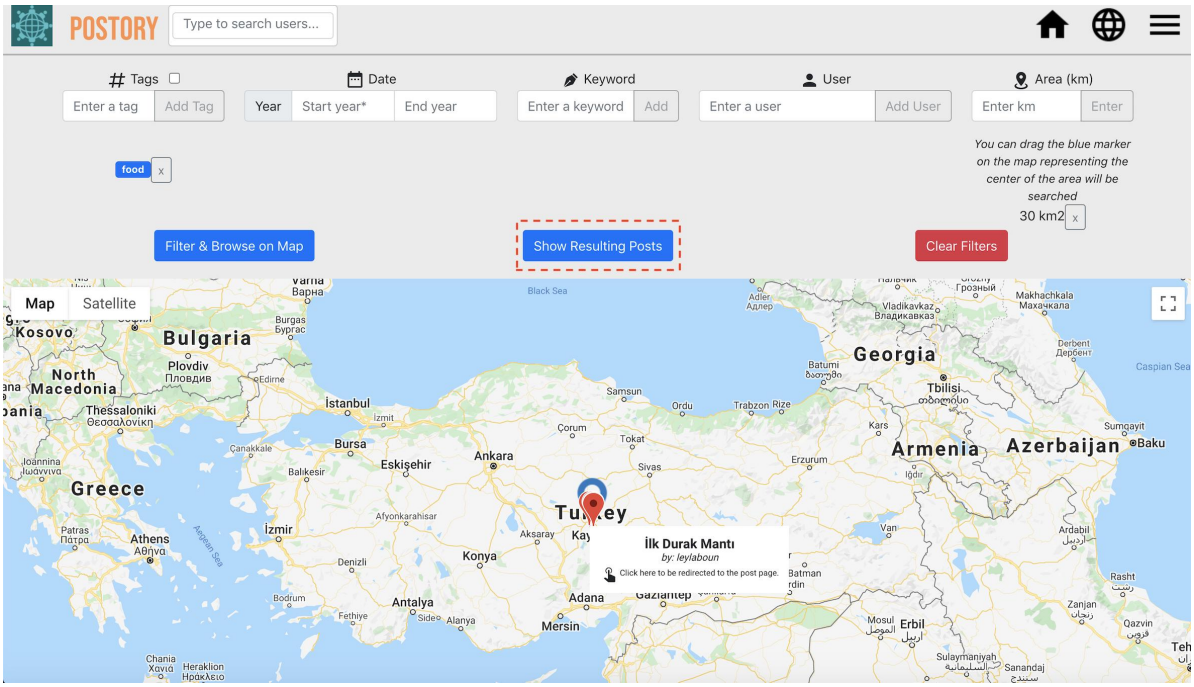


Fig4: Discover Page - 4

Users are able to see the resulting posts after applying the filters on a different page, in order to see the full posts like in the Home Page (*see* Home Page), by clicking the `Show Resulting Posts` button (*see* red dashed button on *Fig5*). They will be directed to a page like a Home Page, to see the resulting posts of a filter operation (*see Fig5*).

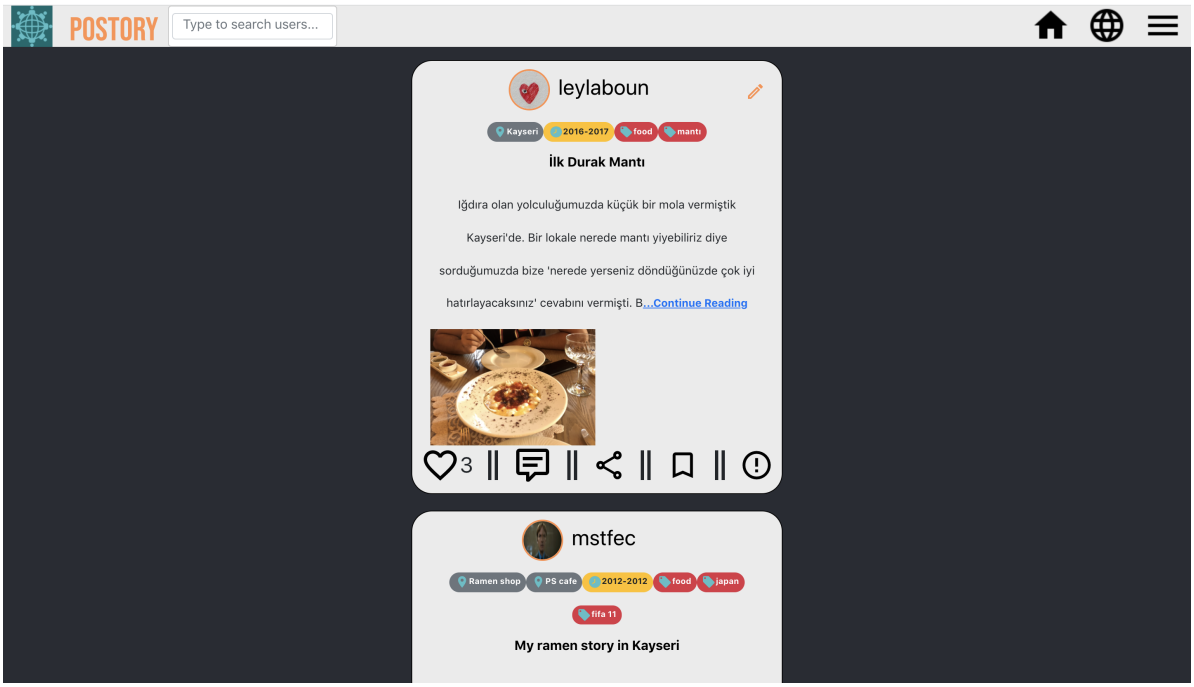


Fig5: Discover Page - 5

Related Search

Users can also expand their search space when they want to filter posts using a tag (i.e., they can add more related tags to their search). Users can see the related tags of the tags that they have added to their search by clicking on a tag (*see* red dashed area on *Fig6*). The related tags will pop-up on the bottom of the screen (*see* blue dashed area on *Fig6*).

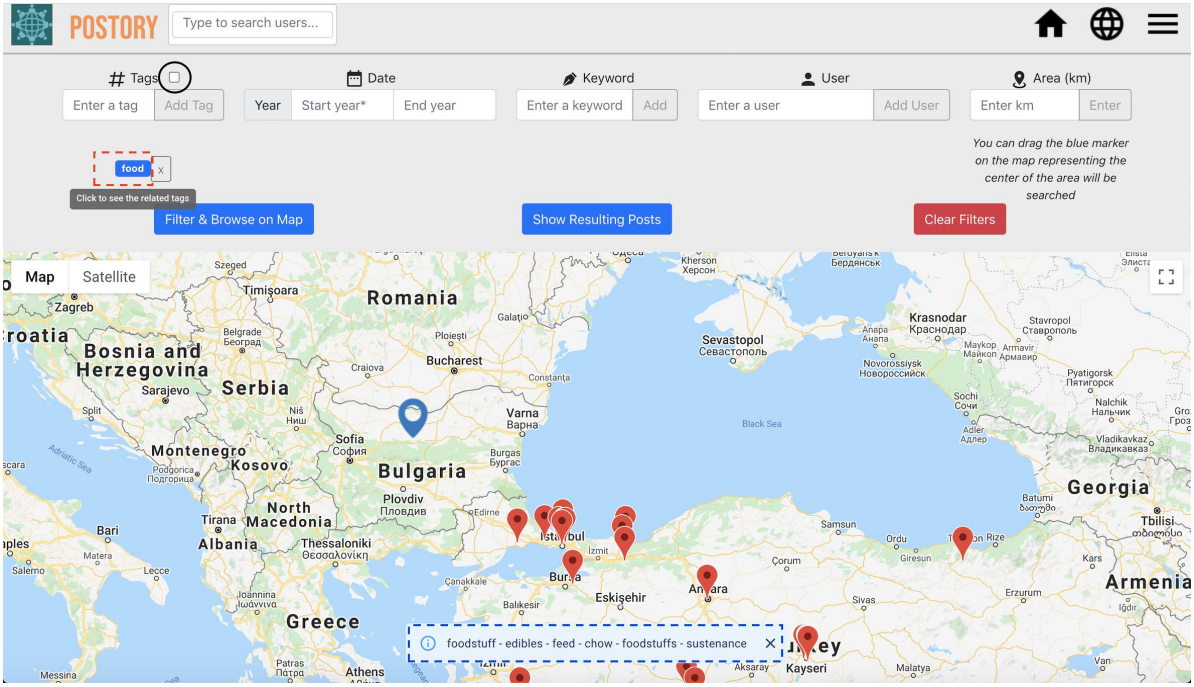


Fig6: Discover Page - 6

When users clicks on the checkbox next to tags section (*see* black circle on *Fig6*), system automatically includes all the related tags to the search.

Clear Filters

Users can clear filters either by deleting all the filters one by one (i.e., clicking the x button near each added filter - *see* red circles on *Fig7*) or by clicking the `Clear Filters` (*see* blue dashed circle on *Fig7*) button and clearing all of them at once.

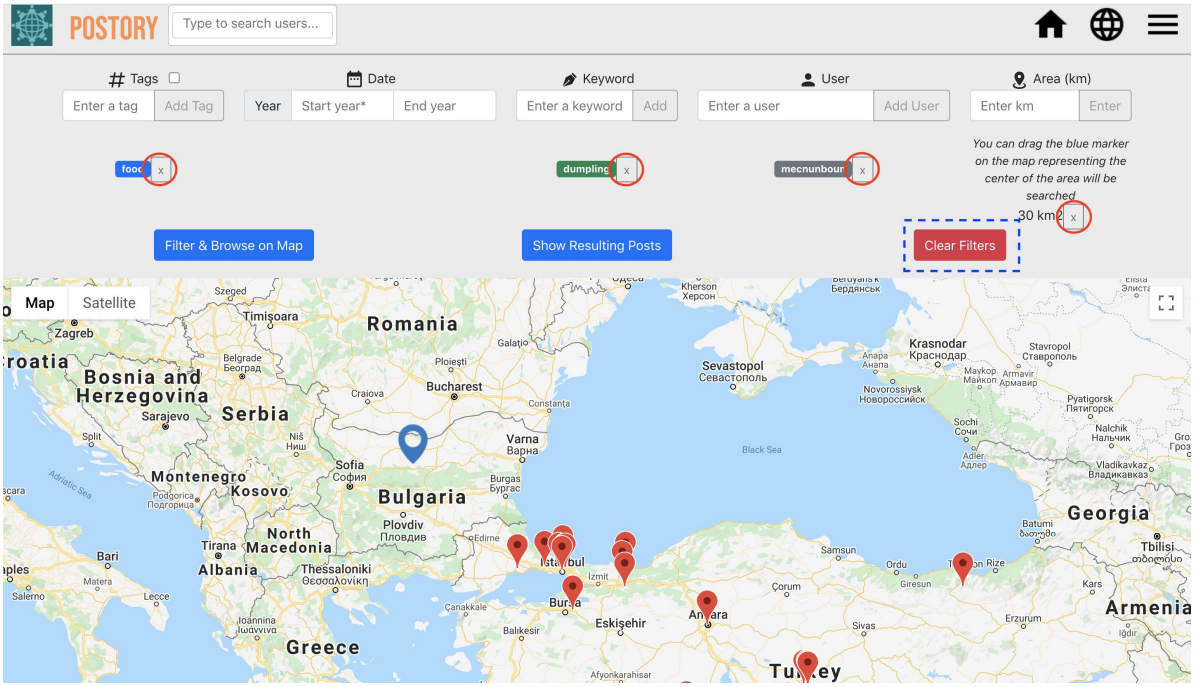


Fig7: Discover Page - 7

Profile Page

The profile page is where users can see information (i.e., username, profile picture, number of followers, number of followings, and number of posts) of public users and private users they follow, and their posts (*see Fig9*).

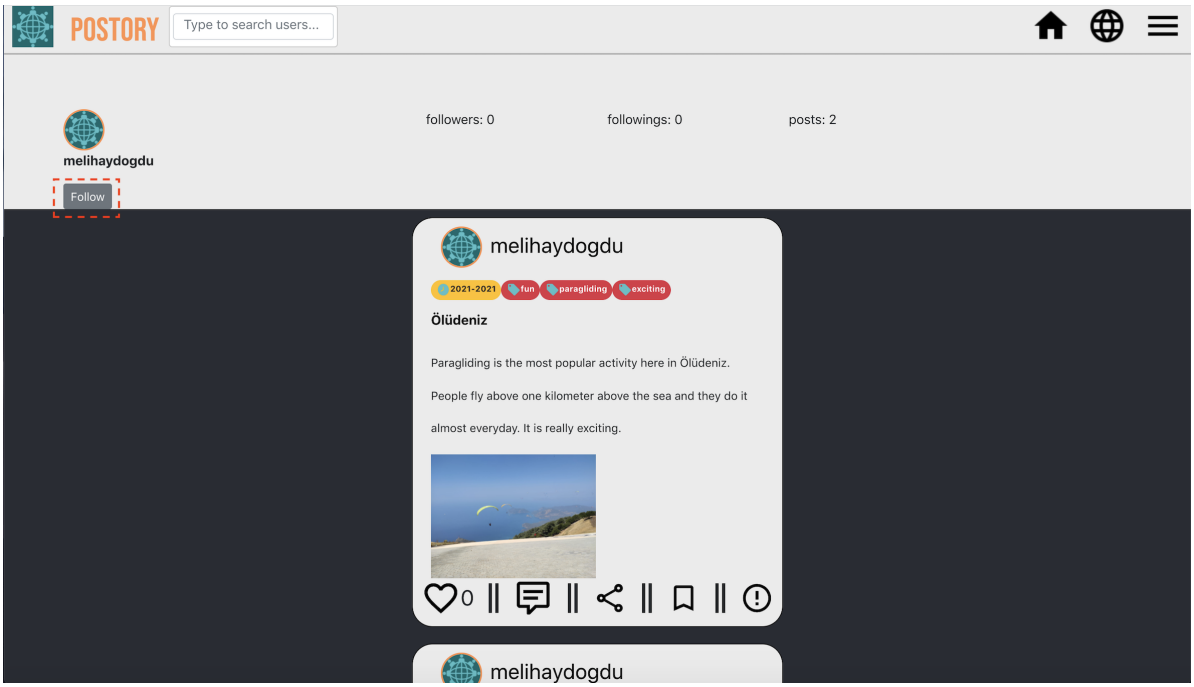


Fig9: Profile Page - 1

Users can interact with users (i.e., follow/unfollow) by clicking the `Follow` button (*see* red dashed button on *Fig9*). If it is a private profile, a follow request will be sent to that user. If a user already follows another one, when she navigates to that user's profile, she sees the `Unfollow` button (*see* red dashed button on *Fig10*)

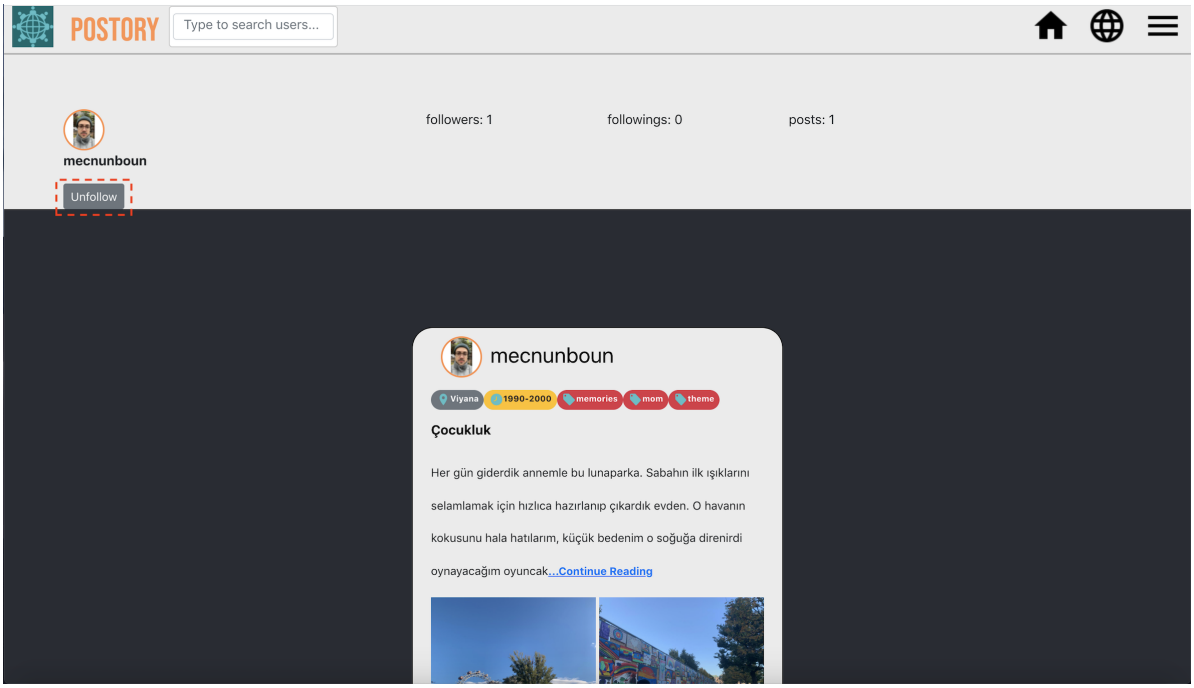


Fig10: Profile Page - 2

When a user is on her own profile page, she does not see a `Follow` or `Unfollow` (*see Fig11*). Users see a `Make my application Private` button (*see* red dashed button on *Fig11*), which makes their account private (i.e., invisible by non-followers) on click.

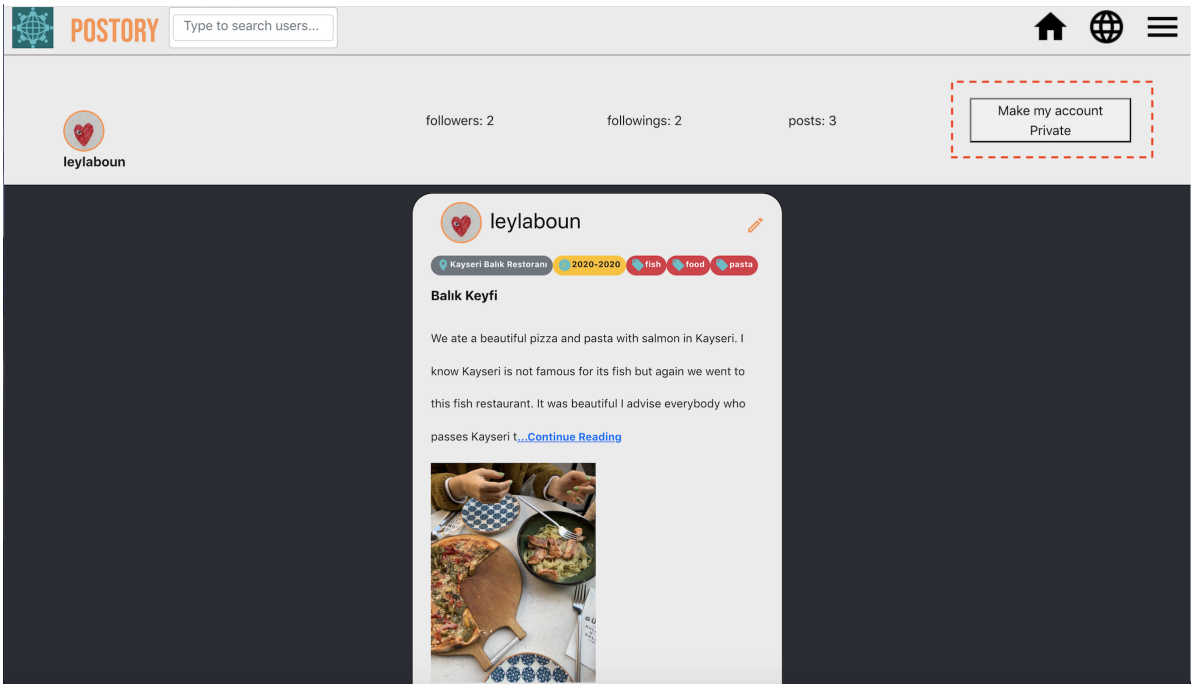


Fig11: Profile Page - 3

Users can change their profile pictures by clicking on their profile pictures. A plus symbol will appear when they move the mouse on the profile picture (*see* the red dashed circle on *Fig12*). Clicking on that plus symbol navigates users to the picture selection interface (i.e., users can select a picture from their device).

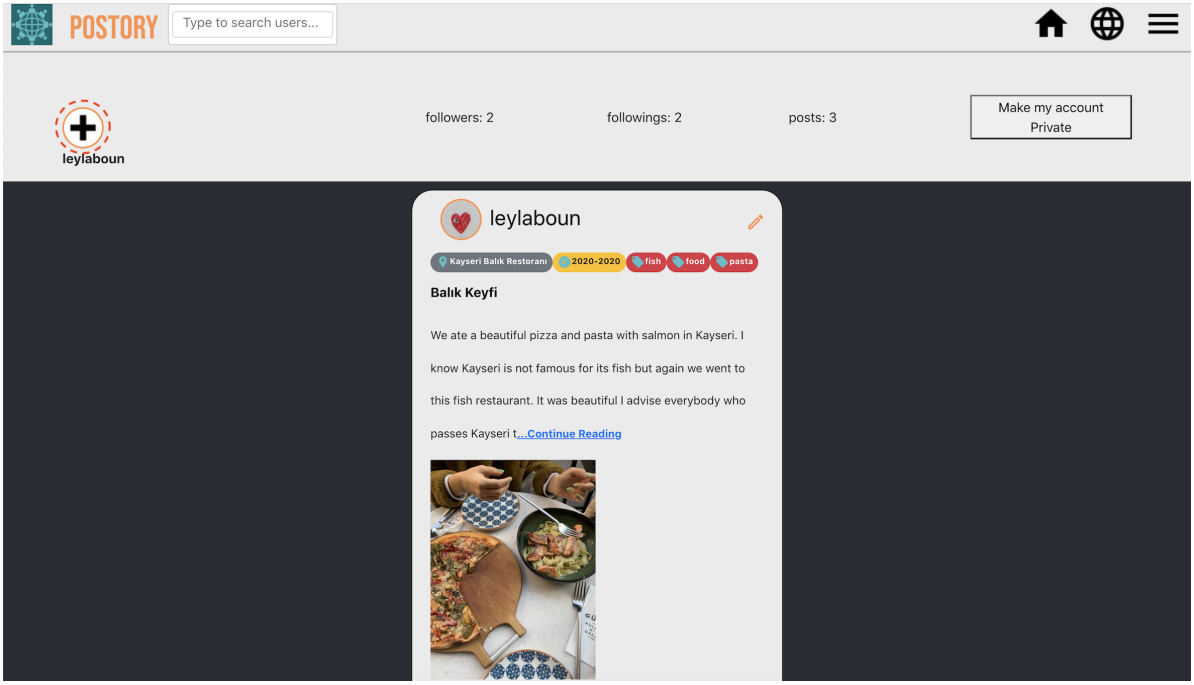


Fig12: Profile Page - 4

View Post Page

View post page is where users can see the details of a post (i.e., geolocations, pictures in high resolution, and comments) (*see Fig13*).

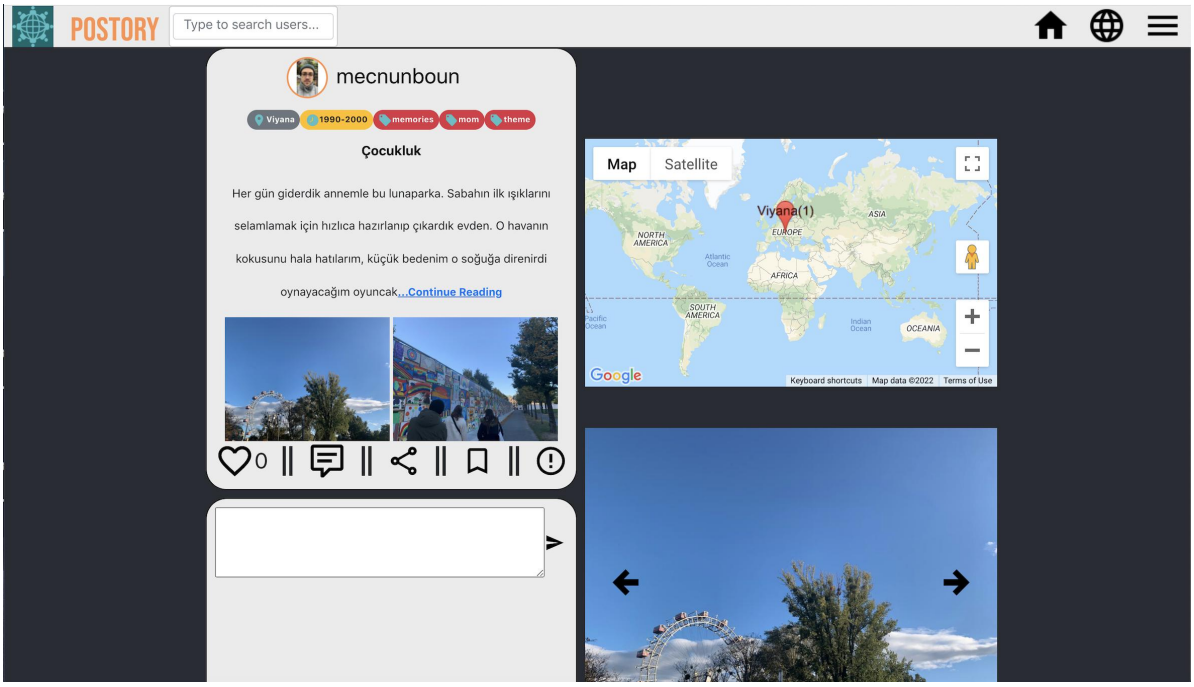


Fig13: View Post Page - 1

Users can see all of the pictures of a post by clicking the arrow buttons (*see* yellow dashed circles on *Fig14*). Also, they can comment on that specific post by entering the comment into the input area and then clicking on the arrow button (*see* red dashed rectangle area in *Fig14*)

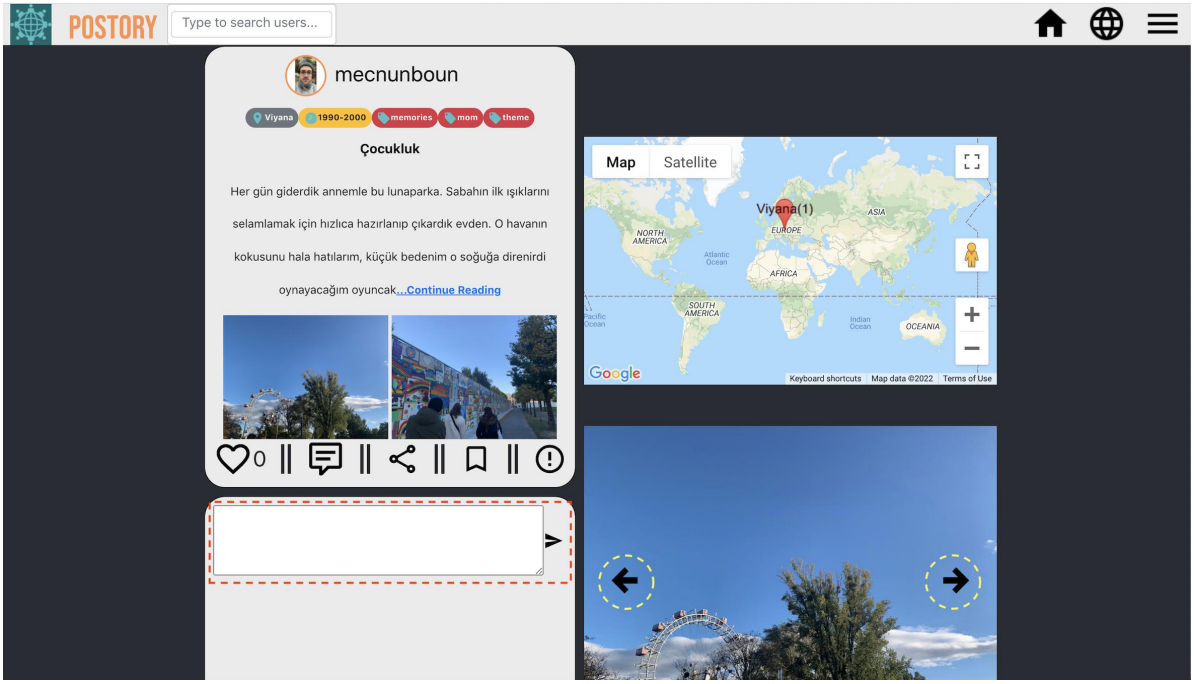


Fig14: View Post Page - 2

An **example** comment action is described below:

1. A user first enters the comment text to the input space and then hits the arrow button next to it (i.e., red dashed rectangle area on *Fig14*). (*see Fig15*)
2. The comment then will appear in the comment section (*see Fig16*)

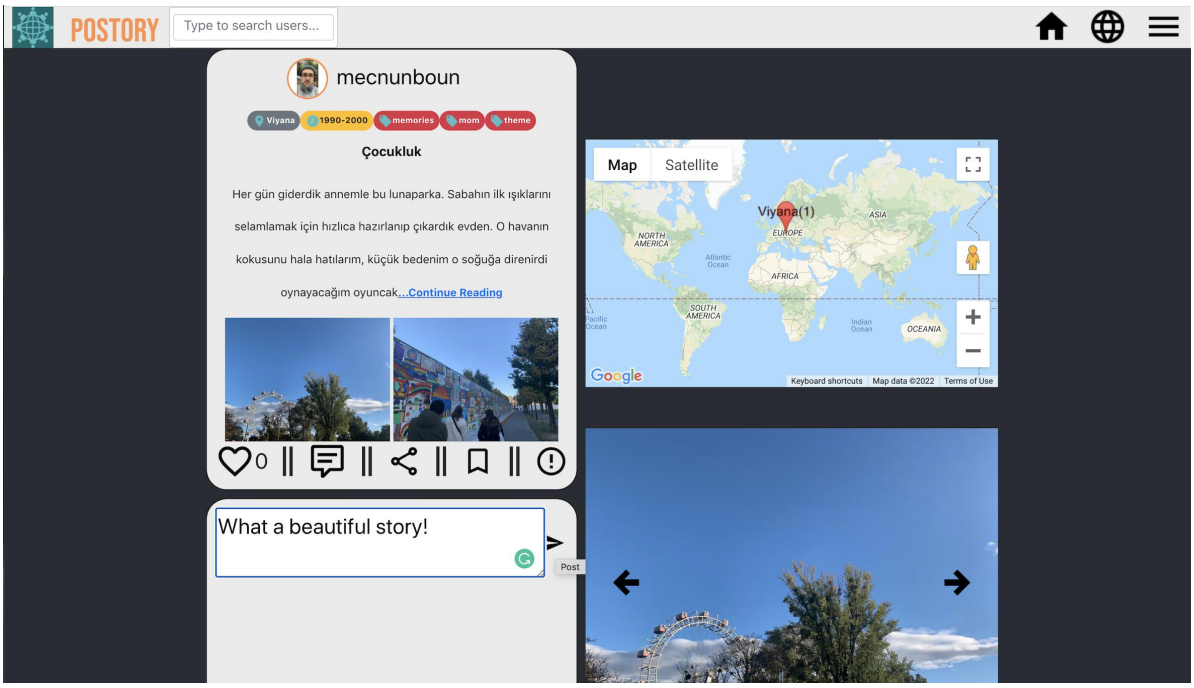


Fig15: View Post Page - 3

qq

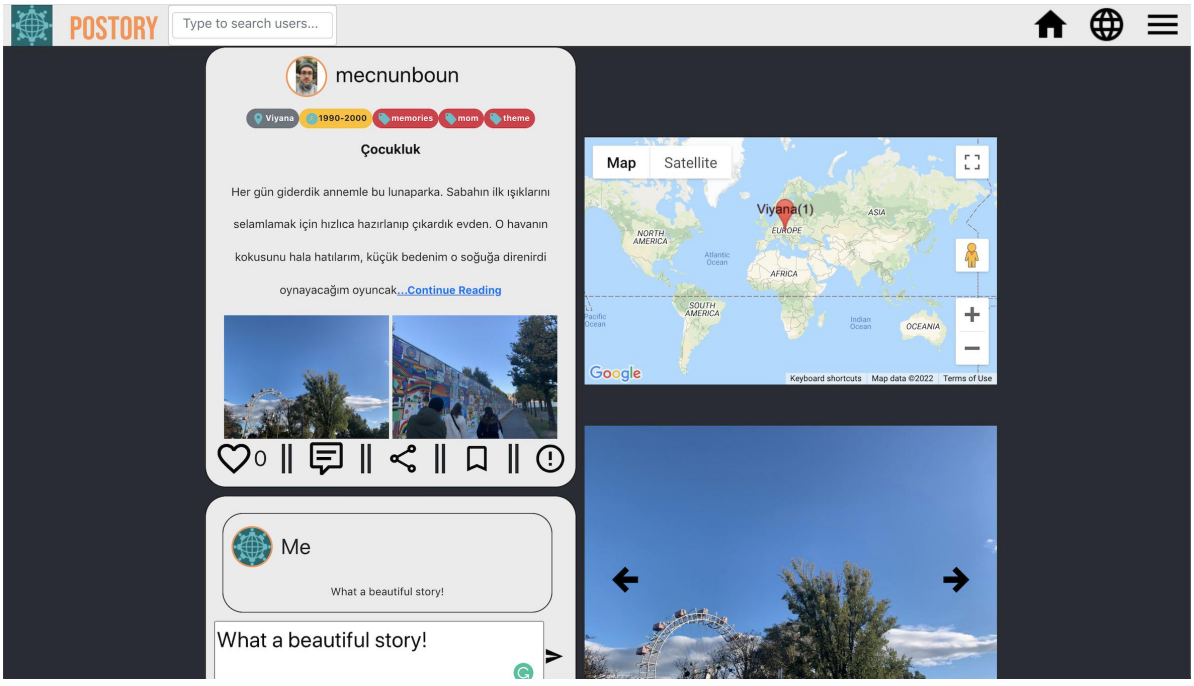


Fig16: View Post Page - 4

Homepage

Homepage is where a logged-in user lands in when he/she opens up the application. In the homepage, the user can browse through posts of followed people, and he/she can travel to other places of the app.

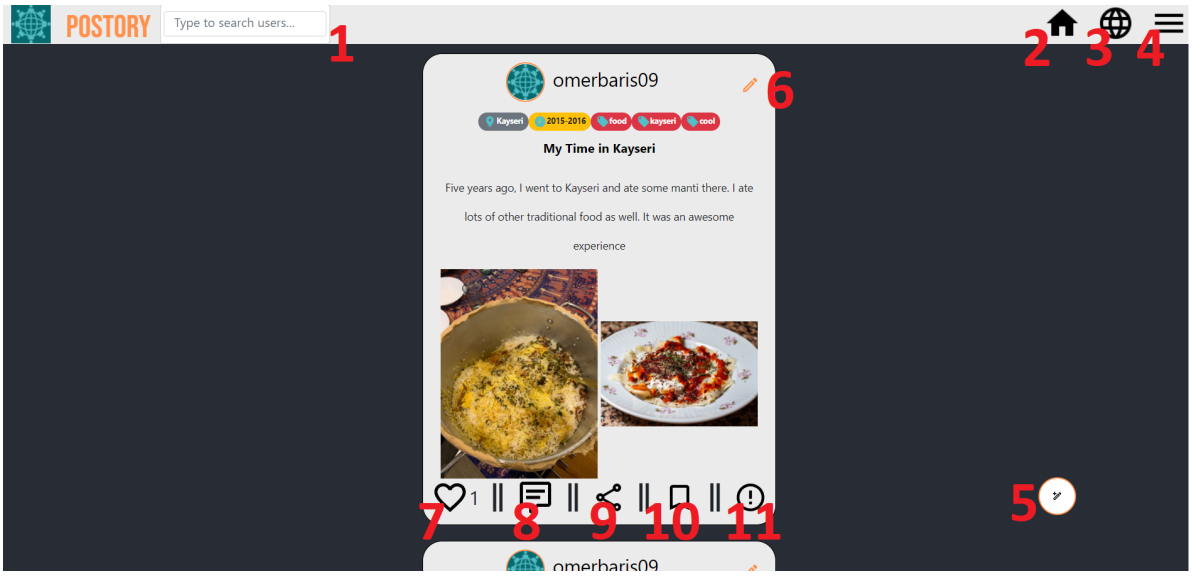


Fig17: Homepage

1. Search Bar: In the search bar, user can search for public or private profiles. When user enters input text to the search bar, user profiles that match the query pops up below the bar.
2. Homepage button: Redirects to Homepage from anywhere else in the app.
3. Discover button: Redirects to the Discover page.
4. Menu: From the menu, the user can further interact with the app and travel to other pages.
5. Create post button: Redirects to Create Post page, the place where a user can enter the input for his/her post and eventually create it.
6. Edit button: Lets user edit a post. A user can edit only his/her own posts.
7. Like button: Likes a post
8. Comment button: Redirects to View Post page where the user can see details of the post and comment on it.
9. Share button: Copies post link to clipboard so that the user can share it with someone else.
10. Save button: Saves a post
11. Report button: Reports a post to the admins.

Create/Edit Post page

Create Post page is the page where a user can enter the input for his/her post and eventualluy create it. The user can enter its title, body, locations, time, tags, and images; and then create it.

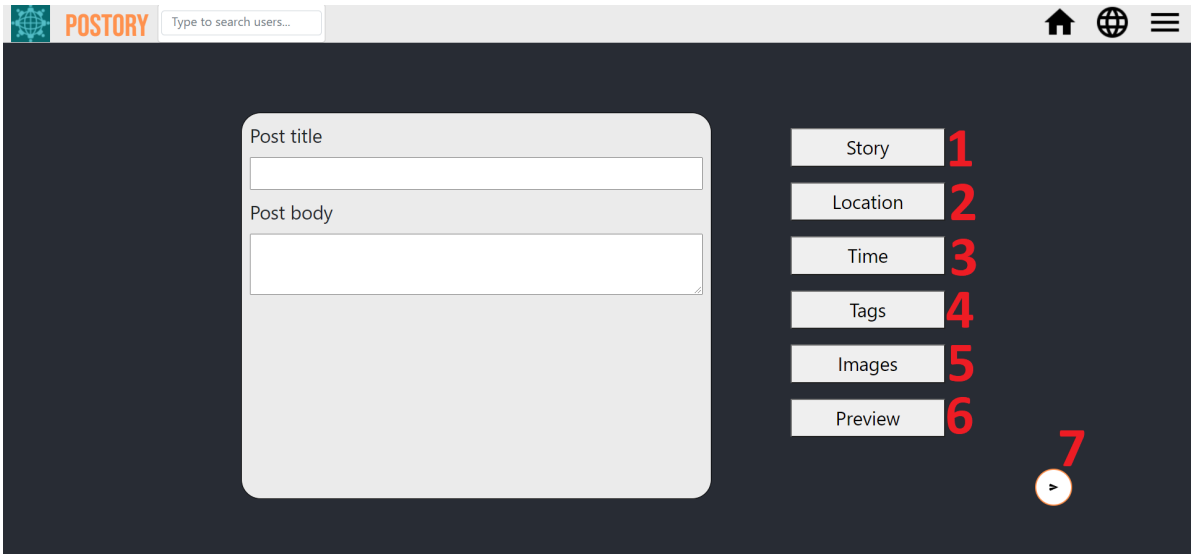


Fig18: Create Post page - Overview

1. Lets user enter the title and the body of the post.

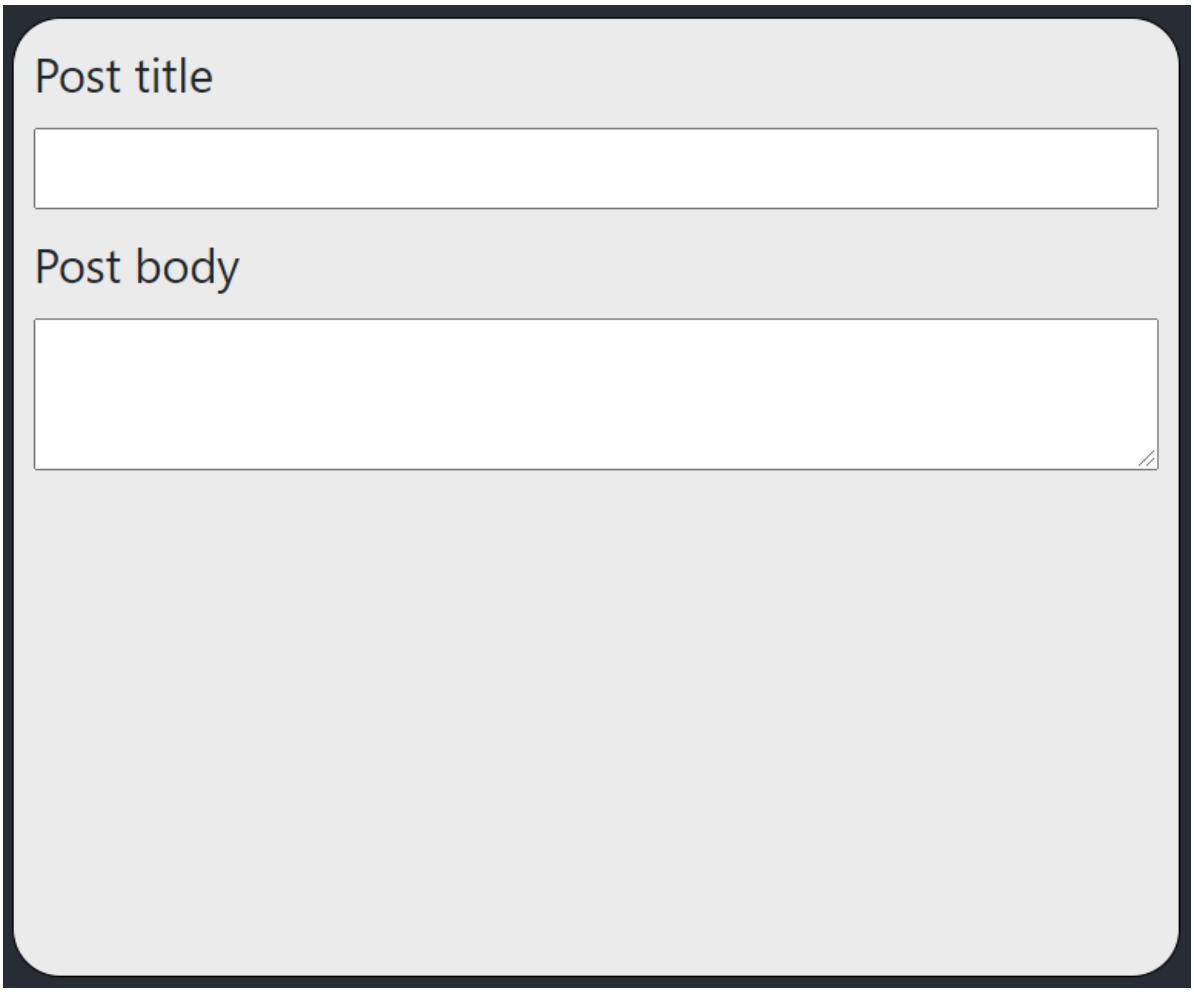


Fig19: Create Post page - Entering title and body

2. Lets user enter locations of the post. User can browse the map and click on locations to add them to their post. User can also enter names for their locations.

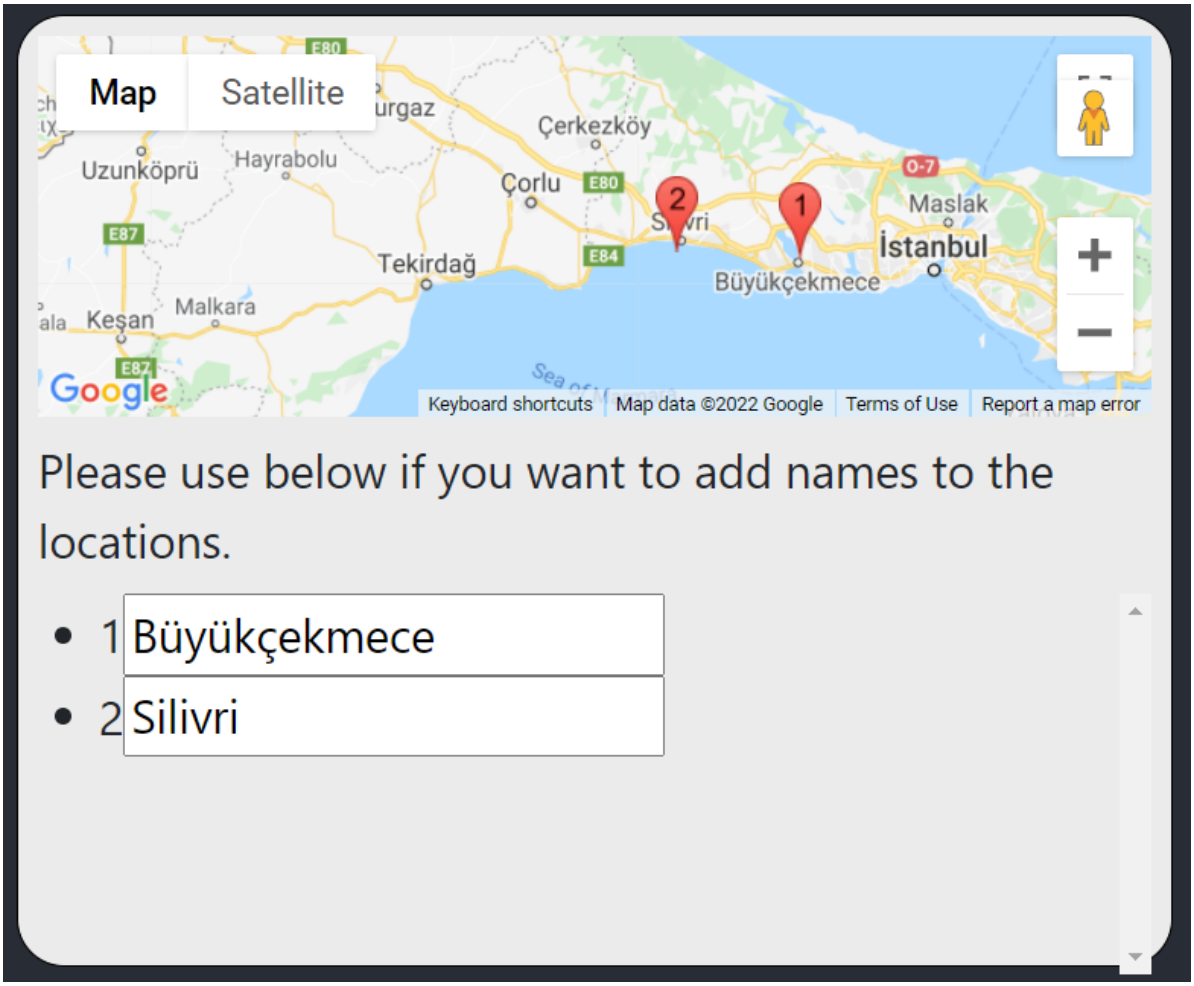


Fig20: Create Post page - Adding locations

3. Lets user enter time of the post. User can add time with the resolution of their choice. More specific time unit becomes available only when the user enters more general time unit (e.g: when user enters the year range of their post as, month range appears). User can also clear their selection using **Clear** buttons.

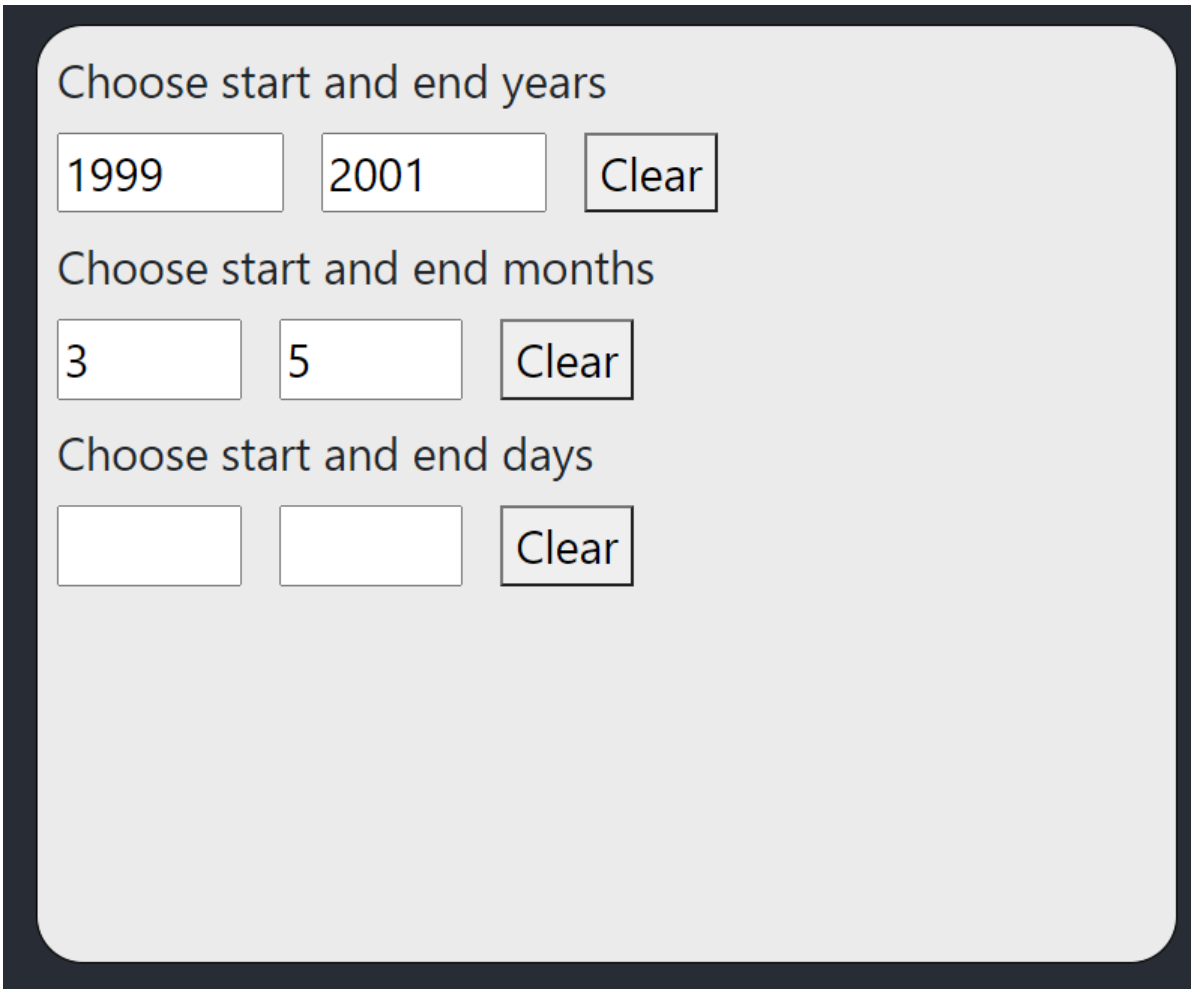


Fig21: Create Post page - Adding time

4. Lets user enter tags of the post. User can enter the text for their tag and click the **Add** button to add the tag. User can also clear their tags by pressing the **x** button near their tags, or by pressing the **Clear All** button to clear all tags.

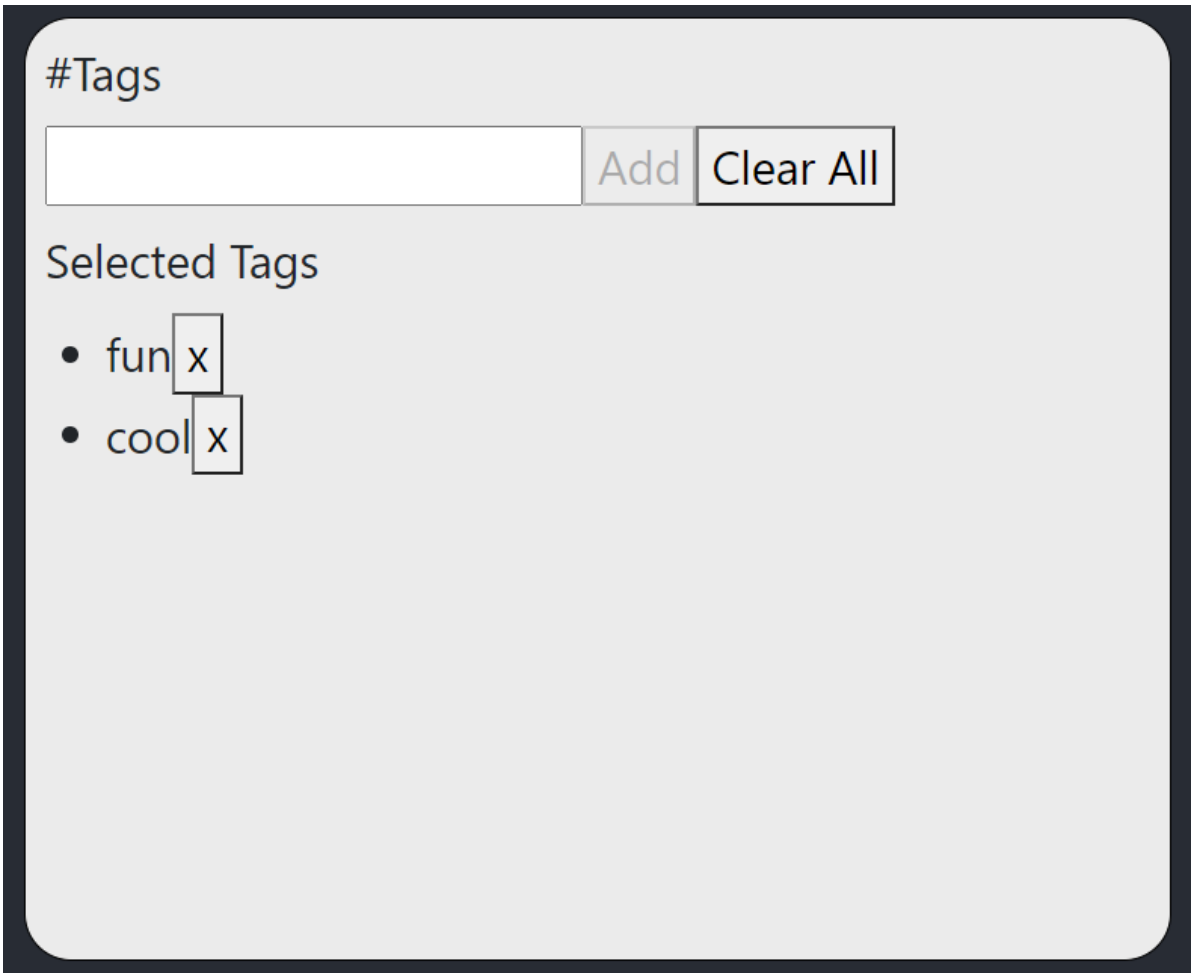


Fig22: Create Post page - Adding tags

5. Lets user add images to the post. User can click the **Choose Files** button to choose and enter files to their post.

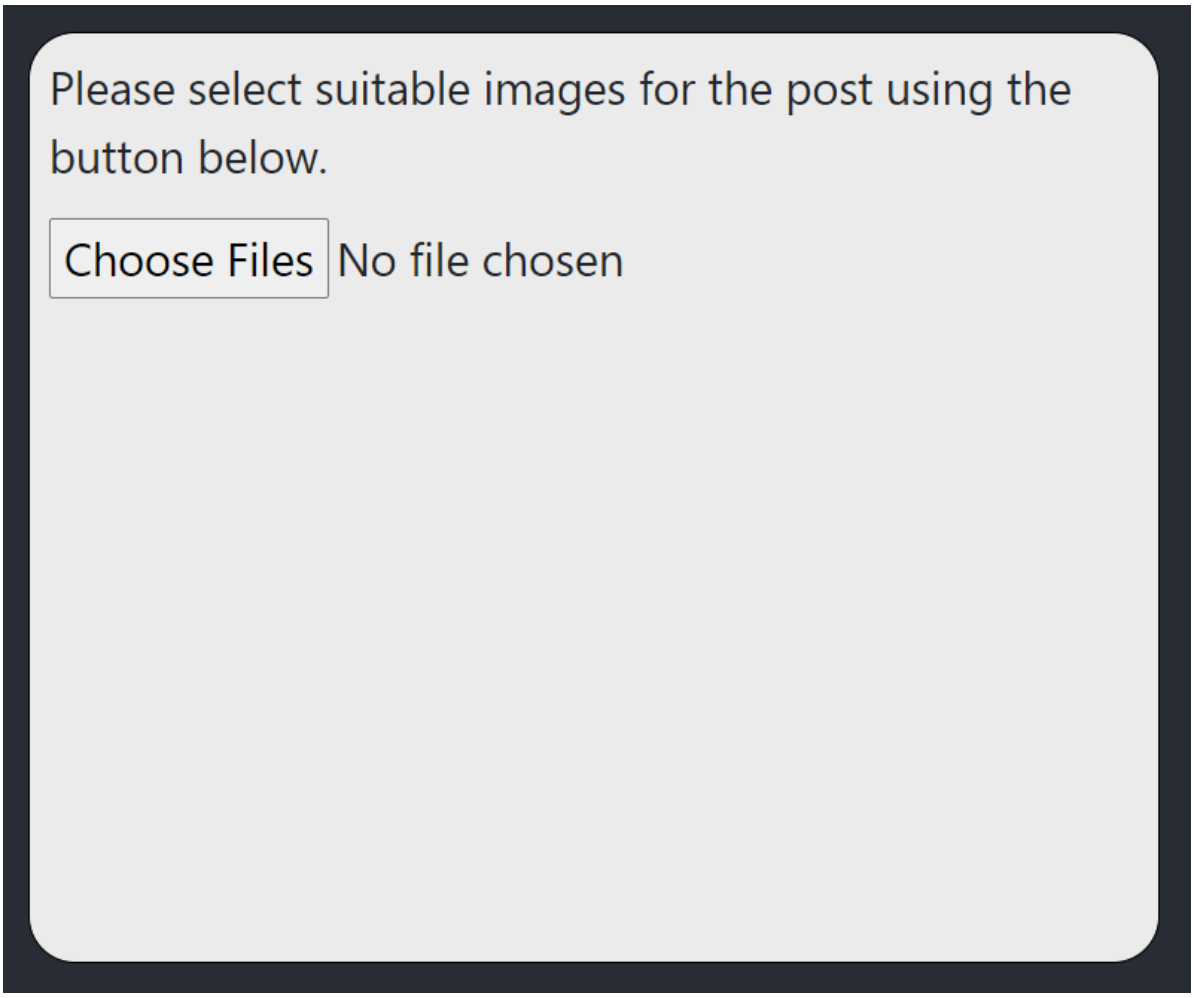


Fig23: Create Post page - Adding images

6. Lets user see preview of the post before creating it.
7. Create Post button: Creates the post with the entered input.

Editing a post

Editing a post can be done by clicking the pencil button while viewing a post on View Post page or Homepage. Editing a post brings user the same page as creating a post. From there, the user can edit its title, body, locations, time, tags, and images; and then re-post it

Activity Stream page

Activity stream page lets a user see their own activities, activities of followed people, activities of all public accounts, and follow requests.

Fig24: Activity Stream page

1. Own tab: Shows activities of the currently logged in user
2. Followed tab: Shows activities of followed people.
3. All tab: Shows activities of all public profiles.
4. Follow Requests tab: Shows incoming follow requests to the currently logged in user. There the user can see the incoming follow requests and accept/reject them.
5. Activity: An example activity. Actor and object of the activity are interactable so that the user can click and see the actor or the object of an activity.

Android

Sign In page

Sign In page lets user sign in to Postory with their e-mail and password.

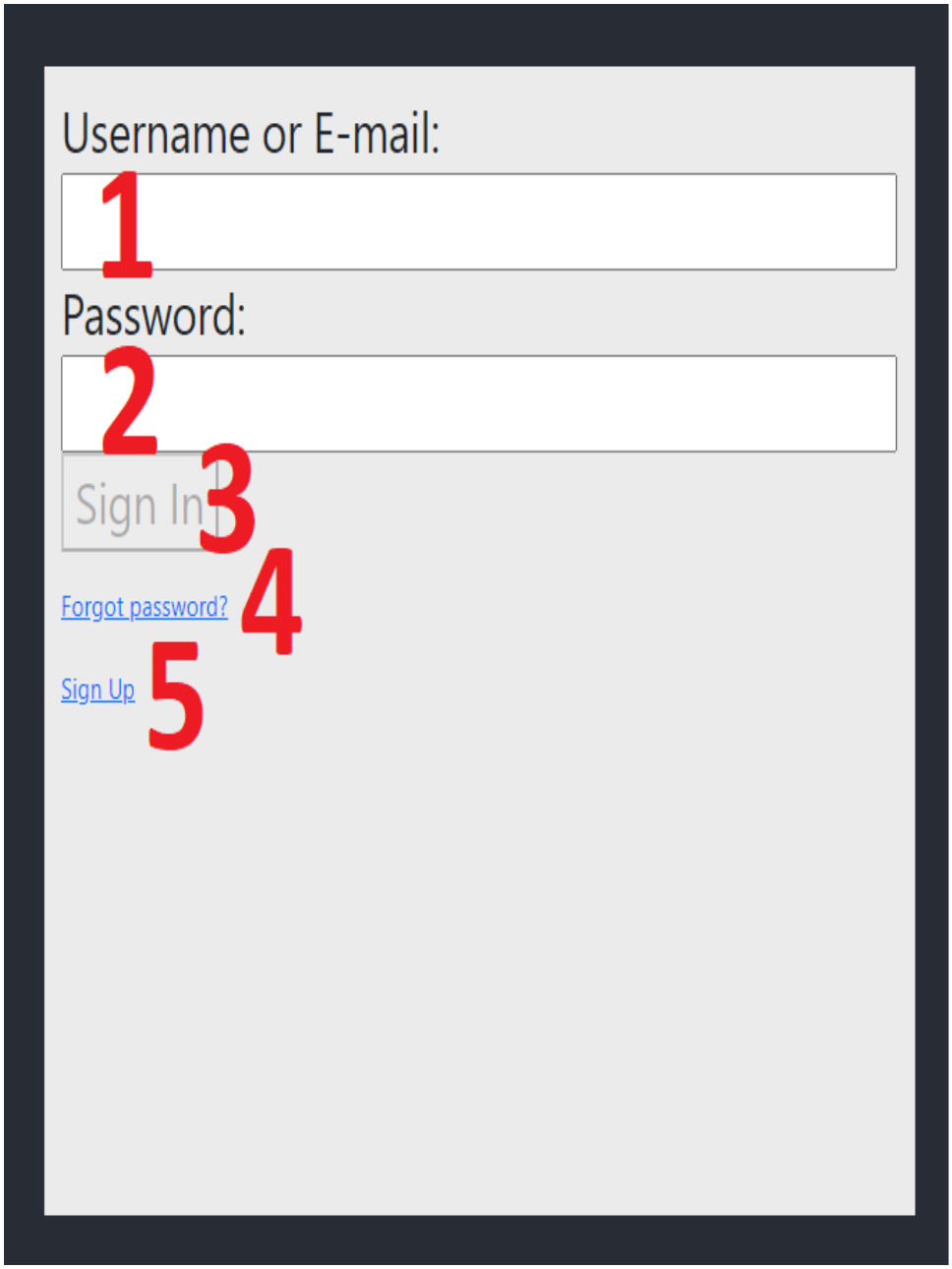


Fig25: Sign In page

1. E-mail/Username field: Lets user enter their e-mail
2. Password field: Lets user enter their password
3. Sign In: Lets user submit their sign in with the entered info
4. Forgot Password button: Prompts user for their e-mail and initiates a password reset request.
5. Sign Up button: Redirects to Sign Up page

Sign Up page

Sign Up page lets people sign up to Postory with their information. It also checks for validity of the entered input.

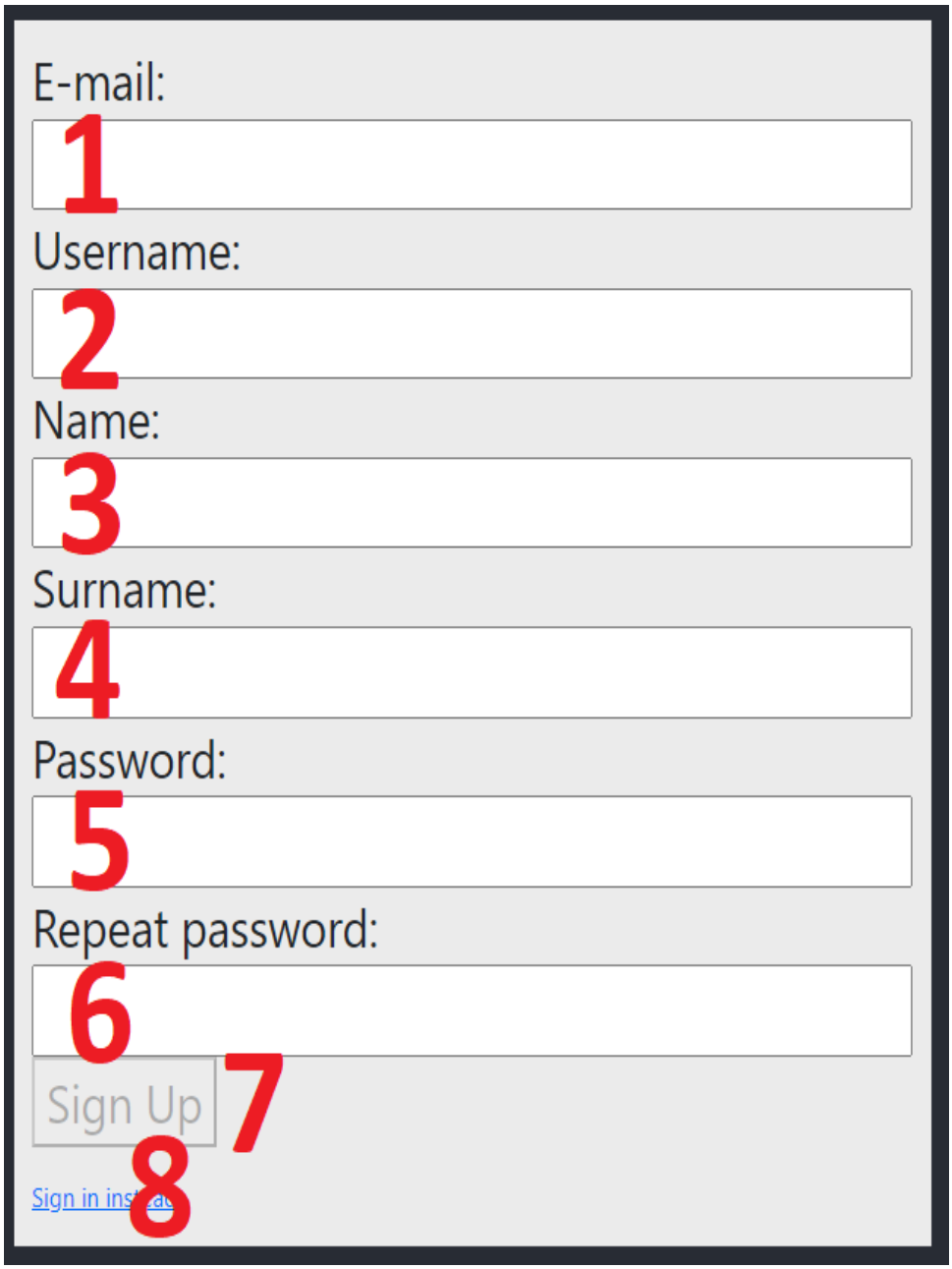


Fig26: Sign Up page

1. E-mail field: Lets person enter their e-mail for the new account
2. Username field: Lets person enter their username for the new account
3. Name field: Lets person enter their name for the new account.
4. Surname field: Lets person enter their surname for the new account.
5. Password field: Lets person enter their password for the new account.
6. Repeat password field: Lets person enter their password confirmation.
7. Sign Up button: Submits the sign up with the entered information.
8. Sign In button: Redirects to Sign In page.

Discover Page

Filter and Explore

On the discover page, users are able to see their own posts, posts of people they follow along with posts from users whose profiles are not private. Posts are denoted with a little red marker on the map, stating their locations. Below is a figure of an examplary discover page. (see Fig1).

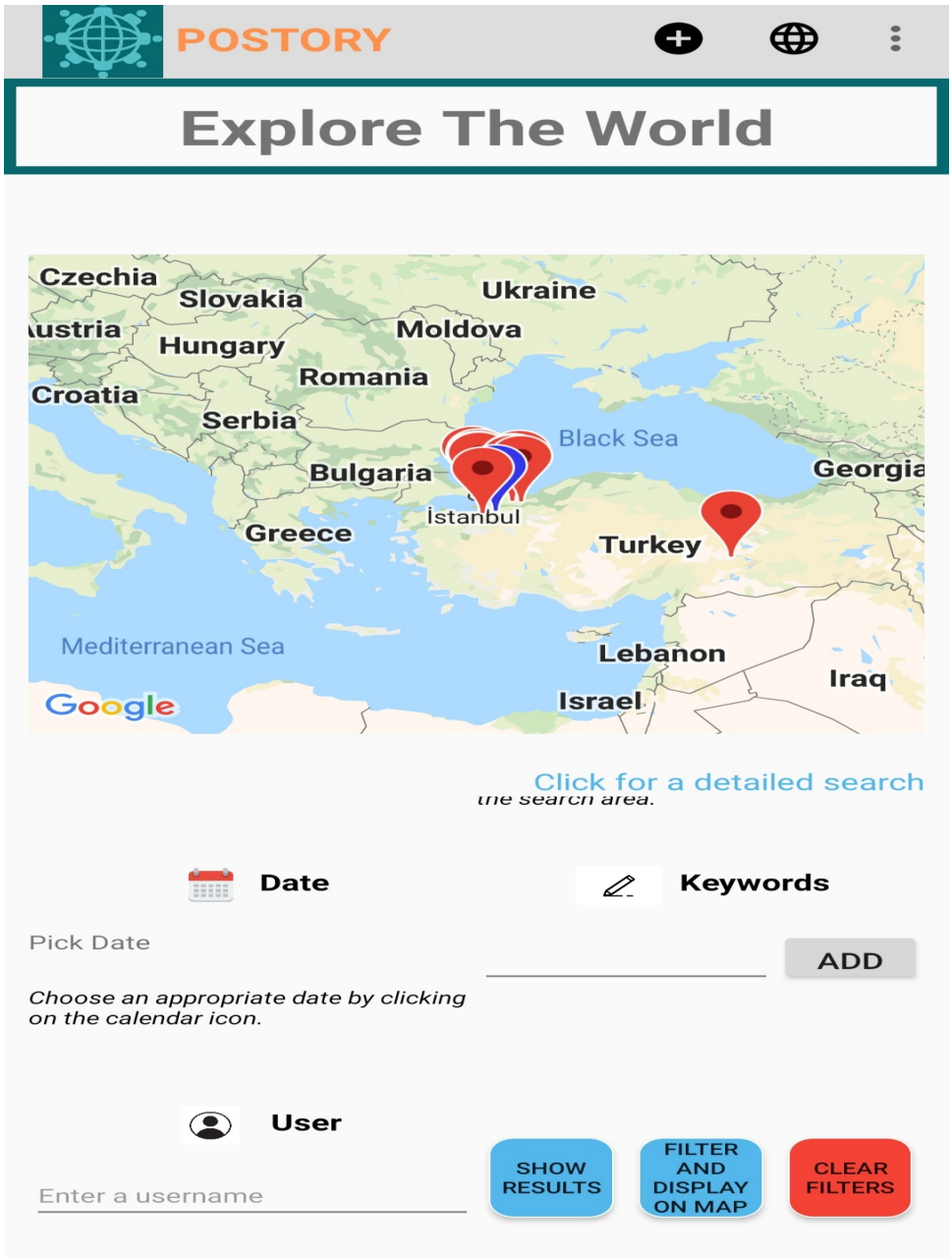


Fig1: Discover Page - 1

An example discovery page can be seen above (see *Fig1*). Users can specify their filters below the map. Here, users can enter tag, date, keyword, user, and area to filter the posts. After specifying the filters of choice, users can see the resulting posts either on the map (i.e., by clicking the `Filter and Display on Map` button) or on a different page (i.e., by clicking the `Show Results` button).

An **example sequence** of filtering is explained below step by step:

1. Here, the user first enters a tag `fun` into `Enter a tag` input space, clicks on `Add`, and clicks on the `Filter and Display on Map` button (see red dashed button on *Fig2*). The resulting markers represent the posts containing the tag `fun`.

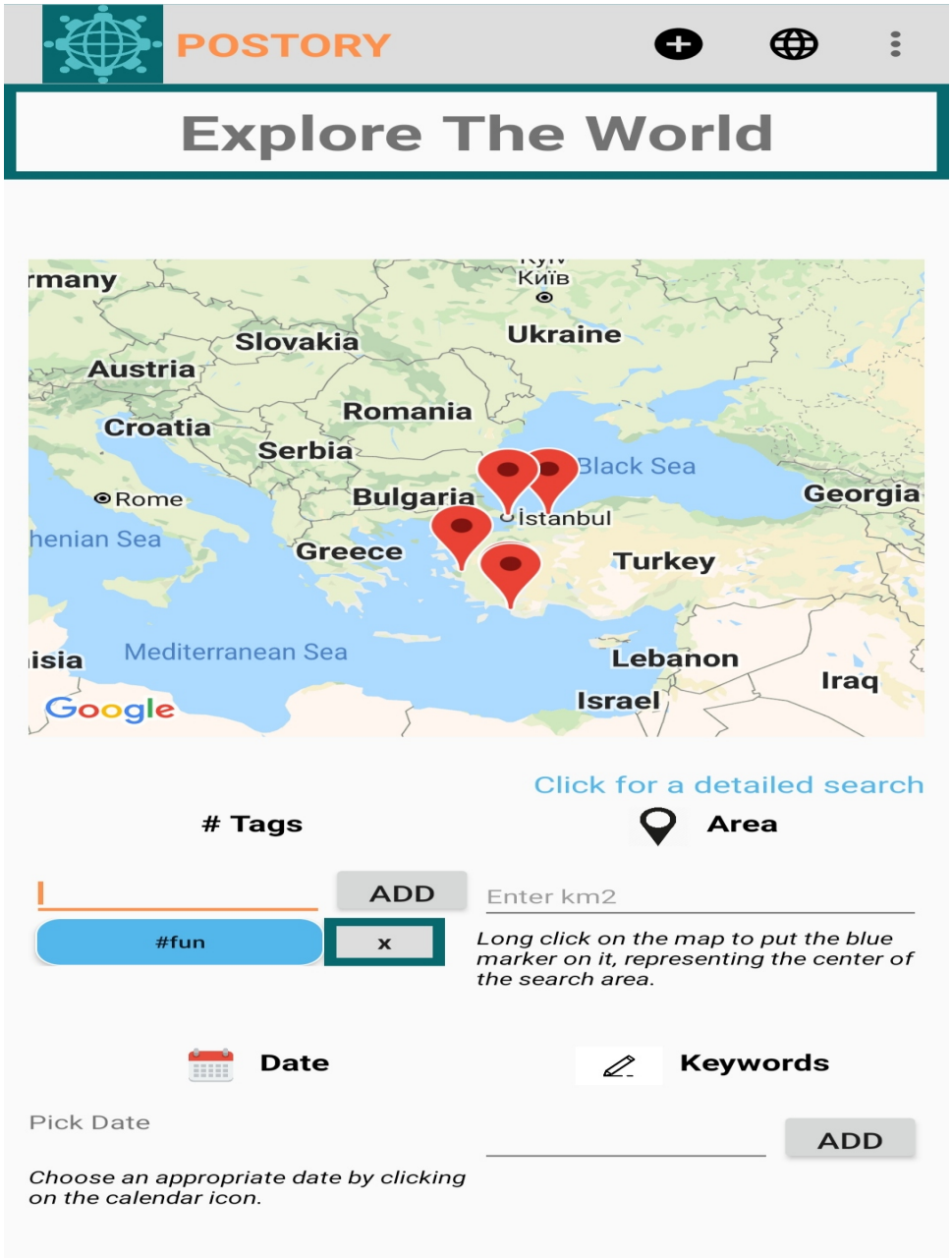


Fig2: Discover Page - 2

2. Later the user filters the results by specifying an area of choice. Here, the user first puts a blue marker on the map by long clicking on it (*see* the blue marker on *Fig1*) which represents the center of the area that is specified. Then the user enters `100` into `Enter km2` input space and clicks on `Enter`.(*see Fig3*)

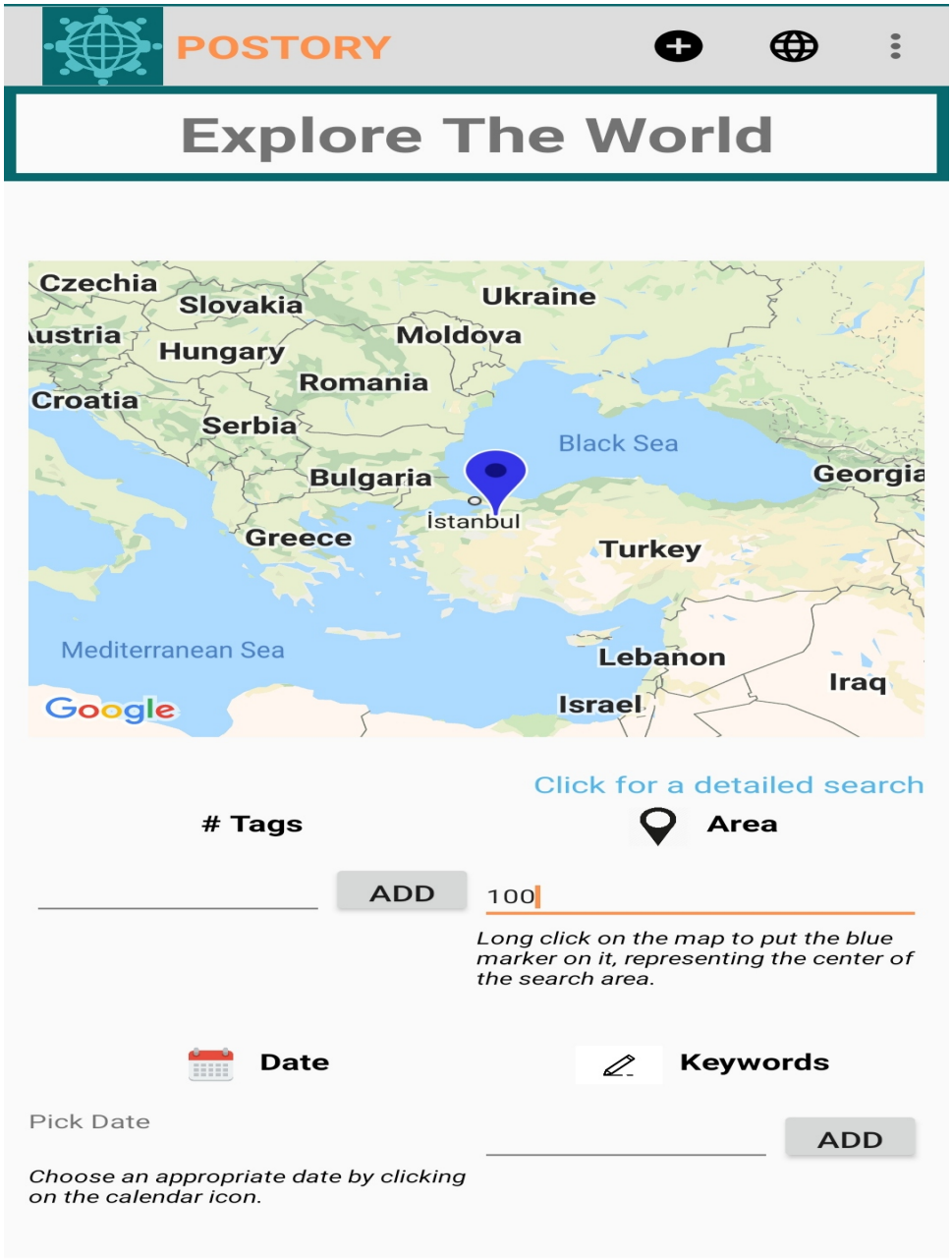


Fig3: Discover Page - 3

4. Finally, the user clicks on the `Filter and Display on Map` button again and sees the resulting posts on the map. (*see Fig4*)

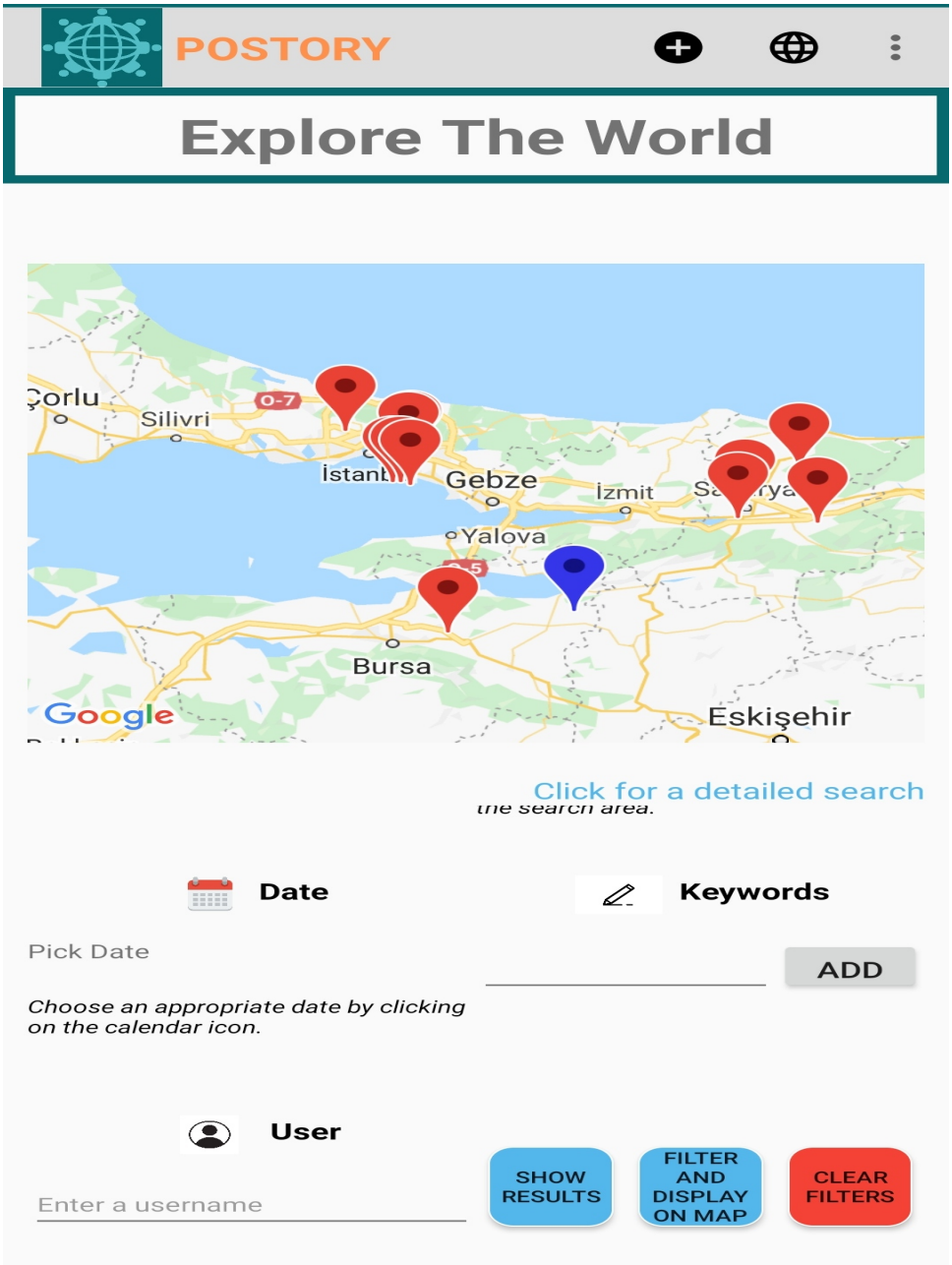


Fig4: Discover Page - 4

Users can click on the red markers, representing the posts, and see the brief information about the posts (i.e., its title and author) in a pop-up (*see Fig4*). They can click on the pop-up to see the full posts on a View Page (*see* View Post Page)

Users can also clear all the filters, using the red `Clear Filters` button (*see Fig4*).

Users are able to see the resulting posts after applying the filters on a different page, in order to see the full posts like in the Home Page (*see* Home Page), by clicking the `Show Results` button (*see* button on *Fig4*). They will be directed to a page like a Home Page, to see the resulting posts of a filter operation (*see Fig5*).

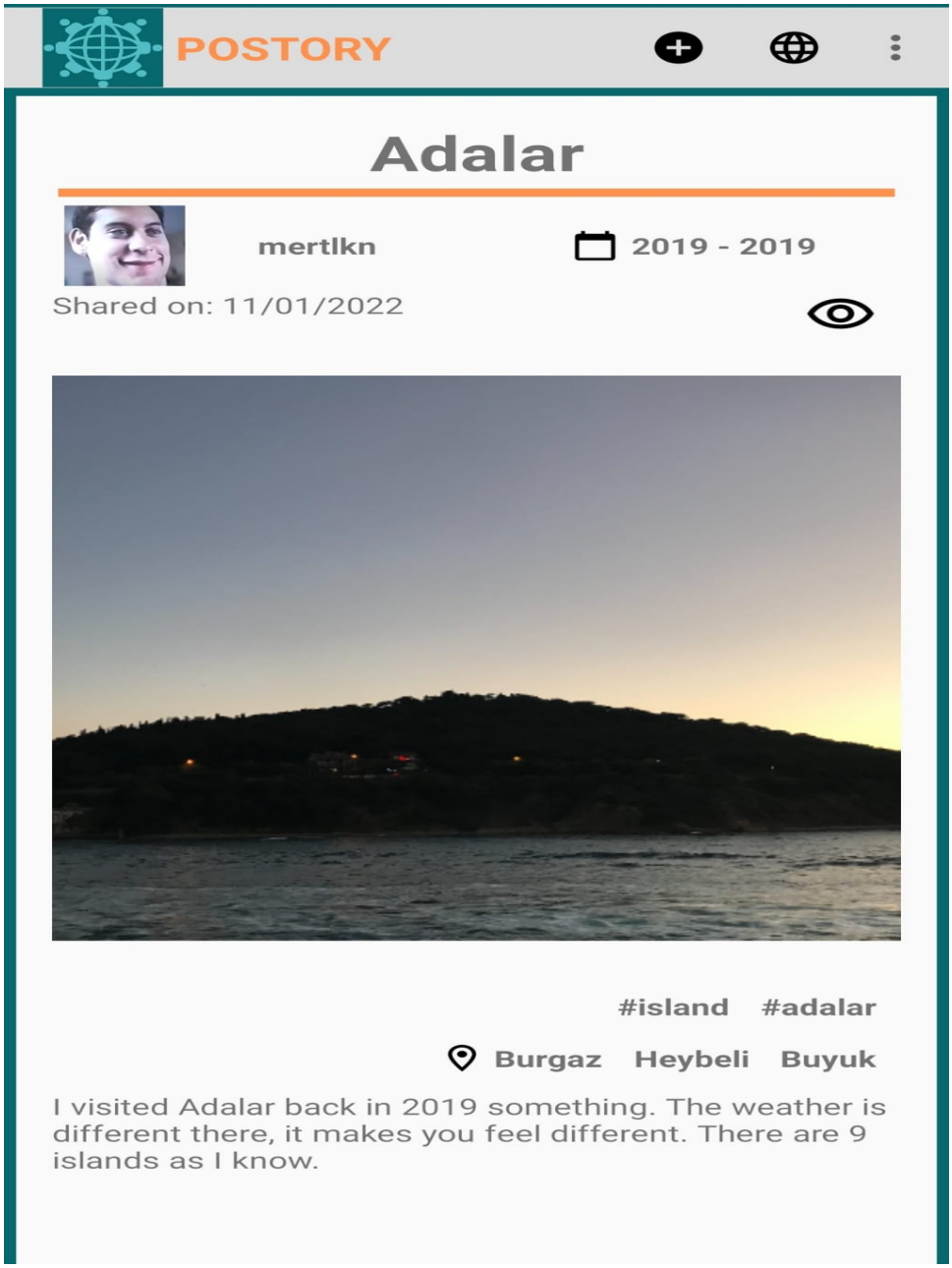


Fig5: Discover Page - 5

Related Search

When the users specifies a tag, the app also includes the tags related to the post, by making use of a WikiData based service. This service returns all related tags, and they are included in the search. These can can be seen below the page, when the user clicks on it. (*see Fig6*)

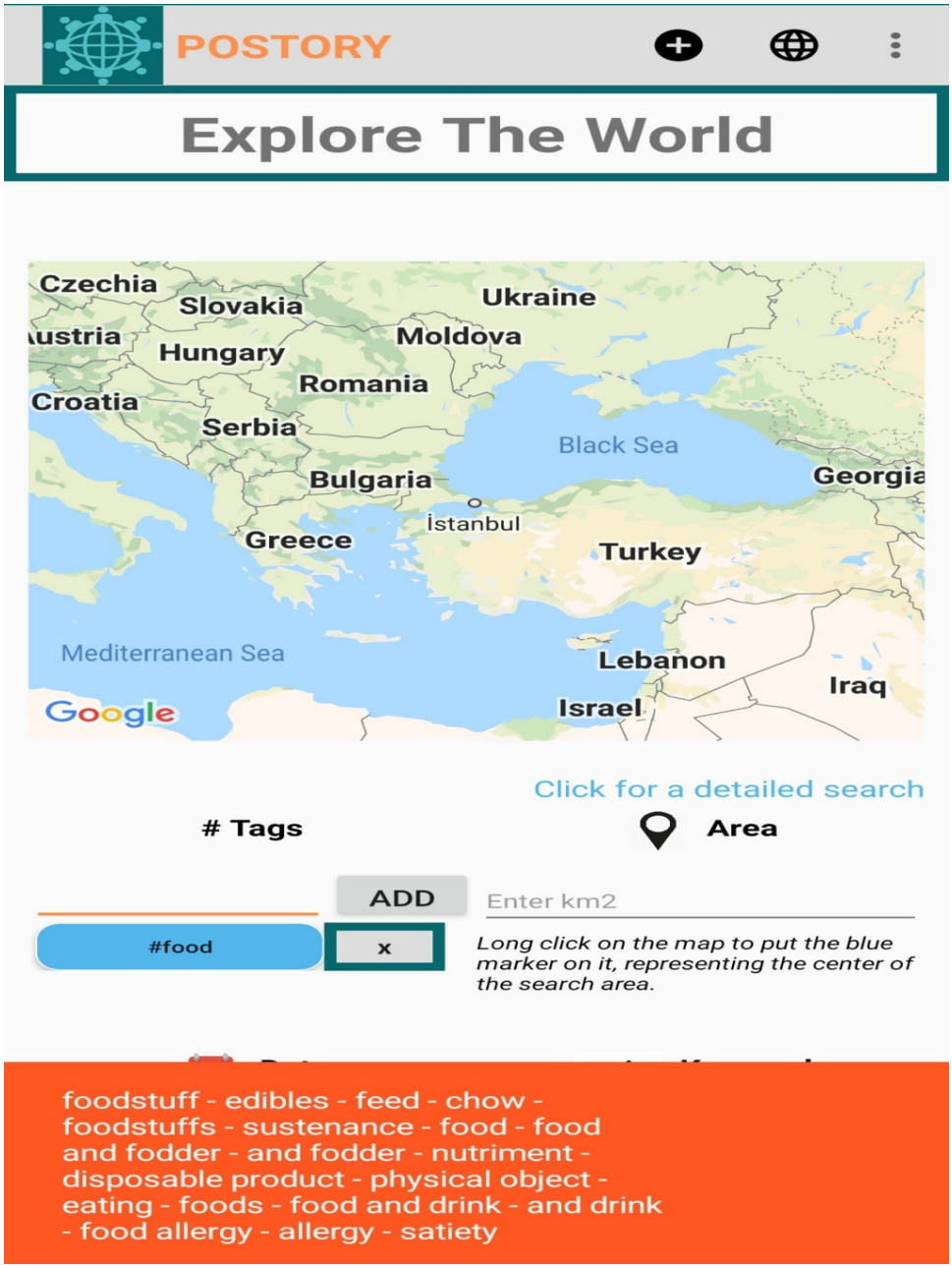


Fig6: Discover Page - 6

Clear Filters

Users can also clear all the filters, using the red `Clear Filters` button (*see Fig4*). The default state of the discovery page is shown when this is done and all the filters are removed.

Profile Page

The profile page is where users can see information (i.e., username, profile picture, number of followers, number of followings, and number of posts) of public users and private users they follow, and their posts. There are differences between the profile page that is seen by the owner and profile page seen by the others.

Profile Page of Other Users

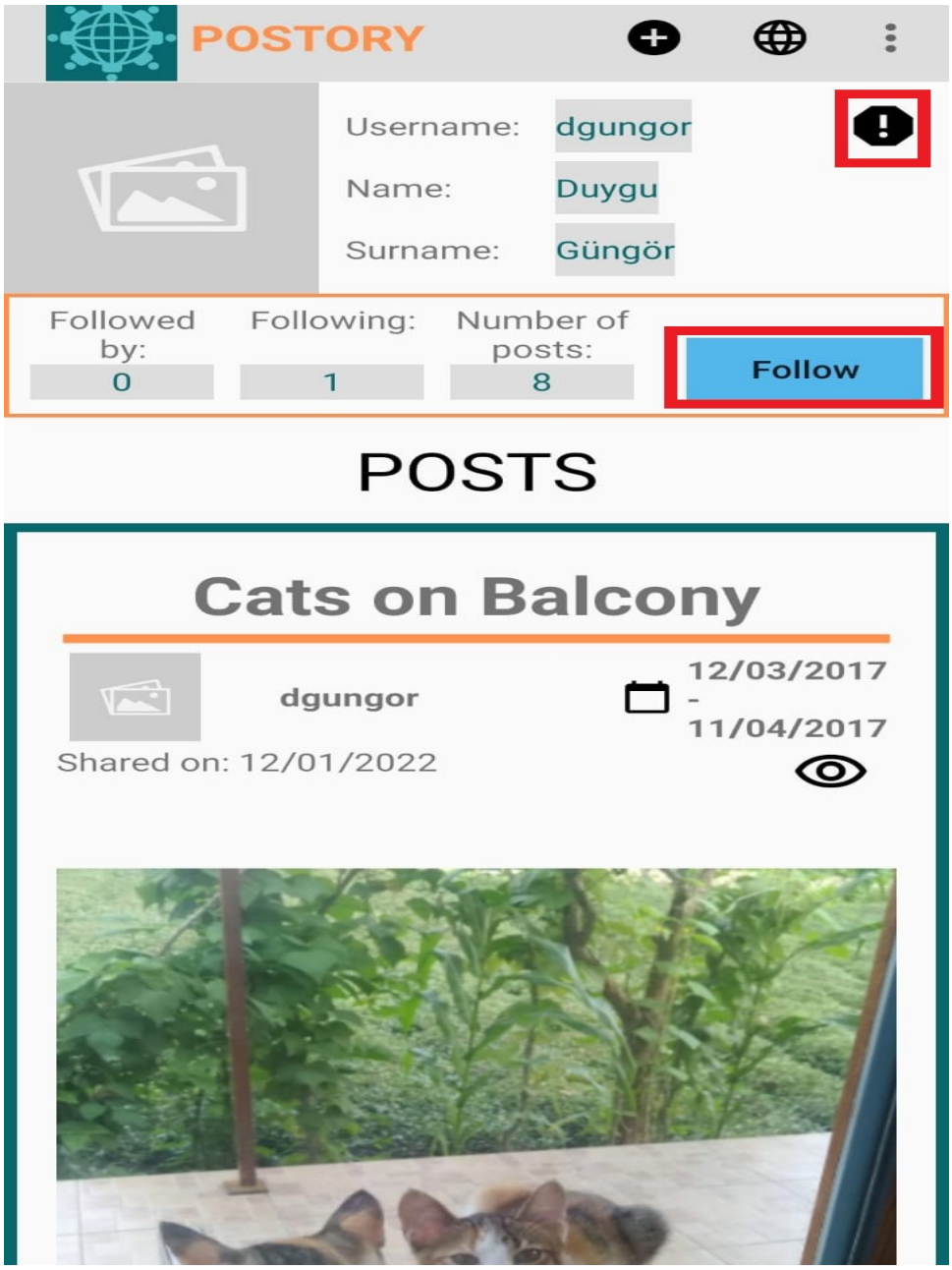


Fig7: Profile Page - 1

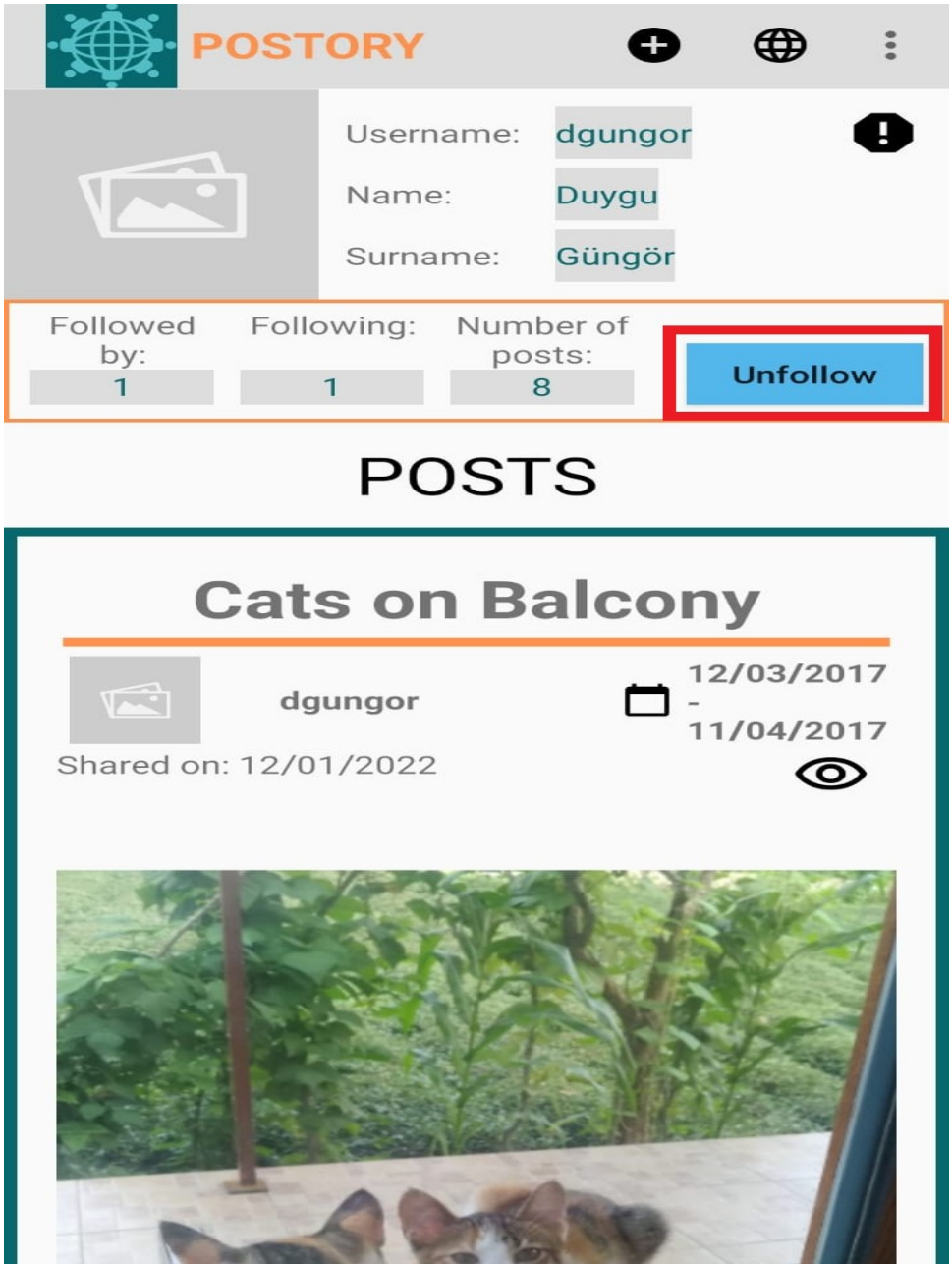


Fig8: Profile Page - 2

Users can interact with users (i.e., follow/unfollow) by clicking the `Follow` button (*see button in red square on Fig7*). If it is a private profile, a follow request will be sent to that user. If a user already follows another one, when she navigates to that user's profile, she sees the `Unfollow` button (*see* button in red square on *Fig8*)

The user can also report other users from their profile page. (*see Exclamation mark in red square on Fig7*)

Profile Page Seen by the Owner

When a user is on her own profile page, she does not see a `Follow` or `Unfollow` (*see Fig9*). The user sees a switch to adjust the privacy option of
(*see Switch inside red square*), which makes their account public/private on click.

There is also a button to see the pending follow requests that are received by the user (*see 'Requests button' inside red square*), which leads to the follow requests page.

Fig9: Profile Page - 3

View Post Page

In the view post page, users can see more information (location, comments, likes) about the post along with the information they saw on the main page. They can like a post and comment on it or report it (*see Fig10, Fig11, Fig12, Fig13,Fig14*).

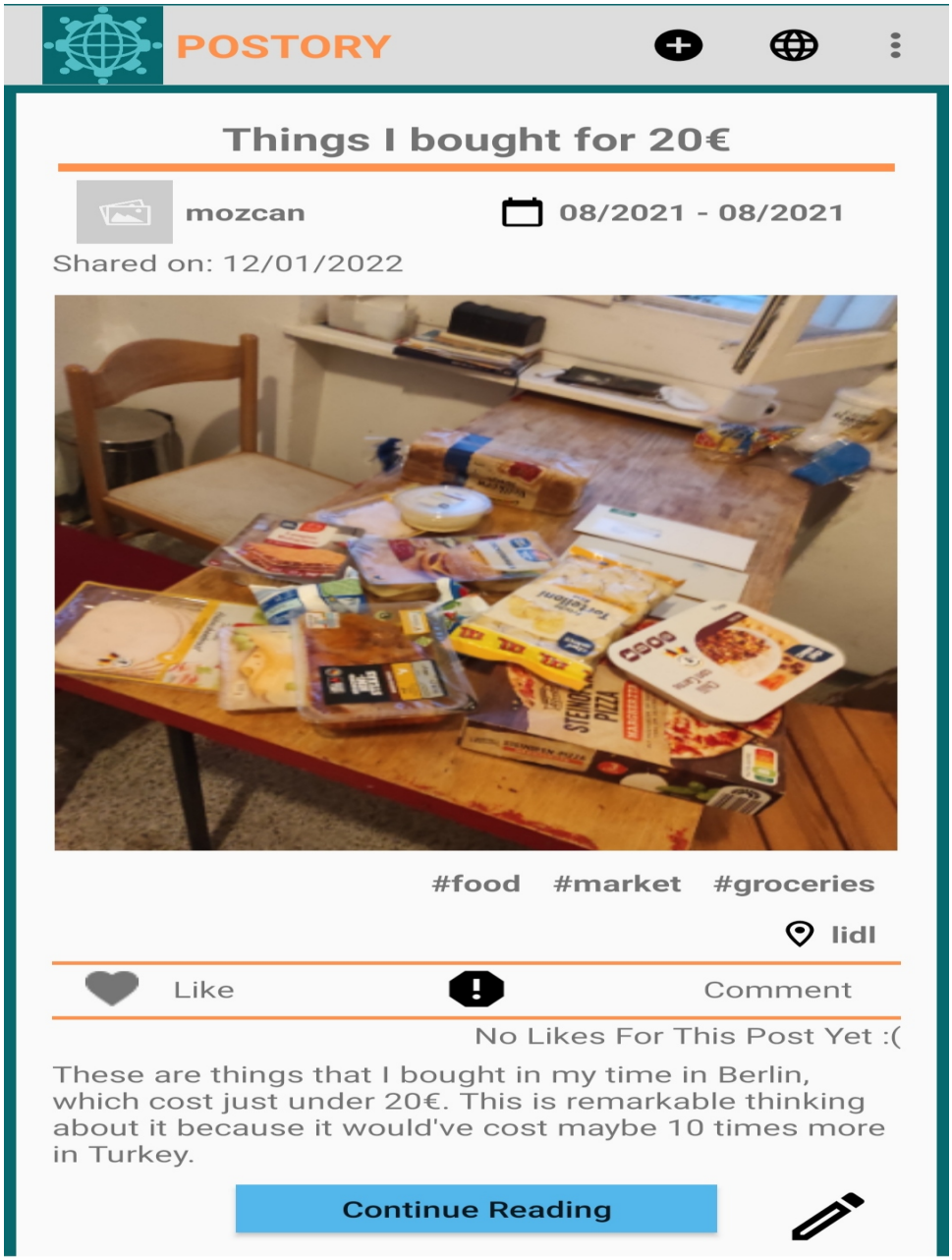


Fig10: View Post Page - 1

Users can see the rest of the body of the post by clicking the `Continue Reading` button (*see Fig10*). They can comment on that specific post by entering the comment into the input area and then clicking on the `Confirm` button (*see Fig11, Fig12*).

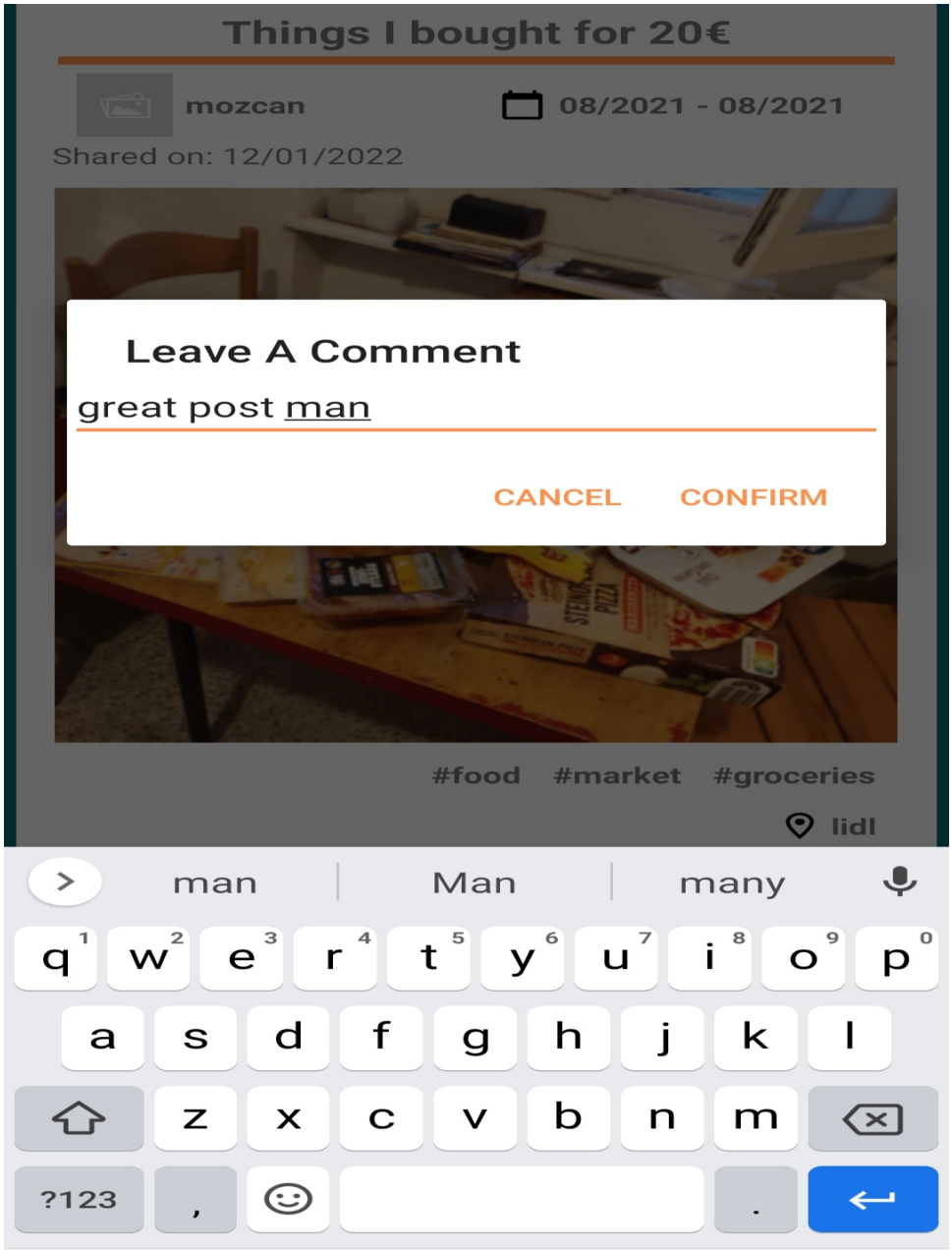


Fig11: View Post Page - Comment

Fig12: View Post Page - Comment - 2

Users can also like a post by clicking the heart shaped button (*see Fig13*) and see the number of likes increase.

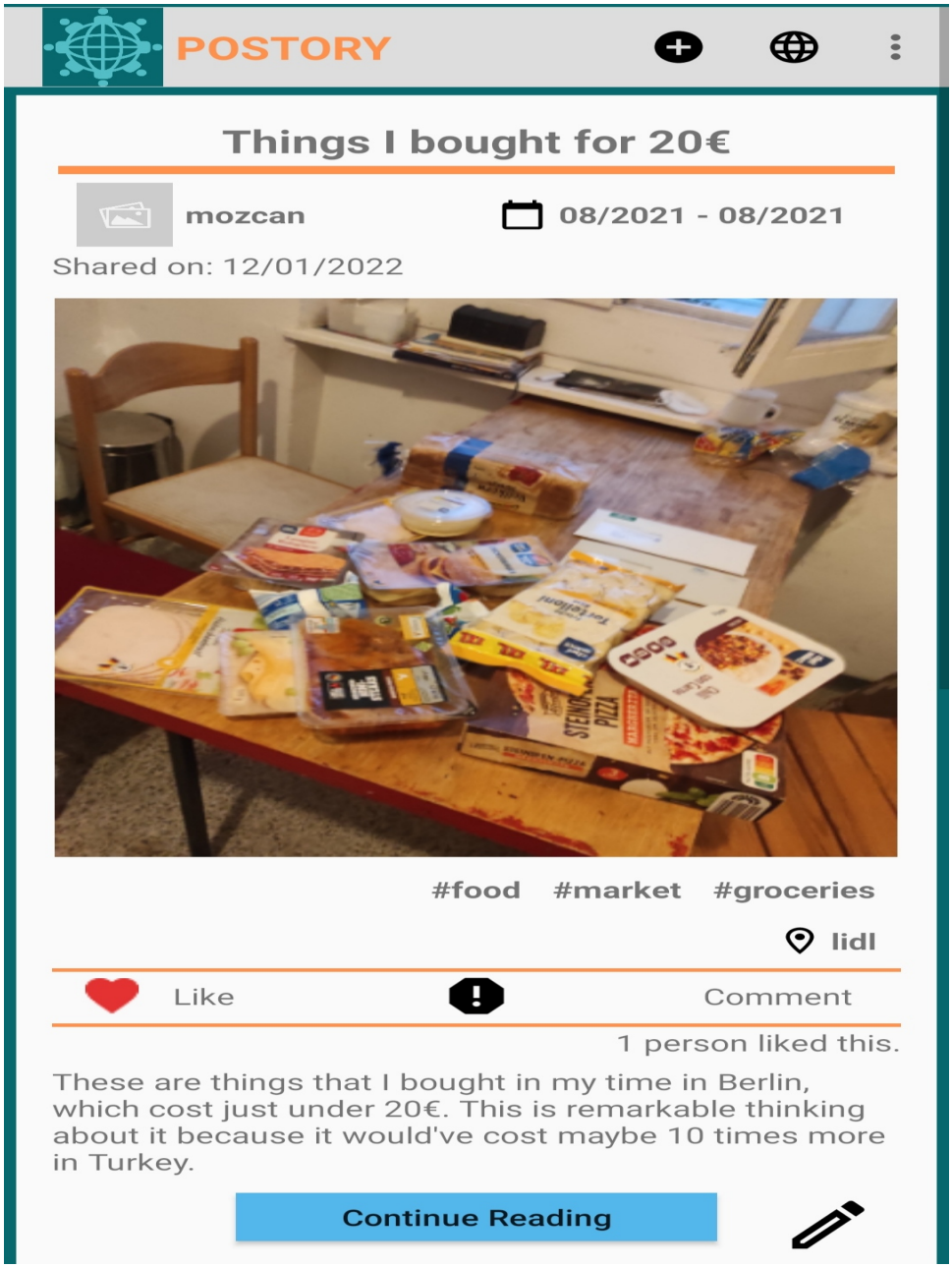


Fig13: View Post Page - Like

One last feature is reporting a post.

An **example** report action is described below:

1. A user sees a post he/she doesn't like.
2. User goes to the View Post Page.
3. Clicks on the button with the report icon. (*see Fig10*)
4. User sees the prompt confirming his/her action was succesful. (*see Fig14*)

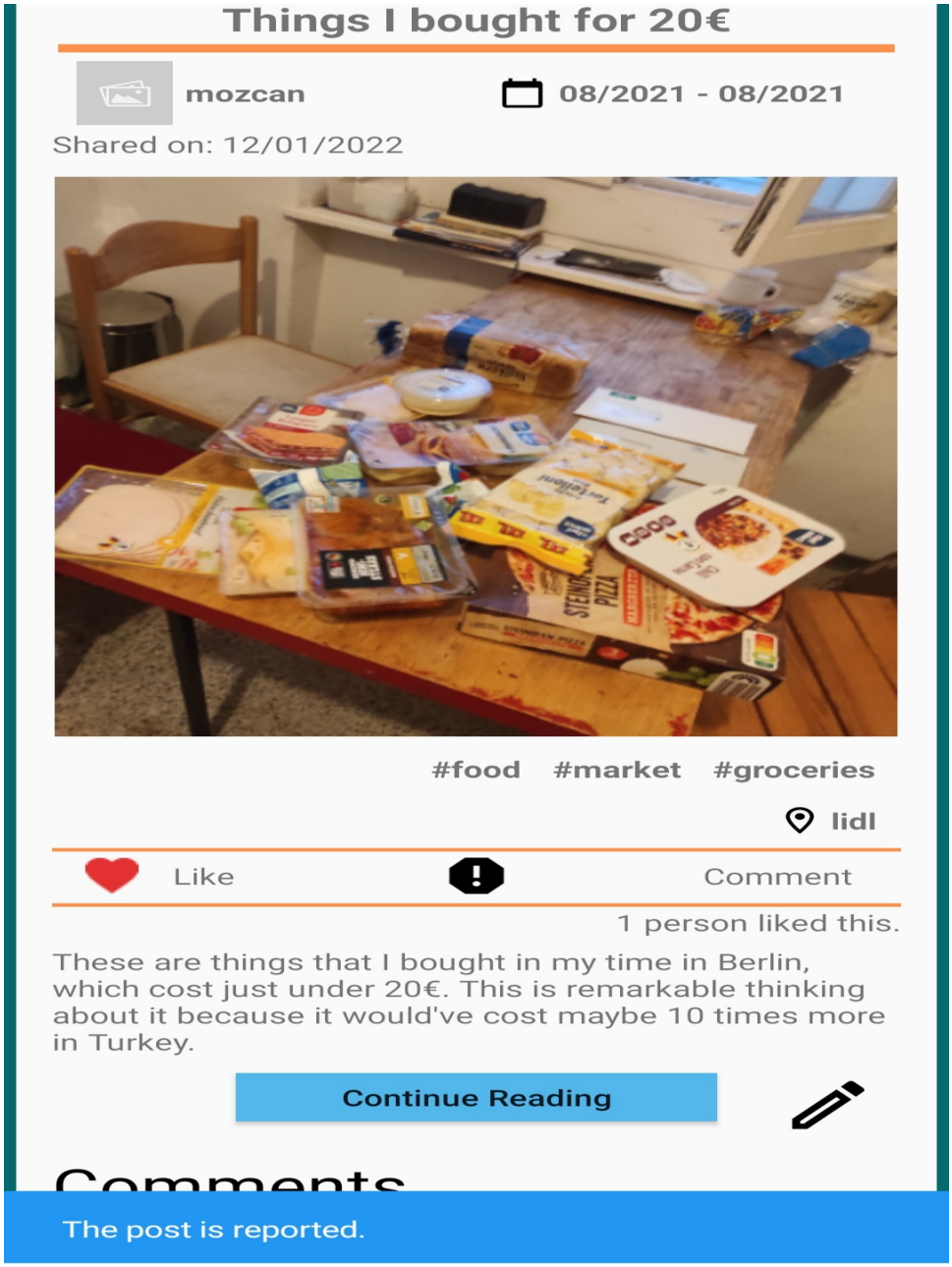


Fig14: View Post Page - Report

Homepage

In the homepage, users are able to see posts from his/her followed users and his/her own. Users can click on the eye button on the right side of the posts to go the View Post page to have a closer look. Users can also click on the profile of the post owner to go to his/her profile page. User can also click on the plus button on top of the page to create a new post, click the world icon to go to the discover page and click the pen icon to edit the post, if they are allowed to do so. (*see Fig15*)

Fig15: Homepage

Create/Edit Post Page

After entering various inputs, specifying a post photo, a title, a body, tags and geolocations, users can create posts. Below, you can see the initial look of the Create Post page. (*see Fig16*)

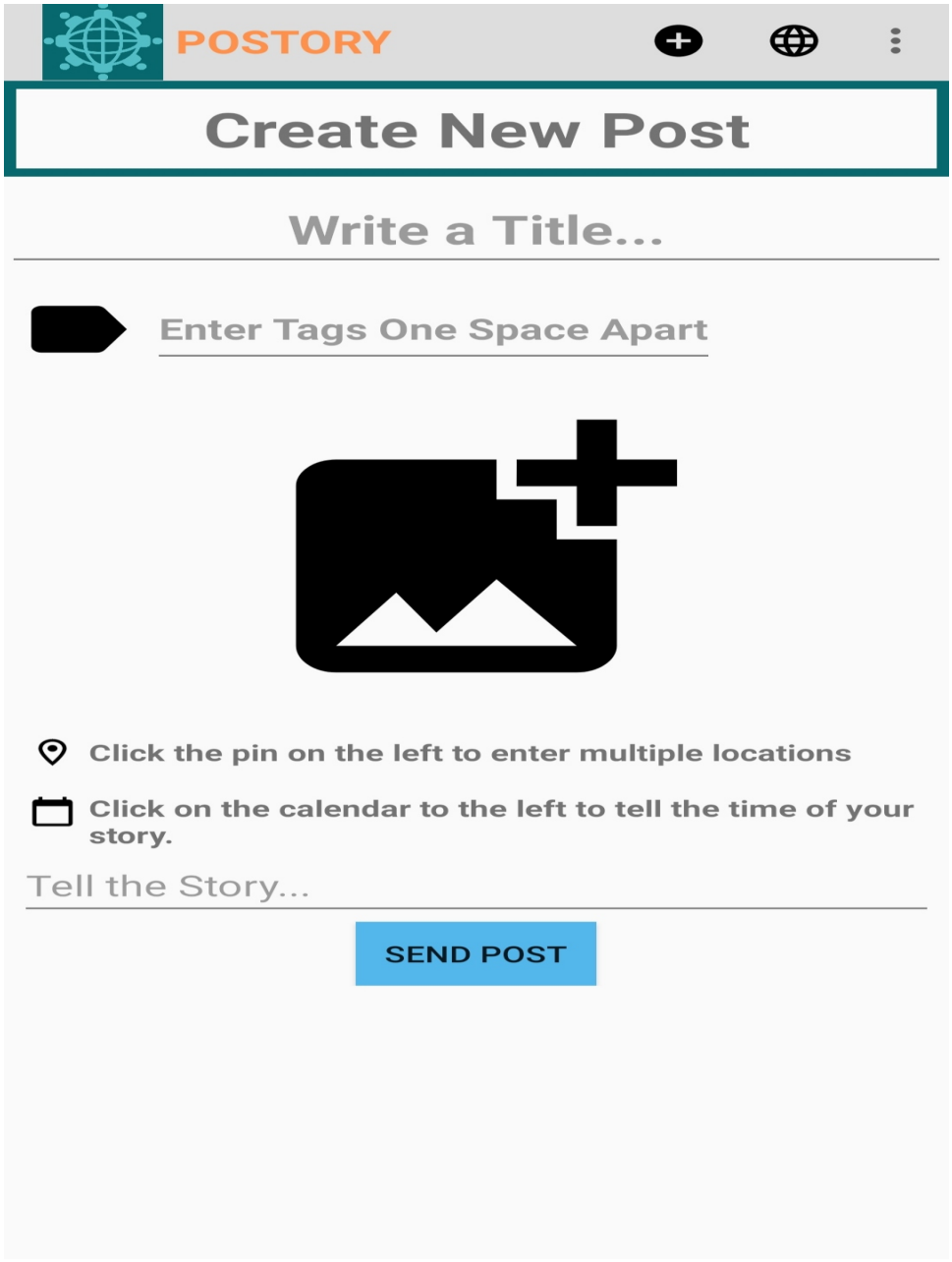


Fig16: Create Post - Initial look

1. When the user wants to add a photo for the post, he/she should click on the placeholder image in the center and choose on the following from the appearing tab to add a photo from the gallery or to take a photo.

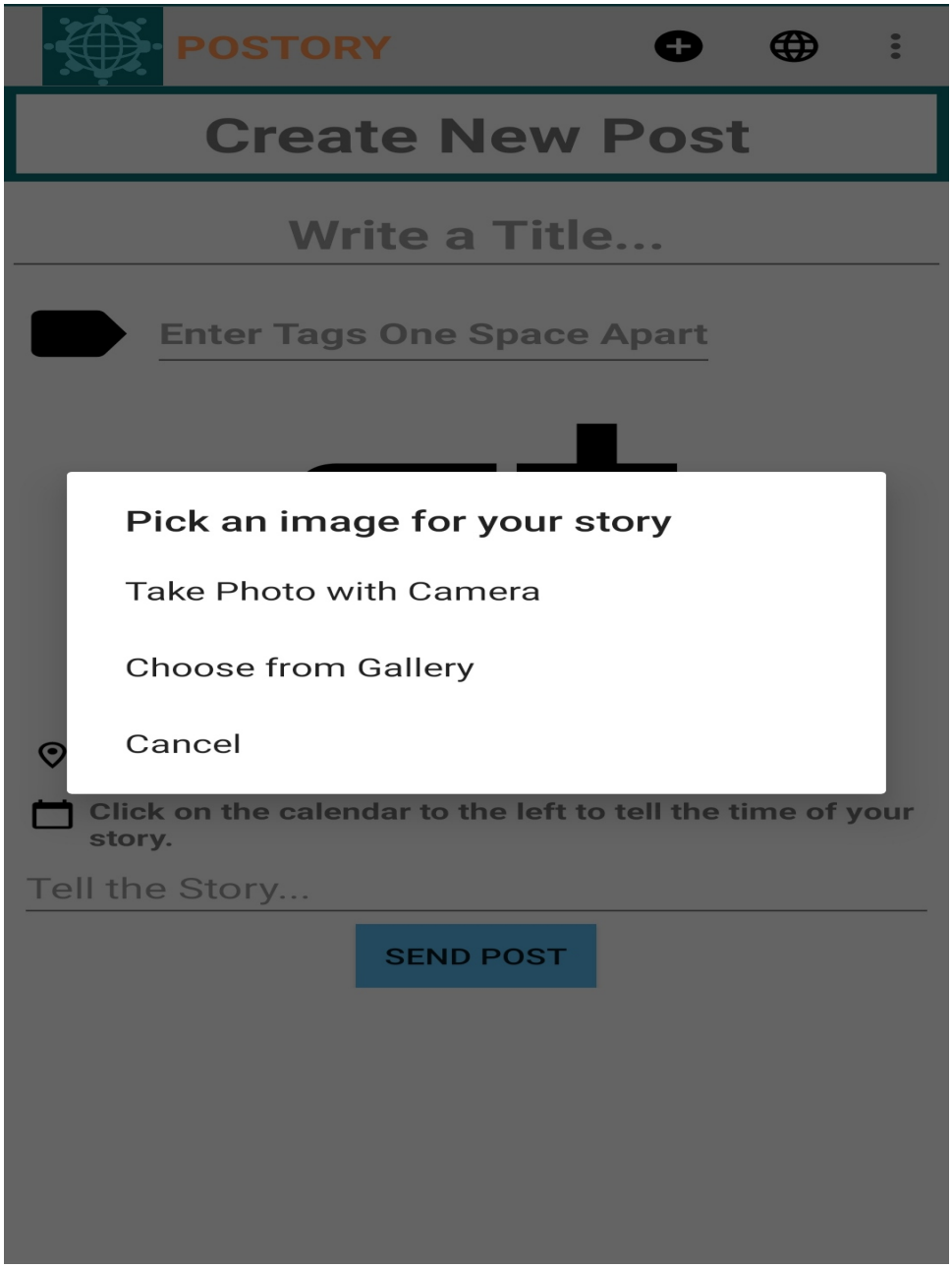


Fig17: Create Post - Add Photo

2. After that if the user wants to add locations to the post, he can click the pin icon, which prompts him/her to a map.(*see Fig18*) After long clicking a location, a text box appears which the user can write a location name to. (*see Fig19*) With this, choosing location is done.

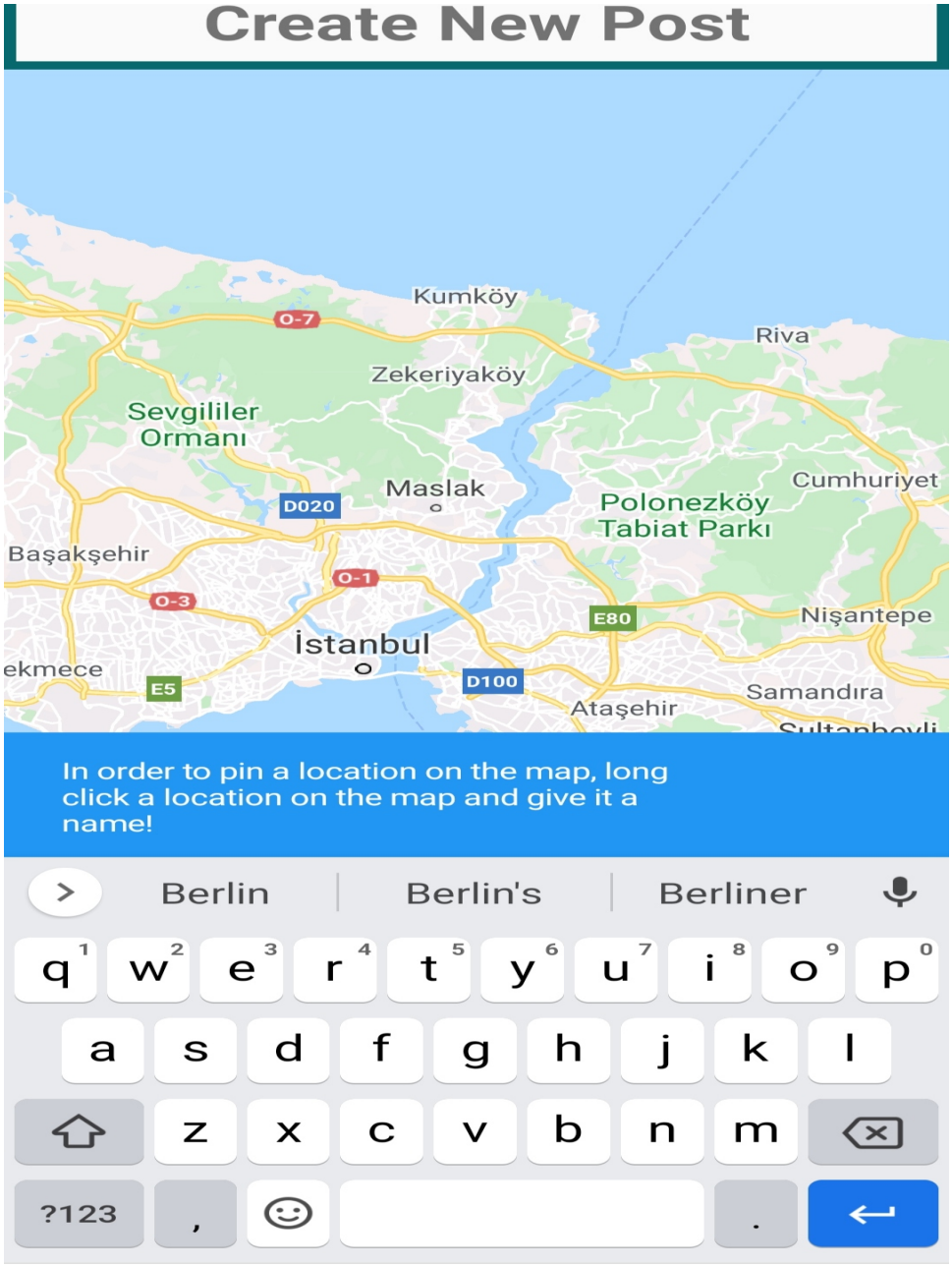


Fig18: Create Post - Choose Location

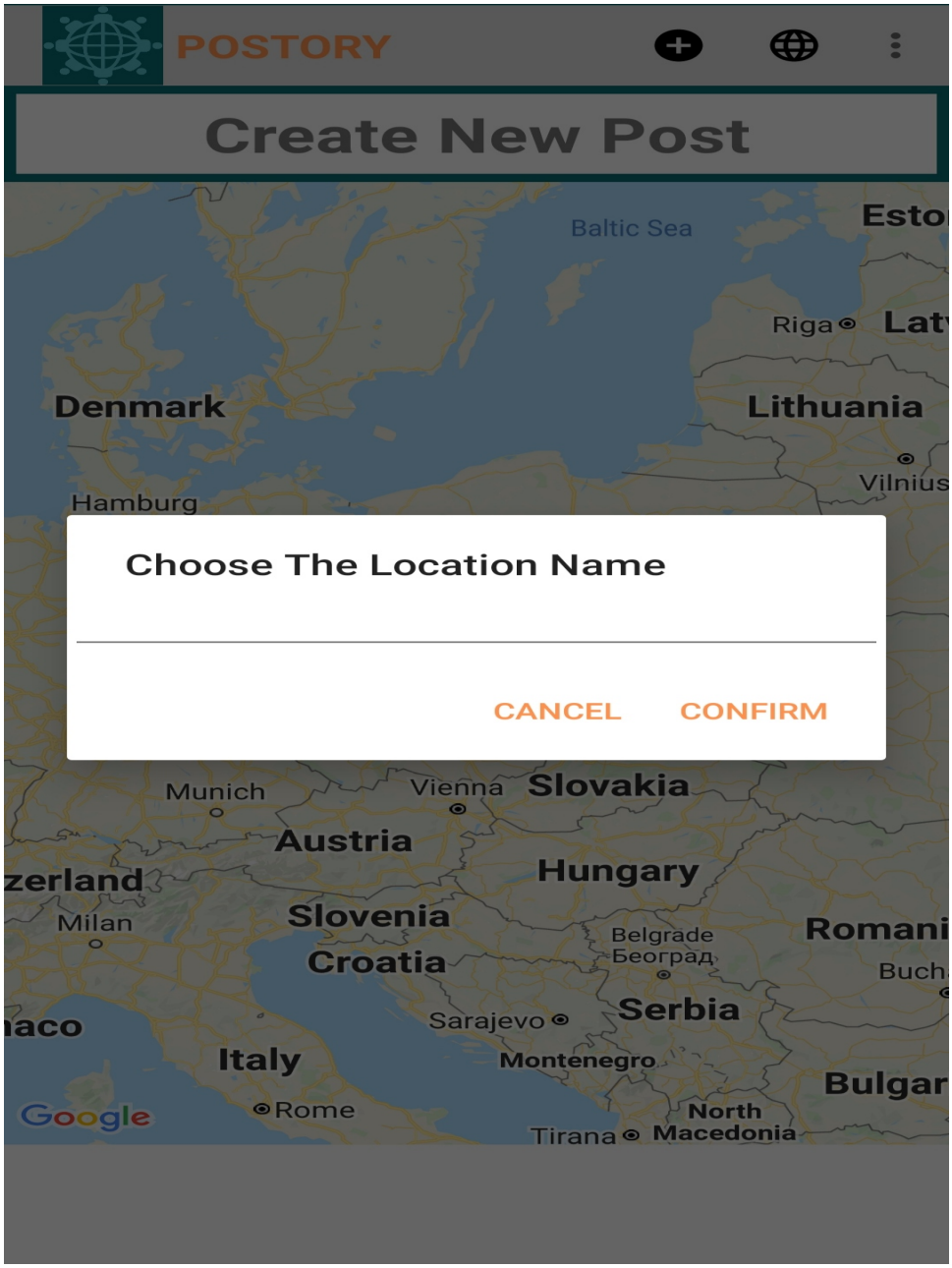


Fig19: Create Post - Name Location

3. User can also choose a timeframe for the post, by clicking the calendar icon. (*see Fig16*). Another fragment shows up and there the user can choose a time interval by specifying the year, month and so on. (*see Fig20*)

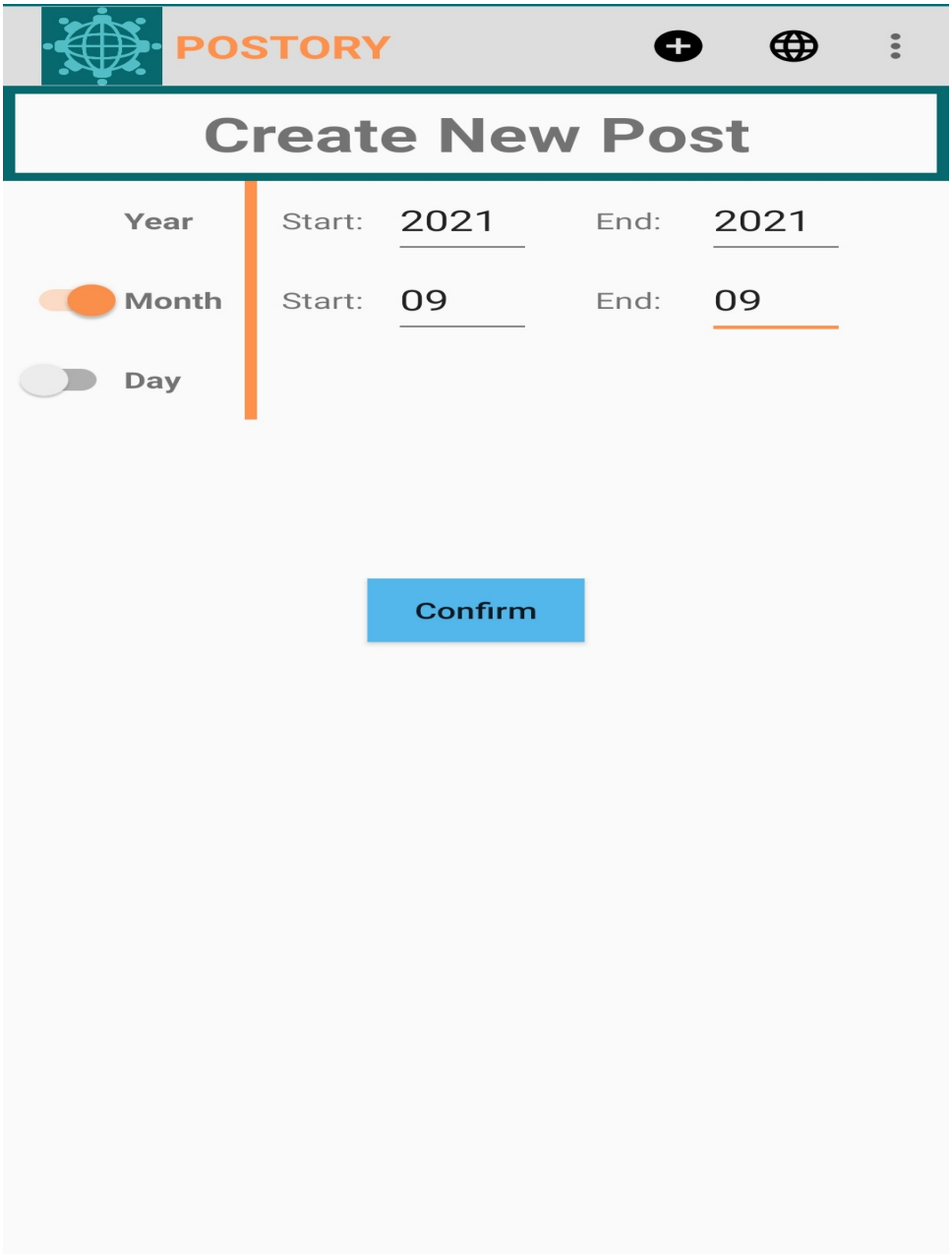


Fig20: Create Post - Add Time

4. After writing a title and a body to the story, user can add tags to the post by writing words to the tags section with one blank space between each of them. (*see Fig21*) If the user is content with the post, he/she can click the `Send Post` button (*see Fig21*) and post it. After the post is succesfully created, he/she will be redirected to the homepage.(*see Fig22*).

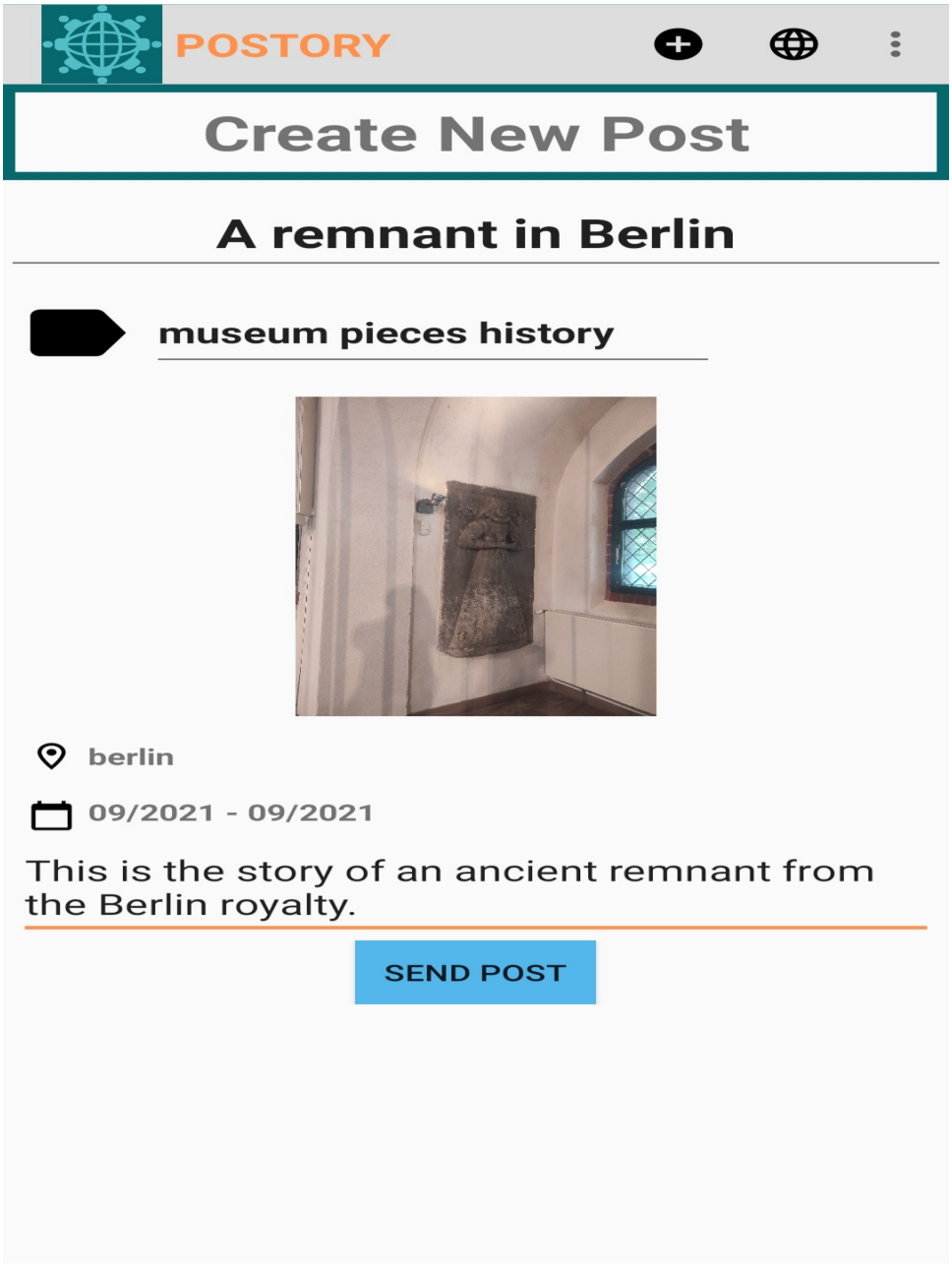


Fig21: Create Post - Final Look

Fig22: Create Post - Post on the homepage

Editing a post

Editing a post can be done by clicking the pen icon on the homepage or the View post page. This functionality uses the same page as the Create Post page but for editing, the features of a post are already displayed. User can edit various features of a post, including the tags, geolocation, the title, the date and the story. Upon changing the desired fields, user can click the `Send Post` button (*see Fig23*) so that the post will be updated and the user will be directed to the homepage.

Fig23: Editing a post

Activity Stream Page

Activity stream page lets a user see their own activities and activities of followed people.

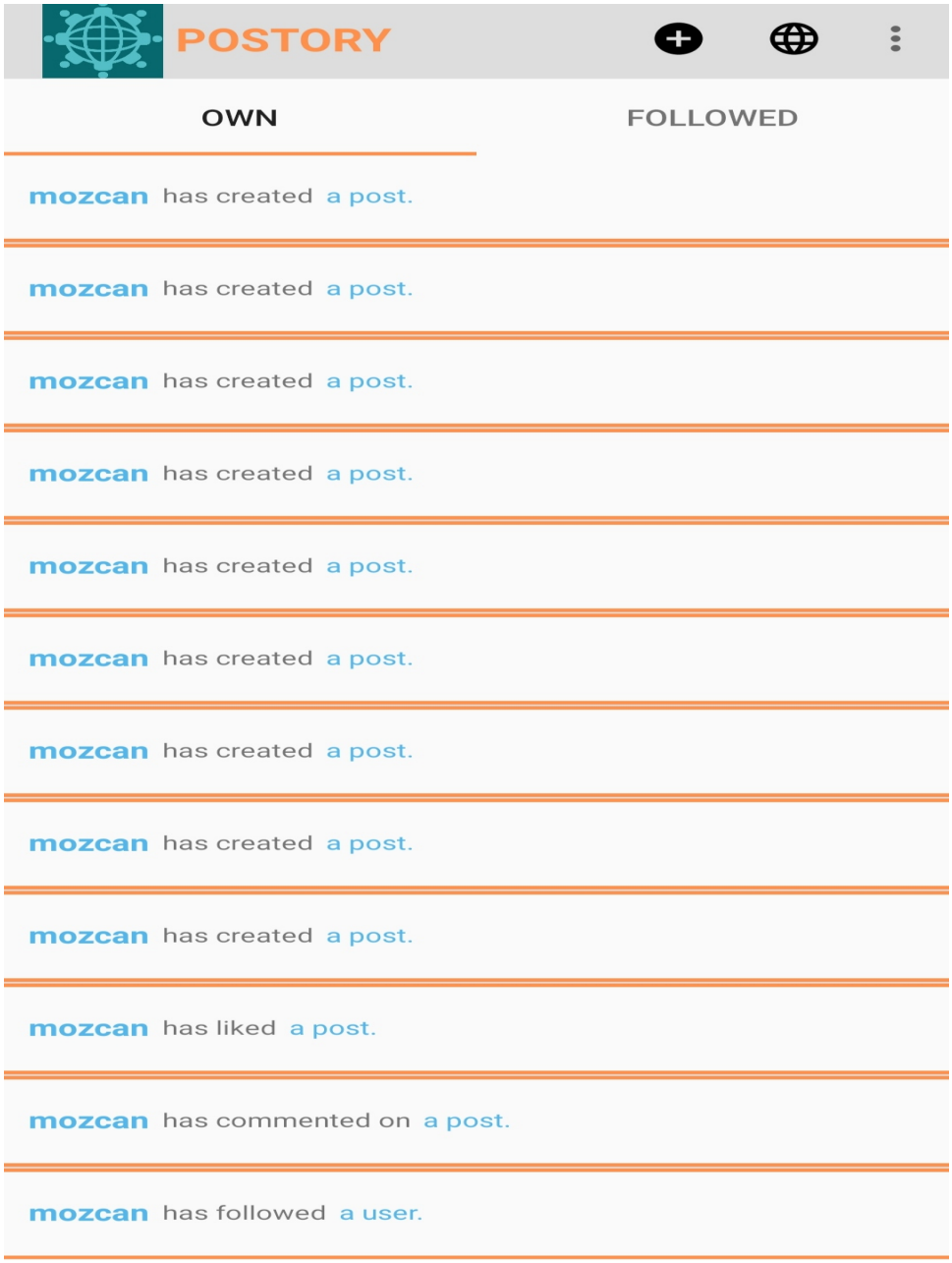


Fig24: Activity Stream

1. Own tab: Show activities committed by the current user. Activities include creating, editing, liking and commenting on posts and following other users.
2. Followed tab: Shows activities of followed people.

User Search page

User Search page lets users see the results of user search by nickname feature.

In *Fig25 below,* you can see the result of search with the term "me".

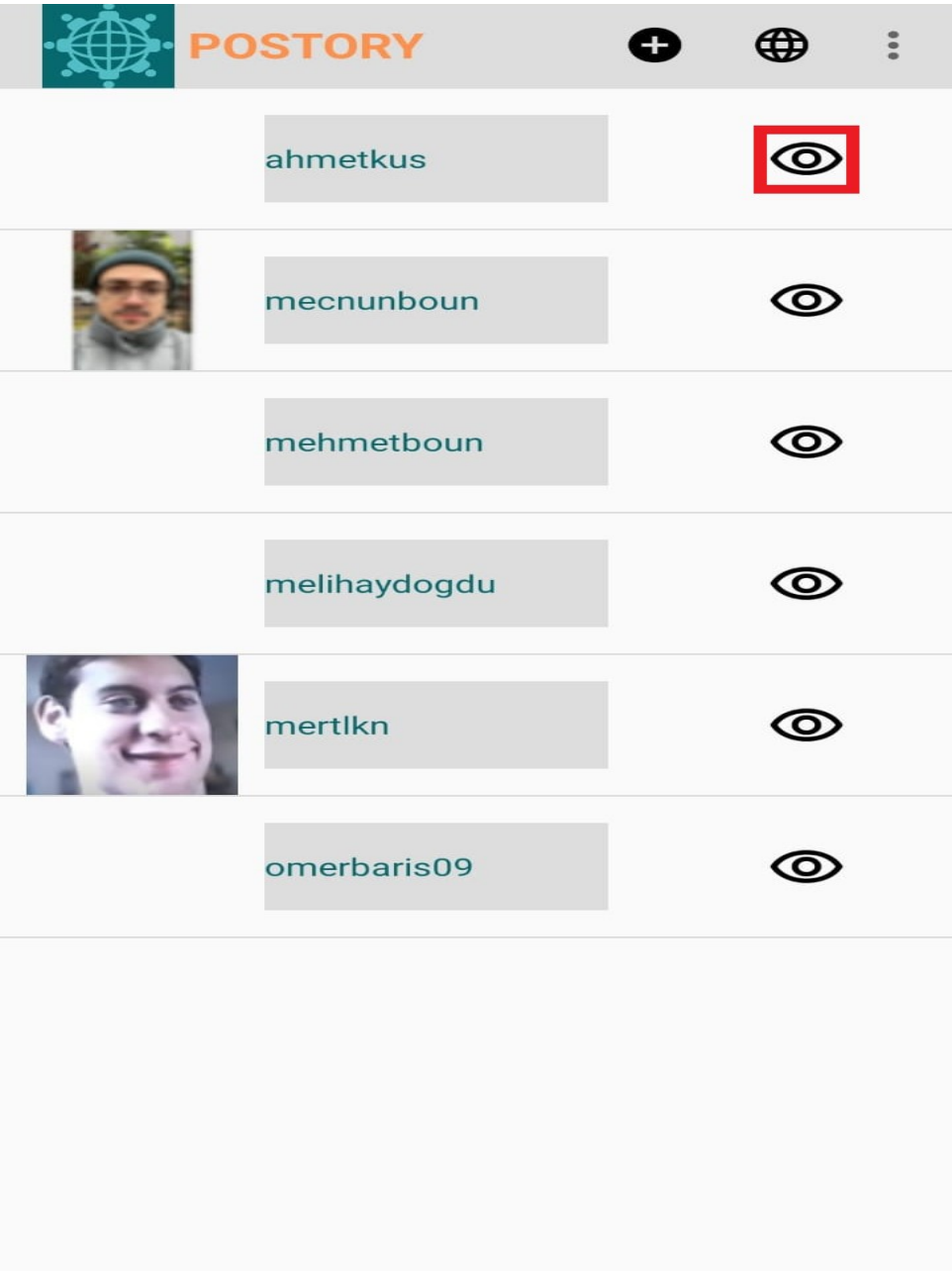


Fig25: User Search

In *Fig25* all the users with nicknames that include "me" are retrieved. The user can navigate to profiles of these users by clicking the "eye button" that is marked inside the red square.

Sign In Page

Sign In page lets user sign in to Postory with their e-mail and password.

Fig26: Sign In page

1. E-mail field: Lets user enter their e-mail
2. Password field: Lets user enter their password
3. Sign In: Lets user submit their sign in with the entered info
4. Forgot Password button: Prompts user for their e-mail and initiates a password reset request.
5. Sign Up button: Redirects to Sign Up page

Forgot Password Page

Forgot password page lets user to reset her password via writing her e-mail address. When the user successfully sends a reset request, a link for password reset is sent to her email.

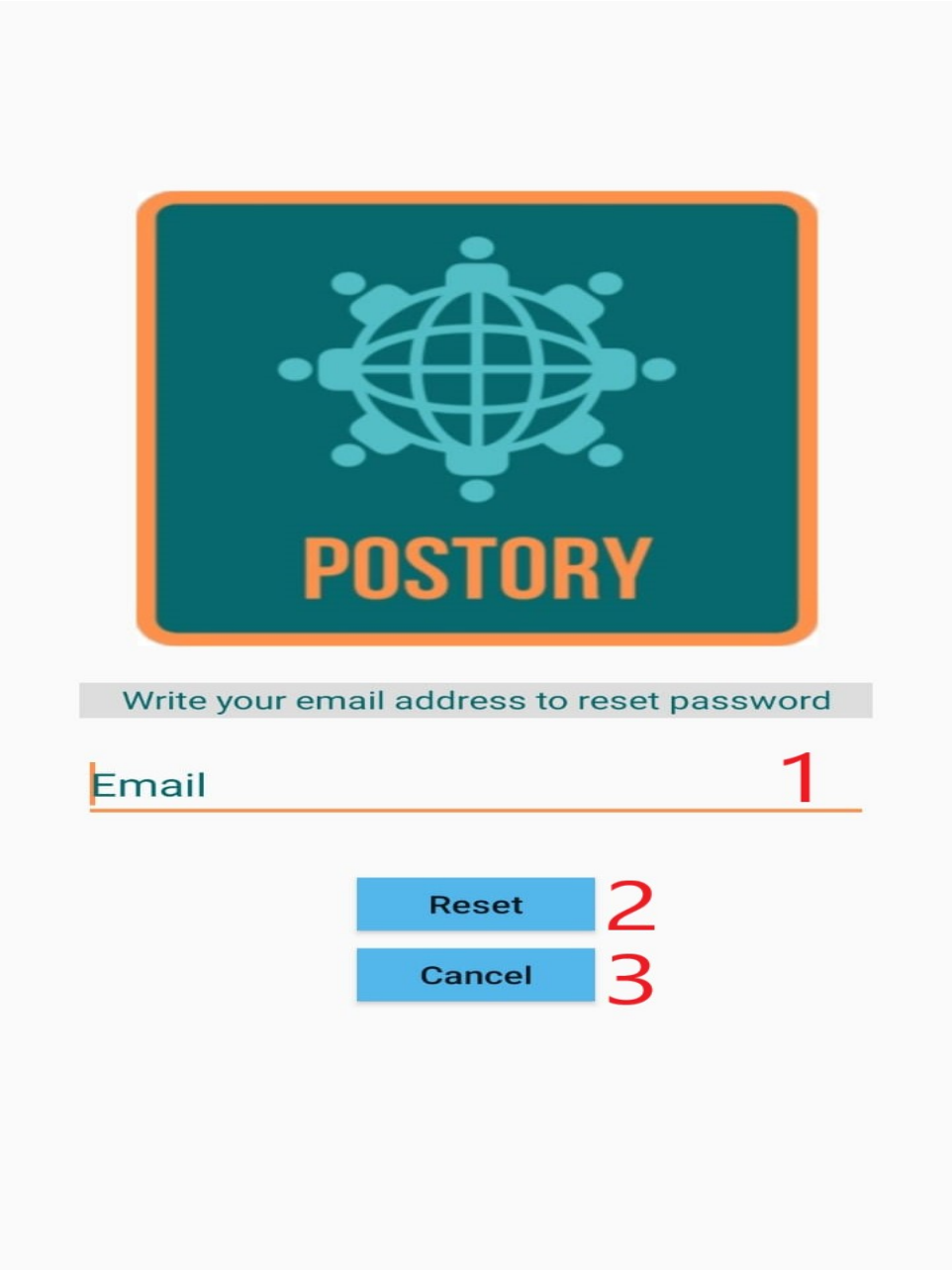


Fig27: Forgot Password Page

1. E-mail field: Lets user enter her e-mail address
2. Reset button: Lets user send request for a password reset. After reset request is sent, the user is redirected to Sign In page.
3. Cancel button: Lets user close this page and redirect back to Sign In page.

Sign Up Page

Sign Up page lets people sign up to Postory with their information. It also checks for validity of the entered input.

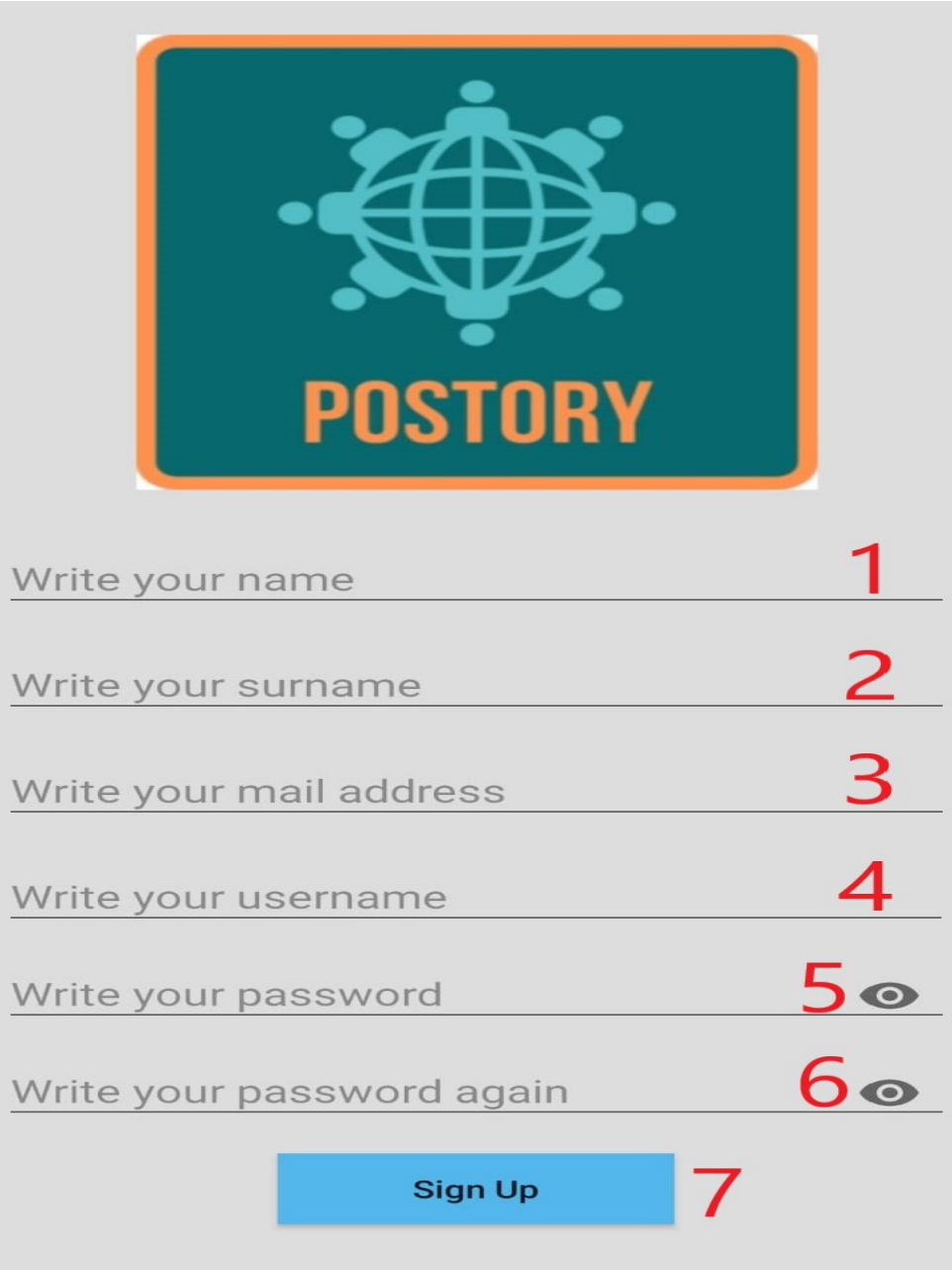


Fig28: Sign Up page

1. Name field: Lets person enter their name for the new account.
2. Surname field: Lets person enter their surname for the new account.
3. E-mail field: Lets person enter their e-mail for the new account
4. Username field: Lets person enter their username for the new account
5. Password field: Lets person enter their password for the new account.
6. Repeat password field: Lets person enter their password confirmation.
7. Sign Up button: Submits the sign up with the entered information.

After the sign up button is pressed, an email is sent to the given mail address for activation. The user can activate her account with that link.

Toolbar

Toolbar becomes visible after the user signs in. It is used for navigation among activities.

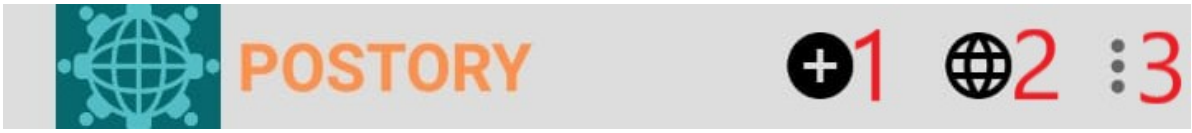


Fig29: Toolbar - 1

In *Fig29:*

1. Lets user go to Create Post Page.
2. Lets user go to Discovery Page.
3. Lets user extend the toolbar.

When the toolbar is extended, new options are visible, which are visible in *FigYY*

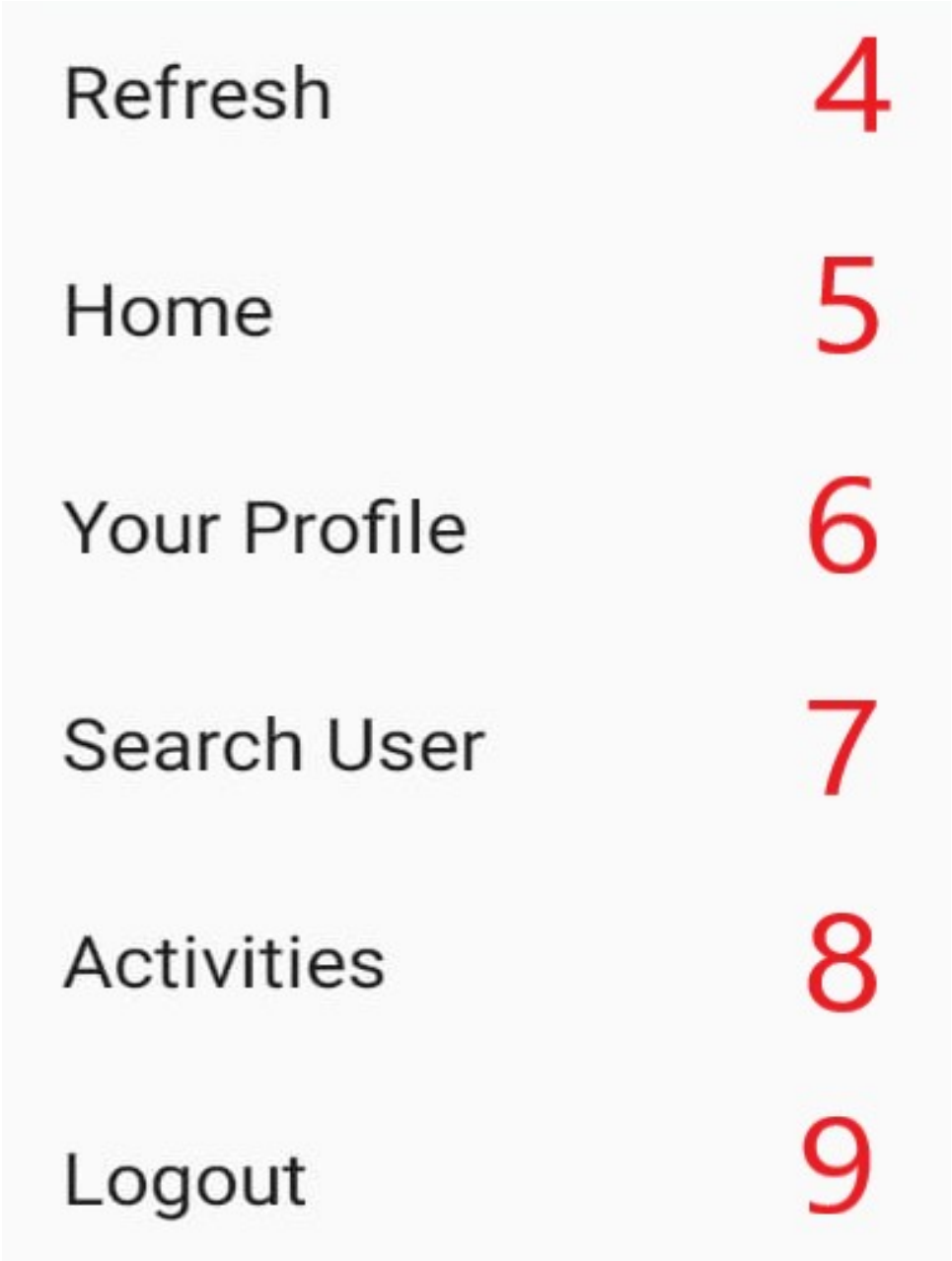


Fig30: Toolbar - 2

In *Fig30:*

4. Lets user refresh the page.
5. Lets user go to homepage.
6. Lets user go to her profile page.

7. Lets user search for other users.
8. Lets user go to Activity Streams page.
9. Lets user sign out.

When '7' is clicked, the toolbar provides a search bar for searching users as it can be seen in *Fig31:*

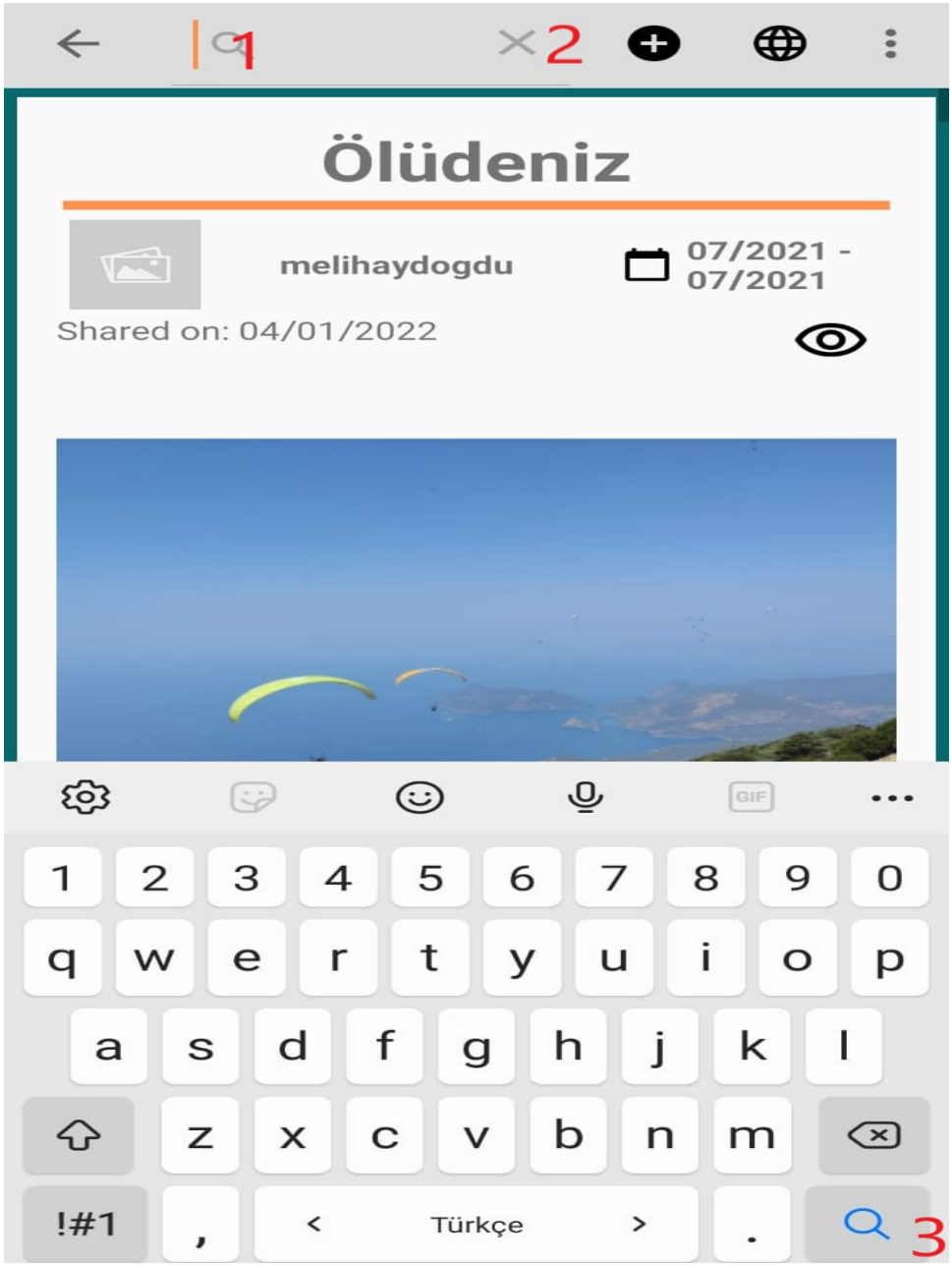


Fig31: Toolbar - 3

In *Fig31:*

1. Is the field for writing username.
2. Lets user cancel the search operation.
3. Lets user complete the search.

System manual

Backend

Requirements

- Docker *(available here)*
- docker-compose *(available here)*
- git *(available here)*
- The provided .env file of the backend.

Steps to Build and Run

- Run the following on the command line:

1. `git clone https://github.com/bounswe/2021SpringGroup9`
2. `cd postory/backend`
3. `Copy the provided .env file of the backend to the directory`
4. `docker-compose build`
5. `docker-compose up -d`

After running these commands, the backend will be available at: http://localhost:8000/.

More information on how to run locally and deploy the app can be found in backend/README.md.

Frontend

Requirements

- Docker *(available here)*
- docker-compose *(available here)*
- git *(available here)*
- The provided .env file of the frontend.
- Change the `REACT_APP_BACKEND_API` variable on the .env file if needed.

Steps to Build and Run

- Run the following on the command line:

1. Clone the repo if not cloned already\
   `git clone https://github.com/bounswe/2021SpringGroup9`
2. `cd postory/frontend`
3. Copy the provided .env file of the frontend to the frontend directory (2021SpringGroup9/postory/frontend)
4. `docker-compose build`
5. `docker-compose up -d`

After running these commands, the frontend will be available at: http://localhost:3000/.
You can also access the readme file at frontend/README.md.

Android

This manual is a guide for running the Postory Android APK. APK is a file format that is used for distribution of Android applications. Our apk is located at *2021SpringGroup9/postory/android/postory.apk* on master branch.

Requirements

- Android Studio
- Android Studio AVD Manager

Clone the Code and APK

1. Clone the repo if not cloned already\
   `git clone https://github.com/bounswe/2021SpringGroup9`
2. `cd postory/android`
3. The apk is named `postory.apk`

Run on your Android smartphone

The apk file should be transferred to your smartphone. This could be done with a USB connection. After the transfer, click on the APK file on your phone. Defaultly, Android smartphones do not install apk files for security reasons.

- If your Android version is recent, it might prompt you to permit installation of unknown apps. After you allow it, the application will be installed and opened.
- If you are not prompted for giving permissions to unknown apps, you may need to do it manually.

  1- Go to Settings -> Security -> Install Unknown Apps
  2- Allow installation of unknown apps.

Note: The steps for giving manual permissions may vary among devices. This is a summary of the process.

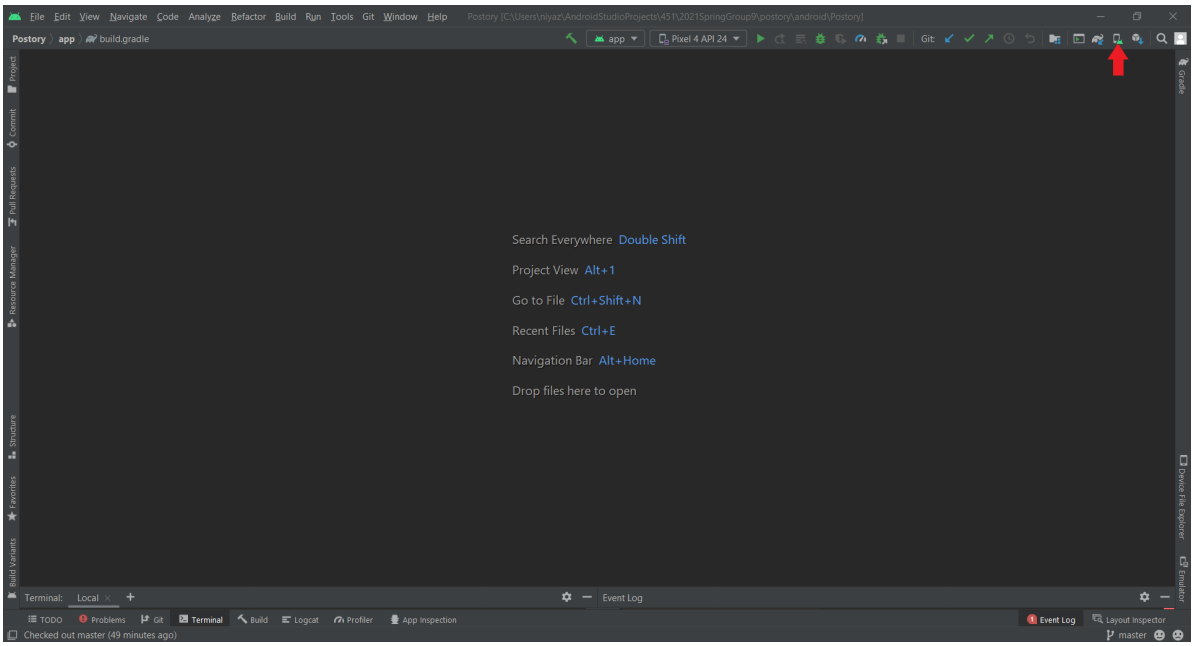
Run on Emulator

To run an APK on emulator, you need to have Android Studio along with AVD Manager. To see the formal documentation, click here.

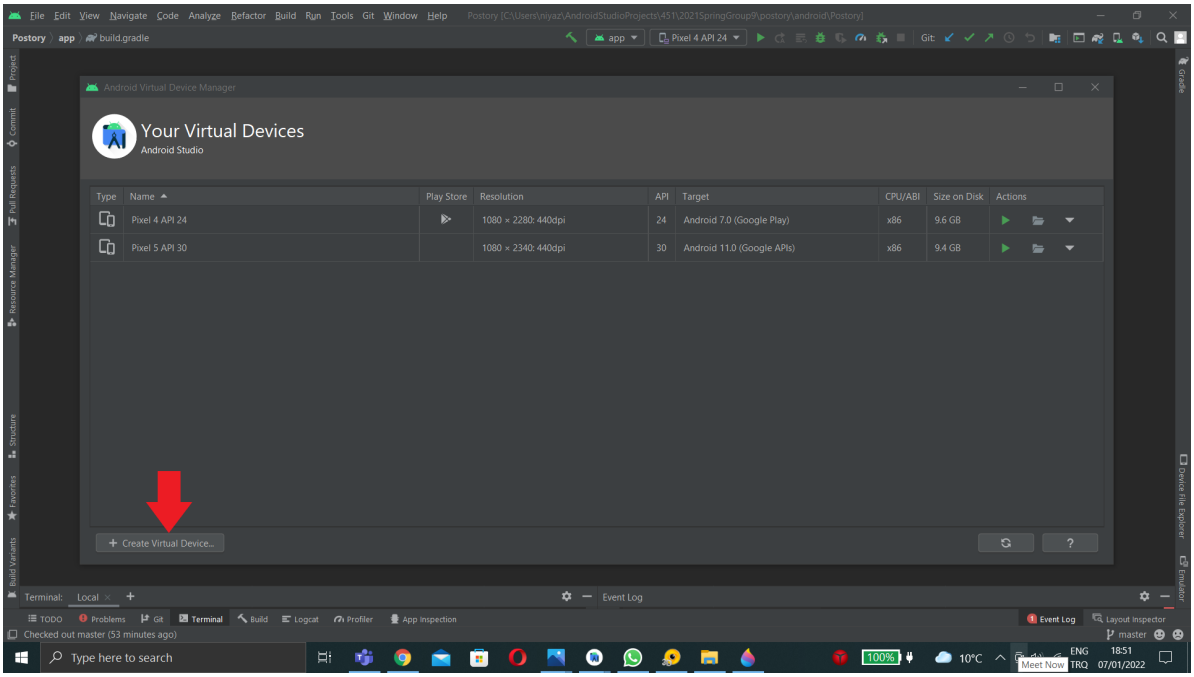
After the AVD Manager is successfully installed, an emulator device should be created (or you can use another emulator that you created before)

To create an emulator(you can skip if you have a suitable emulator):

1- Click on AVD Manager



2- Click on Create Virtual Device



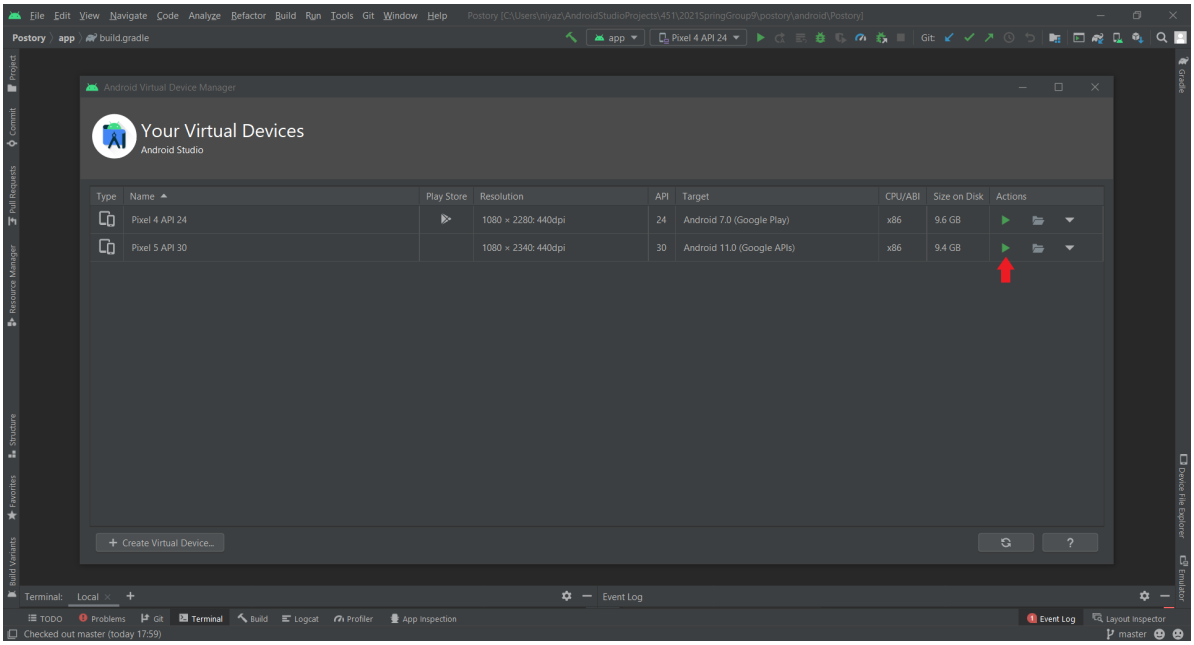
3- Pick a device size. Note that the application works best on smartphone ratios.

4- Select a system image. You need to download the image if it does not exist in your system. This might take some time. Note that you should pick an API level in accordance with the limitations stated in README. A higher API level is usually better but you might need to download more data.

After you have a suitable emulator, run the emulator.

1- Open AVD Manager

2- Run the emulator you want.

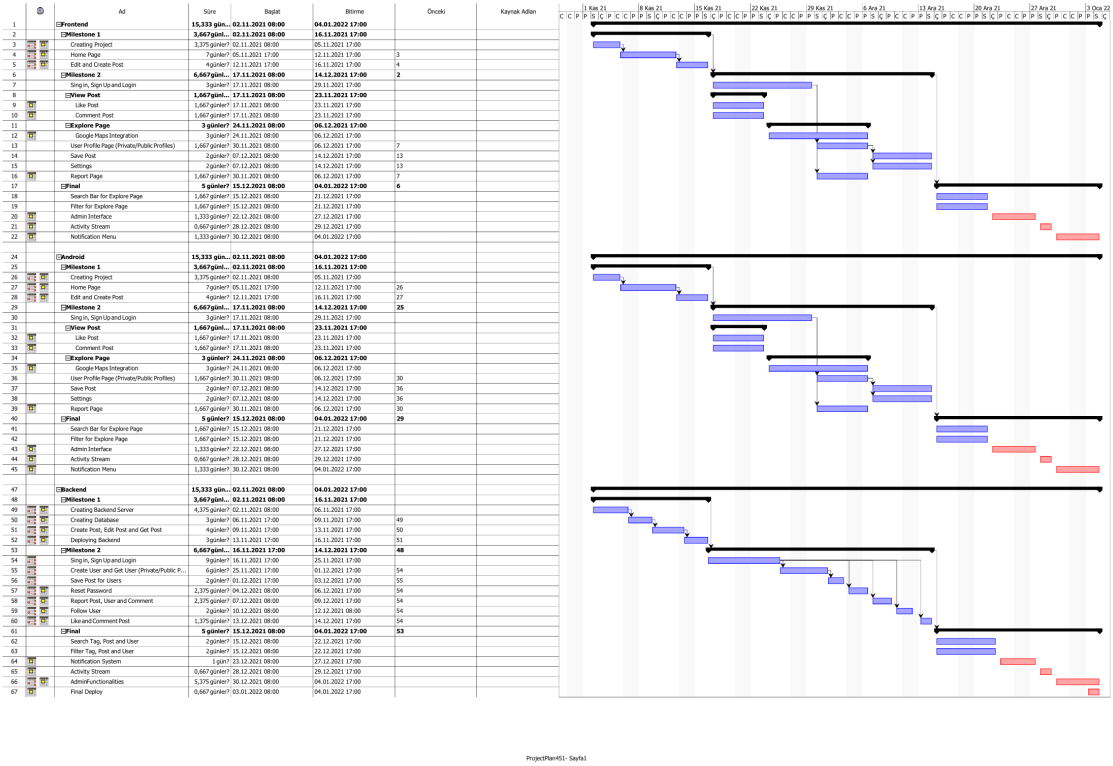


3- After the emulator is on, drag and drop the apk file onto its screen.

4- The application might start running or you might need to click on the application at the emulator to run it.

For detailed documentation, click here.

Project Plan



	Ad	Süre	Başlat	Bitirme	Önceki	Kaynak Adları
1	Frontend	15,333 gün...	02.11.2021 08:00	04.01.2022 17:00		
2	Milestone 1	3,667günl...	02.11.2021 08:00	16.11.2021 17:00		
3	Creating Project	3,375 günler?	02.11.2021 08:00	05.11.2021 17:00		
4	Home Page	7günler?	05.11.2021 17:00	12.11.2021 17:00	3	
5	Edit and Create Post	4günler?	12.11.2021 17:00	16.11.2021 17:00	4	
6	Milestone 2	6,667günl...	17.11.2021 08:00	14.12.2021 17:00	2	
7	Sing in, Sign Up and Login	3günler?	17.11.2021 08:00	29.11.2021 17:00		
8	View Post	1,667günl...	17.11.2021 08:00	23.11.2021 17:00		
9	Like Post	1,667 günler?	17.11.2021 08:00	23.11.2021 17:00		
10	Comment Post	1,667 günler?	17.11.2021 08:00	23.11.2021 17:00		
11	Explore Page	3 günler?	24.11.2021 08:00	06.12.2021 17:00		
12	Google Maps Integration	3günler?	24.11.2021 08:00	06.12.2021 17:00		
13	User Profile Page (Private/Public Profiles)	1,667 günler?	30.11.2021 08:00	06.12.2021 17:00	7	
14	Save Post	2günler?	07.12.2021 08:00	14.12.2021 17:00	13	
15	Settings	2günler?	07.12.2021 08:00	14.12.2021 17:00	13	
16	Report Page	1,667 günler?	30.11.2021 08:00	06.12.2021 17:00	7	
17	Final	5 günler?	15.12.2021 08:00	04.01.2022 17:00	6	
18	Search Bar for Explore Page	1,667 günler?	15.12.2021 08:00	21.12.2021 17:00		
19	Filter for Explore Page	1,667 günler?	15.12.2021 08:00	21.12.2021 17:00		
20	Admin Interface	1,333 günler?	22.12.2021 08:00	27.12.2021 17:00		
21	Activity Stream	0,667 günler?	28.12.2021 08:00	29.12.2021 17:00		
22	Notification Menu	1,333 günler?	30.12.2021 08:00	04.01.2022 17:00		
24	Android	15,333 gün...	02.11.2021 08:00	04.01.2022 17:00		
25	Milestone 1	3,667günl...	02.11.2021 08:00	16.11.2021 17:00		
26	Creating Project	3,375 günler?	02.11.2021 08:00	05.11.2021 17:00		
27	Home Page	7günler?	05.11.2021 17:00	12.11.2021 17:00	26	
28	Edit and Create Post	4günler?	12.11.2021 17:00	16.11.2021 17:00	27	
29	Milestone 2	6,667günl...	17.11.2021 08:00	14.12.2021 17:00	25	
30	Sing in, Sign Up and Login	3günler?	17.11.2021 08:00	29.11.2021 17:00		
31	View Post	1,667günl...	17.11.2021 08:00	23.11.2021 17:00		
32	Like Post	1,667 günler?	17.11.2021 08:00	23.11.2021 17:00		
33	Comment Post	1,667 günler?	17.11.2021 08:00	23.11.2021 17:00		
34	Explore Page	3 günler?	24.11.2021 08:00	06.12.2021 17:00		
35	Google Maps Integration	3günler?	24.11.2021 08:00	06.12.2021 17:00		
36	User Profile Page (Private/Public Profiles)	1,667 günler?	30.11.2021 08:00	06.12.2021 17:00	30	
37	Save Post	2günler?	07.12.2021 08:00	14.12.2021 17:00	36	
38	Settings	2günler?	07.12.2021 08:00	14.12.2021 17:00	36	
39	Report Page	1,667 günler?	30.11.2021 08:00	06.12.2021 17:00	30	
40	Final	5 günler?	15.12.2021 08:00	04.01.2022 17:00	29	
41	Search Bar for Explore Page	1,667 günler?	15.12.2021 08:00	21.12.2021 17:00		
42	Filter for Explore Page	1,667 günler?	15.12.2021 08:00	21.12.2021 17:00		
43	Admin Interface	1,333 günler?	22.12.2021 08:00	27.12.2021 17:00		
44	Activity Stream	0,667 günler?	28.12.2021 08:00	29.12.2021 17:00		
45	Notification Menu	1,333 günler?	30.12.2021 08:00	04.01.2022 17:00		
47	Backend	15,333 gün...	02.11.2021 08:00	04.01.2022 17:00		
48	Milestone 1	3,667günl...	02.11.2021 08:00	16.11.2021 17:00		
49	Creating Backend Server	4,375 günler?	02.11.2021 08:00	06.11.2021 17:00		
50	Creating Database	3günler?	06.11.2021 17:00	09.11.2021 17:00	49	
51	Create Post, Edit Post and Get Post	4günler?	09.11.2021 17:00	13.11.2021 17:00	50	
52	Deploying Backend	3günler?	13.11.2021 17:00	16.11.2021 17:00	51	
53	Milestone 2	6,667günl...	16.11.2021 17:00	14.12.2021 17:00	48	
54	Sing in, Sign Up and Login	9günler?	16.11.2021 17:00	25.11.2021 17:00		
55	Create User and Get User (Private/Public P...	6günler?	25.11.2021 17:00	01.12.2021 17:00	54	
56	Save Post for Users	2günler?	01.12.2021 17:00	03.12.2021 17:00	55	
57	Reset Password	2,375 günler?	04.12.2021 08:00	06.12.2021 17:00	54	
58	Report Post, User and Comment	2,375 günler?	07.12.2021 08:00	09.12.2021 17:00	54	
59	Follow User	2günler?	10.12.2021 08:00	12.12.2021 08:00	54	
60	Like and Comment Post	1,375 günler?	13.12.2021 08:00	14.12.2021 17:00	54	
61	Final	5 günler?	15.12.2021 08:00	04.01.2022 17:00	53	
62	Search Tag, Post and User	2günler?	15.12.2021 08:00	22.12.2021 17:00		
63	Filter Tag, Post and User	2günler?	15.12.2021 08:00	22.12.2021 17:00		
64	Notification System	1 gün?	23.12.2021 08:00	27.12.2021 17:00		
65	Activity Stream	0,667 günler?	28.12.2021 08:00	29.12.2021 17:00		
66	AdminFunctionalities	5,375 günler?	30.12.2021 08:00	04.01.2022 17:00		
67	Final Deploy	0,667 günler?	03.01.2022 08:00	04.01.2022 17:00		

ProjectPlan451 - Sayfa1

Software Requirements Specification

To see the requirements, click here.

Software Design Documents

In this section, links to design documents are provided.

Scenarios and Mockups:

- Creating Posts Scenario & Mockup
- Viewing Posts Scenario & Mockup
- Searching and Filtering Scenario & Mockup

UML Diagrams:

- Use Case Diagram
- Class Diagram
- Sequence Diagrams

Coding Work Done by Each Team Member

The detailed work is provided in the individual reports. The summary must be consistent with individual reports.

Name	Work Done
Mert Alkan	• Nearby stories function implementation Issue#502 • Implementation of activity stream Issue#503,PR#506 • Adding public users' activities endpoint Issue#583,PR#548 • Bugfixes and further developments in activity stream. Issue#584, Issue#585, PR#519, PR#520, PR#571 • Implementing tests for the functionalities that I wrote. Issue#504, Issue#581,PR#505, PR#552 • Document endpoints and tests written by me. Issue#603, PR#592 • Individual milestone 3 report. Issue#605
Ahmet Melih Aydoğdu	• Attended weekly meetings and team meetings • Implemented filter and search mechanism • Implemented admin functionality of Django • Implemented a unit test for filter functionality • Implemented Wikidata API requests to find related tags • Implemented save post functionality • Implemented ban control functionalities • Wrote README file for deploying the backend of the application • Took dump of the database and wrote the steps to restore the database in README file • Updated backend code to be able to be deployable to every environment • Wrote the individual report for Milestone 3 • Created 12 posts for final milestone • Worked on issues: #498, #499, #500, #601 • Opened pull requests: #512, #517 • Reviewed pull requests: #505, #506, #513, #514
Zehranaz Canfes	• Implement the unfollow user functionality: Issue #484 • Implement the follow request and add model and serializer for follow request: Issue #515, PR #513. • Implement the accept/decline follow request: Issue #515, PR #513. • The unit tests I have written can be found here. • Implement the get user helper function: Issue #549, PR #513. • Implement the get follow requests: Issue #547](https://github.com/bounswe/2021SpringGroup9/issues/547), PR #513. • Implemented the report user/story functionality: Issue #516, PR #514. • Added change account settings functionality: Issue #516 • Fixed bugs: Issue #531, PR #533, PR #524, Issue #544, PR #541, Issue #556, PR #541. • Added swagger for API documentation: PR #587 • All of the Pull Requests assigned to me for the final milestone can be found here. • All of the Pull Requests reviewed by me for the final milestone can be found here.

Name	Work Done
Mustafa Emir Çolak	• Implement the user searching functionality of the navigation bar.Issue #485 - PR #491 • Implement the dropdown menu component on navigation bar.Issue #501 - PR #525 • Implement unfollowing a user in the profile pageIssue #494 - PR #497 • Implement changing the user account as private in profile page Issue #507 - PR #526 • Various bugfixes for the create post page.PR #553PR #567 • Updated the dockerfileCommitPR #589 after the demo • Listing part of status of requirements on this report • Implemented tests for PostView Post PageNavbar menuSearch userCreate Post
Ömer Barış Erkek	• Attended Discord meetings of our group • Attended Lab meetings of our group • Implemented my works in separate branches, created and managed issues for them, and opened pull requests for them when the work is completed • Tested and reviewed pull requests of my teammates • Written my Milestone 3 personal report • Contributed to Milestone 3 group report • Presented some of the Milestone 3 presentation • Created 12 example posts for our application • Implemented UI tests for Activation, ActivityStream, ForgotPassword, ForgotPasswordConfirm, SignIn, SignUp, TextChooser, and TimeChooser components • Documented all the components, functions, and tests that I implemented throughout the semester • Implemented ActivityStream component for the frontend as well as its backend integrations • Added ban checking functionality to SignIn component

Name	Work Done
Melih Özcan	• Attended the weekly meetings of the group, both in the LAB session and in Discord. • Implemented the taking photos with the phone camera functionality. Issue #459 • Implemented the post preview on the discover page functionality. Issue# 515 - Pull request #511 • Implemented filtering posts functionality. Issue #529 - Pull request #530 • Implemented the Activity Stream functionality. Issue #554 - Pull request #555 • Implemented the related tags functionality. Issue #557 • Implemented the clear filters functionality. Issue #574 - Pull request #575 • Did the presentation in the name of the Android department • Wrote a test for parsing the Post JSON response correctly The class can be seen here: PostModelConverterTest • Documented the code. Issue #597 - Pull request #598 • Helped with writing the user manual for the Android application. Issue #607 • Wrote the README file to show how to run the app. Issue #600
Ahmet İbrahim Şentürk	• Research and Implement a Test Issue #391 - PR #539 • Implement Filtering Functionality (Layout) - Issue #490 - PR #509 - PR #560 • Write Tests For Filtering Functionality - Issue #496 - PR #509 • Integrate Filtering Functionality with Backend - Issue #532 - PR #535 • Implement Wiki Data Functionalities - Issue #508 - PR #523 • Write Tests For Filtering Functionality/ Profile Page - Issue #540 - PR #545 • Guideline for the Final Presentation - Issue #536 • Document the Code and Tests - Issue #588 - PR #590 • Write Web User Manual for the pages: Discover, Proifle, and View Post
Niyazi Ülke	• Implemented unfollow functionality for Android. #488- PR 495 • Implemented private profile for Android. #527 - PR 537 • Connected activity streams page to the toolbar. #561 - PR 562 • Implemented follow request accept/reject page for Android. #576 - PR 573 • Implemented report functionality for Android. #580 - PR 534 • Implemented user search functionality for Android. #489 - PR 522 • Solved some bugs. #558 - PR 559, PR 569 • Wrote tests for HourMinuteHandler class. (Omitted in the last milestone) #582 - PR 563 • Wrote documentation for the codes. #599 - PR 604