CMPE 451

Project Development in Software Engineering

Fall 2021

Final Customer Milestone Report

12/01/2022

rebound

Group 2 - reBOUNd

Android Team	Backend Team	Frontend Team
Ahmet Necip Görgülü	Bengisu Özaydın	Berkay Döner
Mehmet Saraçoğlu	Ege Can Kaya	Doğukan Akar
Ömer Arslan	İbrahim Kağan Bayat	Şefika Akman
	Kerem Zaman	

Final Release:

https://github.com/bounswe/2021SpringGroup2/releases/tag/v0.9
https://github.com/bounswe/2021SpringGroup2/tree/v0.9

Report prepared by:

Ahmet Necip Görgülü
Berkay Döner
Ege Can Kaya
Doğukan Akar
Şefika Akman
Ömer Arslan

Executive Summary **3**
Status of Deliverables 3
Final Release Notes: 4
Changes we have planned/implemented to improve the development process: 5
Final Project Evaluations: 5

Summary of Work Done by Each Team Member **6**

Progress According to Requirements **10**
Glossary 11
1. Functional Requirements 11
1.1 User Requirements 11
1.2 System Requirements 12
2. Non-Functional Requirements 15

API Endpoints **17**

Implementation of Core Functionality **72**
Web 72
Android 76

User Interface/User Experience **76**
Homepage and navigation bar/header 76
Signup 79
Login 80
Forgot password 82
Profile Page 83
Create Event Page 85
Search Events 89
Comments & Answers Page 94

Standards **95**
W3C Activity Streams 2.0 95
Wikidata 96
User Manual 97
Frontend Application 97
1-) Homepage 97
2-) Sign up 97
3-) Login 99
4-) Forgot password 100
5-) Profile page 101
6-) New event 101
7-) Searching events 103
8-) Event pages 106
9-) Viewing applications 109
10-) Giving badges 111
11-) New equipment post 112

12-) Searching equipment posts 113
13-) Equipment pages 114
14-) Searching users 115
Android Application 116
1-) Homepage 116
2-) Sign up 117
3-) Login 118
4-) Forgot password 119
5-) Profile page 120
6-) New event 121
7-) Searching events 122
8-) Event pages 123
9-) Viewing applications 124
10-) Giving badges 125
11-) New equipment post 126
12-) Equipment pages 127
13-) Searching users 128
14-) User Page 129
SYSTEM MANUAL 130
Project plan 130
UML Diagrams 132
Class Diagram 132
Use Case Diagram 133
Sequence Diagrams 134
Scenarios and Mockups 145
1. Creating an Event 145
2. User Searches for an Event 152
3. Player & Spectator Application 158
4. Event Creator Selecting from Applicants 169
5. Discussion 177

Executive Summary

Status of Deliverables

Deliverable	Status
Frontend Application	Completed (with some missing requirements)
Android Application	Completed (with some missing requirements)
Backend Application	Completed (with some missing requirements)
System Manual	Completed
User Manual	Completed
API documentation	Completed
Requirements Specification	Completed
Software Design Documents	Completed
Project Plan	Completed
Milestone Report	Completed

Final Release Notes:

Version: v0.9
Date: January 6, 2022 (Initial tag was released on January 4, 2022 but we forgot to add Android files, we have just added them on January 6.)
This is the final release of reBOUNd and it is now ready for its users.

New Features Until Previous Release:

- User Search
- Discussion
- Applying to events
- Creating Equipment

- Follow/Unfollow (Android)
- Viewing events a user has participated in
- Viewing events a user has created
- Viewing players of an event
- Viewing spectators of an event
- Viewing followings
- Giving, receiving, viewing badges
- Related Equipment
- Viewing, accepting, rejecting event applicants

Changes we have planned/implemented to improve the development process:

For the past 2 milestones we have been struggling to meet our implementation goals. We were stalling behind our project plan and we needed to find a way to resolve this problem. We needed to not only keep up the pace, but we needed to work extra time to deliver the requirements we were meant to deliver. In order to do that we decided to tackle 2 main issues we were facing throughout this semester.

First one was problems arising from task distribution. For the earlier milestones we were advancing very slowly since only a portion of our team was able to handle their load and the rest needed help from them. This made it hard for us to run parallel tasks. As time went on and the competence of our team members have risen, our task distribution became a lot more efficient.

The other problem was our backend falling behind other sub-groups. For the past 2 milestones this has been a major problem since it made it hard for other 2 sub-groups to proceed without the necessary API's or the API documentation. Thankfully, they have made huge progress during the preparation for this final milestone and started leading the android-frontend subteams towards the end. This made everything proceed a lot faster and we managed to implement the majority of our requirements.

Final Project Evaluations:

We have managed to implement the core functionalities of our system and we were able to show them in the Final Demo. We were also able to use the W3C Activity Stream 2.0 and Wikidata standards in our system through API calls and badge functionality. Unfortunately, we realized that we misunderstood some of the requirements and encountered minor problems during the demo. For example, we had not implemented a search functionality based on the numeric distance and a circle area, we implemented a rectangular search functionality based on coordinates and did not take numeric distance as input. Such misunderstandings show that we could ask more questions to get more feedback during the customer meetings.

We realized for big presentations like this, we need meticulous preparation. Since we had a very tight schedule to meet every requirement, it was not possible for us to prepare for this Final Milestone Presentation sufficiently. This resulted in some errors during the presentation. It is ultimately a planning error and taught us about the importance of good planning.

Summary of Work Done by Each Team Member

Doğukan Akar	• Fixing bugs on the search and signup pages • Creating card views for all entities • Creating search pages for event, equipment and users • Reviewing pull requests • Fixing bugs on the routing • Creating profile page • Implementing controllers for search and recommendation
Şefika Akman	• Applied and owned events for profile page added. (issue:#354, pr:#361) • Equipment cards for event page added. (issue:#351, pr:#361) • Event cards for equipment page added. (issue:#339, pr:#361) • Sport type for search events added and fixed. (issue:#333, pr:#286) • Images of event pages are added and fixed. (issue:#332, pr:#338) • Search button for logged in user added. (issue:#325, pr:#326) • Navigation restart function added. (issue:#314, pr:#332) • View equipment page created. (issue:#287, pr:#315) • Access and refresh token working fixed. (issue:#278, pr:#313) • Pull requests reviewed and added comments for them. • Organized navigation buttons for logged in and logged out users.

Ömer Arslan	• FragmentGiveABadge class is created. GiveBadge button is added and connected to the network. Badge name, desc and images implemented from the getBadges response, showLoading added. • FragmentRelatedEvent class is created. New cardview is generated for showing the related events. • Badges fields added to the profile and searched user's profiles to show the badges of the users. • ViewBadgesInDetailFragment is created. ViewBadges buttons added to profile and searched users' profiles. getBadgeByName endpoint is created and connected. • acceptOrReject and getApplicants endpoints are created and connected. • A new cardview is generated for viewing the applicants. Accept, reject and showProfile button is added to the cardview and connected to the endpoints. • New recyclerview adapter is implemented for showing the cardviews. • Another recyclerview adapter is implemented for showing the spectator cardviews. Accept, reject and showProfile button is added to the cardview and connected the endpoints. • Logout button is generated, placed in the profile page and implemented to redirect to the login page.
İbrahim Kağan Bayat	• Implementing get comments by id function. Commit Pull request issue • Implementing get answers of a comment function. Commit Pull Request Issue • Implementing posting an answer to a comment function. Commit Pull Request Issue

	• Writing test functions for comments and answers Commit Issue Pull Request • Reviewing pull request for applying event Pull Request • Reviewing pull request for getting profiles with both username or user id. Pull Request • Reviewing pull request for get post events converted to activity stream response. Pull Request • Reviewing pull request for comments. Also models.py is written by me. Pull Request • Reviewing pull request for searching user by their usernames. Pull Request • Reviewing pull request of searching events by their applicant, players, spectators and also a bug fix in date. Pull Request
Berkay Döner	• [Frontend] Implemented the create equipment page. **Related Issues: #365** **Related Pull Requests: #328** • [Frontend] Improving forgot password page. **Related Issues: #367** **Related Pull Requests: #304** • [Frontend] Implementing badge related functionalities. **Related Issues: #295** **Related Pull Requests: #320** • [Frontend] Implemented the event application related functionalities. **Related Issues: #300** **Related Pull Requests: #321, #327** • [Frontend] Integrated the comment and answers to event&equipment pages. **Related Pull Requests: #347** • [Frontend] Implemented forgot password unit tests, #304 • [Frontend] Implemented create equipment page unit tests, #350

	• [Frontend] Implemented badge unit tests, #352. • [Frontend] Implemented event application unit tests, #355. • Reported bugs #294 and #306. • reviewed #266, #296, #297, #307, #326, #329, #331, #366 • Wrote user manual for frontend application, contributed to all sections of the milestone report.
Ahmet Necip Görgülü	• Adjusting View All Events and Search Events functionalities so they are compatible with pagination • Adding working OnClickListeners to Recycle Adapters • Implementing View Events a User Has Created • Implementing User Search Functionality • Implementing Detailed User Result Page • Implementing Discussion Functionality • Implementing Create Equipment Functionality • Implementing Follow – Unfollow functionalities • Reworking the User Search API • Implementing “View Events a User Has Participated In” functionality • Implementing “View players-spectators In an Event” functionality • QOL changes on MapView • Related Equipment On Detailed Event Page • Writing the Android User Manual
Ege Can Kaya	• Implemented Equipments API • Implemented user lookup by ID • Implemented Badges API • Implemented profile privacy settings • Implemented Comments/Answers API • Implemented User Profile search • Added additional search fields for

	event searching • Reviewed and contributed to the implementation of the Follow/Block API • Numerous bug fixes • Review of Docker file update • Wrote unit tests for Equipments • Wrote the API documentation for Equipments and Badges, reviewed and improved all other API documentations • Wrote the API documentation and Standards parts of the milestone report • Details of the above work, along with the relevant issues and pull requests are already given in my individual report.
Bengisu Özaydın	• Writing API documentation for follow & block functionalities. Wiki + Issue • Implementing Follow & Block API. Issue + (Also linked commits in this issue) • Reviewing the pull request for Comments & Answers API. PR + Issue • Revision of Follow & Block API documentation. Wiki + Issue
Mehmet Saraçoğlu	• Added the View Event Detailed part. • Reverted the old - broken files. • Reverted the broken commits and pull requests. • PR: 273, 283 • Issue: 260
Kerem Zaman	• Writing API documentation for applying to events and viewing/accepting/rejecting applicants • Implementing the functionalities for applying to an event as player or

	spectator and viewing/accepting/rejecting applicants (Issue #275 , PR #276) • Writing unit tests for event application functionalities (Issue #275 , PR #276) • Updating Dockerfile for recently added Django applications (Issue #302, PR #303) • Reviewing equipment post functionality and making some changes (PR #292) • Reviewing badges API (PR #297) • Reviewing comments & answer functionalities (PR #309) • Reviewing privacy settings implementation (PR #298) • Deploying the application several times

Progress According to Requirements

Addressed requirements are displayed in green, if they are fully implemented and in yellow if they are implemented partially. Some addressed requirements have explanations at the end.

Glossary

1. Functional Requirements

1.1 User Requirements

1.1.1. Sign Up

1.1.1.1. Users shall register to be a registered user.
1.1.1.2. Users shall specify their email, password and username while registering.
1.1.1.3. Users shall accept the terms and conditions document to register. (Android team implemented a checkbox to accept these but we do not have such documents yet.)
1.1.1.4. Users shall fill personal information form while registering.
1.1.1.5. Users should select their interests and skills to get suggestions

from the system (Users can select their interests however the suggestion feature is missing).

1.1.2. Sign In

1.1.2.1. Registered users shall enter their email or username and password to sign in.

1.1.3. Registered Users

1.1.3.1. Registered users shall have a personal profile page.
1.1.3.2. Registered users shall not be blocked by the user to view a user's profile page.
1.1.3.3. User should be registered in order to write comments under posts.

1.1.4. Search

1.1.4.1 Registered users shall use the search feature to find users or posts.

1.1.5. Post

1.1.5.1 Event Post

1.1.5.1.1 Registered users shall enter the values of the event to create an event post.
1.1.5.1.2 Event creators shall specify the type of sports events, manage location and type of sports events.
1.1.5.1.3 Event creators shall select the participants among the applicants.
1.1.5.1.4 Event creators shall report game results along with analytics.
1.1.5.1.5 Users shall indicate the type of sport and location in order to filter events.
1.1.5.1.6 Registered users shall specify their interests and skills to be able to get event recommendations.

1.1.5.2 Equipment Post

1.1.5.2.1 Registered users shall enter the values of the equipment to create an equipment post.
1.1.5.2.2 Registered users shall select the proper post and enter the contents of a comment to enter a comment.
1.1.5.2.3 Users shall indicate the type of sport and location in order to filter equipment.

1.1.5.2.4 Registered users shall specify their interests and skills to be able to get equipment recommendations.

1.1.6. Notification

1.1.6.1 Registered users shall select which notifications that they will see.
1.1.6.2 Registered users shall enable or disable all or a subset of notifications.

1.2 System Requirements

1.2.1 Sign Up

1.2.1.1. System shall ask for username, email and password for regular signup.
1.2.1.2. System shall send verification mail after regular signup.
1.2.1.3. System shall redirect users to the profile creation page immediately after signup (Signup page is also the profile creation page).
1.2.1.4. System shall not allow you to sign up with the same email more than once at the same time.

1.2.2 Sign In

1.2.2.1. System shall allow users to sign in with email/username.
1.2.2.2. System shall allow users to logout by clicking a button.

1.2.3 User Types

1.2.3.1. Banned User
1.2.3.2. Regular User
1.2.3.3. Administrator

1.2.4 User Profile

1.2.4.1. System shall ask user their interests and skills while creating profile
1.2.4.2. Interests and skills shall be updated according to games registered to system by admins
1.2.4.3. User profile shall display personal information, skills, badges, followers, followings and post history.
1.2.4.4. System shall allow users to edit their profile page and password. (Edit functionality for profile page is implemented but has

bugs also password cannot be changed for now.)
1.2.4.5. System shall allow users to delete their profiles.

1.2.5 Events

1.2.5.1. System should recommend sports events to the users according to their interests.
1.2.5.2. System shall send emails and push notifications when a registered event approaches.
1.2.5.3. System shall allow event creators to specify event type, needed number of players, location of event, event time, skills of participants, age range of participants.
1.2.5.4 Events with needed users registered shall be signed as full, so other people can filter them.
1.2.5.5 System shall allow guests to view the events list and individual event pages, without participating in them.
1.2.5.6 All users shall be allowed to list all events on the events page.
1.2.5.7 Event pages shall display the event creator, event location, sports field, applicants, participants and spectators of the event.
1.2.5.8 System shall allow all users to view the profile pages of the event creator, any of the applicants, participants and the spectators.
1.2.5.9 System shall allow only registered users to apply to events.

1.2.6 Searching

1.2.6.1 Event Search

1.2.6.1.1. Events shall be filtered by user according to event type, appropriate age range, location, date, hours, skill levels of attendees and availability . (availability is missing)
1.2.6.1.2. Events shall be sorted with respect to name, time and location (e.g. from closer events to far away ones).

1.2.6.2 User Search

1.2.6.2.1. Users shall be filtered according to their ages, skills and interested sports.
1.2.6.2.2. Users shall be sorted with respect to their names and ages.

1.2.7 Sport Equipment Post

1.2.7.1. System shall recommend some websites to the users for their equipment needs. (Websites are not recommended; however, there are equipment recommendations for the same sport in the event pages.)

1.2.7.2. System shall allow filtering and sorting the equipment websites according to the name of the website and name of the sport.
1.2.7.3. System shall allow equipment posts to be edited by their posters.

1.2.8 Reporting

1.2.8.1. System shall allow users to report some problems to event creators in the event.
1.2.8.2. System shall allow users to report some bugs to administrators.
1.2.8.3. System shall allow users to report some users for misusage.

1.2.9 Discussions

1.2.9.1. System shall allow users to discuss under event or equipment post by writing comments.
1.2.9.2. System shall also allow users to answer/reply discussion comments.
1.2.9.3. Users shall not be able to reply to the answers/replies, thus replies shall be the deepest level of conversations.

1.2.10 Notifications

1.2.10.1. System shall notify users when their player/spectator applications are approved/rejected.
1.2.10.2. System shall notify users for any related event happening around if they share their locations.
1.2.10.3. Users shall be notified if some people withdraw from an event after the event is signed as full. (Both registered users and users signed as want to attend but can't register because of capacity)
1.2.10.4. Users shall be notified if the event they applied/selected to be a player/spectator is delayed/cancelled.
1.2.10.5. Users shall be notified when a user follows them.
1.2.10.6. Users shall be notified when a user comments on the discussion page of an event they created.

1.2.11 Badges

1.2.11.1. There shall be badges that are given by the system after some milestones, that are given by the event creators at the end of the event and that are given by the users.
1.2.11.2. There should be badges that indicate bad behavior.

1.2.11.3 Only registered users shall be allowed to give badges to the creator of a post or to fellow players of an event.

1.2.12 User Interactions

1.2.12.1. System shall allow registered users to follow other users, so that they can be notified about the events created by the users they follow (Android supports following users but there are no notifications. Frontend doesn't have follow feature).
1.2.12.2. System shall allow users to block other users.

1.2.13 Analytics

1.2.13.1. System shall ask for and store analytics for players, event creators and sports fields after events.
1.2.13.2. System shall allow users to see the analytics for players, event creators and sports fields.

2. Non-Functional Requirements

2.1. Availability

2.1.1. System shall be available as a web application for modern browsers such as Chrome and Firefox.
2.1.2. System shall be available for Android based devices.
2.1.3. English language shall be supported.

2.2. Deployment

2.2.1. System should be containerized using Docker.

2.2.2. System should run on an Amazon EC2 server. (We decided to try Google Cloud and deployed our application there, we might change it to Amazon EC2 in the future.)

2.3. Performance

2.3.1. Response time for any request shall be below 5 seconds.

2.4. Privacy

2.4.1. All personal data shall be stored and processed according to KVKK and GDPR.
2.4.2. Users shall accept the privacy policy when signing up to the system.

2.4.3. Users shall be able to choose which of their personal data to be seen publicly.

2.5. Security

2.5.1. Passwords shall be at least 8 characters long and be a mixture of lowercase characters, uppercase characters and digits.
2.5.2. Passwords shall be encrypted using SHA-256 before storing.
2.5.3. Users shall verify their login information via email after 3 unsuccessful login attempts.
2.5.4. All inputs from users shall be checked to prevent SQL injections. (We have implemented various input validation methods; however, we are not sure if we cover all inputs.)
2.5.5. There shall be a password reset facility that sends a verification code via email or SMS.
2.5.6. System should only allow predetermined known file types to be uploaded. (e.g. PNG and JPG for images)
2.5.7. System should hide the passwords while typing on the password bar.

2.6. Standards

2.6.1. System shall follow the standards introduced by W3C Activity Stream Standard 2.0. (Not every API follows Activity Stream)
2.6.2. The semantic taggings should be supported with Wikidata.

API Endpoints

Link to the API: http://34.68.66.109/

Login (obtain JWT token)

POST 'api/token/obtain/'

- **Author:** Ege Can Kaya
- **Definition:** Takes a set of user credentials and returns an access and refresh JSON web token pair to prove the authentication of those credentials.
- **Example Response Value:**

Example Response Value:

{

"refresh": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0b2tlbl90eXBlIjoicmVmcmVzaCIsImV4cCI6MTYzODAzMzQ0OCwiaWF0IjoxNjM2ODIzODQ4LCJqdGkiOiJiNDkxZjIwZjU2OTY0YTJiYjI2OTIyNWJjMmZkNTc3YSIsInVzZXJfaWQiOjR9.tCGeBNzTXXBA8fTC2eqxyy2wfa1tYEN8nL0mSCzftP0",
"access": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjM2ODI0MTQ4LCJpYXQiOjE2MzY4MjM4NDgsImp0aSI6ImY0ZWIyNDY5ZWIwZDQyMDQ4NzI1NTAyNDQ5YjZlYTM2IiwidXNlcl9pZCI6NH0.HgN9BIi3F-1lnio_s32IHdsHNDUlAuNcWaE_s9A11vY"
}

Example Failure Value:

{
"detail": "No active account found with the given credentials"
}

- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Username	Body	String
password	Password	Body	String

- **Response Messages**
  - 200: OK.
  - 401: Unauthorized.

Login (refresh JWT token)

POST 'api/token/obtain/'

- **Author:** Ege Can Kaya
- **Definition:** Takes a refresh type JSON web token and returns an access type JSON web token if the refresh token is valid.

- **Example Response Value:**

Example Response Value:

{
"access": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjM2ODI0NDUxLCJpYXQiOjE2MzY4MjM4NDgsImp0aSI6IjY3ZTRlZDE2NzQyYTQyYTZiOGY2OTBiZTFmMTUzZjQ1IiwidXNlcl9pZCI6NH0.J9WhNq04miuDSABsobZtPHS5drcXxcq0eQE6MH44-zU",
"refresh": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0b2tlbl90eXBlIjoicmVmcmVzaCIsImV4cCI6MTYzODAzMzc1MSwiaWF0IjoxNjM2ODI0MTUxLCJqdGkiOiI0NjQ0MDFjNGY2ZjA0OTA3YjZlZTA3NDA0ZmQ0ZmI0MiIsInVzZXJfaWQiOjR9.r4q3jpcilwT6bqXupFqhSnu7pc58IzPNJXilnfwhwwM"
}

Example Failure Value:

{
"detail": "Token is invalid or expired",
"code": "token_not_valid"
}

- **Parameters**

Parameter	Description	Parameter Type	Data Type
refresh	Refresh type JWT token	Body	String

- **Response Messages**
  - 200: OK.
  - 401: Unauthorized.

Register

POST 'api/user/create/'

- **Author:** Ege Can Kaya
- **Definition:** Allows a user to register
- **Example Response Value:**

Example Response Value:

```
{
  "email": "egekaya@somemail.com",
  "username": "egekaya"
}
```

Example Failure Value:

```
{
  "username": [
    "This field is required."
  ],
  "password": [
    "This field is required."
  ]
}
```

- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
username	Username of registering user	Body	String
email	Email address of registering user	Body	String
password	Password of registering user	Body	String

- **Response Messages**
    - 201: Created.
    - 401: Bad request.

Reset password (send mail)

POST 'api/password/reset/'

- **Author:** Ege Can Kaya
- **Definition:** Provides a method to request a password reset token based on an e-mail address
- **Example Response Value:**

Example Response Value:


```
{
  "status": "OK"
}
```


Example Failure Value:


```
{
  "email": [
   "We couldn't find an account associated with that email. Please try a different e-mail address."
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
Email	Email address of user trying to reset their password	Body	String

- **Response Messages**

    - 200: OK.
    - 401: Bad request.

Reset password (confirmation)

POST 'api/password/reset/'

- **Author:** Ege Can Kaya
- **Definition:** Provides a method to reset a password based on a unique token
- **Example Response Value:**

Example Response Value:

```
{
  "status": "OK"
}
```

Example Failure Value:

```
{
  "detail": "Not found."
}
```

- **Parameters**

Parameter	Description	Parameter Type	Data Type
Password	New password for the user	Body	String
Token	Token received by email by the user allowing them to choose a new password	Body	String

- **Response Messages**

    - 200: OK.
    - 401: Bad request.

View profile

GET 'api/users/<username>/'

- **Author:** Ege Can Kaya
- **Definition:** Views the profile of a user
- **Example Response Value:**

Example Response Value:

```
{
  "first_name": "Ege",
  "last_name": "Kaya",
  "bio": "Hello, I love sports.",
  "fav_sport_1": "football",
  "fav_sport_2": "basketball",
  "fav_sport_3": "tennis",
  "location": "Istanbul"
}
```

Example Failure Value:

```
{
  "detail": "Not found."
}
```

- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Username of profile to view	Path	String

- **Response Messages**
    - 200: OK.
    - 404: Not Found.

Edit profile

PUT 'api/users/<username>/'

- **Author:** Ege Can Kaya
- **Definition:** Allows a user to edit their profile
- **Example Response Value:**

Example Response Value:


```
{
    "first_name": "Lewis",
    "last_name": "Hamilton",
    "bio": "7-time world champion",
    "fav_sport_1": "Formula 1",
    "fav_sport_2": "Formula 2",
    "fav_sport_3": "Formula E",
    "location": "Monaco",
    "avatar": "https://upload.wikimedia.org/wikipedia/commons/1/18/Lewis_Hamilton_2016_Malaysia_2.jpg",
    "privacy": false
}
```


Example Failure Value:


```
{
    "detail": "Unauthorized."
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type

username	Username of profile to edit	Path	String
first_name	New value for first name	Body	String
last_name	New value for last name	Body	String
bio	New value for bio	Body	String
fav_sport_1	New value for favorite sport 1	Body	String
fav_sport_2	New value for favorite sport 2	Body	String
fav_sport_3	New value for favorite sport 3	Body	String
location	New value for location	Body	String
avatar	New value for avatar	Body	String
privacy	New value for privacy	Body	Boolean

- **Response Messages**
  - 200: OK.
  - 401: Unauthorized.

Search profiles

GET 'api/users/?query=<query>'

- **Author:** Ege Can Kaya
- **Definition:** Searches profiles by username according to given query
- **Example Response Value:**

Example Response Value:


```
{
   "results": [
      {
         "id": 4,
         "username": "egecky",
         "avatar": "",
         "privacy": false
      }
   ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
query	query string for the search	queru	String

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Get event by ID

GET 'api/posts/<id>/'

- **Author:** Doğukan Akar

- **Definition:** Allows a user to fetch an event from database
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Sally created an event",
  "type": "Create",
  "actor": {
    "type": "Person",
    "name": "Sally"
  },
  "object": {
    "type": "Event",
    "postId": "",
    "ownerId": "",
    "content": "",
    "title": "Beginner friendly tennis game",
    "creationDate": "2014-11-31T23:00:00-08:00",
    "lastUpdateDate": "2014-11-31T23:00:00-08:00",
    "numberOfClicks": 0,
    "location": {
      "name": "Etiler Tennis Club",
      "type": "Place",
      "longitude": 12.34,
      "latitude": 56.78,
      "altitude": 90,
      "units": "m"
    },
    "eventDate": "2014-12-31T23 00 00-08 00",
    "eventSport": "Tennis",
    "eventMinAge": 16,
    "eventMaxAge": 18,
    "eventMinSkillLevel": 1,
    "eventMaxSkillLevel": 5,
    "eventPlayerCapacity": 12,
    "eventSpectatorCapacity": 12,
    "eventApplicants": [1,2,3],
    "eventPlayers": [1,2,3]
```

```
  }
}
```


Example Failure Value:


```
{
  "detail": "Not found."
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
id	ID of the searched event	Path	Int

- **Response Messages**
  - 200: OK.
  - 404: Not Found.

Get all events

GET 'api/posts/'

- **Author:** Doğukan Akar
- **Definition:** Returns a list of post ids using filters.
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Object history",
  "type": "OrderedCollection",
  "totalItems": 2,
  "totalPages": 2,
  "orderedItems": [
    {
```

```
        "type": "Create",
        "actor": {
            "type": "Person",
            "name": "Sally"
        },
        "object": "/api/posts/1"
    },
    {
        "type": "Create",
        "actor": {
            "type": "Person",
            "name": "Sally"
        },
        "object": "/api/posts/2"
    }
  ]
}
```


- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
query	Query string	Query	String
creation_date	Creation date of the searched event	Body	DateTime
location	Location of the searched event	Body	String
date	Date of the searched event	Body	DateTime

duration	Duration of the searched event	Body	DateTime
sport	Sport type of the searched event	Body	DateTime
age_group	Which age group is preferred for the searched event	Body	Integer
player_capacity	Player capacity of the searched event	Body	Intege
spec_capacity	Spectator capacity of the searched event	Body	Integer
players	Players of the searched event	Body	Integer
spectators	Spectators of the searched event	Body	Integer
latitude	Latitude of the searched event	Body	Float
longitude	Longitude of the searched event	Body	Float

owner	Owner of the searched event	Body	User

- **Response Messages**
  - 200: OK.
  - 500: Internal server error.

Get events in pages

GET 'api/posts/?page=<page>'

- **Author:** Kerem Zaman
- **Definition:** Returns a list of post ids using filters.
- **Example Response Value:**

Example Response Value:


```
{
   "count": 26,
   "next": null,
   "previous": "http://localhost:9000/api/posts/?page=2",
   "results": [
      {
         "id": 25,
         "content": "Samba!",
         "title": "Let the Rhythm",
         "creation_date": "2022-01-12T19:44:17.802799Z",
         "location": "Rio",
         "sport": "Dance",
         "latitude": 41.015740309484656,
         "longitude": 28.96322853241089,
         "date": "2022-01-14T21:30:37Z",
         "duration": 389,
         "min_age": 18,
         "max_age": 60,
         "player_capacity": 15,
         "spec_capacity": 0,
         "players": [],
         "player_applicants": [],
```

"spec_applicants": [],
"spectators": [],
"min_skill_level": 1,
"max_skill_level": 3,
"owner": 15
},
{
"id": 14,
"content": "We're going to have a Moonboard session at Boulderhane. We're going to be battling Moonboard problems ranging from V4 to v6.",
"title": "Quick Moonboard Sesh",
"creation_date": "2022-01-12T17:41:34.373028Z",
"location": "Boulderhane",
"sport": "Climbing",
"latitude": 41.09022,
"longitude": 29.00322,
"date": "2022-01-20T16:00:07Z",
"duration": 120,
"min_age": 15,
"max_age": 60,
"player_capacity": 10,
"spec_capacity": 0,
"players": [],
"player_applicants": [],
"spec_applicants": [],
"spectators": [],
"min_skill_level": 2,
"max_skill_level": 4,
"owner": 18
},
{
"id": 20,
"content": "Not fighting, just having fun.",
"title": "Box Games",
"creation_date": "2022-01-12T19:30:27.361420Z",
"location": "Istanbul",
"sport": "Boxing",
"latitude": 41.0082,
"longitude": 28.9784,
"date": "2022-01-13T19:30:14Z",

"duration": 15840,
"min_age": 18,
"max_age": 50,
"player_capacity": 12,
"spec_capacity": 5,
"players": [],
"player_applicants": [],
"spec_applicants": [],
"spectators": [],
"min_skill_level": 0,
"max_skill_level": 0,
"owner": 6
},
{
"id": 21,
"content": "Join us.",
"title": "Foooooootballllllll",
"creation_date": "2022-01-12T19:31:07.508868Z",
"location": "Istanbul",
"sport": "Football",
"latitude": 41.0082,
"longitude": 28.9784,
"date": "2022-01-27T19:30:50Z",
"duration": 5760,
"min_age": 15,
"max_age": 80,
"player_capacity": 50,
"spec_capacity": 100,
"players": [],
"player_applicants": [],
"spec_applicants": [],
"spectators": [],
"min_skill_level": 1,
"max_skill_level": 4,
"owner": 6
},
{
"id": 22,
"content": "Somebody finds a surfing place, we are there.",
"title": "Surfing",

"creation_date": "2022-01-12T19:32:19.500976Z",
"location": "Mugla",
"sport": "Surfing",
"latitude": 37.1642053,
"longitude": 28.2624288,
"date": "2022-01-26T19:32:01Z",
"duration": 221760,
"min_age": 18,
"max_age": 45,
"player_capacity": 20,
"spec_capacity": 0,
"players": [],
"player_applicants": [],
"spec_applicants": [],
"spectators": [],
"min_skill_level": 3,
"max_skill_level": 5,
"owner": 6
},
{
"id": 26,
"content": "We are cycling to draw attention to the climate crisis!",
"title": "Let's cycle for the environment",
"creation_date": "2022-01-12T19:50:58.299250Z",
"location": "Bibliothèque de SciencesPo",
"sport": "Road cycling",
"latitude": 48.85427,
"longitude": 2.328145176292373,
"date": "2022-01-14T12:30:00Z",
"duration": 180,
"min_age": 18,
"max_age": 50,
"player_capacity": 15,
"spec_capacity": 0,
"players": [],
"player_applicants": [],
"spec_applicants": [],
"spectators": [],
"min_skill_level": 1,
"max_skill_level": 5,

```
        "owner": 14
      }
    ],
    "totalPages": 3
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
page	Page of the list of events	query	Integer

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Create an event

POST 'api/posts/'

- **Author:** Doğukan Akar
- **Definition:** Allows a user to create an event
- **Example Response Value:**

Example Response Value:


```
{
    "@context": "https://www.w3.org/ns/activitystreams",
    "summary": "Sally created an event",
    "type": "Create",
    "actor": {
        "type": "Person",
        "name": "Sally"
    },
    "object": {
        "type": "Event",
```

```
        "name": "A Simple Event",
        "postId": "",
        "ownerId": "",
        "content": "",
        "title": "Beginner friendly tennis game",
        "creationDate": "2014-11-31T23:00:00-08:00",
        "lastUpdateDate": "2014-11-31T23:00:00-08:00",
        "numberOfClicks": 0,
        "location": {
            "name": "Etiler Tennis Club",
            "type": "Place",
            "longitude": 12.34,
            "latitude": 56.78,
            "altitude": 90,
            "units": "m"
        },
        "eventDate": "2014-12-31T23 00 00-08 00",
        "eventSport": "Tennis",
        "eventMinAge": 16,
        "eventMaxAge": 18,
        "eventMinSkillLevel": 1,
        "eventMaxSkillLevel": 5,
        "eventPlayerCapacity": 12,
        "eventSpectatorCapacity": 12,
        "eventApplicants": [1,2,3],
        "eventPlayers": [1,2,3]
    }
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
query	Query string	Query	String

creation_date	Creation date of the searched event	Body	DateTime
location	Location of the created event	Body	String
date	Date of the created event	Body	DateTime
duration	Duration of the created event	Body	DateTime
sport	Sport type of the created event	Body	DateTime
age_group	Which age group is preferred for the created event	Body	Integer
player_capacity	Player capacity of the created event	Body	Intege
spec_capacity	Spectator capacity of the created event	Body	Integer
players	Players of the created event	Body	Integer

spectators	Spectators of the searched event	Body	Integer
latitude	Latitude of the created event	Body	Float
longitude	Longitude of the created event	Body	Float
owner	Owner of the created event	Body	User

- **Response Messages**
  - 201: Created.
  - 400: Bad request.

Apply to an event

POST 'api/posts/<id>/apply/'

- **Author:** Kerem Zaman
- **Definition:** Allows a user to apply for an event as a spectator or a player
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "1 applied to event 1 as spectator",
  "type": "Join",
  "actor": {
    "type": "Person",
    "name": "kerem"
  },
  "object": {
```

```
    "type": "Event",
    "name": "A Simple Event",
    "postId": 1,
    "ownerId": 1,
    "content": "nba fans gathering for the lakers vs miami match",
    "title": "basketball event",
    "creationDate": "2021-12-28T19:44:28.254597Z",
    "numberOfClicks": 0,
    "location": {
      "name": "Istanbul",
      "type": "Place",
      "longitude": 28.9784,
      "latitude": 41.0082,
      "units": "m"
    },
    "eventDate": "2022-01-03T10:00:00Z",
    "eventSport": "Basketball",
    "eventMinAge": 18,
    "eventMaxAge": 80,
    "eventMinSkillLevel": 1,
    "eventMaxSkillLevel": 4,
    "eventPlayerCapacity": 3,
    "eventSpectatorCapacity": 3,
    "eventApplicantsAsPlayer": [],
    "eventApplicantsAsSpectator": [
      1
    ],
    "eventPlayers": []
  }
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type

id	ID of the event to apply	path	Integer
user	ID of the user applying to the event	body	Integer
type	Type of the applicant (spectator/player)	body	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get list of users applied to an event

GET'api/posts/<id>/applicants'

- **Author:** Kerem Zaman
- **Definition:** Allows users to see spectator/player applicants for an event
- **Example Response Value:**

Example Response Value:

```
{
  "applicants": [
    1,
    2
  ]
}
```

- **Parameters**

Parameter	Description	Parameter Type	Data Type

id	ID of the event	path	Integer
type	Type of the applicant (spectator/player)	query	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Accept or reject users applied to an event

POST 'api/posts/<id>/applicants'

- **Author:** Kerem Zaman
- **Definition:** Allows event owners to accept or reject spectator/player applicants
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "1 accepted 3 to event 1 as player",
  "type": "Accept",
  "actor": {
    "type": "Person",
    "name": "3"
  },
  "object": {
    "type": "Event",
    "name": "A Simple Event",
    "postId": 1,
    "ownerId": 1,
    "content": "nba nba",
    "title": "basketball event",
    "creationDate": "2021-12-28T19:44:28.254597Z",
    "numberOfClicks": 0,
```

```
    "location": {
      "name": "Istanbul",
      "type": "Place",
      "longitude": 28.9784,
      "latitude": 41.0082,
      "units": "m"
    },
    "eventDate": "2022-01-03T10:00:00Z",
    "eventSport": "Basketball",
    "eventMinAge": 18,
    "eventMaxAge": 80,
    "eventMinSkillLevel": 1,
    "eventMaxSkillLevel": 4,
    "eventPlayerCapacity": 3,
    "eventSpectatorCapacity": 3,
    "eventApplicantsAsPlayer": [],
    "eventApplicantsAsSpectator": [
      1,
      2
    ],
    "eventPlayers": [
      3
    ]
  }
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
id	ID of the event	path	Integer

type	Type of the applicant (spectator/player)	query	String
user	ID of the user applying to the event	Body	Integer
owner	ID of the user hosts the event	Body	Integer
accept	status of the applicant	Body	Boolean

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get comment to a post by id

GET 'api/posts/<post_id>/comments/<comment_id>/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to fetch a comment made to a post
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Sally created a comment",
  "type": "Create",
  "actor": {
    "type": "Person",
    "name": "Sally"
  },
  "object": {
```

"type": "Comment",
"postId": "",
"ownerId": "",
"content": "A Simple Comment",
"creationDate": "2014-11-31T23:00:00-08:00"
}
}

- **Parameters**

Parameter	Description	Parameter Type	Data Type
postId	ID of the searched post	path	Integer
commentId	ID of the comment	path	Integer

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get all comments to a post

GET 'api/posts/<post_id>/comments/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to fetch all comments to a post
- **Example Response Value:**

Example Response Value:

{
"@context": "https://www.w3.org/ns/activitystreams",
"summary": "Object history",
"type": "Collection",

```
  "totalItems": 1,
  "items": [
    {
      "@context": "https://www.w3.org/ns/activitystreams",
      "summary": "egecky created a comment",
      "type": "Create",
      "actor": {
        "type": "Person",
        "name": "egecky"
      },
      "object": {
        "type": "Comment",
        "postId": 1,
        "id": 1,
        "ownerId": 4,
        "content": "i would love to join but i can't afford baklava if i lose",
        "creationDate": "2022-01-04T09:20:00.305006Z"
      }
    }
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
postId	ID of the searched post	path	Integer

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Create a comment to a post

POST 'api/posts/<post_id>/comments'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to add a comment to a post
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Sally created a comment",
  "type": "Create",
  "actor": {
    "type": "Person",
    "name": "Sally"
  },
  "object": {
    "type": "Comment",
    "postId": "",
    "ownerId": "",
    "content": "A Simple Comment",
    "creationDate": "2014-11-31T23:00:00-08:00"
  }
}
```


- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
postId	ID of the searched post	path	Integer
name	Username of the comment poster	Body	String

ownerId	ID of the comment owner	Body	Integer
content	Comment content	Body	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get all answers to a comment

GET 'api/posts/<post_id>/comments/<comment_id>/answers/'

- **Author:** İbrahim Kağan Bayat
- **Definition:** Allows a user to fetch all answers to a comment made to a post
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Object history",
  "type": "Collection",
  "totalItems": 2,
  "items": [
    {
      "type": "Create",
      "actor": {
        "type": "Person",
        "name": "Sally"
      },
      "object": {
        "answer": "I would like to participate",
        "creationDate": "2014-11-31T23:00:00-08:00"
      }
    },
    {
      "type": "Create",
```

```
        "actor": {
            "type": "Person",
            "name": "Sally"
        },
        "object": {
            "answer": "I wish I was available!",
            "creationDate": "2014-11-31T23:00:00-12:00"
        }
    }
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
postId	ID of the post	path	Integer
commentId	ID of the comment	path	Integer

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Create an answer to a comment

POST 'api/posts/<post_id>/comments/<comment_id>/answers/'

- **Author:** İbrahim Kağan Bayat
- **Definition:** Allows a user to add an answer to a comment made to a post
- **Example Response Value:**

Example Response Value:

```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Sally created an answer",
  "type": "Create",
  "actor": {
    "type": "Person",
    "name": "Sally"
  },
  "object": {
    "answer": "I wish I was available!",
    "creationDate": "2014-11-31T23:00:00-12:00"
  }
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
postId	ID of the post	path	Integer
commentId	ID of the comment	path	Integer
ownerId	Id of the answer poster	Body	Integer
content	answer content	Body	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get equipment by id

GET 'api/equipments/<id>/'

- **Author:** Ege Can Kaya
- **Definition:** Allows a user to fetch an equipment from database
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "MaxVerstappen posted an equipment",
  "type": "Create",
  "actor": {
    "type": "Person",
    "name": "MaxVerstappen"
  },
  "object": {
    "type": "Equipment",
    "postId": "",
    "ownerId": "",
    "content": "I can't drive it so I'm selling it",
    "title": "Unused F1 car with Honda engine",
    "creationDate": "2014-11-31T23:00:00-08:00",
    "lastUpdateDate": "2014-11-31T23:00:00-08:00",
    "numberOfClicks": 0,
    "location": {
      "name": "Red Bull Racing HQ, Milton Keynes",
      "type": "Place",
      "longitude": 52.007877,
      "latitude": -0.692203,
      "altitude": 76,
      "units": "m"
    },
    "url":"https://www.redbullracing.com/adverts",
    "sport": "F1",
    "equipmentType": "Car"
  }
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
id	ID of the searched equipment	path	Integer

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get all equipments

GET 'api/equipments/'

- **Author:** Ege Can Kaya
- **Definition:** Returns a list of post ids using filters.
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Object history",
  "type": "OrderedCollection",
  "totalItems": 2,
  "totalPages": 2,
  "orderedItems": [
    {
      "object": "/api/equipments/1/"
    },
    {
      "object": "/api/equipments/2/"
    }
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
query	Query string	Query	String
creation_date	Creation date of the searched equipment	Body	DateTime
location	Location of the searched equipment	Body	String
sport	Sport type of the searched equipment	Body	Char
latitude	Latitude of the searched event	Body	Float
longitude	Longitude of the searched event	Body	Float
owner	Owner of the searched event	Body	User
equipment_type	Equipment type of the searched equipment	Body	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get equipments in pages

GET 'api/equipments/?page=<page>'

- **Author:** Ege Can Kaya
- **Definition:** Returns a list of equipment ids using filters.
- **Example Response Value:**

Example Response Value:


```
{
    "count": 6,
    "next": null,
    "previous": null,
    "results": [
        {
            "id": 1,
            "content": "Frisbee in good condition from my grandmother",
            "title": "Frisbee in good condition",
            "creation_date": "2022-01-04T09:32:29.513208Z",
            "location": "Bangkok",
            "sport": "Ultimate Frisbee",
            "latitude": 13.019945,
            "longitude": 99.9979,
            "url": "frisbee.com",
            "equipment_type": "Frisbee",
            "owner": 5
        },
        {
            "id": 2,
            "content": "Only the best!",
            "title": "Football",
            "creation_date": "2022-01-04T10:14:44.812102Z",
            "location": "Bangkok Football Store",
            "sport": "Soccer",
            "latitude": 13.7525438,
            "longitude": 100.4934734,
            "url": "www.football.com",
            "equipment_type": "Ball",
            "owner": 8
```

},
{
"id": 3,
"content": "Only the best!",
"title": "Frisbee",
"creation_date": "2022-01-04T10:16:08.718667Z",
"location": "Bangkok Frisbee Store",
"sport": "Ultimate Frisbee",
"latitude": 13.8245796,
"longitude": 100.6224463,
"url": "www.frisbee.com",
"equipment_type": "Frisbee",
"owner": 8
},
{
"id": 4,
"content": "I own my skiing equipment for years but I did not have the opportunity to use it often so it is almost good as new! I bought the equipment from the link below three years ago.",
"title": "I am selling my skiing equipment",
"creation_date": "2022-01-12T17:02:27.722834Z",
"location": "Beykoz",
"sport": "Skiing",
"latitude": 41.1239355,
"longitude": 29.1083151,
"url": "https://www.hepsiburada.com/elan-maxx-siyah-kirmizi-qs-el-4-5-p-HBCV000011391H",
"equipment_type": "Skiing Gear",
"owner": 8
},
{
"id": 5,
"content": "I have a spare unopened mouthpiece that I can give someone free of charge!",
"title": "Spare unopened mouthpiece",
"creation_date": "2022-01-12T17:37:09.622921Z",
"location": "Boxing gym",
"sport": "Boxing",
"latitude": 41.113503,
"longitude": 28.9991,

"url": "none",
"equipment_type": "Mouthpiece",
"owner": 18
},
{
"id": 6,
"content": "I was eager to learn golf but couldn't continue, so I'm selling my golf sticks, they were rarely used and in good condition.",
"title": "2 years old golf sticks but rarely used",
"creation_date": "2022-01-12T20:00:00.554507Z",
"location": "Esenkent",
"sport": "Golf",
"latitude": 41.04156850593763,
"longitude": 28.673046107869602,
"url": "https://www.sahibinden.com/listing/ikinci-el-ve-sifir-alisveris-spor-doga-sporlari-ping-4-iron-tertemiz-990368232/detail",
"equipment_type": "Golf Stick",
"owner": 2
}
],
"totalPages": 1
}

- **Parameters**

Parameter	Description	Parameter Type	Data Type
page	Page of the list of events	Query	Integer

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Create an equipment

POST 'api/equipments/'

- **Author:** Ege Can Kaya
- **Definition:** Allows a user to create an event
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "MaxVerstappen posted an equipment",
  "type": "Create",
  "actor": {
    "type": "Person",
    "name": "MaxVerstappen"
  },
  "object": {
    "type": "Equipment",
    "postId": "",
    "ownerId": "",
    "content": "I can't drive it so I'm selling it",
    "title": "Unused F1 car with Honda engine",
    "creationDate": "2014-11-31T23:00:00-08:00",
    "lastUpdateDate": "2014-11-31T23:00:00-08:00",
    "numberOfClicks": 0,
    "location": {
      "name": "Red Bull Racing HQ, Milton Keynes",
      "type": "Place",
      "longitude": 52.007877,
      "latitude": -0.692203,
      "altitude": 76,
      "units": "m"
    },
    "url":"https://www.redbullracing.com/adverts",
    "sport": "F1",
    "equipmentType": "Car"
  }
}
```

- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
content	Content of the created equipment	Body	String
title	Title of the created equipment	Body	String
location	Location of the created equipment	Body	String
sport	Sport type of the created equipment	Body	String
latitude	Latitude of the created equipment	Body	Float
longitude	Longitude of the created equipment	Body	Float
owner	Owner of the created equipment	Body	User
equipment_type	Type of the created equipment	Body	String

url	URL of the created equipment	Body	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Follow user

GET 'api/users/<username>/follow/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to follow another user
- **Example Response Value:**

Example Response Value:

```
{
  "message": "Successfully followed user."
}
```

- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
username	Name of followed user	Path	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Unfollow user

GET 'api/users/<username>/unfollow/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to unfollow another user
- **Example Response Value:**

Example Response Value:

```
{
   "message": "Successfully unfollowed user."
}
```

- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Name of unfollowed user	Path	String

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Get followings of user

GET 'api/users/<username>/followings/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to see users followed by a user
- **Example Response Value:**

Example Response Value:

```
{
   "@context": "https://www.w3.org/ns/activitystreams",
   "summary": "Users Sally follows.",
   "type": "Collection",
   "total_items": 1,
   "items": [
```

```
    {
      "@context": "https://www.w3.org/ns/activitystreams",
      "summary": "Sally followed John",
      "type": "Follow",
      "actor": {
        "type": "Person",
        "name": "Sally"
      },
      "object": {
        "type": "Person",
        "identifier": "John"
      }
    }
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Name of user to get followings of	Path	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get followers of user

GET 'api/users/<username>/followers/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to see users following a user
- **Example Response Value:**

Example Response Value:


```
{
```

```
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Users following Sally.",
  "type": "Collection",
  "total_items": 2,
  "items": [
    {
      "@context": "https://www.w3.org/ns/activitystreams",
      "summary": "John followed Sally",
      "type": "Follow",
      "actor": {
        "type": "Person",
        "name": "John"
      },
      "object": {
        "type": "Person",
        "identifier": "Sally"
      }
    },
    {
      "@context": "https://www.w3.org/ns/activitystreams",
      "summary": "James followed Sally",
      "type": "Follow",
      "actor": {
        "type": "Person",
        "name": "James"
      },
      "object": {
        "type": "Person",
        "identifier": "Sally"
      }
    }

  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type

username	Name of user to get followers of	Path	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Block user

GET 'api/users/<username>/block/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to block another user
- **Example Response Value:**

Example Response Value:

```
{
  "message": "Successfully blocked user."
}
```

- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
username	Name of blocked user	Path	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Unblock user

GET 'api/users/<username>/unblock/'

- **Author:** Bengisu Özaydın

- **Definition:** Allows a user to block another user
- **Example Response Value:**

Example Response Value:


```
{
  "message": "Successfully unblocked user."
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Name of unblocked user	Path	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get blockings of user

GET 'api/users/<username>/blockings/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows a user to see the blocked users
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Users Sally blocked.",
  "type": "Collection",
  "total_items": 1,
  "items": [
    {
      "@context": "https://www.w3.org/ns/activitystreams",
```

```
        "summary": "Sally blocked John",
        "type": "Block",
        "actor": {
            "type": "Person",
            "name": "Sally"
        },
        "object": {
            "type": "Person",
            "identifier": "John"
        }
    }
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Name of user to get blockings of	Path	String

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Get blockers of user

GET 'api/users/<username>/blockers/'

- **Author:** Bengisu Özaydın
- **Definition:** Allows to get users that blocked a user
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Users that blocked Sally.",
```

```
  "type": "Collection",
  "total_items": 1,
  "items": [
    {
      "@context": "https://www.w3.org/ns/activitystreams",
      "summary": "John blocked Sally",
      "type": "Block",
      "actor": {
        "type": "Person",
        "name": "John"
      },
      "object": {
        "type": "Person",
        "identifier": "Sally"
      }
    }
  ]
}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
username	Name of user to get blockers of	Path	String

- **Response Messages**
  - 200: OK.
  - 401: Bad request.

Get badge by name

GET 'api/badges/<badge_name>/'

- **Author:** Berkay Döner
- **Definition:** Allows a user to fetch a badge by name
- **Example Response Value:**

Example Response Value:


```
{
    "@context": "https://www.wikidata.org/wiki/Q491",
    "type": "Badge",
    "name": "Friendly",
    "content": "This user is friendly towards other team players",
    "icon": {
        "type": "Image",
        "content":<base64_string>
    }

}
```


- **Parameters**

Parameter	Description	Parameter Type	Data Type
badge_name	Name of badge	Path	String

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Get all badges

GET 'api/badges/'

- **Author:** Berkay Döner
- **Definition:** Allows a user to fetch all badges in the database
- **Example Response Value:**

Example Response Value:


```
{
    "@context": "https://www.w3.org/ns/activitystreams",
    "summary": "Badge list",
```

```
  "type": "Collection",
  "totalItems": 2,
  "items": [
    {
      "object": {
        "type": "Badge",
        "name": "Friendly",
        "@context": "https://www.wikidata.org/wiki/Q491",
        "content": "This user is friendly towards other team players",
        "icon": {
            "type": "Image",
            "content":<base64_string>
        }
      }
    },
    {
      "object": {
        "type": "Badge",
        "name": "Fair play",
        "@context": " https://www.wikidata.org/wiki/Q995616",
        "content": "This user plays fairly",
        "icon": {
          "type": "Image",
          "content":<base64_string>
        }
      }
    }
  ]
}
```


- **Response Messages**
    - 200: OK.

Get all related events

GET 'api/users/<username>/related-events/'

- **Author:** Berkay Döner
- **Definition:** Allows a user to fetch all events of the target user where the user can give a badge

- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Event list",
  "type": "Collection",
  "totalItems": 2,
  "items": [
    {
      "object": {
        "type": "Event",
        "postId": 1,
        "title": "Beginner friendly tennis game",
        "eventSport": "Tennis",
        "eventDate": "2014-12-31T23 00 00-08 00"
      }
    },
    {
      "object": {
        "type": "Event",
        "postId": 2,
        "title": "Beginner friendly football game",
        "eventSport": "Football",
        "eventDate": "2014-12-31T23 00 00-08 00"
      }
    }
  ]
}
```


- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
target	Username of the target user	Path	String

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Give a badge to a user

POST 'api/users/<username>/badges/'

- **Author:** Berkay Döner
- **Definition:** Allows a user to give a badge to another user
- **Example Response Value:**

Example Response Value:


```
{
  "@context": "https://www.w3.org/ns/activitystreams",
  "summary": "Sally gave badge to Lewis",
  "type": "Offer",
  "actor": {
    "type": "Person",
    "name": "Sally",
    "id":  "1"
  },
  "object": {
    "type": "Badge",
    "name": "Friendly",
    "attributedTo": [
    {
      "type": "Event",
      "id": "2"
    }
  ]
  },
  "target": {
    "type": "Person",
    "name": "Lewis",
    "id":  "2"
  }
}
```

- **Parameters**

Parameter	**Description**	**Parameter Type**	**Data Type**
target	Username of the target user	Path	String
badge_name	Name of the badge to be given	Body	String
event_id	ID of the event where badge is awarded	Body	Integer

- **Response Messages**
    - 200: OK.
    - 401: Bad request.

Example API calls:

GET http://34.68.66.109/api/posts/1/comments/

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
{
    "username":"ali-arslan",
    "email":"aliarslan@gmail.com",
    "password":"12345678"
}
```

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize JSON


```
{
    "@context": "https://www.w3.org/ns/activitystreams",
    "summary": "Object history",
    "type": "Collection",
    "totalItems": 1,
    "items": [
        {
            "@context": "https://www.w3.org/ns/activitystreams",
            "summary": "egecky created a comment",
            "type": "Create",
            "actor": {
                "type": "Person",
                "name": "egecky"
            },
            "object": {
                "type": "Comment",
                "postId": 1,
                "id": 1,
                "ownerId": 4,
                "content": "i would love to join but i can't afford baklava if i lose",
                "creationDate": "2022-01-04T09:20:00.305006Z"
            }
        }
    ]
}
```

POST http://34.68.66.109/api/equipments/

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1  {
2      "title":"ball for sale",
3      "content":"i can't use my ball so i'm selling it",
4      "owner":8,
5      "sport":"Football",
6      "location":"Istanbul",
7      "equipment_type":"ball",
8      "url":"www.google.com"
9  }
```

Body Cookies Headers (11) Test Results

Pretty Raw Preview Visualize JSON

```
1  {
2      "@context": "https://www.w3.org/ns/activitystreams",
3      "summary": "egecky posted an equipment",
4      "type": "Create",
5      "actor": {
6          "type": "Person",
7          "name": "egecky"
8      },
9      "object": {
10         "type": "Equipment",
11         "postId": 2,
12         "ownerId": 8,
13         "content": "i can't use my ball so i'm selling it",
14         "title": "ball for sale",
15         "creationDate": "2022-01-03T22:20:15.835445Z",
16         "numberOfClicks": 0,
17         "location": {
18             "name": "Istanbul",
19             "type": "Place",
20             "longitude": 1.0,
21             "latitude": 1.0,
22             "units": "m"
23         },
```

POST http://34.68.66.109/api/posts/

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
{
    "title":"an empty event",
    "content":"oh no i forgot to fill it out",
    "owner":3,
    "sport":"f1"
}
```

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize JSON


```
{
    "@context": "https://www.w3.org/ns/activitystreams",
    "summary": "berkay123 created an event",
    "type": "Create",
    "actor": {
        "type": "Person",
        "name": "berkay123"
    },
    "object": {
        "type": "Event",
        "name": "A Simple Event",
        "postId": 12,
        "ownerId": 3,
        "content": "oh no i forgot to fill it out",
        "title": "an empty event",
        "creationDate": "2021-12-14T11:48:47.235892Z",
        "numberOfClicks": 0,
        "location": {
            "name": "",
            "type": "Place",
            "longitude": 1.0,
            "latitude": 1.0,
            "units": "m"
        }
    },
    "eventDate": "2021-12-14T11:48:47.236310Z",
    "eventSport": "f1",
    "eventMinAge": 18,
    "eventMaxAge": 75,
    "eventMinSkillLevel": 0,
    "eventMaxSkillLevel": 0,
    "eventPlayerCapacity": 10,
    "eventSpectatorCapacity": 0,
    "eventApplicants": [],
    "eventPlayers": []
}
```


Implementation of Core Functionality

Web

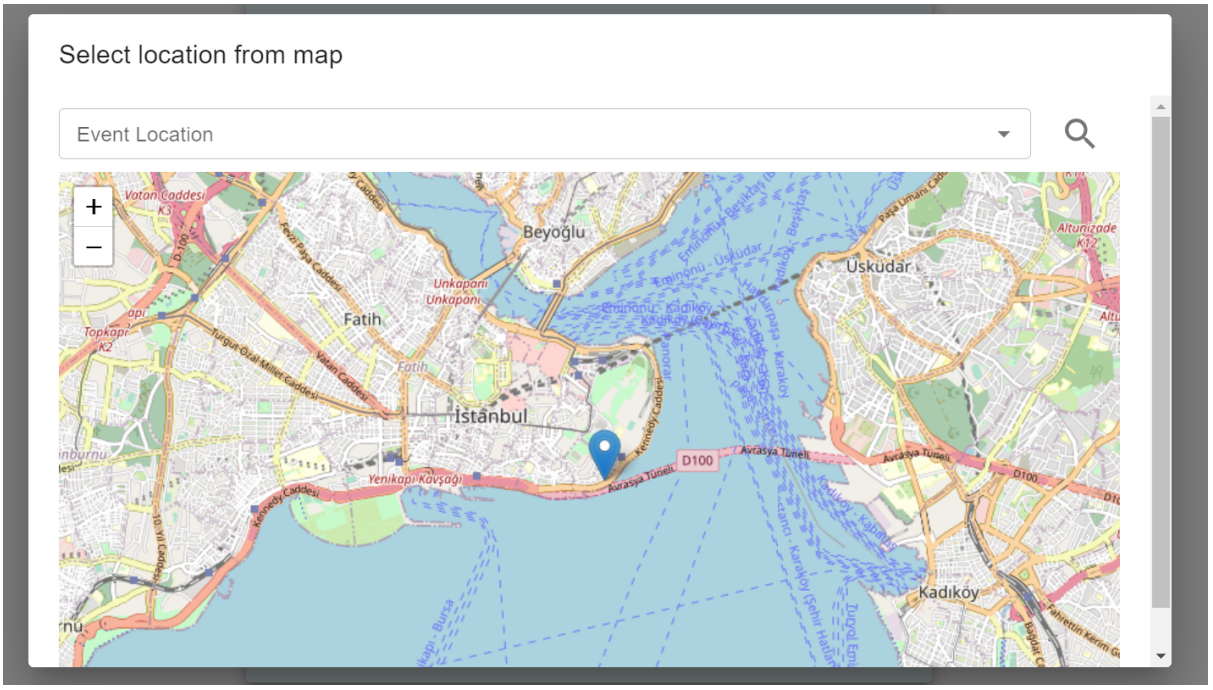
Status: We already had the functionality to signup login and create events. We had a map to choose location while creating events. We also have a map for searching events in a specified area, so Raul can search for the events near Marsi Hotel now. He can apply for the game. The event creator can accept or reject his application. In the event page they can see the event players (Raul also). So they can click to the Rauls avatar and go to Raul's profile page. Then, they can see the events that Raul played on his timeline. We haven't implemented the following functionality yet.

Detailed Work: For both maps, we used the leaflet and react-leaflet libraries, which support OpenStreetMap. We tried to use open source libraries in our application since our application itself is open source and we want to encourage collaboration.

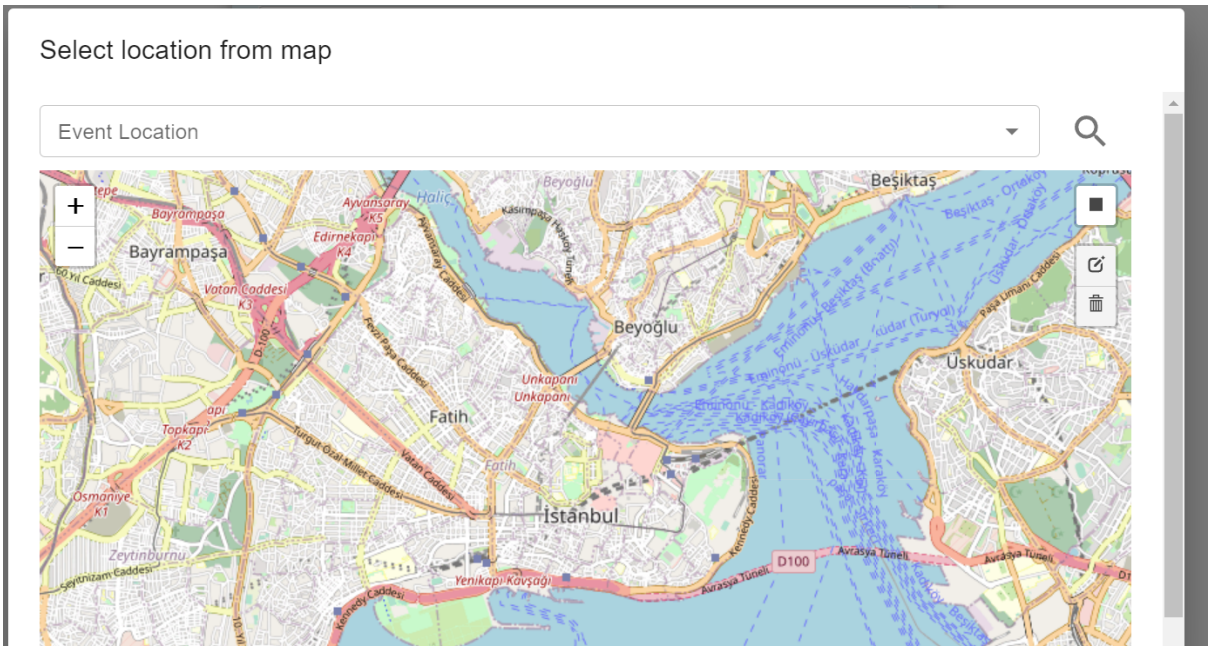
In the search page, we implemented rectangular area selection using the leaflet-draw and react-leaflet-draw libraries in addition to leaflet. Users can select the rectangle button on the right to start the drawing process. When a rectangle is drawn, the drawing button gets disabled since users are only allowed to specify one area.

In the profile page, we used material ui card components to show the events. It uses the search API with parameters to get events that Raul played.

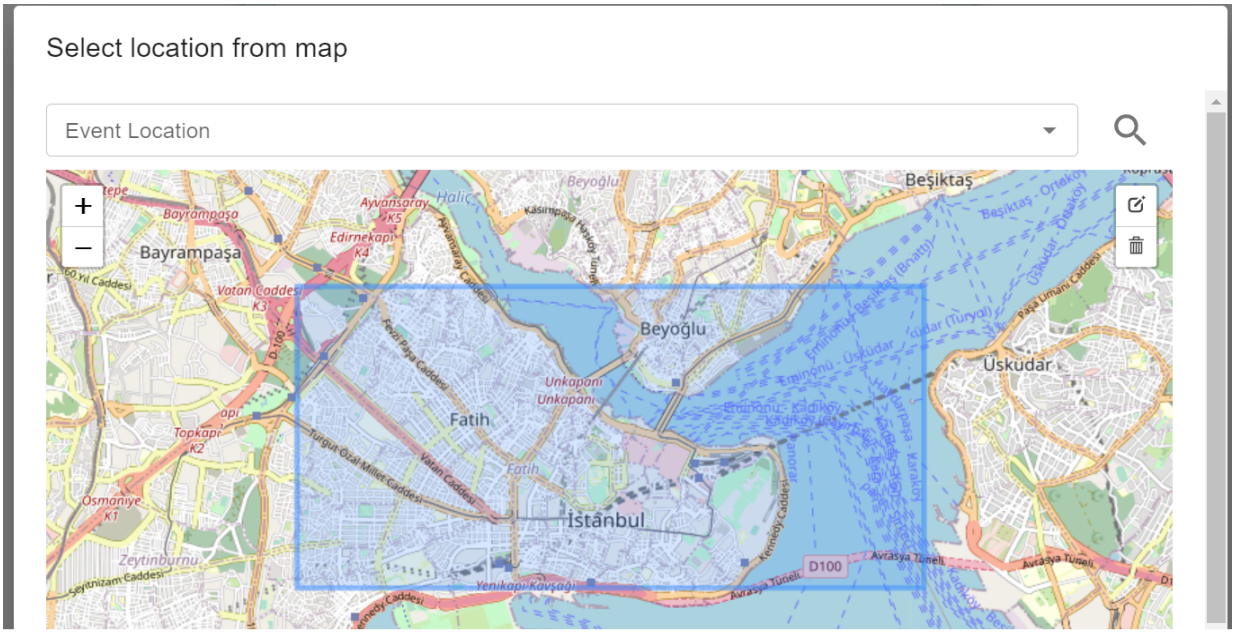
Future work: We can add the following users functionality. We could add a button to the profile page and users could follow each other using that button. Then some other functionality like homepage feed and notifications could be implemented.



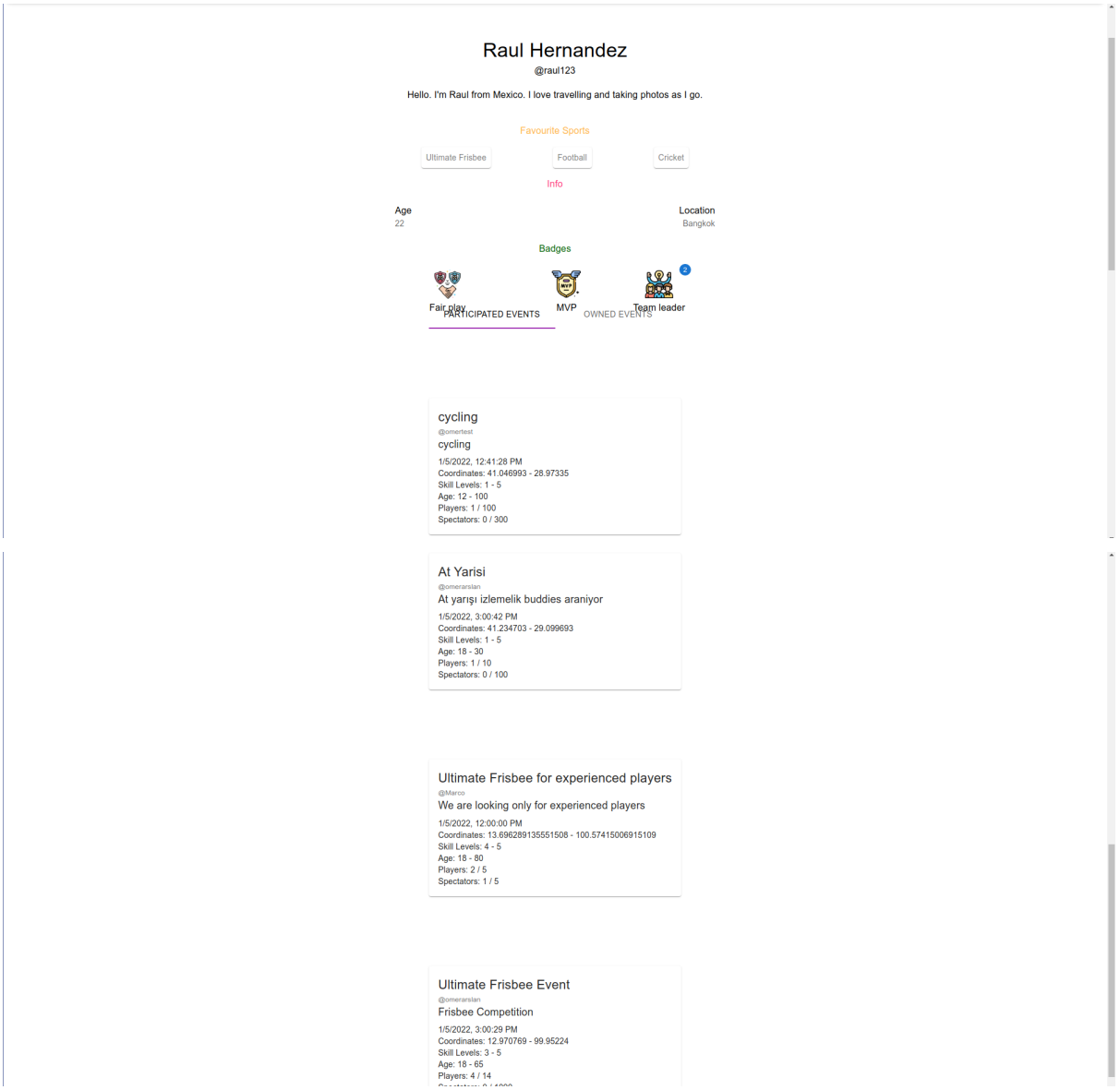
Map in the create event page



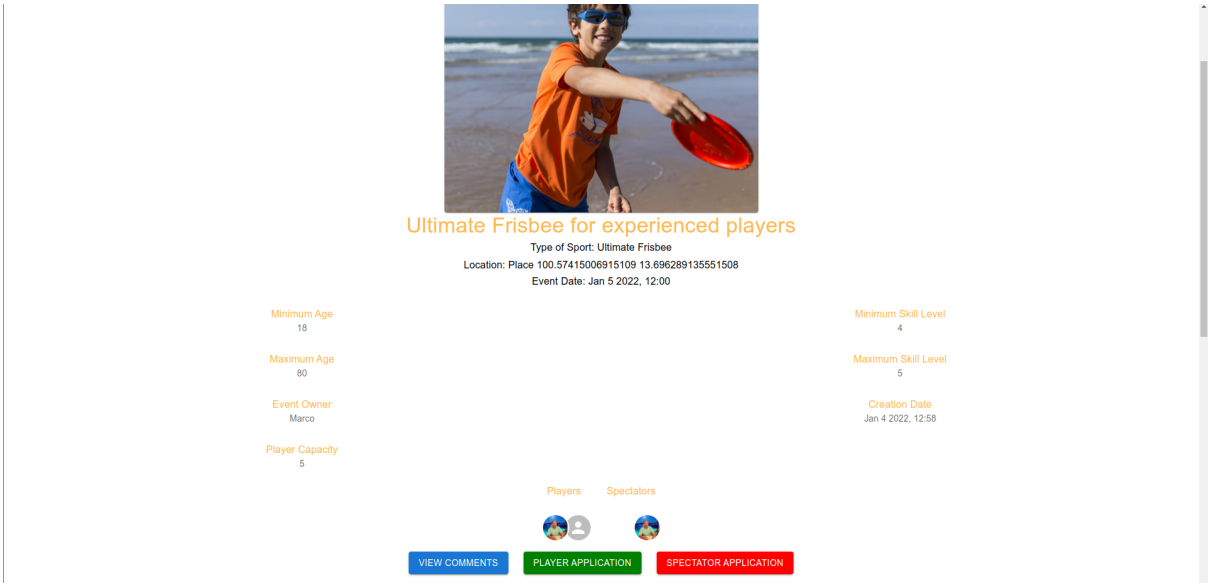
Map in the search page, before choosing a rectangular area



Map in the search page, after choosing a rectangular area



Profile page of Raul. Here is the events he participated



Event Page (You can see the player application button)

Android

Our application right now has all the functionalities Raul needs to complete the core scenario. Raul can search for an event around Bangkok using our Event Search functionality. Then he can apply for the event. He can see all the players participating in the event, he can look into their profile and follow them. He can see the previous events these players have participated in. He can give badges to these players he has played with and he can also receive badges from them.

User Interface/User Experience

Homepage and navigation bar/header

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/Home

Android Code:
2021SpringGroup2/FragmentViewAllEvents.kt at android · bounswe/2021SpringGroup2 (github.com)

rebound

HOME | SEARCH | SIGN UP | LOG IN

Discover Sports

Soccer

Sport that is played mostly outdoors, on grass, soccer (or football in Europe) opposes two teams. The objective is simple: send, by using mostly your feet, the ball to the opposing team's goal to score points. Ready to use a quick feint to shimmy past your opponent?

SEE SOCCER EVENTS

Basketball

Competitive and recreative sport opposing two teams, basketball is played on a court where two hoops are suspended in height at each extremity. With your team, your goal is to score points by throwing the ball into the basket. You will have to do some feints and dribble the ball to progress on the court and get as close as possible to the basket before throwing the ball. Can you jump high enough to make a dunker?

SEE BASKETBALL EVENTS

Volleyball

Volleyball is played for fun or during competition, inside or outside, with two teams on each side of a net on a delimited court. By only using your hands and your arms to send the ball on the other side of the net to score points, you'll make wonderful exchanges with your teammates. But for that, you will need to use your calm to receive the ball with precision and power to return it with great propulsion!

SEE VOLLEYBALL EVENTS

Browse Events

Do you even lift bro

@dogukanakar

Come lift with me in uçaksavar

1/8/2022, 4:00:08 PM
Coordinates: 41.0854789 - 29.040732305033927
Skill Levels: 4 - 5
Age: 18 - 80
Players: 0 / 2
Spectators: 0 / 0

Let's do kickboxing and forget out problems

@bikembengisu

We will warm-up extensively, do sparring, do mock-matches and end with an extensive cool-down

1/31/2022, 3:00:47 PM
Coordinates: 41.0640428 - 29.009908569247614
Skill Levels: 2 - 5
Age: 12 - 45
Players: 0 / 12
Spectators: 0 / 20

Advanced Tennis Game

@sefikaakman

I am organizing a professional tennis event. If you are a self confident person,

1/5/2022, 12:28:51 PM
Coordinates: 41.093435205265195 - 29.051799774169925
Skill Levels: 3 - 5
Age: 18 - 45
Players: 0 / 20
Spectators: 0 / 50

rebound

HOME | SEARCH | NEW EVENT | NEW EQUIPMENT | PROFILE | LOG OUT

Discover Sports

Soccer

Sport that is played mostly outdoors, on grass, soccer (or football in Europe) opposes two teams. The objective is simple: send, by using mostly your feet, the ball to the opposing team's goal to score points. Ready to use a quick feint to shimmy past your opponent?

SEE SOCCER EVENTS

Basketball

Competitive and recreative sport opposing two teams, basketball is played on a court where two hoops are suspended in height at each extremity. With your team, your goal is to score points by throwing the ball into the basket. You will have to do some feints and dribble the ball to progress on the court and get as close as possible to the basket before throwing the ball. Can you jump high enough to make a dunker?

SEE BASKETBALL EVENTS

Volleyball

Volleyball is played for fun or during competition, inside or outside, with two teams on each side of a net on a delimited court. By only using your hands and your arms to send the ball on the other side of the net to score points, you'll make wonderful exchanges with your teammates. But for that, you will need to use your calm to receive the ball with precision and power to return it with great propulsion!

SEE VOLLEYBALL EVENTS

Browse Events

Do you even lift bro
@dogukanakar
Come lift with me in uçaksavar
1/8/2022, 4:00:08 PM
Coordinates: 41.0854789 - 29.040732305033927
Skill Levels: 4 - 5
Age: 18 - 80
Players: 0 / 2
Spectators: 0 / 0

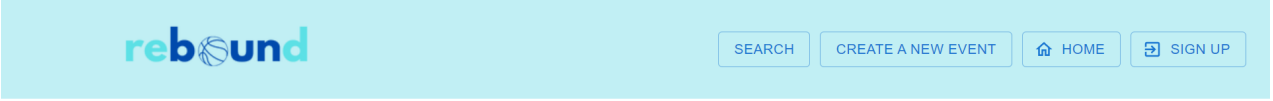
Let's do kickboxing and forget out problems
@bikembengisu
We will warm-up extensively, do sparring, do mock-matches and end with an extensive cool-down
1/31/2022, 3:00:47 PM
Coordinates: 41.0640428 - 29.009908569247614
Skill Levels: 2 - 5
Age: 12 - 45
Players: 0 / 12
Spectators: 0 / 20

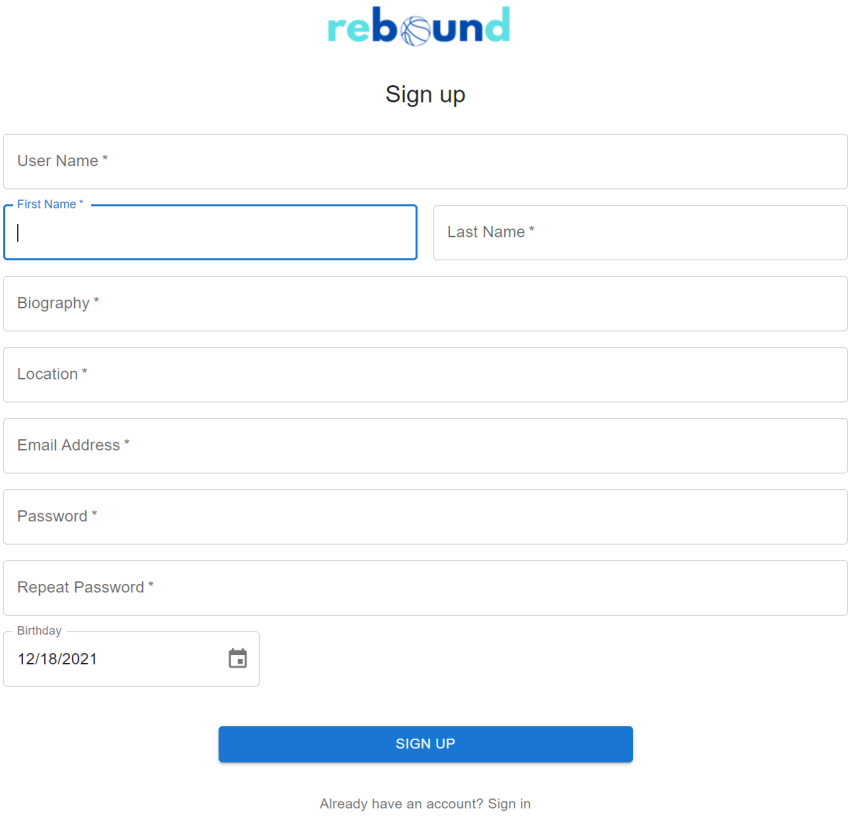
Advanced Tennis Game
@sefikaakman
I am organizing a professional tennis event. If you are a self confident person,
1/5/2022, 12:28:51 PM
Coordinates: 41.093435205265195 - 29.051799774169925
Skill Levels: 3 - 5
Age: 18 - 45
Players: 0 / 20
Spectators: 0 / 50

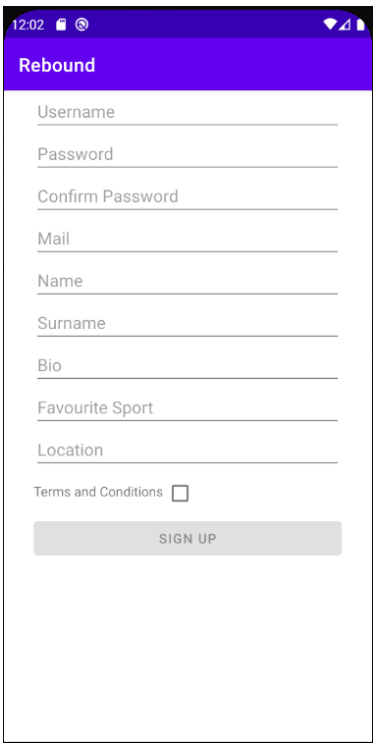
Signup

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/Signup

Android Code:
2021SpringGroup2/SignupActivity.kt at android · bounswe/2021SpringGroup2 (github.com)



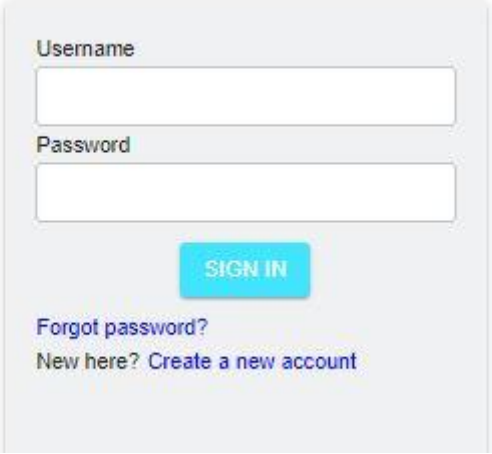


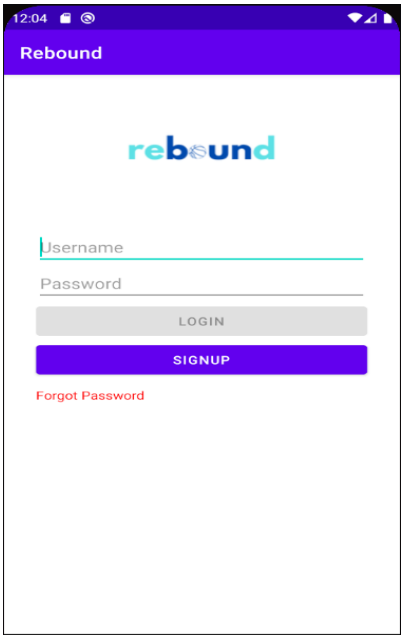


Login

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/Login

Android Code:
2021SpringGroup2/LoginActivity.kt at android · bounswe/2021SpringGroup2 (github.com)

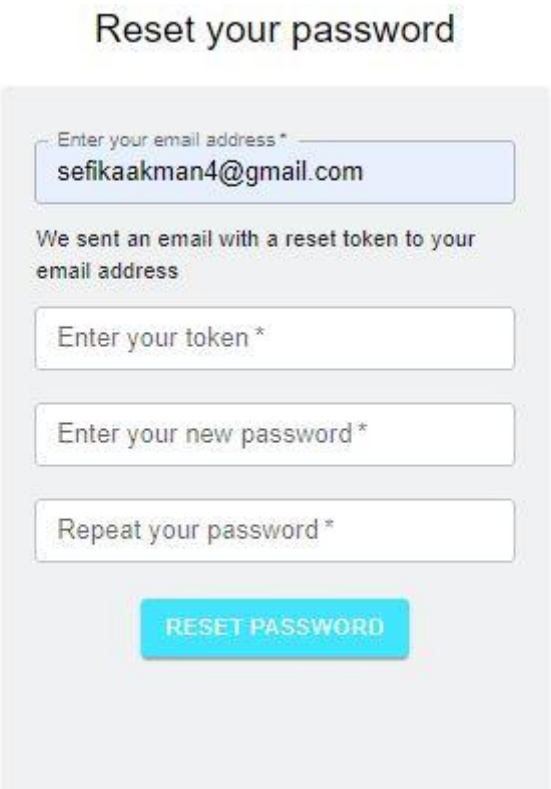

Username
Password
SIGN IN
Forgot password?
New here? Create a new account

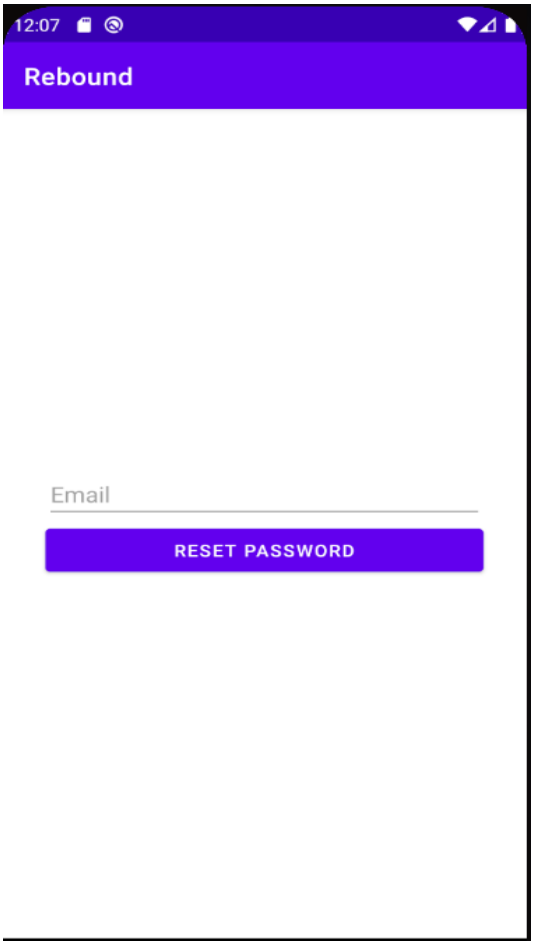

12:04
Rebound
rebound
Username
Password
LOGIN
SIGNUP
Forgot Password

Forgot password

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/blob/development/frontend/src/Views/Login/ForgotPassword.js

Android Code:
2021SpringGroup2/ForgotPasswordActivity.kt at android · bounswe/2021SpringGroup2 (github.com) - 2021SpringGroup2/ResetPasswordActivity.kt at android · bounswe/2021SpringGroup2 (github.com)





Profile Page

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/Profile

Android Code:
2021SpringGroup2/FragmentProfile.kt at android · bounswe/2021SpringGroup2 (github.com)

rebound

HOME | SEARCH | NEW EVENT | NEW EQUIPMENT | PROFILE | LOG OUT

Sefika Akman

@sefikaakman

Favourite Sports

Info

Age
22

Location
Istanbul

PARTICIPATED EVENTS | OWNED EVENTS

Ultimate Frisbee Event

@omerarslan

Frisbee Competition

1/5/2022, 3:00:29 PM
Coordinates: 12.970769 - 99.95224
Skill Levels: 3 - 5
Age: 18 - 65
Players: 4 / 14
Spectators: 0 / 1000

rebound

HOME | SEARCH | NEW EVENT | NEW EQUIPMENT | PROFILE | LOG OUT

Sefika Akman

@sefikaakman

Favourite Sports

Info

Age
22

Location
Istanbul

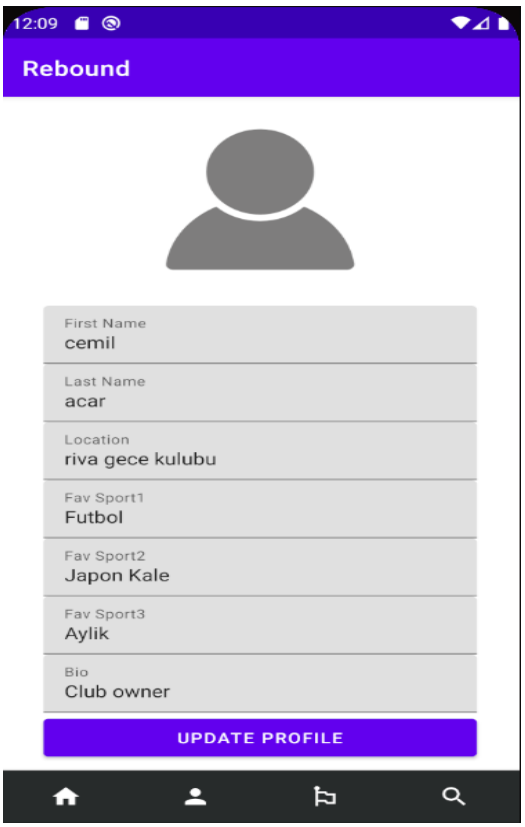
PARTICIPATED EVENTS | OWNED EVENTS

Advanced Tennis Game

@sefikaakman

I am organizing a professional tennis event. If you are a self confident person,

1/5/2022, 12:28:51 PM
Coordinates: 41.093435205265195 - 29.051799774169925
Skill Levels: 3 - 5
Age: 18 - 45
Players: 0 / 20
Spectators: 0 / 50

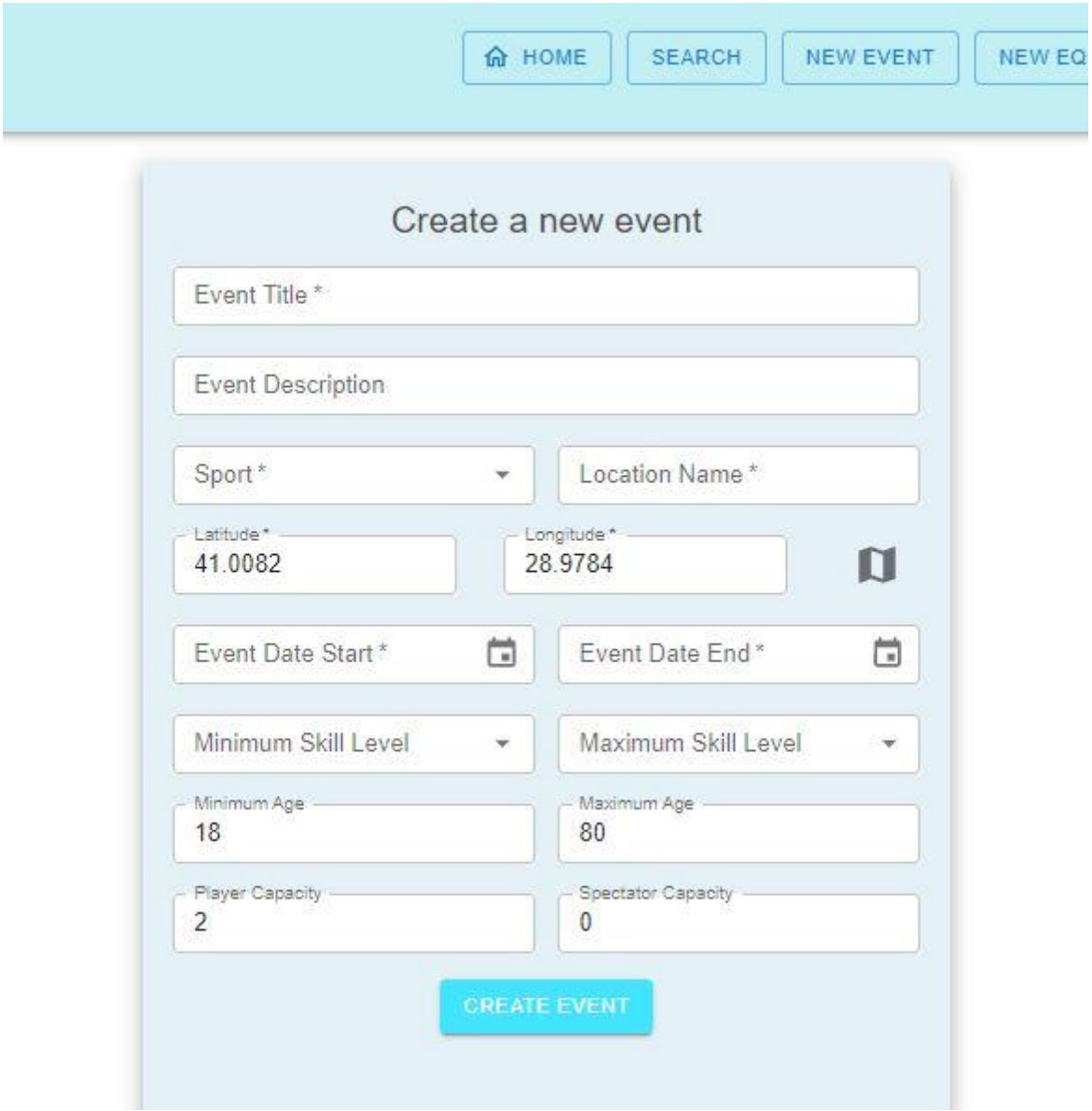


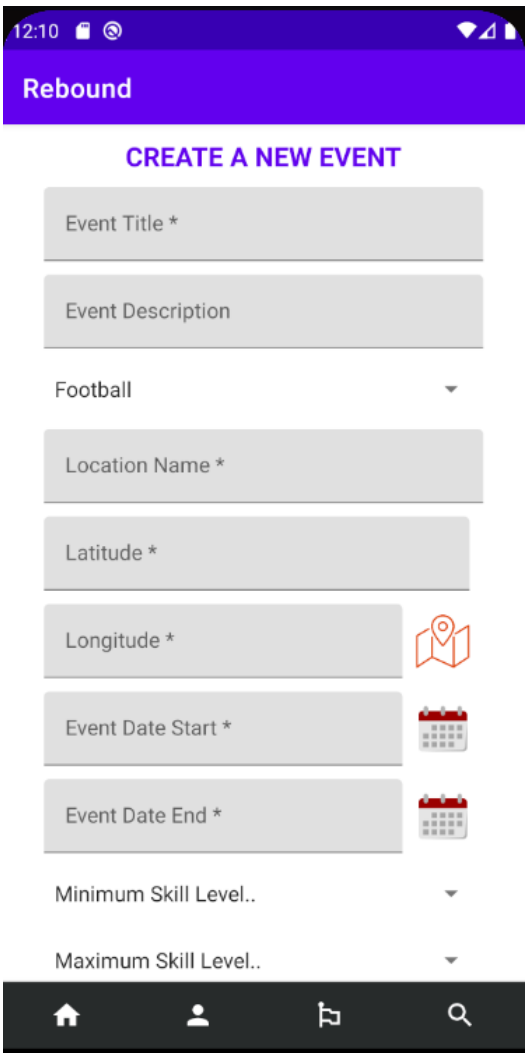
Create Event Page

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/Create%20Event

Android Code:
2021SpringGroup2/FragmentCreateEvent.kt at android · bounswe/2021SpringGroup2 (github.com)

Maps are shown above in the mapping functionality section.

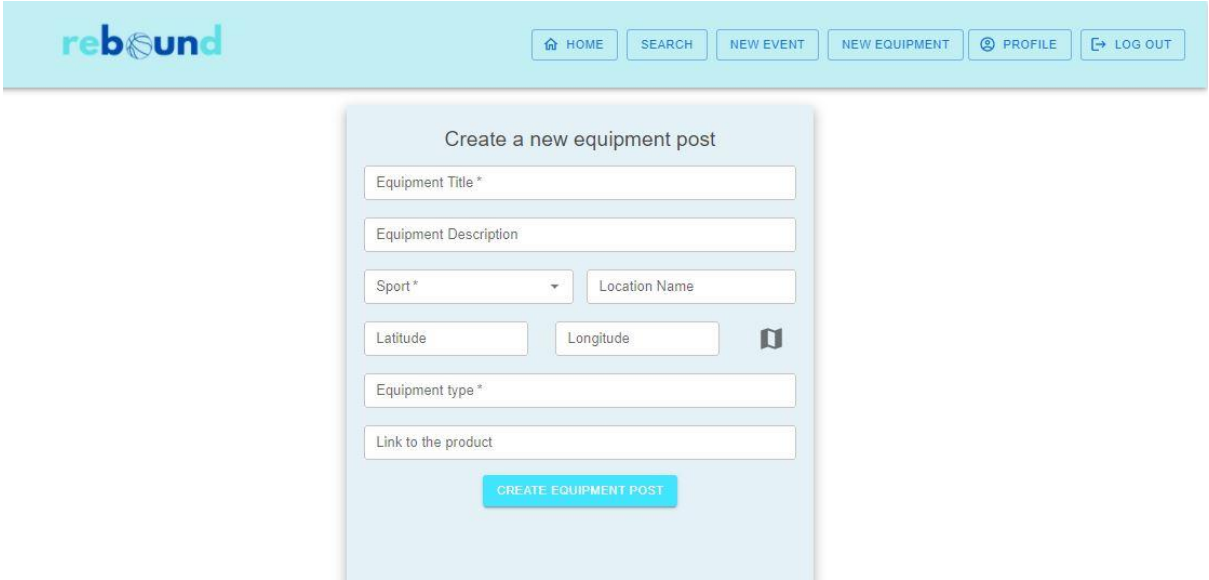


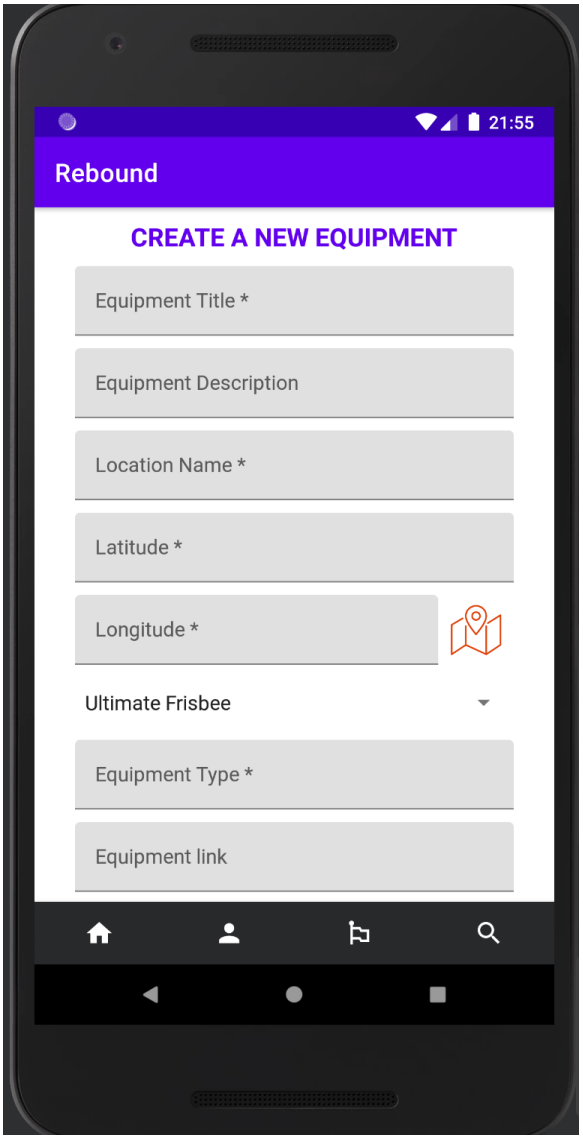


Create New Equipment Page

Frontend Code: [2021SpringGroup2/frontend/src/Views/Create Equipment at development · bounswe/2021SpringGroup2 (github.com)]

Android Code:
[https://github.com/bounswe/2021SpringGroup2/blob/master/FindSportEvents/app/src/main/java/com/bounswe/findsportevents/main/fragments/FragmentCreateEquipment.kt]

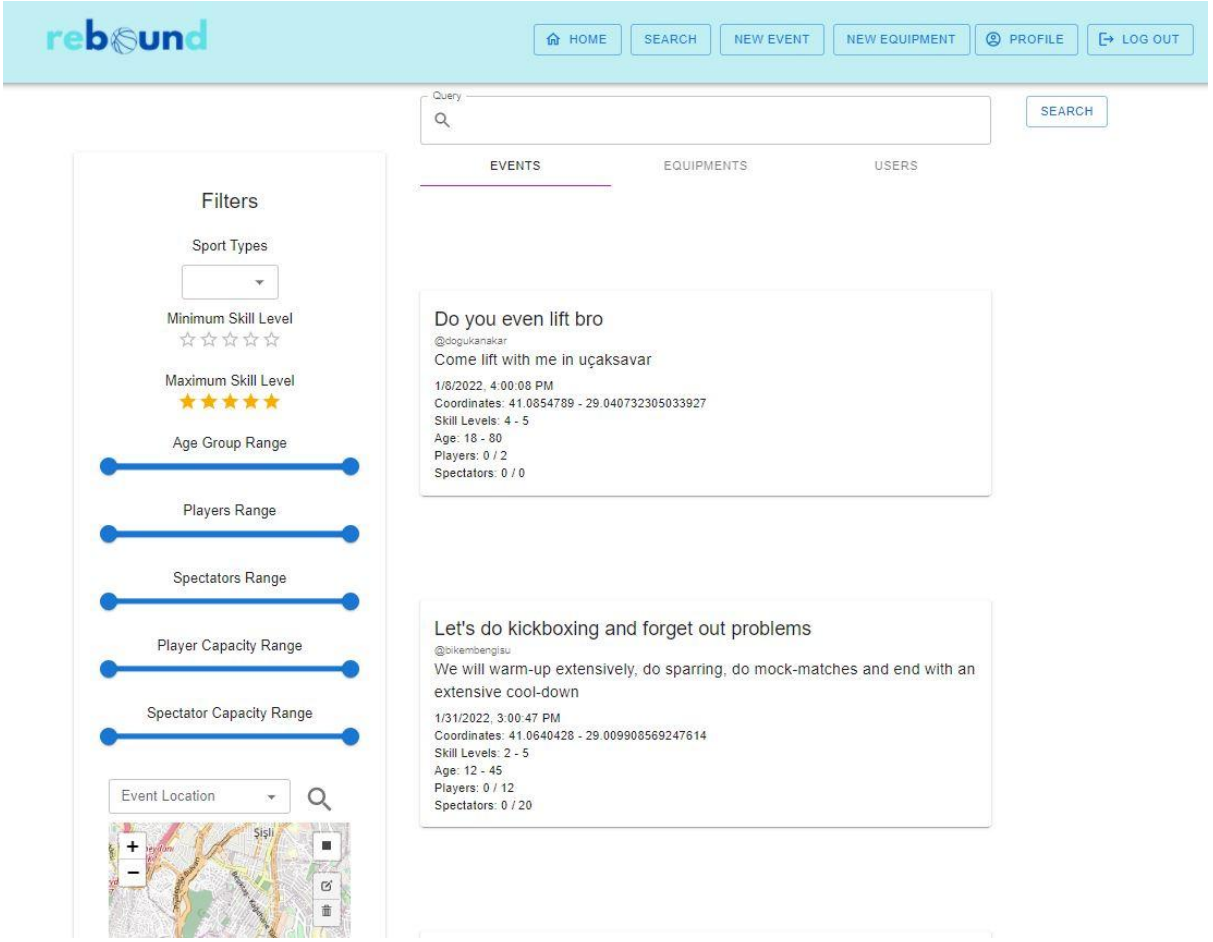

rebound
HOME
SEARCH
NEW EVENT
NEW EQUIPMENT
PROFILE
LOG OUT
Create a new equipment post
Equipment Title *
Equipment Description
Sport *
Location Name
Latitude
Longitude
Equipment type *
Link to the product
CREATE EQUIPMENT POST


21:55
Rebound
CREATE A NEW EQUIPMENT
Equipment Title *
Equipment Description
Location Name *
Latitude *
Longitude *
Ultimate Frisbee
Equipment Type *
Equipment link

Search Events

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/SearchEvents

Android Code:
[2021SpringGroup2/FragmentSearchEvent.kt at android · bounswe/2021SpringGroup2 (github.com)](https)



Minimum Creation Date

01/31/2021

Maximum Creation Date

01/31/2025

Date Range (Start)

01/12/2022

Date Range (End)

01/12/2022

Advanced Tennis Game

@sefikaakman

I am organizing a professional tennis event. If you are a self confident person,

1/5/2022, 12:28:51 PM
Coordinates: 41.093435205265195 - 29.051799774169925
Skill Levels: 3 - 5
Age: 18 - 45
Players: 0 / 20
Spectators: 0 / 50

come explore this random cave with me

@saetang3747

no safety measures whatsoever, we're going in raw

1/5/2022, 3:39:58 PM
Coordinates: 13.785727342053764 - 100.4637908935547
Skill Levels: 4 - 5
Age: 18 - 25
Players: 0 / 5
Spectators: 0 / 0

cycling

@omertest

cycling

1/5/2022, 12:41:28 PM
Coordinates: 41.046993 - 28.97335
Skill Levels: 1 - 5
Age: 12 - 100
Players: 1 / 100
Spectators: 0 / 300

Etiler Natürel Park'ta Halisaha

HOME | SEARCH | NEW EVENT | NEW EQUIPMENT | PROFILE | LOG OUT

Query

SEARCH

EVENTS | EQUIPMENTS | USERS

Frisbee in good condition

@raul123

Frisbee in good condition from my grandmother

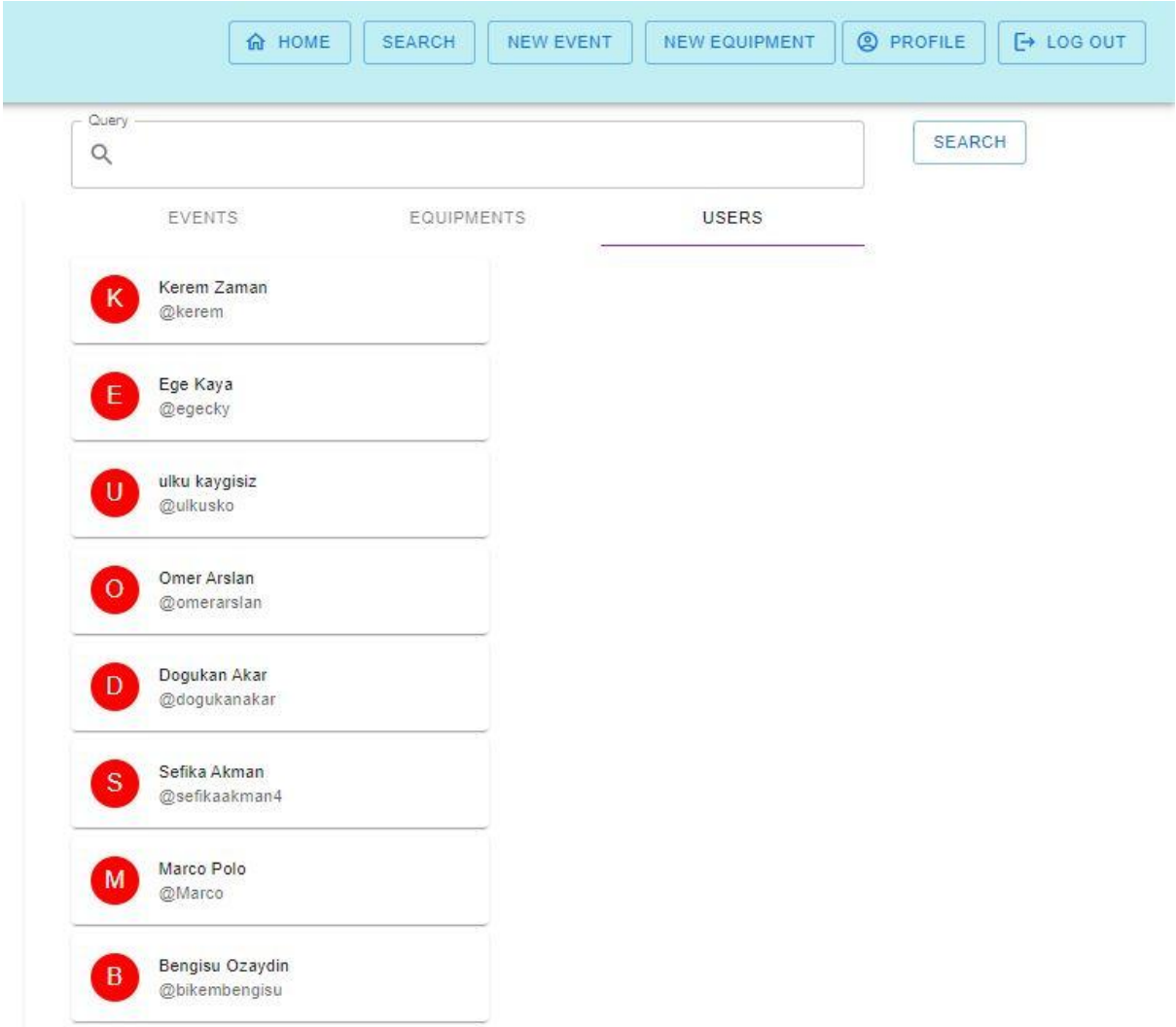
1/4/2022, 12:32:29 PM
Location: Bangkok
Coordinates: 13.019945 - 99.9979
Sport: Ultimate Frisbee
Equipment Type: Frisbee
Url: frisbee.com

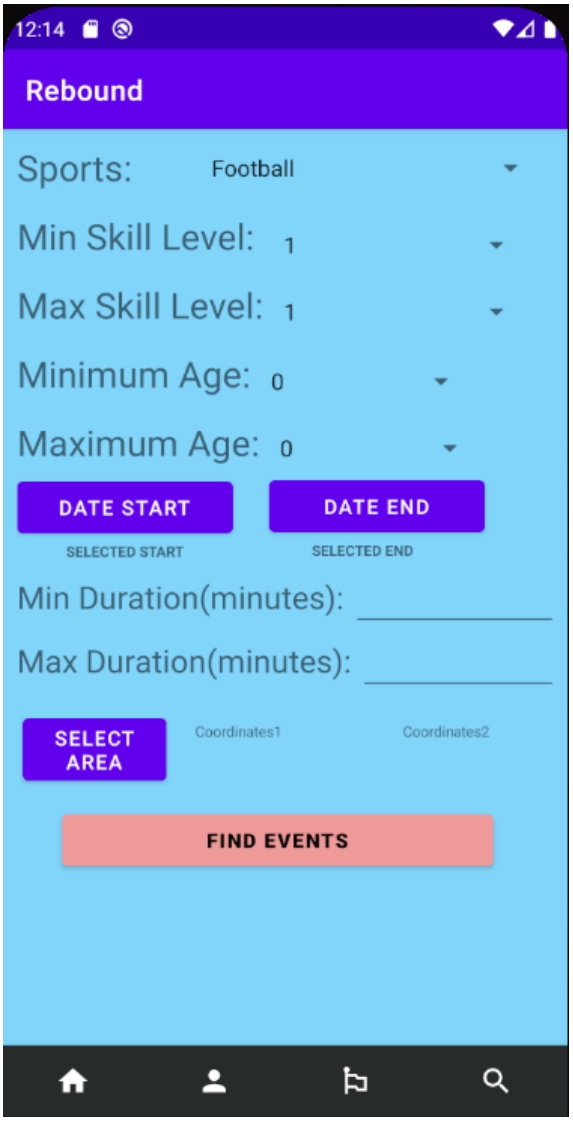
Football

@Marco

Only the best!

1/4/2022, 1:14:44 PM
Location: Bangkok Football Store
Coordinates: 13.7525438 - 100.4934734
Sport: Soccer
Equipment Type: Ball
Url: www.football.com


HOME
SEARCH
NEW EVENT
NEW EQUIPMENT
PROFILE
LOG OUT
Query
SEARCH
EVENTS
EQUIPMENTS
USERS
K
Kerem Zaman
@kerem
E
Ege Kaya
@egecky
U
ulku kaygisiz
@ulkusko
O
Omer Arslan
@omerarslan
D
Dogukan Akar
@dogukanakar
S
Sefika Akman
@sefikaakman4
M
Marco Polo
@Marco
B
Bengisu Ozaydin
@bikembengisu


12:14
Rebound
Sports:
Football
Min Skill Level:
1
Max Skill Level:
1
Minimum Age:
0
Maximum Age:
0
DATE START
DATE END
SELECTED START
SELECTED END
Min Duration(minutes):
Max Duration(minutes):
SELECT AREA
Coordinates1
Coordinates2
FIND EVENTS

Comments & Answers Page

Frontend Code:
https://github.com/bounswe/2021SpringGroup2/tree/development/frontend/src/Views/Comments

Android Code:
https://github.com/bounswe/2021SpringGroup2/blob/master/FindSportEvents/app/src/main/java/com/bounswe/findsportevents/main/fragments/FragmentViewEventDetailed.kt

Standards

W3C Activity Streams 2.0

We used the Activity Streams format for almost all JSON responses originating from the back end. This includes responses for the Event, Equipment, Comment/Answer, Follow/Block and Badges API. Only the User API, which was the first API to be implemented, before the emphasis on Activity Streams was given, does not comply with the Activity Streams format and instead returns a generic JSON response.

We took the liberty of not requiring request bodies to comply with the Activity Streams format, since we believe that the format mostly concerns outputs.

Wikidata

We have used Wikidata for the badges in our application. These badges are given to users by the fellow players of an event, or by the owner of the event that a user has participated in as a player, to highlight their behavior and skills. There are also badges that represent bad behavior. These badges are displayed on the profile pages. We have used the following Wikidata pages in the creation of our badges:

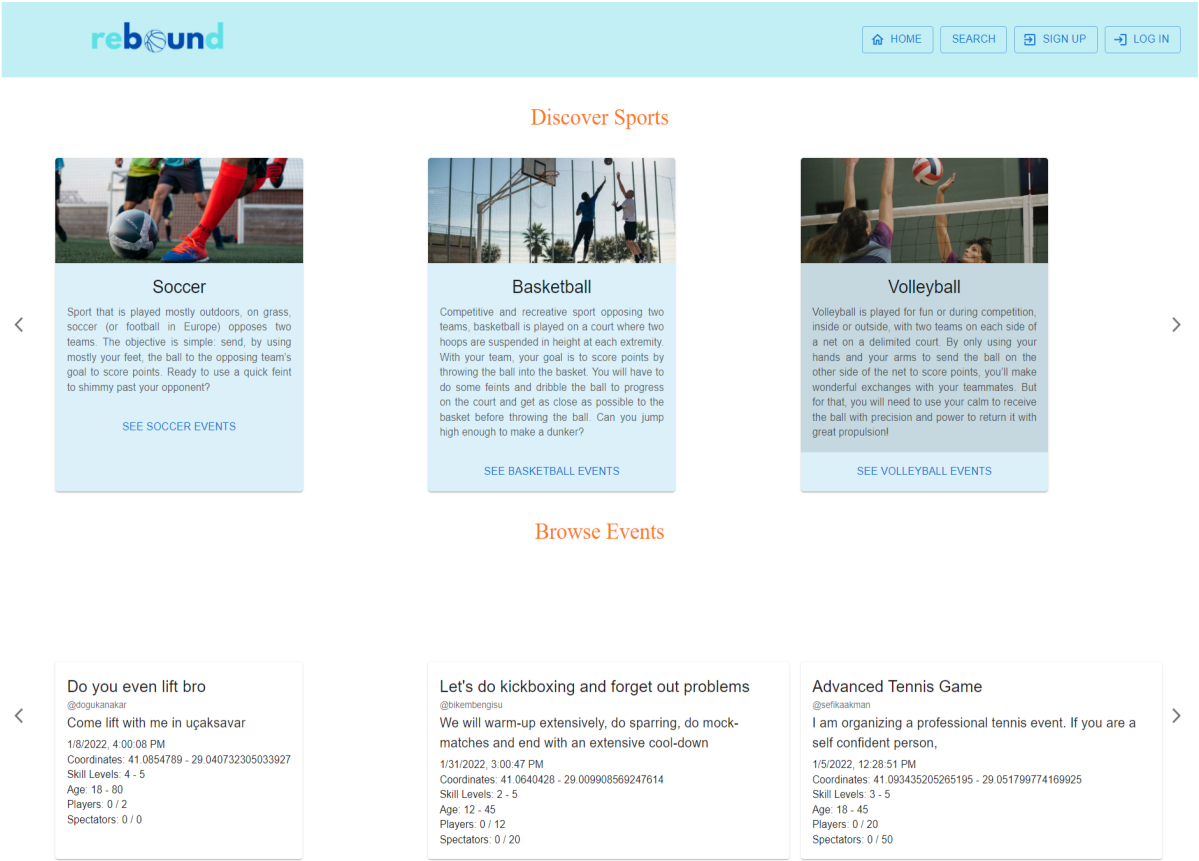
MVP - https://www.wikidata.org/wiki/Q652965
Fair play - https://www.wikidata.org/wiki/Q995616
Friendly - https://www.wikidata.org/wiki/Q491
Team leader - https://www.wikidata.org/wiki/Q743438
Cheater - https://www.wikidata.org/wiki/Q2488105
Agression - https://www.wikidata.org/wiki/Q191797

User Manual

Frontend Application

1-) Homepage

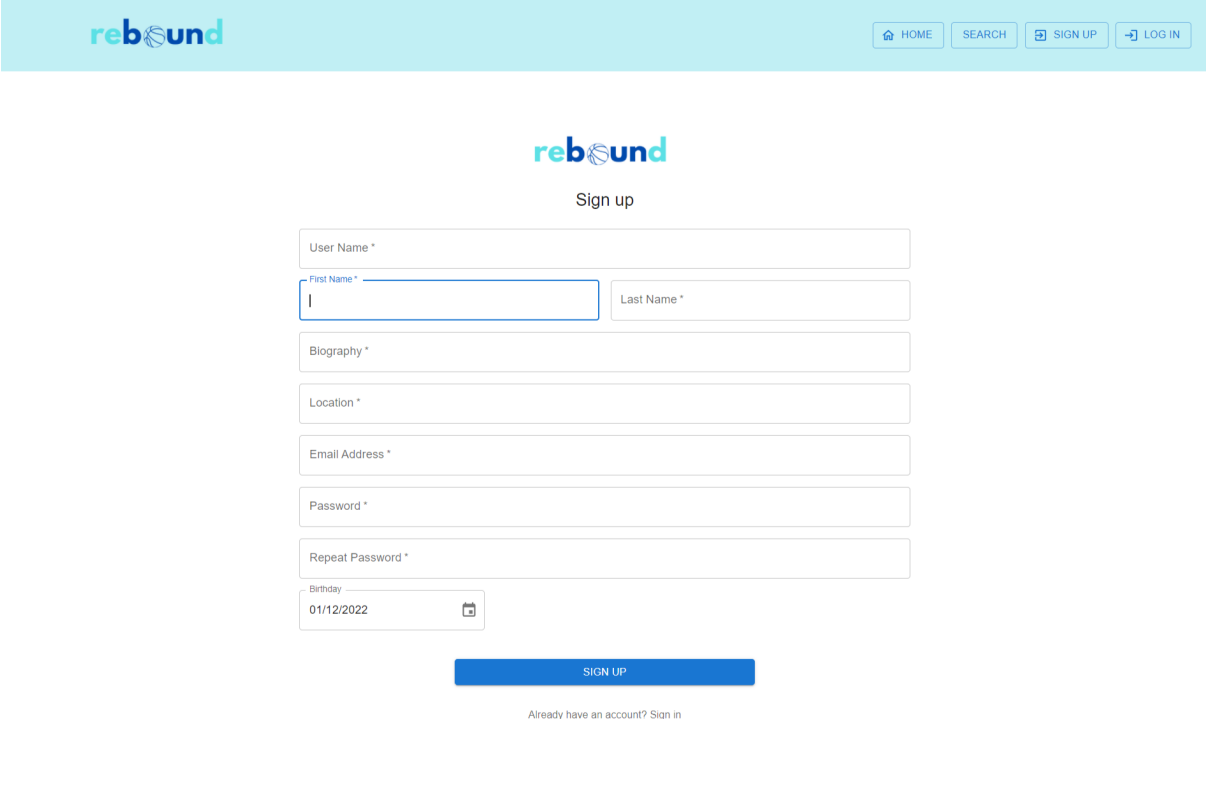
When users visit the website, the homepage is the first page they see. Users can check out the most popular sports in the mini cards and they can find the events for specific sports by clicking the buttons on the bottom of these cards. Users see three sports at a time, they can check up to 12 sports by clicking the forward and backward buttons on the left and right sides. Furthermore, users can check out some events from the mini event cards below. Guest users will see the home, search, sign up and login buttons in the header and can click them to navigate to corresponding pages.

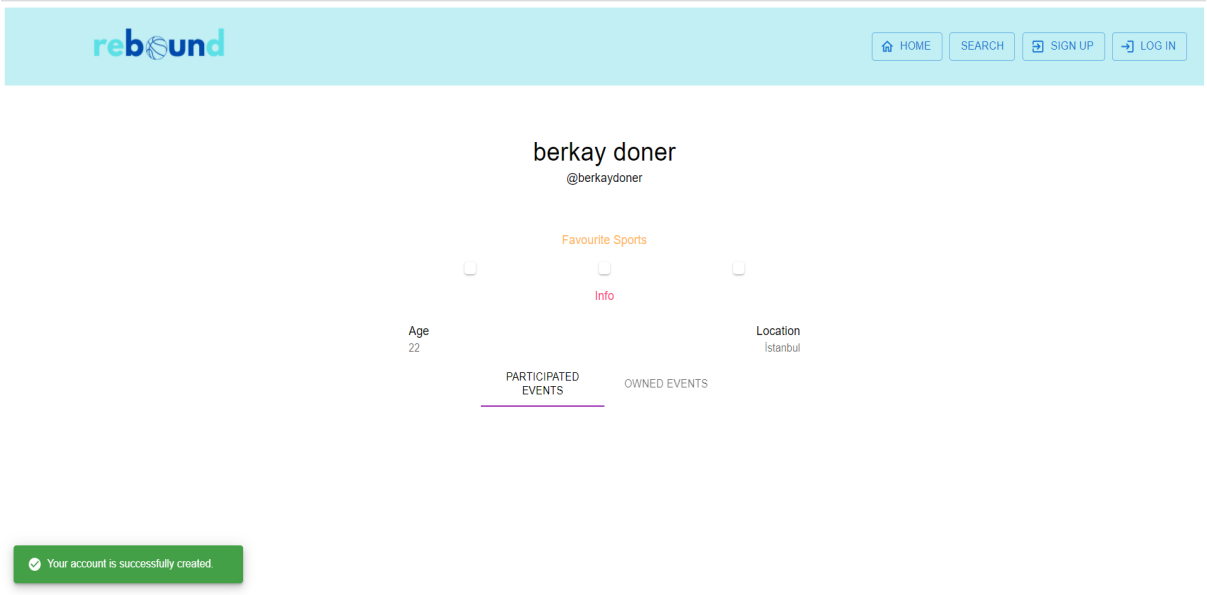


2-) Sign up

Users can navigate to the sign up page by clicking the sign up button in the header. In the sign up, users are expected to type their personal information. For each field, if the user types an invalid input, a warning message is displayed. After filling out the sign up form, users can click on the sign up button below to finalize the process. If the user registers to the system successfully, a notification is displayed on the bottom left corner and the user is

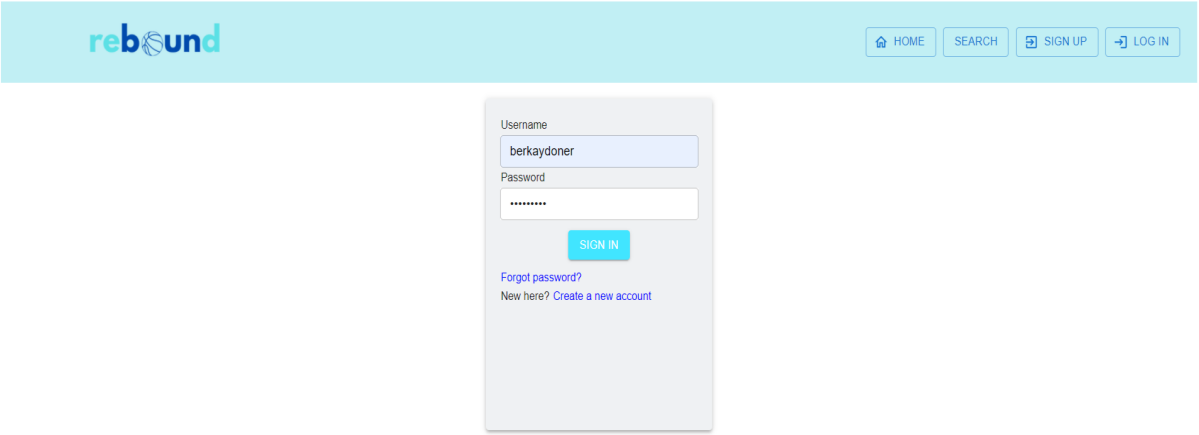
redirected to their profile page. Users can also navigate to the login page by clicking the link below the button.



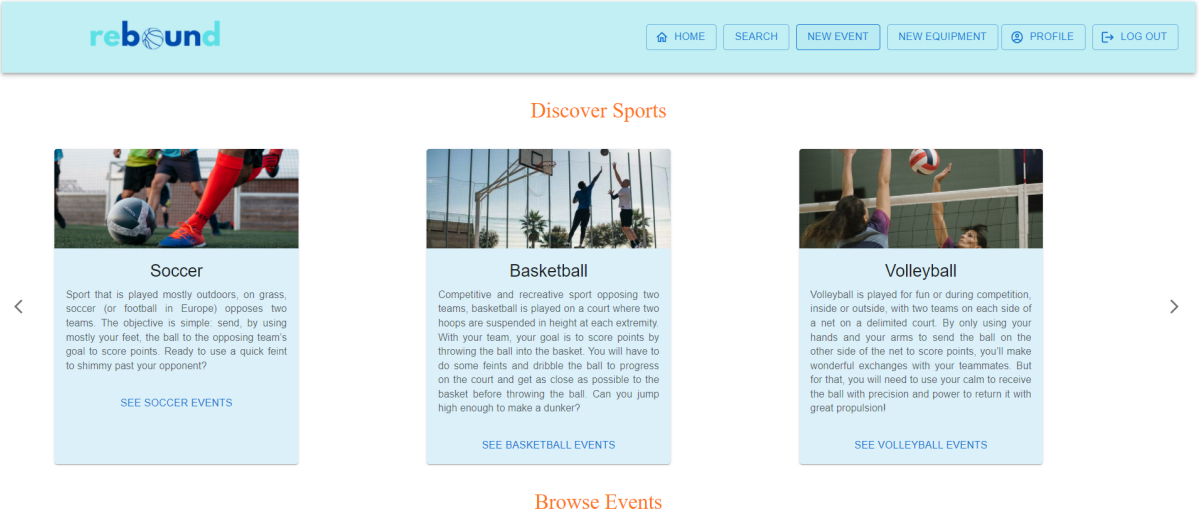


3-) Login

Users can navigate to the login page by clicking the button in the header or to the link in the sign up page. In the login page, users are expected to write their username and password. If the user has forgotten their password, they can navigate to the forgot password page by clicking the link. Similarly, if the user does not have an account yet, they can click the link to navigate to the sign up page. After filling out their information, users can click the sign in button to login. If the information is not correct, users are alerted with a warning message.

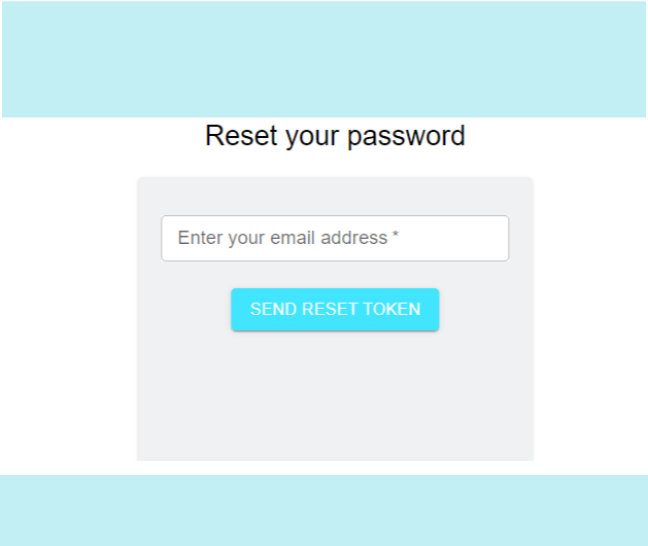


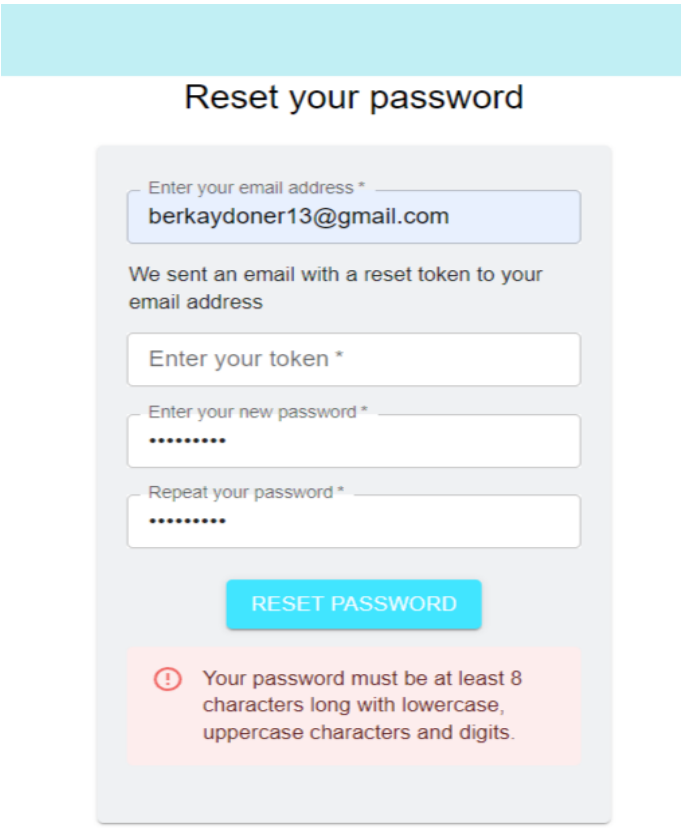
If the user is able to login successfully, they are directed to the homepage and the header now displays create event/create equipment post, profile and logout buttons.



4-) Forgot password

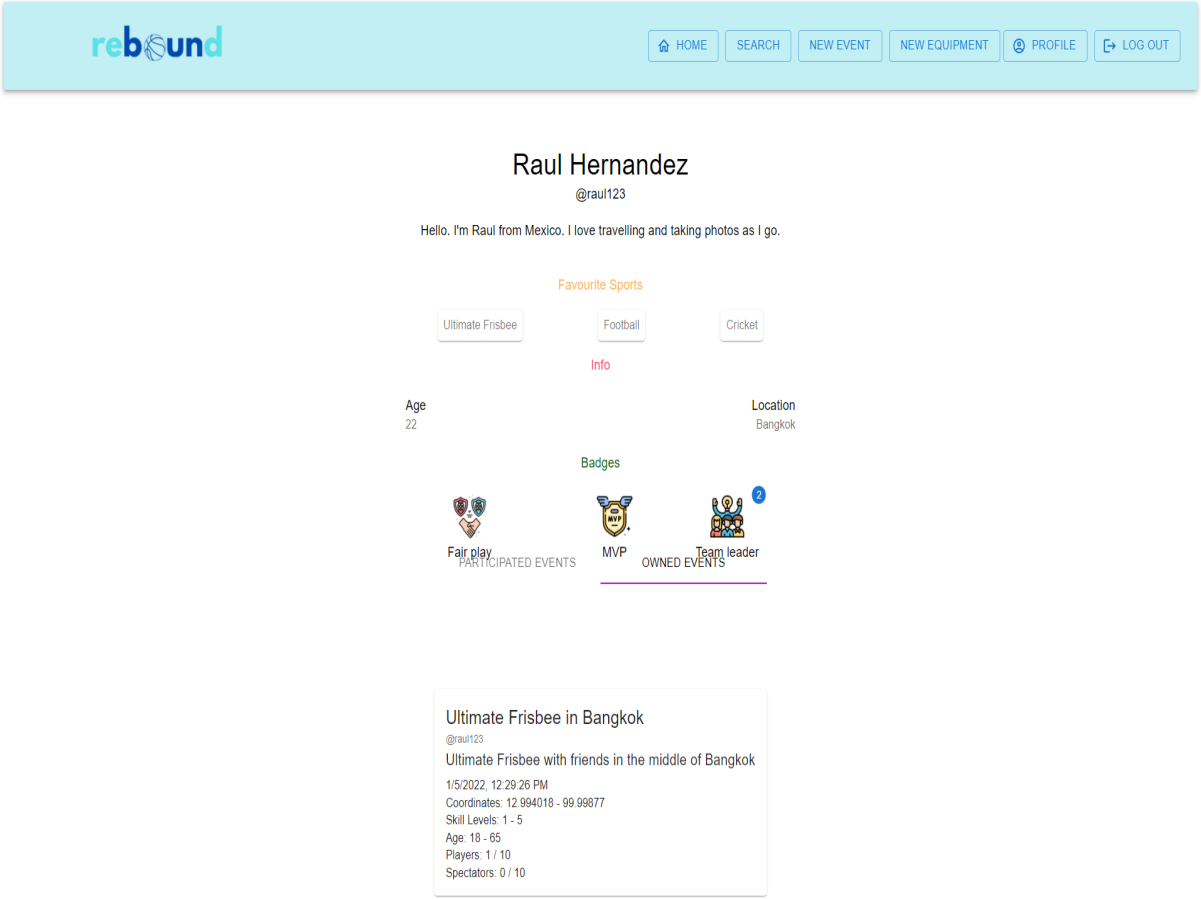
Users can reset their password if they forgot it from the reset password page. To navigate to the page, users have to click the link at the bottom of the login page. In the reset password page, users are expected to first enter their email address. Once users type their email address, a reset token is sent to their email address and users are expected to type the token and their new password along with its repetition to reset their passwords. In all stages, users are alerted with the corresponding error message in case they write invalid inputs such as invalid email, invalid/weak password, different password for repetition.





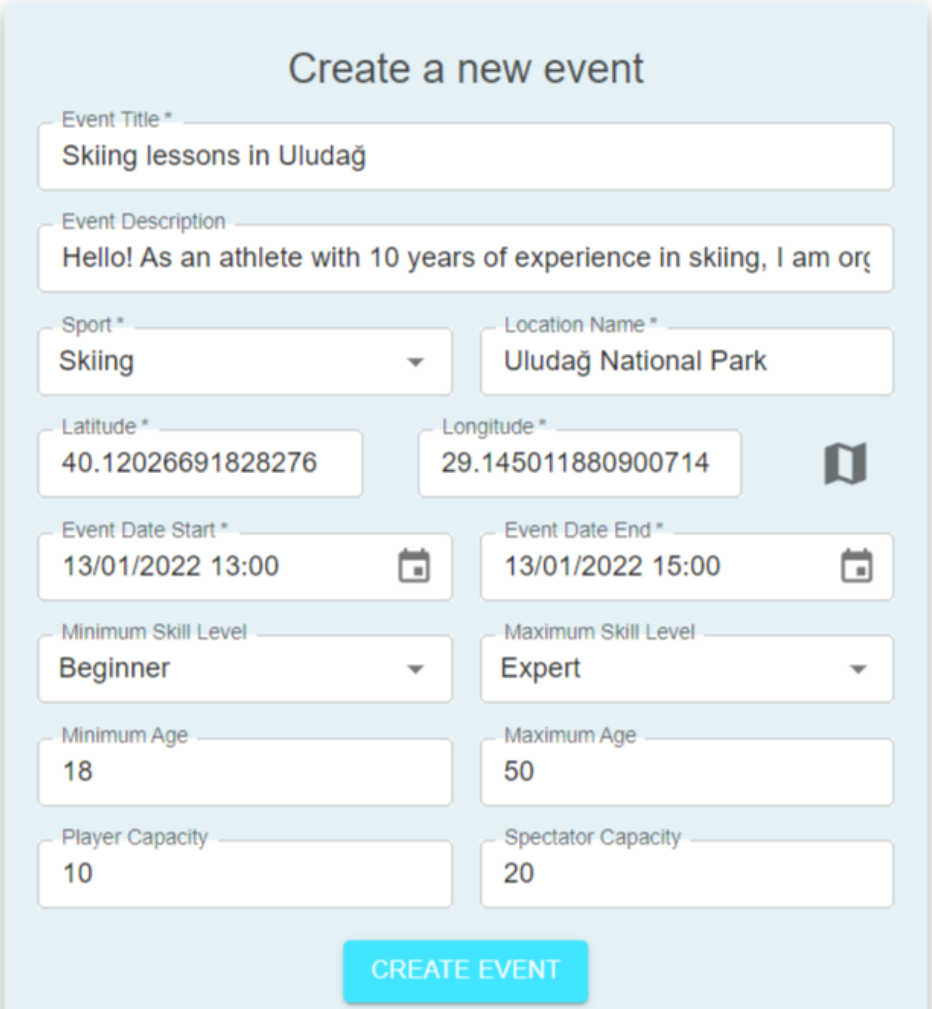
5-) Profile page

Logged in users can view their profile page by clicking the button in the header. All users can access to profile pages of users from the user search page. In these profile pages, users can see the biography, favorite sports, age, location and badges granted to the profile owner. Users can also observe the events created by this user and the events the user has participated as player below and can switch between these tabs.



6-) New event

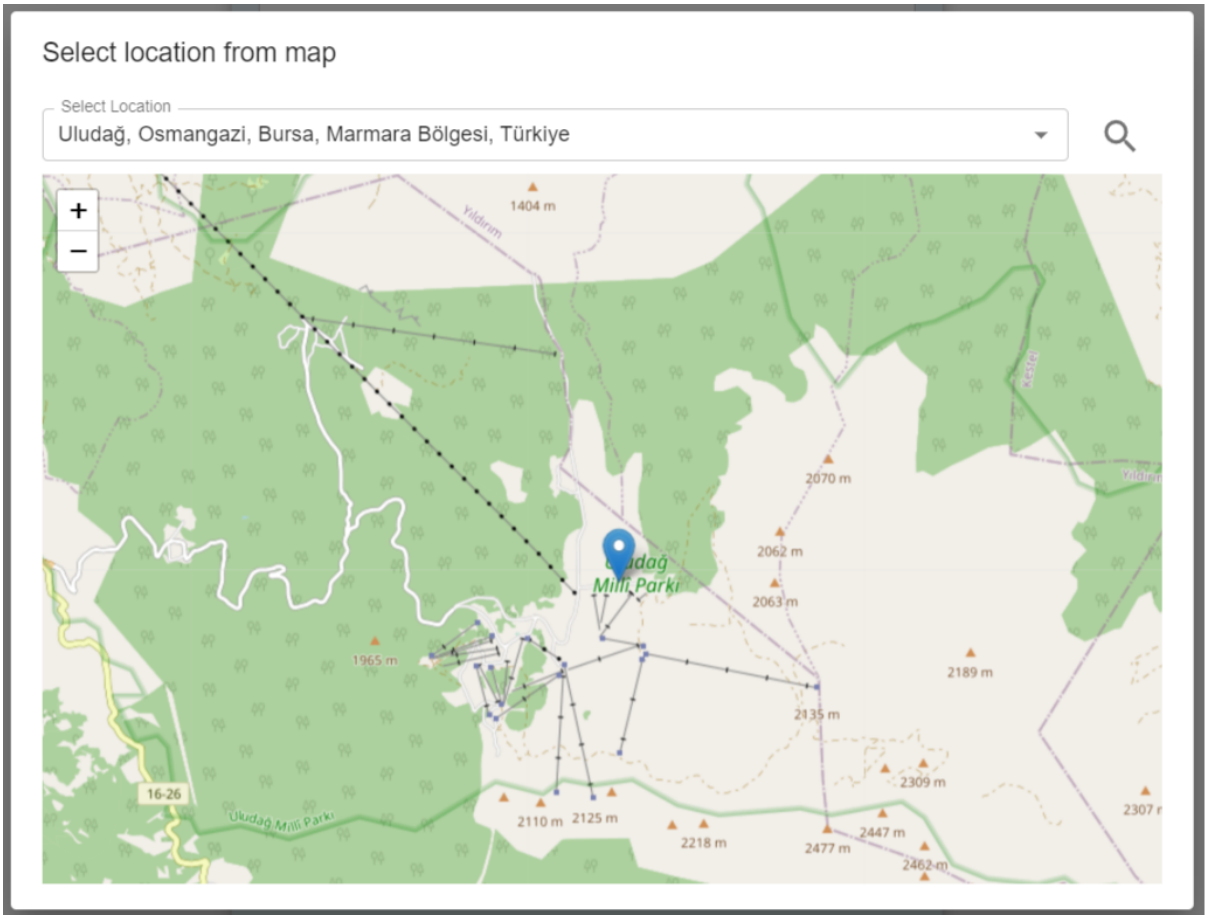
Logged in users can create new event posts by clicking the new event button in the header. In this page, users are expected to fill the necessary information for creating posts. To specify the sport, users can select one of the existing sport types.



Users can directly write the coordinates of the event location and they are also able to select a location from the map, accessed by clicking the map icon to the right of coordinate fields. In the map page, users can drag the marker for specifying the exact location and they can search for general locations in the search bar above. After writing a location name in the search bar and clicking the search button, users can select one of the results displayed to navigate to that location in the map. Coordinate fields and the location of the marker are connected to each other, changing one changes the other and the coordinates are used as the final location.

For any invalid inputs, users are alerted with a warning message. After successfully creating an event post, the page is reset for creating more events.

PS, if you are unable to create events, please refresh your session by logging out and logging in again.

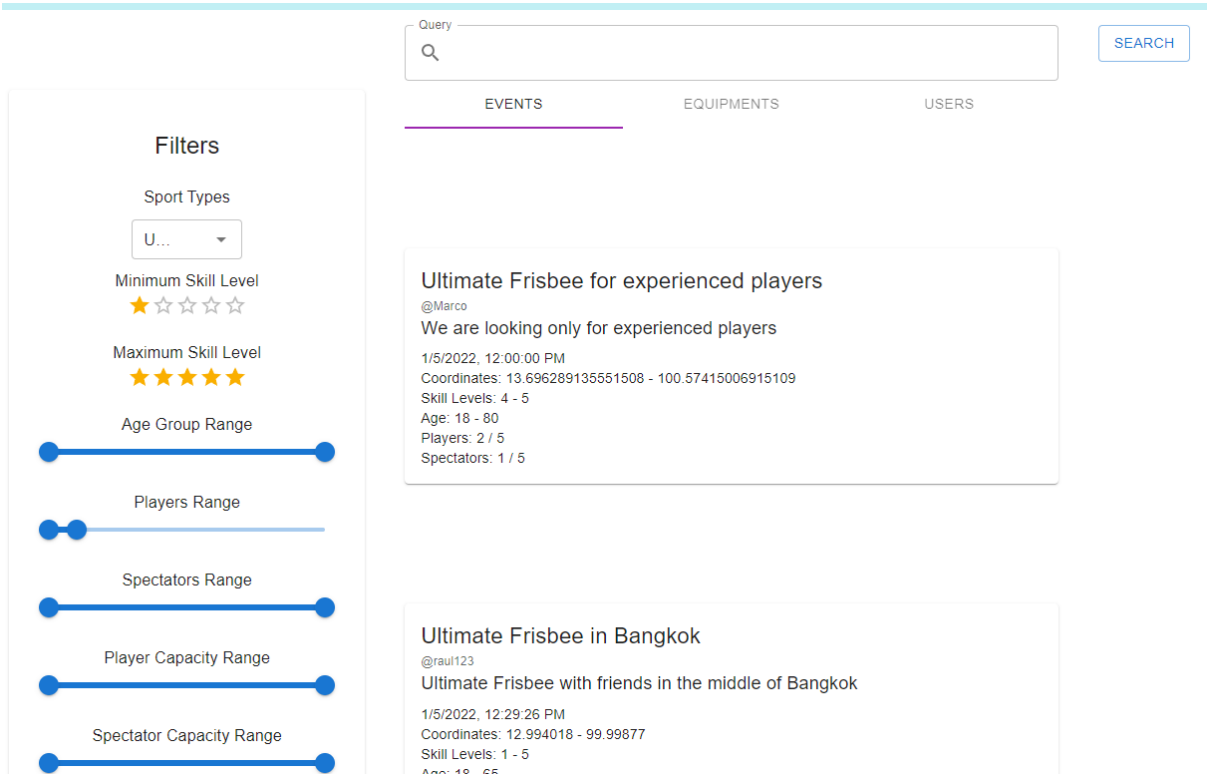


7-) Searching events

All users can navigate to the search page by clicking the button in the header. In the search page, filters are shown in the left and the events are shown at the right with two other tabs for equipment posts and users. Users can directly view all events when the page is first opened. Users can navigate to the detailed event pages by clicking on the mini event cards in the search results.

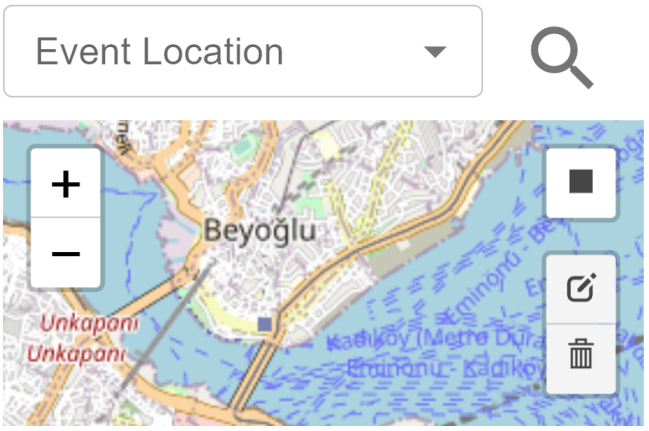
Users can also filter the events by keyword by typing in the search bar above. Also, they can filter the events by sport type by selecting a sport from the filters.

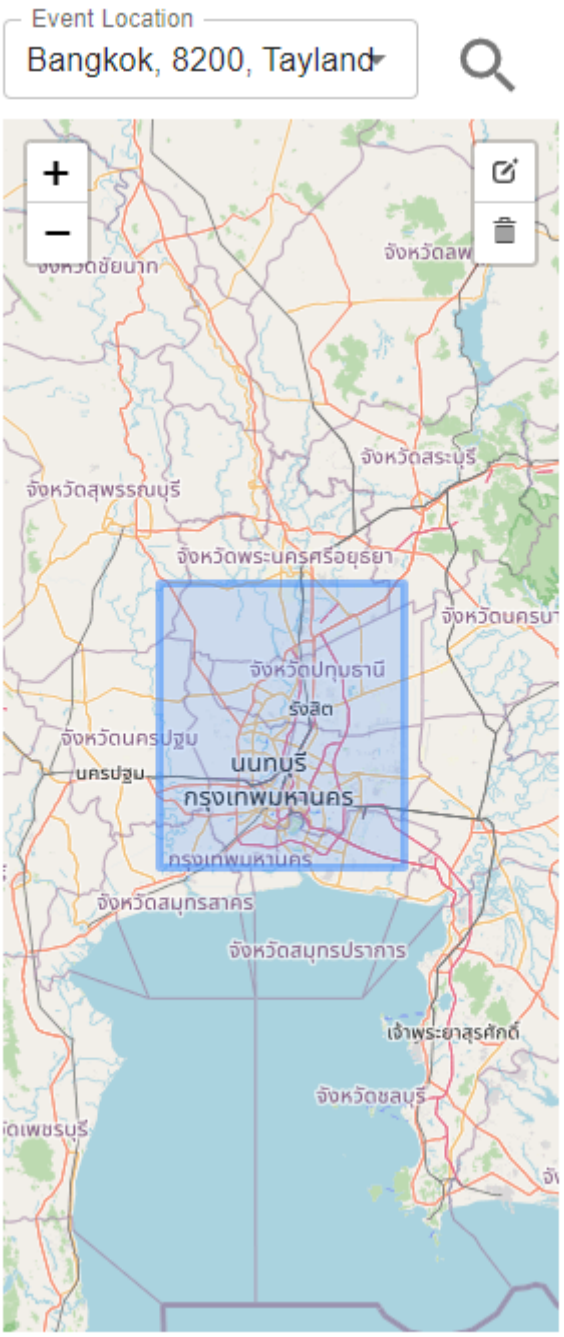
Minimum and maximum skill levels are used to filter the events that have minimum and maximum skill levels in the range specified. For example, with a minimum of 4 and maximum of 5 filters, only events that have minimum skill level >= 4 and maximum skill level <=5 are displayed. Age group filter works similarly, its minimum and maximum values are used as the range to be filtered.



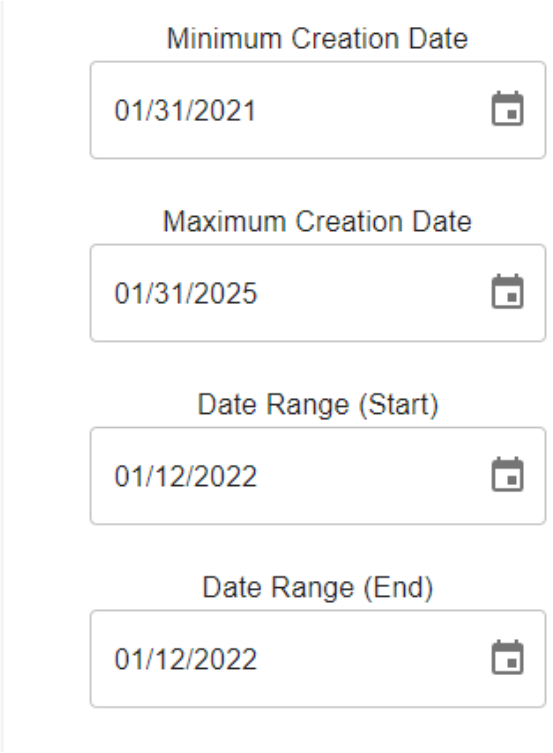
Players, spectators, players capacity and spectator capacity are given as a range filter. When applied, only the events that have number of players/spectators and players/spectator capacity in ranges specified are displayed.

To filter locations, users are expected to draw a rectangle on the map. Users can click the rectangle button to start the drawing process. They can also use the edit/delete buttons to edit/delete existing drawing. Users can only specify one rectangle to filter the events that have coordinates within the rectangle. Rectangle, edit and delete buttons are in the toolbar as follows:





There are also date filters to filter events by their creation date and event dates. Only events within the given ranges are shown.

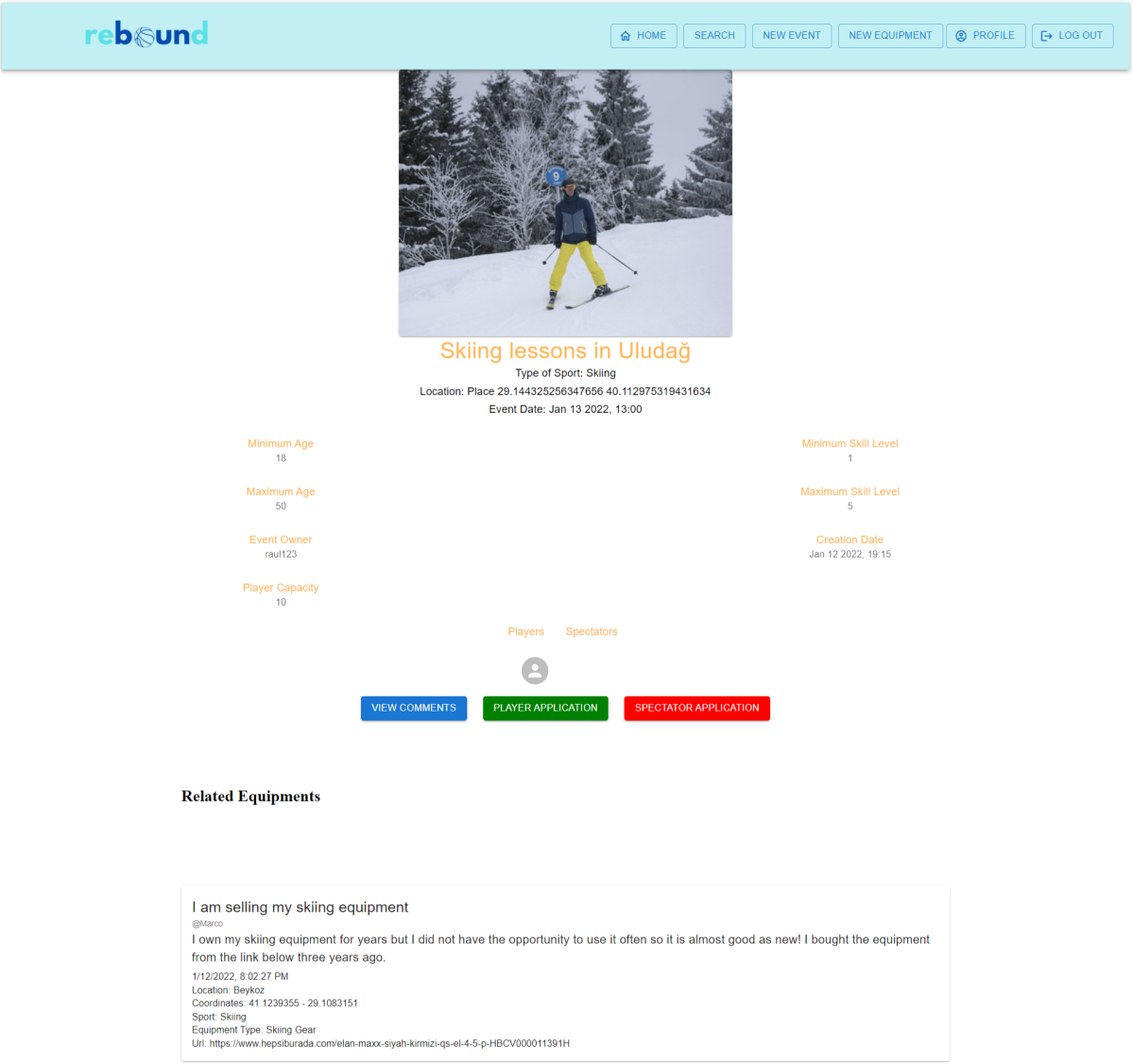


Users can apply these filters by clicking the search button. All these filters are only applied when they are changed, otherwise events are fetched without any filtering, even though default values are displayed. PS, you might have to refresh the page after clicking the search button to get the results due to the slow update of the system.

8-) Event pages

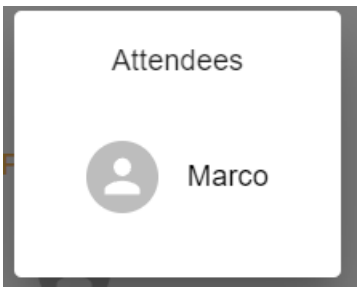
Users can access detailed event pages by clicking the mini event cards from search results, homepage or the participated/owned events displayed in the profile pages. In the detailed event page, users can see the details of the event such as title, sport type, capacity. Furthermore, users are able to see the equipment posts of the same sport as recommendations below. Users can interact with the event using the buttons below.

To apply to the event to be a player/spectator, users can click to corresponding buttons. These buttons will only be enabled if the event capacity is not full, if the user is a logged in user and is not a player/spectator or applicant already. Applying to an event refreshes the page. Note, if you are unable to apply to an event, please refresh your session by logging out and logging in again.



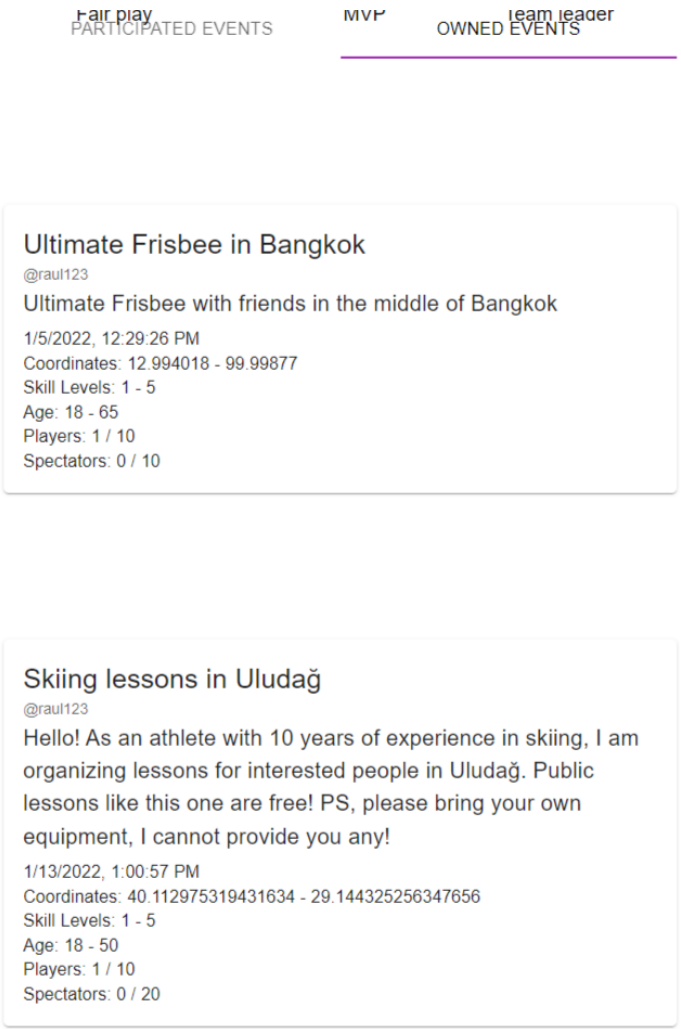
Users are also able to interact with the event by viewing the comments/answers and writing comments/answers themselves by clicking to the view comments button. In the comments page, by writing in the text box below and clicking the share button, users can write comments and by clicking on the reply button below the comments, they can write answers in a similar way.

Other than comments, users are also able to view the players/spectators of the event by clicking the avatar lists displayed under the players/spectators fields. A list of players are displayed with avatars and usernames.

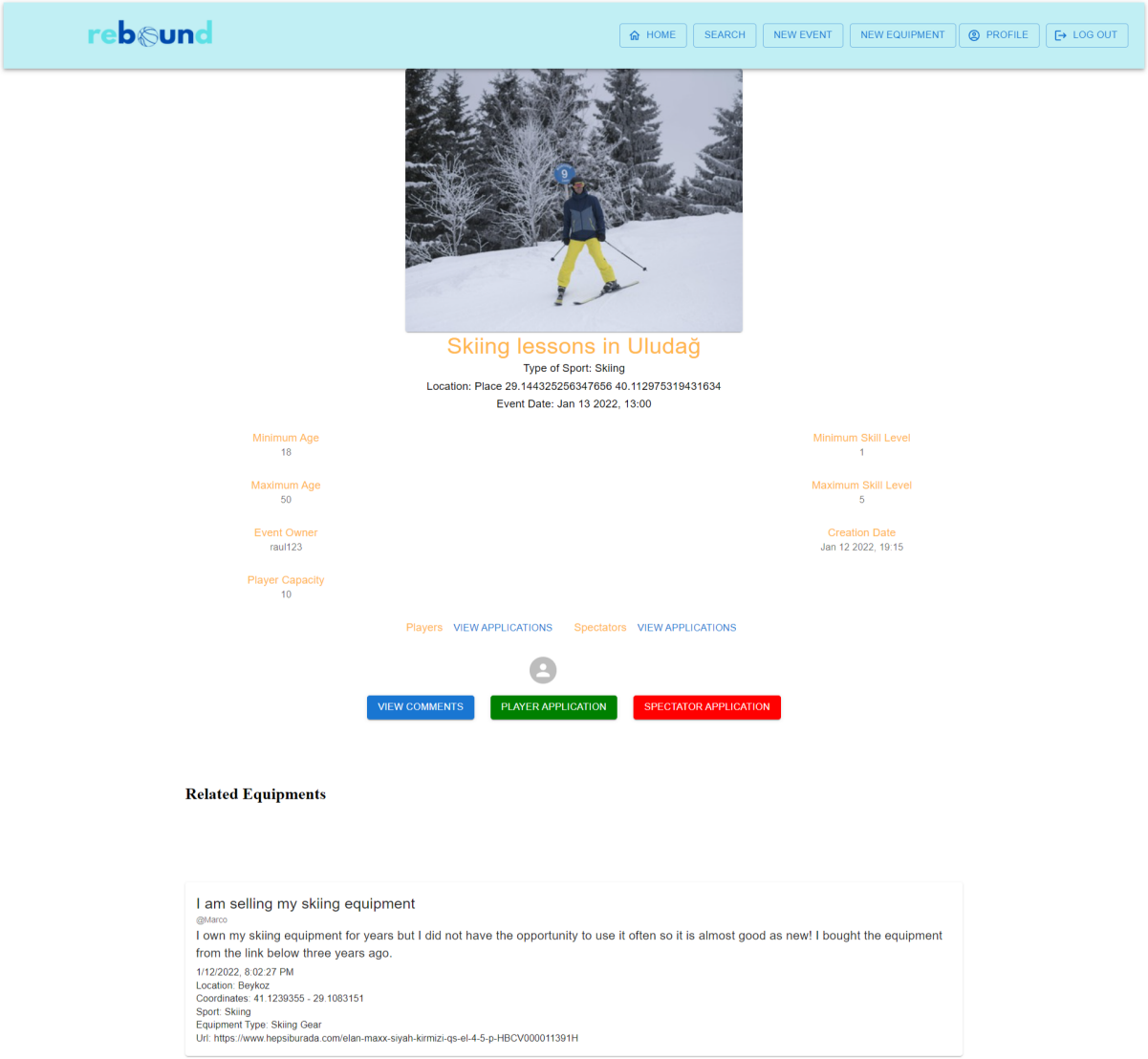


9-) Viewing applications

Event creators are able to view and evaluate applications made to their events from the event pages. To get to their events, users can search for them in the search page and, even better, they can view them from the owned events section in their profile page. From the list of owned events, event creators can click on mini event cards to get to the detailed event page.

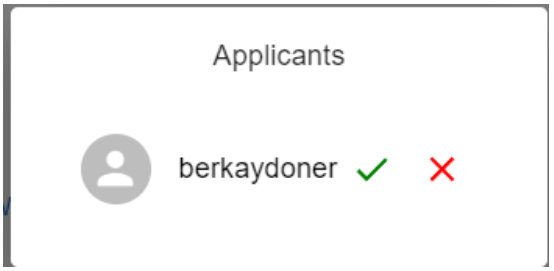


In the detailed event page, event creators are able to view applications through the view applications buttons to the right of players/spectators fields. These buttons are only visible to the event creator and no other visitor.



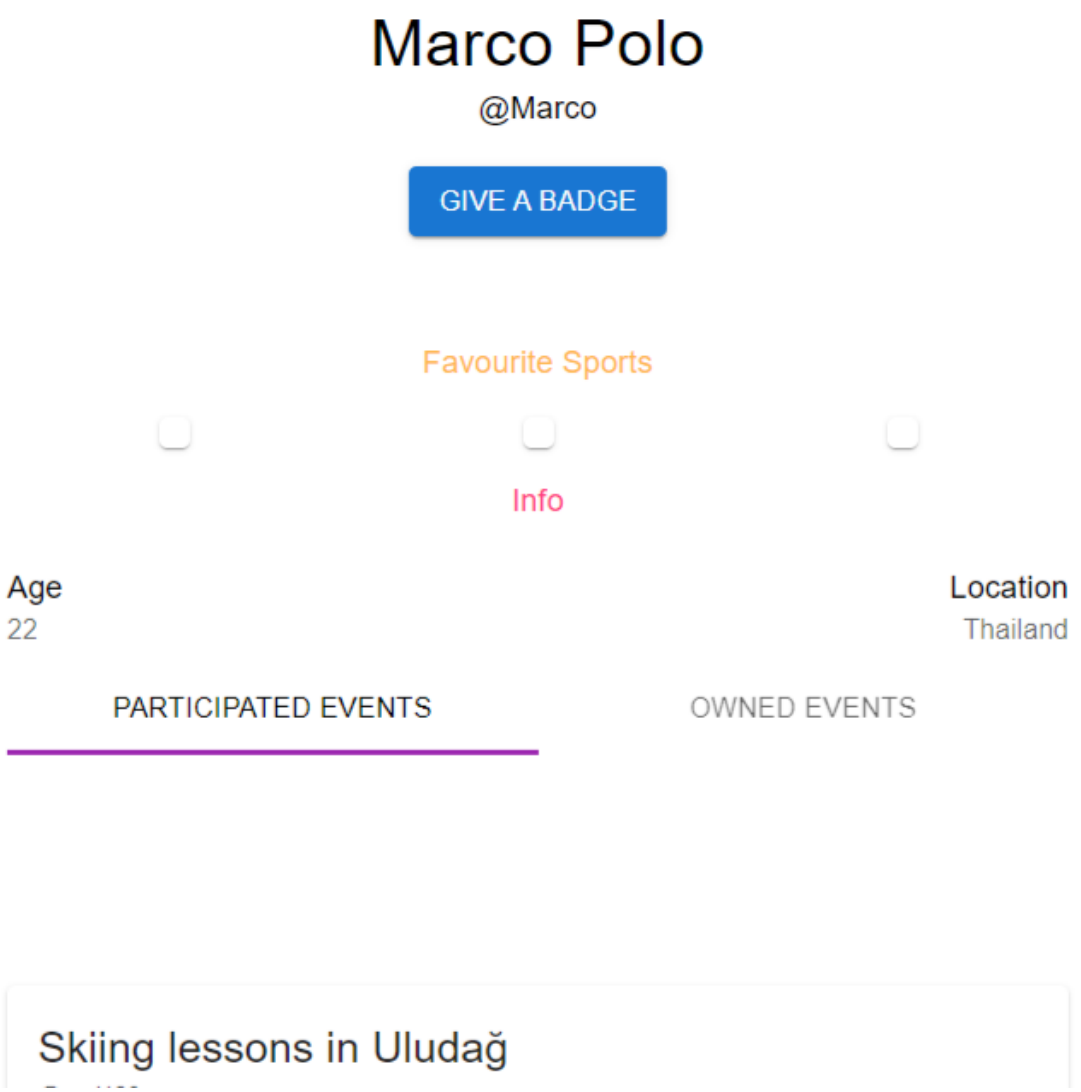
By clicking the view applications buttons, event creators can view the list of applicants with accept and reject buttons. By clicking these buttons, event creators can accept or reject the users. To get a better idea about the applicants, event creators can click on the avatars/usernames of the applicants to get to their profile pages, in which badges can be used as the indicator of the users' behavior. Once the event creator accepts or rejects a user, event page is refreshed to update the players/spectators with the recent change.

Note, if you are unable to evaluate applications, please refresh your session by logging out and logging in again.



10-) Giving badges

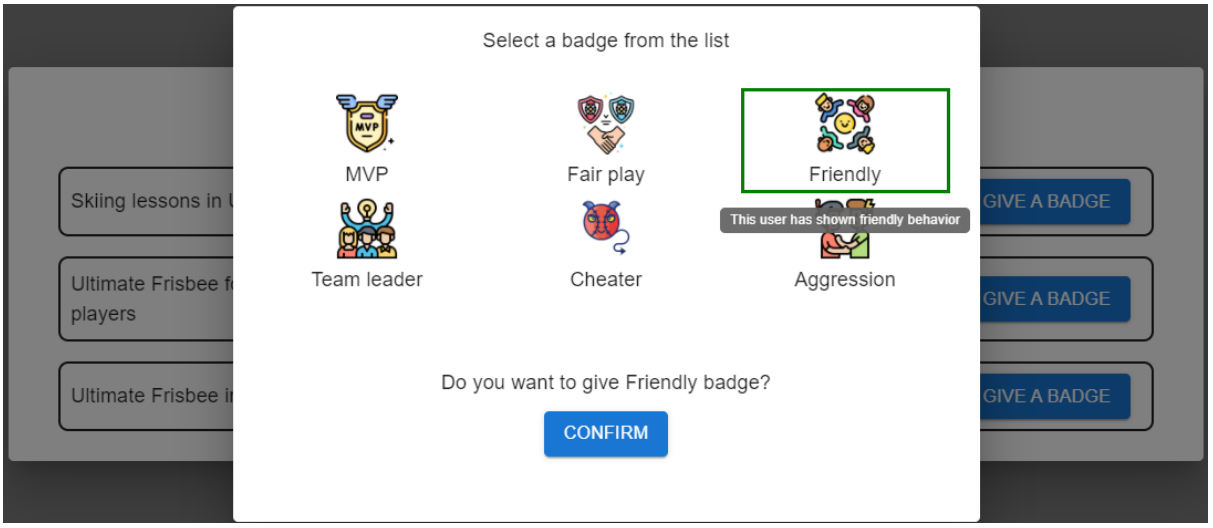
According to the project requirements, players of some events are able to give badges to fellow players and event creators are able to give badges to the players of their events. Users are able to grant badges to other users from their profile pages, a button for gifting badges appears in the profile page only if the visitor user and the profile owner user have common events specified by the requirement. By clicking the button, visitor user can view the events that are available for gifting badges.



From the list of common events, visitor user can click the button of one of them to grant a badge.

After selecting an event, all badges are shown with their titles, icons and descriptions when the mouse is over them. In this page, users are expected to select a badge to grant, the badges become selected and get highlighted with a green window once clicked. After deciding on the badge, users can click the confirm button to grant the badge. This also redirects the user to the profile page, they can see the badge they have just granted.

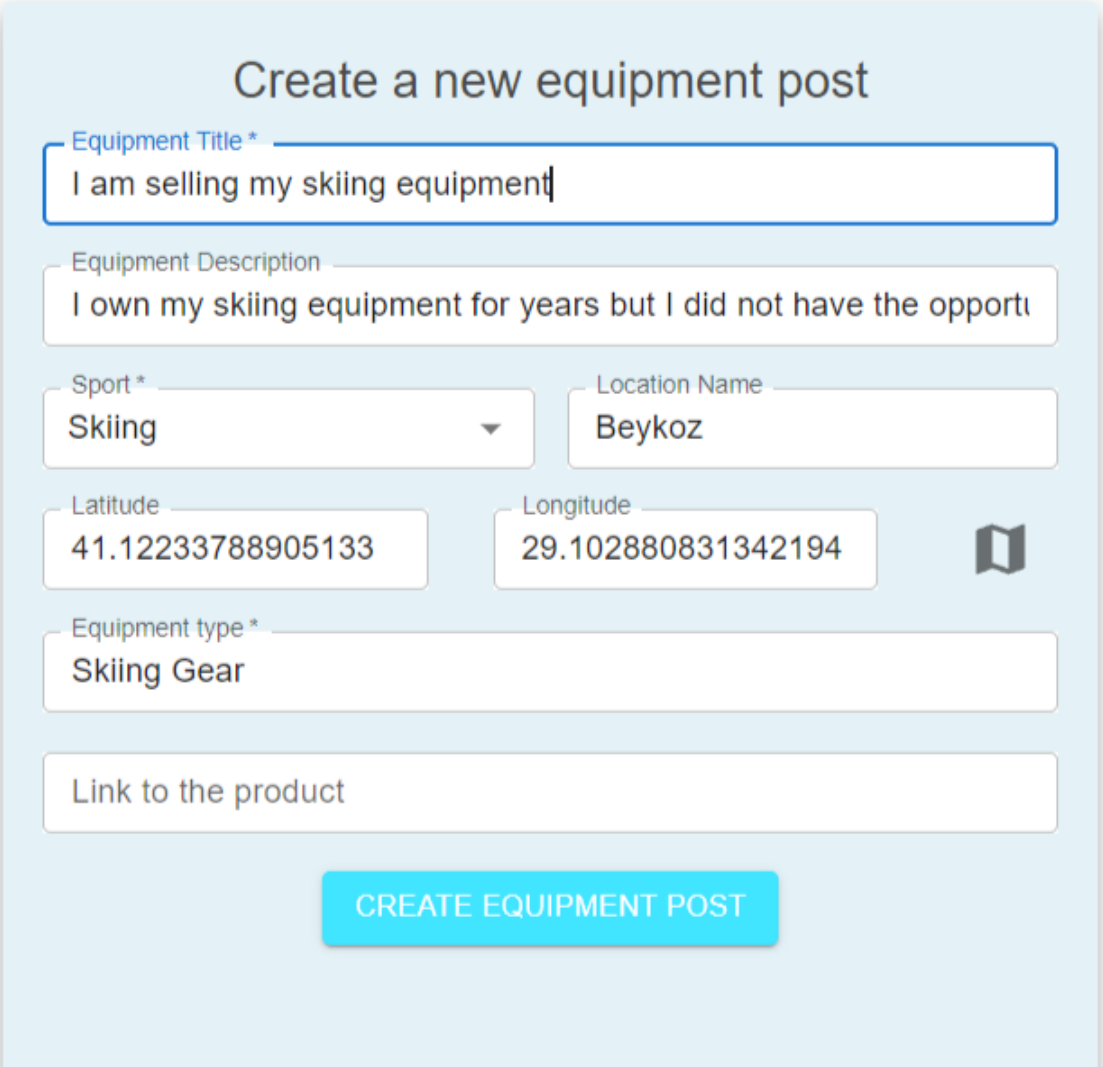
Note, if you are unable to grant badges, please refresh your session by logging out and logging in again.



11-) New equipment post

Logged in users can create new equipment posts by clicking the new equipment button in the header. In this page, users are expected to fill the necessary information for creating posts, very similar to creating events. To specify the sport, users can select one of the existing sport types. Users can also select the location from the map, similar to the creating events. In this page, users are expected to specify the equipment type as text and URL to their equipment. This URL is expected to be in a valid URL type, otherwise a warning is shown.

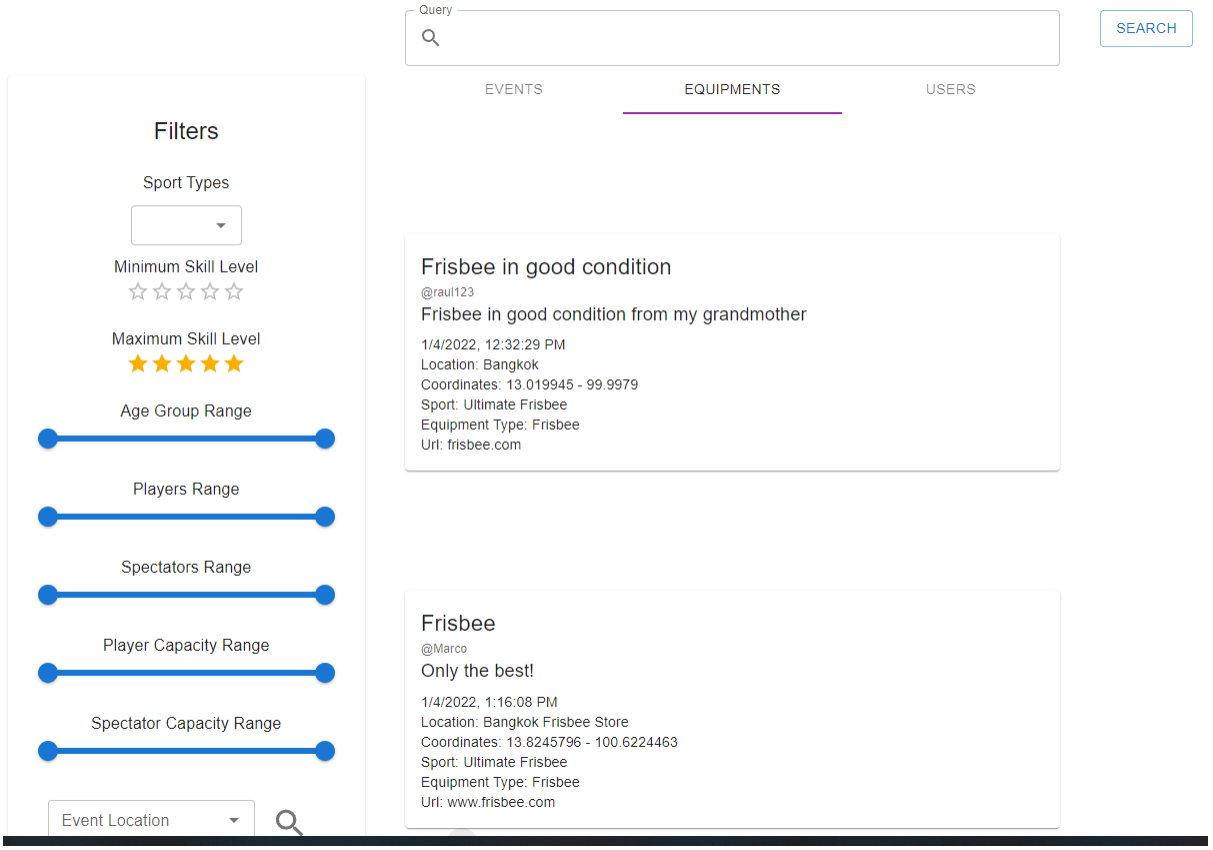
The page resets once the create equipment post is clicked. PS, if you are unable to create equipment posts, please refresh your session by logging out and logging in again.



12-) Searching equipment posts

Users can search equipment posts very similarly to event posts. All users can navigate to the search page by the button in the header. In the search page, equipment posts are displayed on the right along with other tabs and filters are on the left. All equipment posts are shown when the page is first displayed, users can filter equipment by the filters on the left and the search bar at the top.

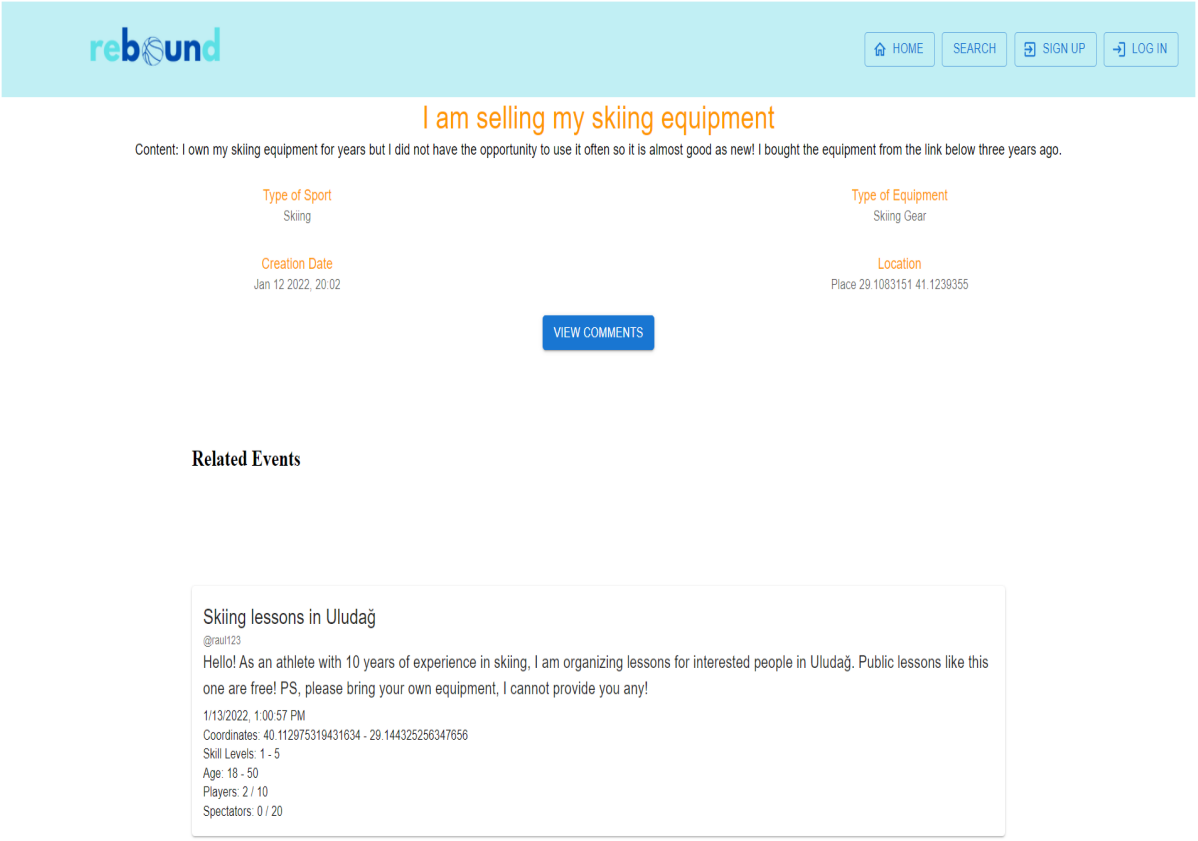
Users can filter equipment posts by keyword, sport type, location and creation date. Other filters do not apply to equipment posts and thus do not have any effect.



13-) Equipment pages

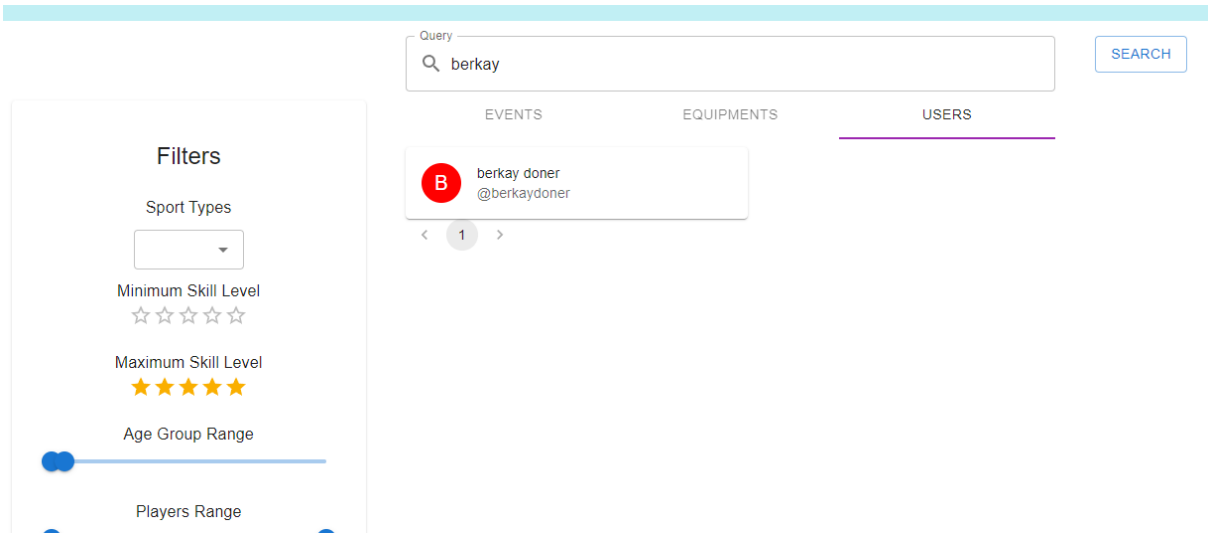
Users can access detailed equipment pages by clicking the mini equipment cards from search results or related equipment posts displayed in the event pages. In the detailed equipment page, users can see the details of the equipment such as title, sport type.

Furthermore, users are able to see the event posts of the same sport as recommendations below in case they want to try their fresh equipment in a sports game. Users can interact with the equipment posts by viewing/writing comments in the comments page. This page is similarly accessed by clicking the view comments button below.



14-) Searching users

Users can search other users from the search page as well. All users are displayed on the right when the page is first opened. Users are only able to filter users by name, other filters do not have any effect. Users can navigate to the profile pages of the users in the result by clicking on them.



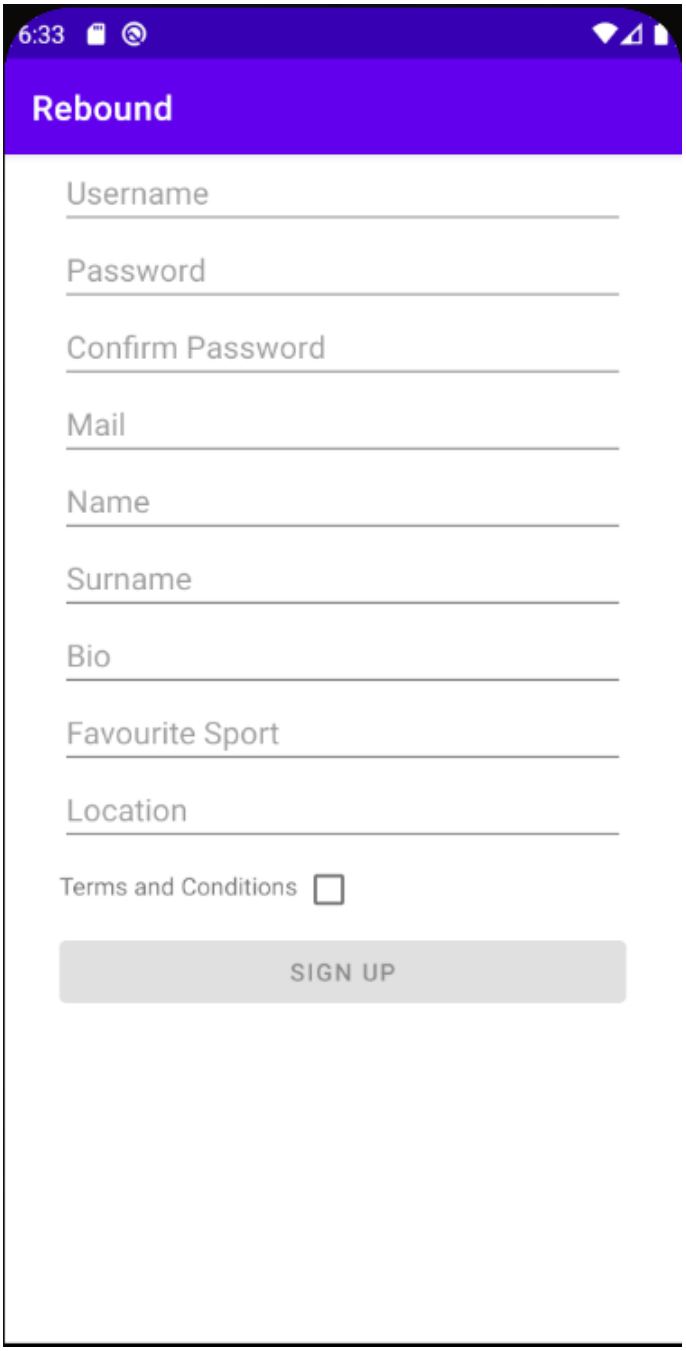
Android Application

1-) Homepage

When users first login to our application, the homepage is going to be the first page they are going to see. Here, they can see all the events created by users and they can click on them if they want further information about them. They can navigate the application using the bottom navigation bar.

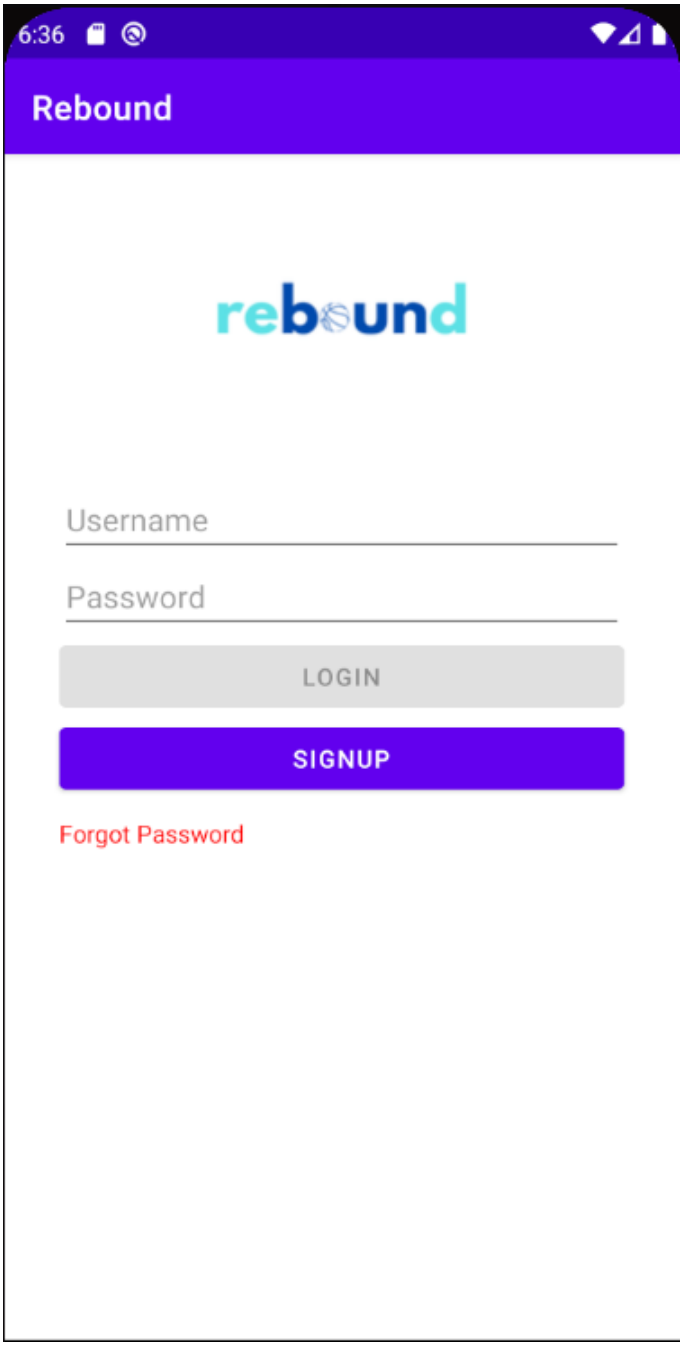
2-) Sign up

On this page users need to fill every single fillable space and accept the Terms and Conditions to sign up.



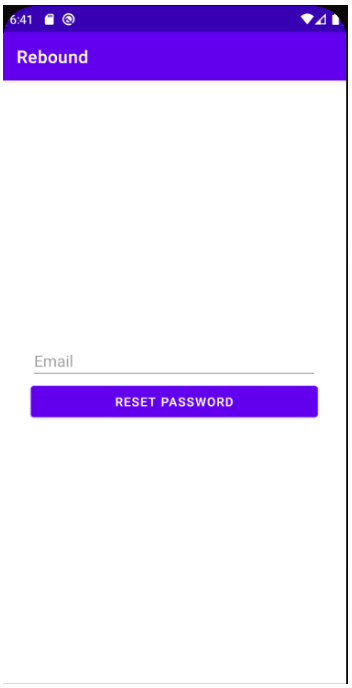
3-) Login

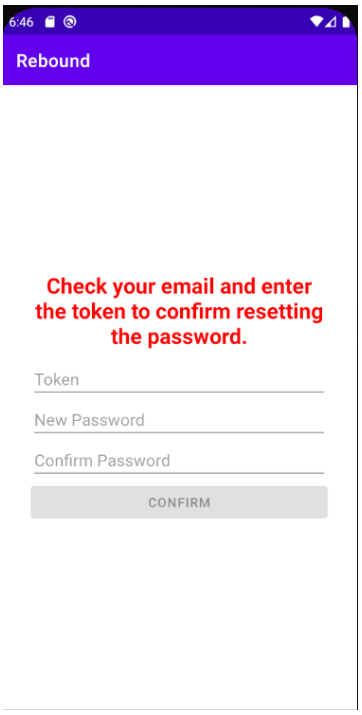
When users first launch our application they are going to meet with the Login Page. Here they can enter their credentials to login if they have signed up before or they can click “SIGNUP” to head to the signup page so they can register. If they happen to forget their password somehow they can head to the password reset page using the “Forgot Password” button on the bottom left.



4-) Forgot password

After clicking the “Forgot Password” button from the Login Page they can enter their mail to reset the password of their account. Clicking the “Reset Password” button is going to send a token to the email address so they can confirm resetting their password.

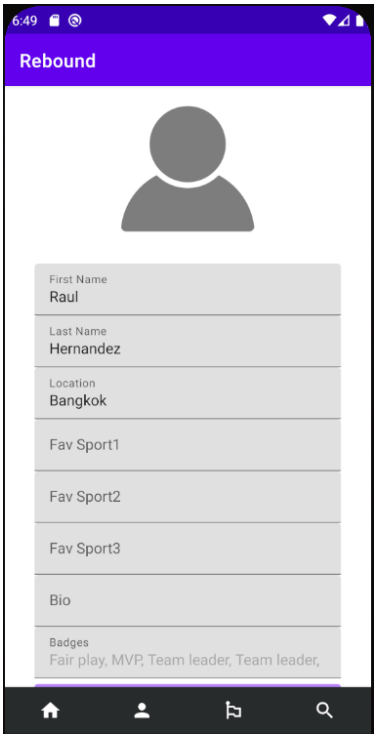


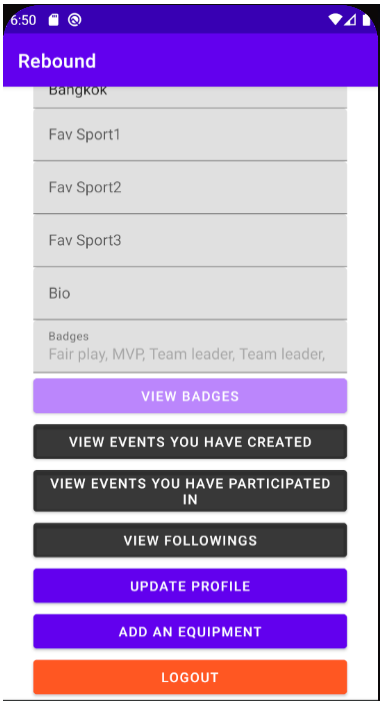


On this page users need to enter the token they receive on their email address in order to change their password.

5-) Profile page

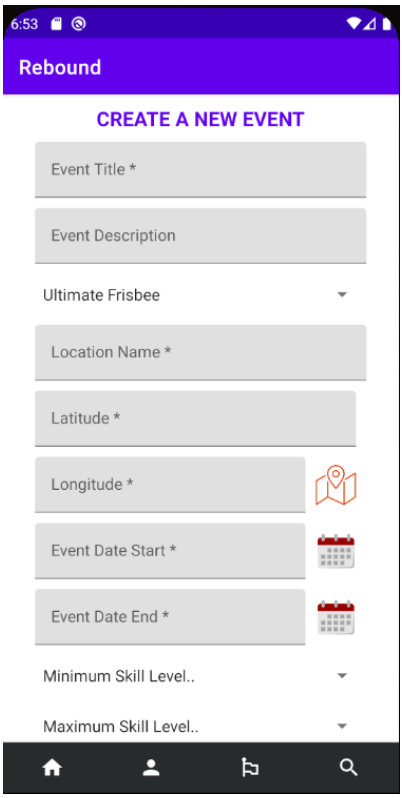
On their own profile page, users can view their badges, events they have created, events they have participated in and their followings by clicking the relevant buttons. They can click “UPDATE PROFILE“ to update their profile page after they have made a change. They can click “ADD AN EQUIPMENT” to head to the equipment creation page and “LOGOUT” button to log out.

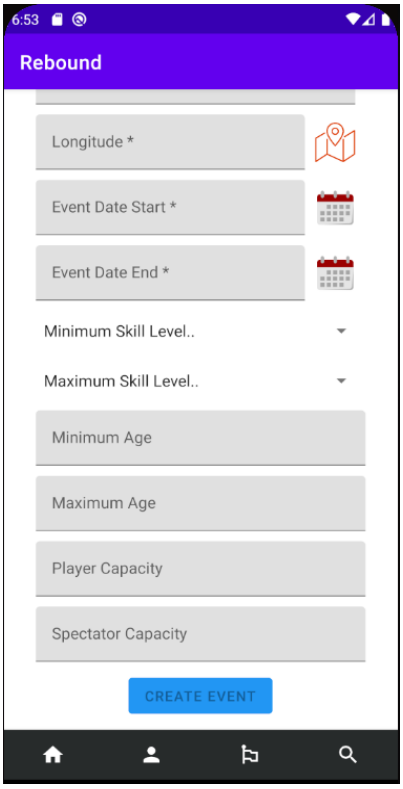




6-) New event

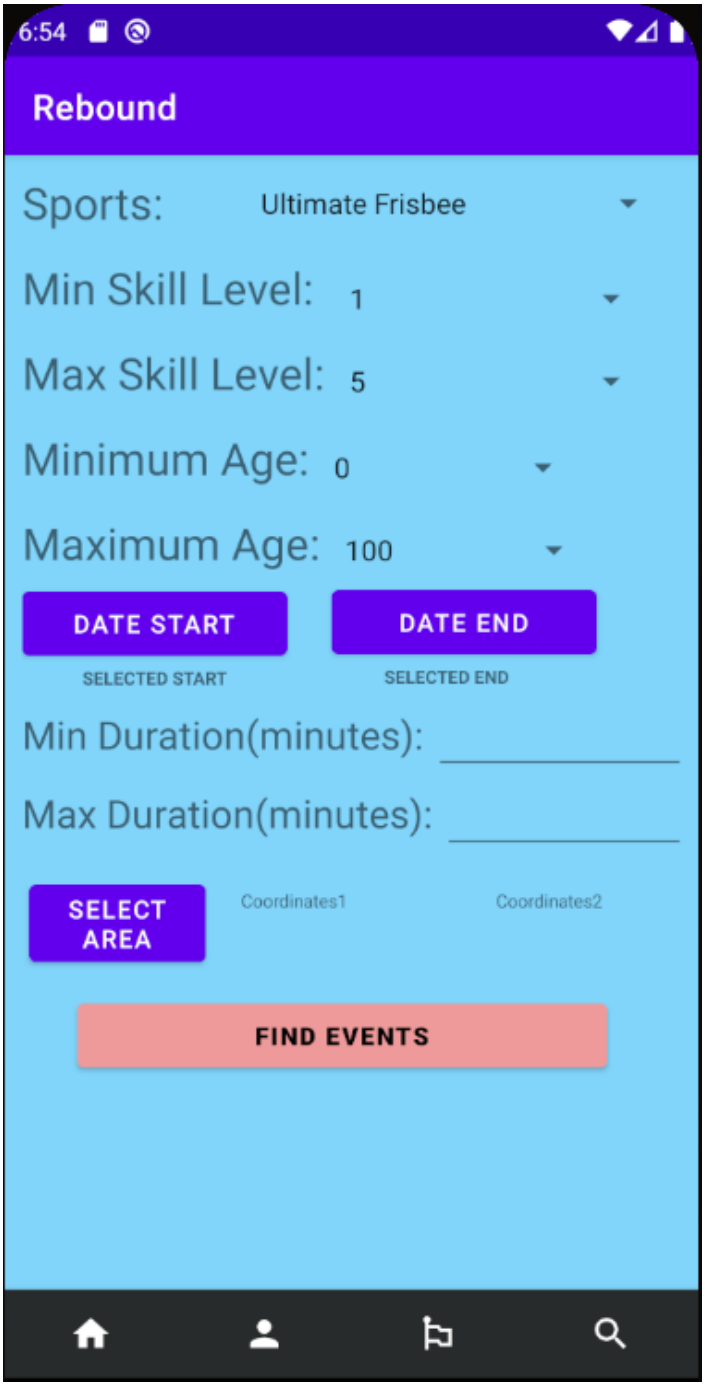
Users need to fill the spaces marked with * in order to create the event. They can click the map icon to select the event location on the map. They can click the calendar icons to select the start and end time of the event. After they click “CREATE EVENT” button they are going to receive a message telling them if event creation was successful or not.





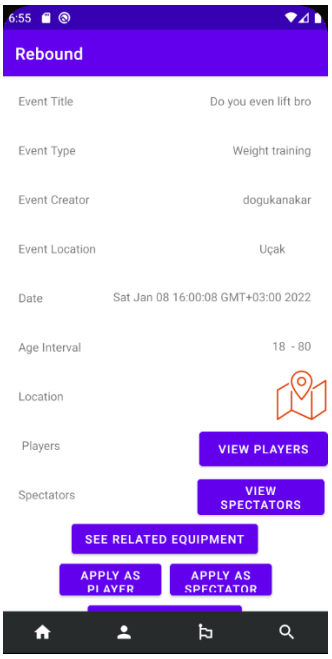
7-) Searching events

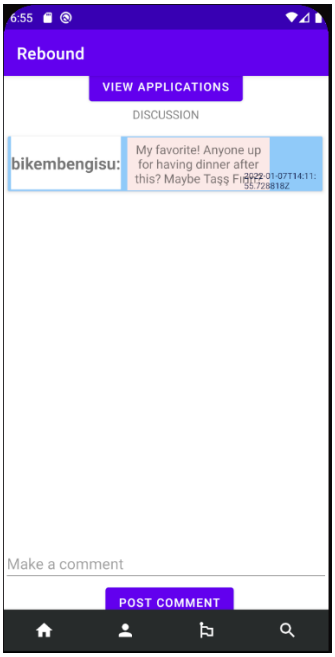
Users need to select the date interval and the area to perform a healthy event search. The other criteria are optional and if left untouched, are going to be filled with default values. After clicking the “FIND EVENTS” button, users are going to be directed to a page where they can see all the events that fit the criteria they have provided.



8-) Event pages

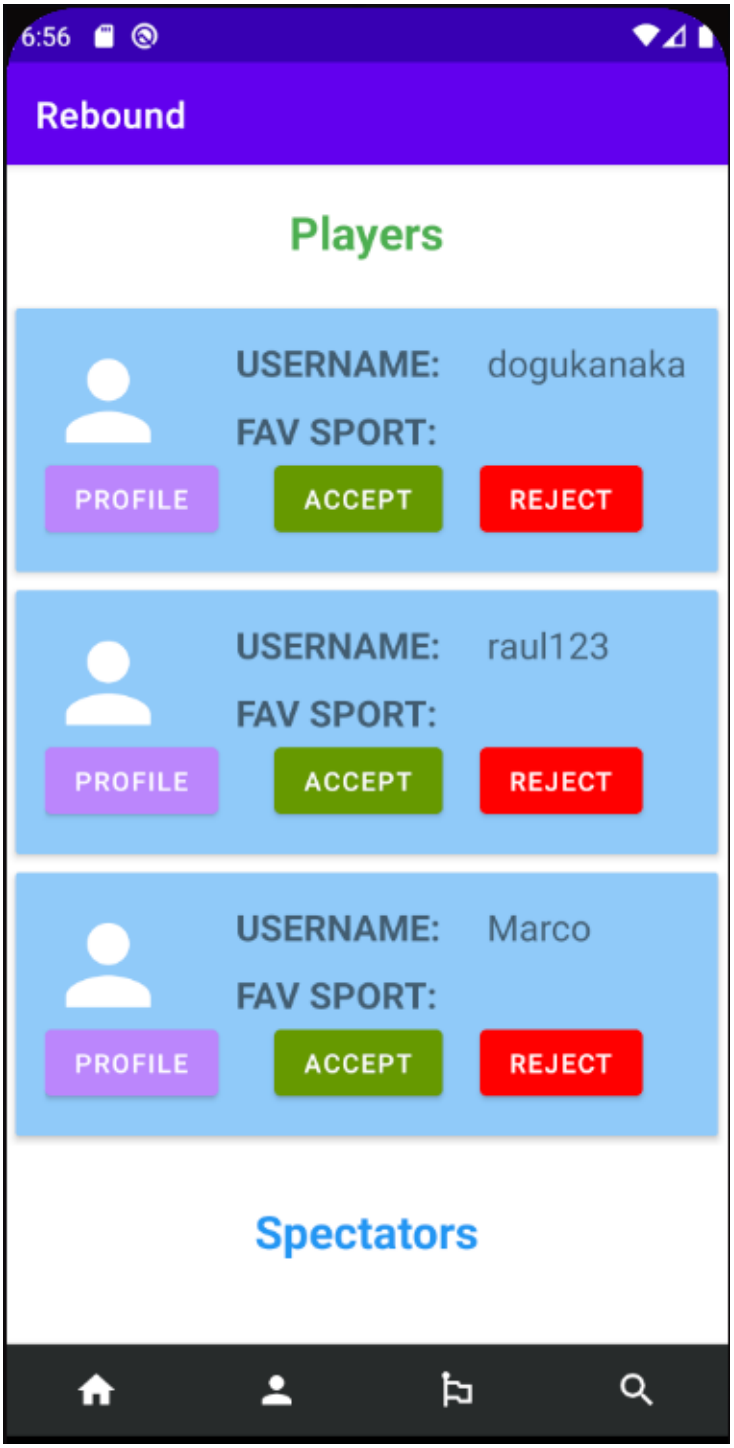
On this page, users can learn more about a specific event. They can see detailed information about it. They can click the map icon to see the event's location on a map view. They can click "VIEW PLAYERS" to view the players of this event and click "VIEW SPECTATORS" to view the spectators. They can see equipment related to the sport by clicking the "SEE RELATED EQUIPMENT" button. They can apply to the event as a player or as a spectator by clicking the buttons "APPLY AS PLAYER" or "APPLY AS SPECTATOR". Owner of this event can click "VIEW APPLICATIONS" to see the applicants of this event. Users can also communicate with other users on the Discussion section at the very bottom. They can read the comments and if they want to read the answers to a specific comment, they can click on the comment. They can also post their own comment at the very bottom by typing their comment and clicking "POST COMMENT".





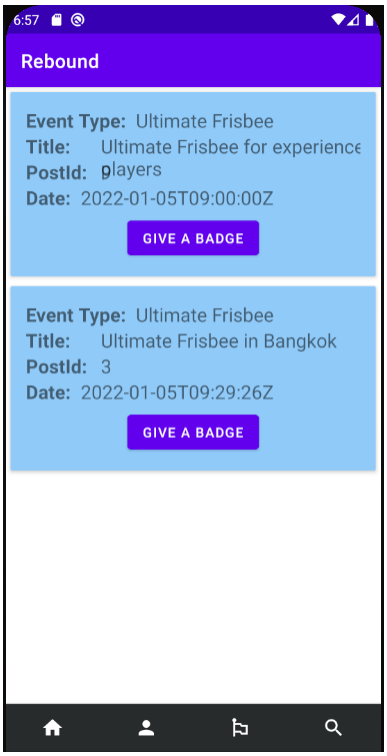
9-) Viewing applications

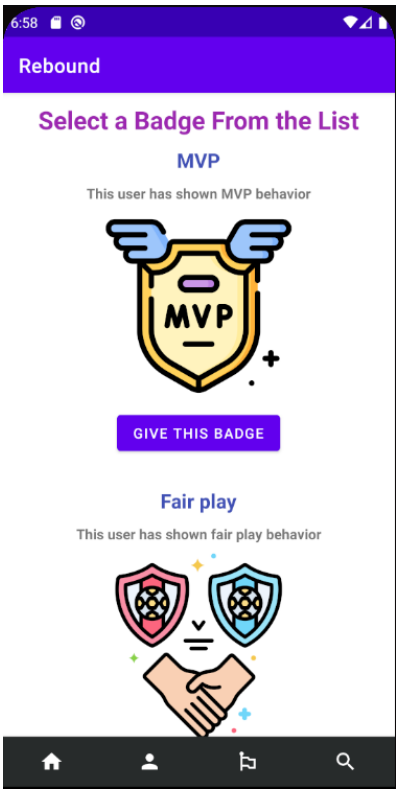
Event creators can accept or reject applicants on this page. They can view these applicants' profiles to decide whether they are going to accept or reject them.



10-) Giving badges

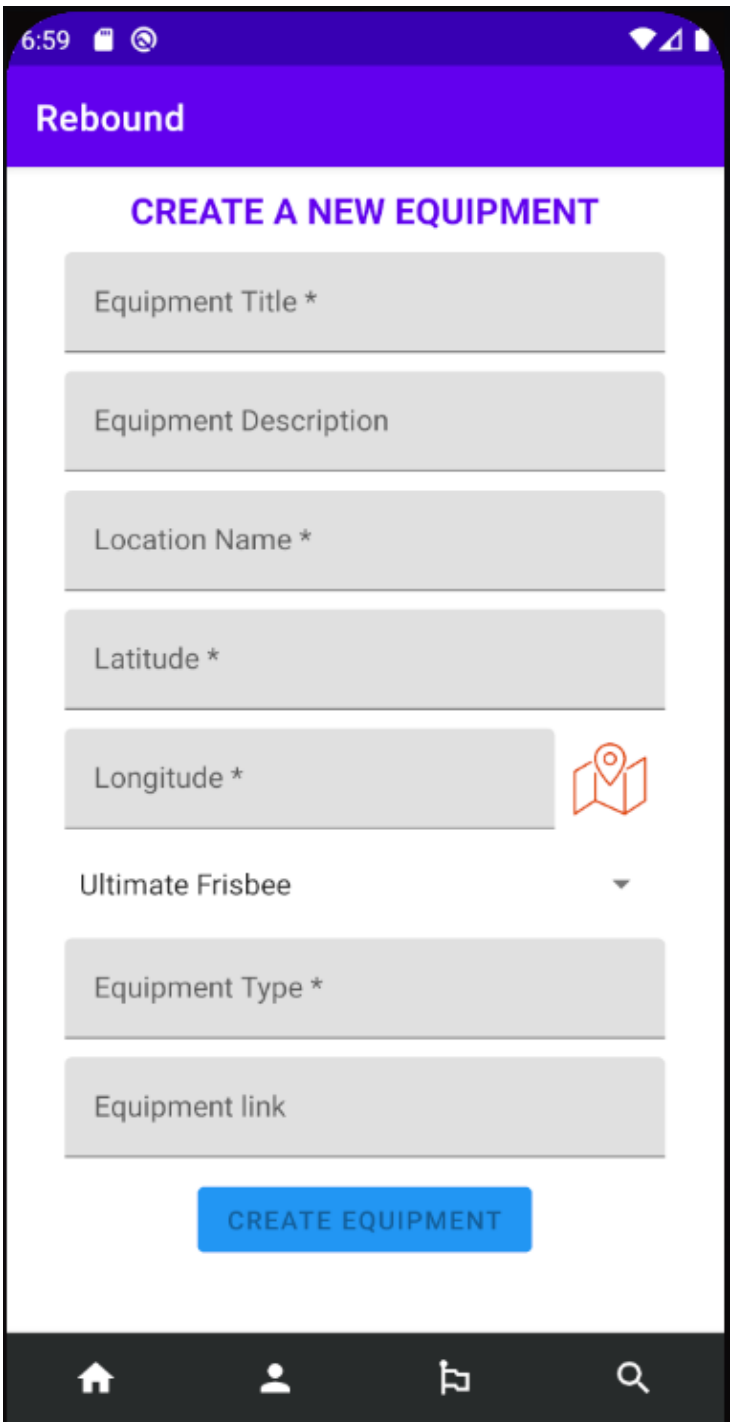
Users can give a badge to other players they have played with by first selecting the event they want to give a badge for. Then they can select a badge to give to the user by clicking “GIVE THIS BADGE” under a specific badge they desire.





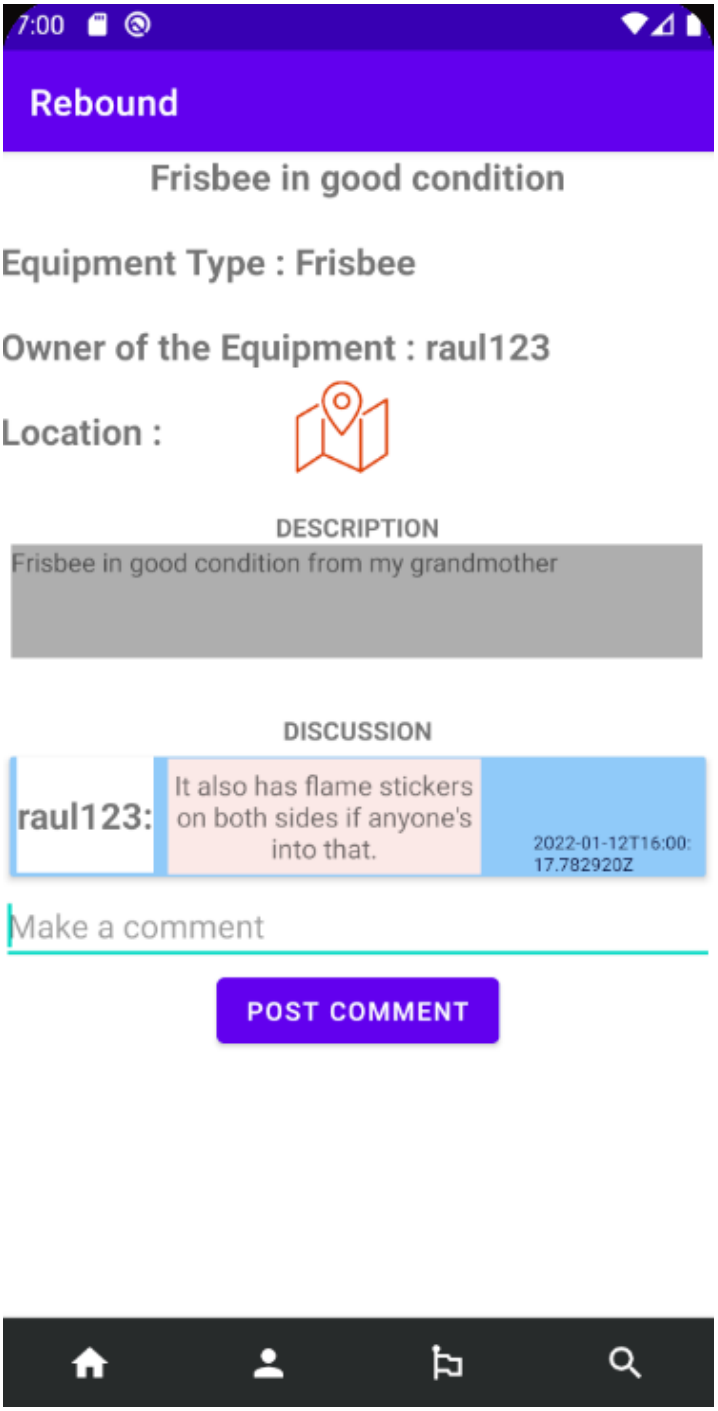
11-) New equipment post

Users need to fill all the fillable spaces in order to create the equipment they want. After clicking the “CREATE EQUIPMENT” button, they are going to receive a message whether the equipment has been successfully created or not.



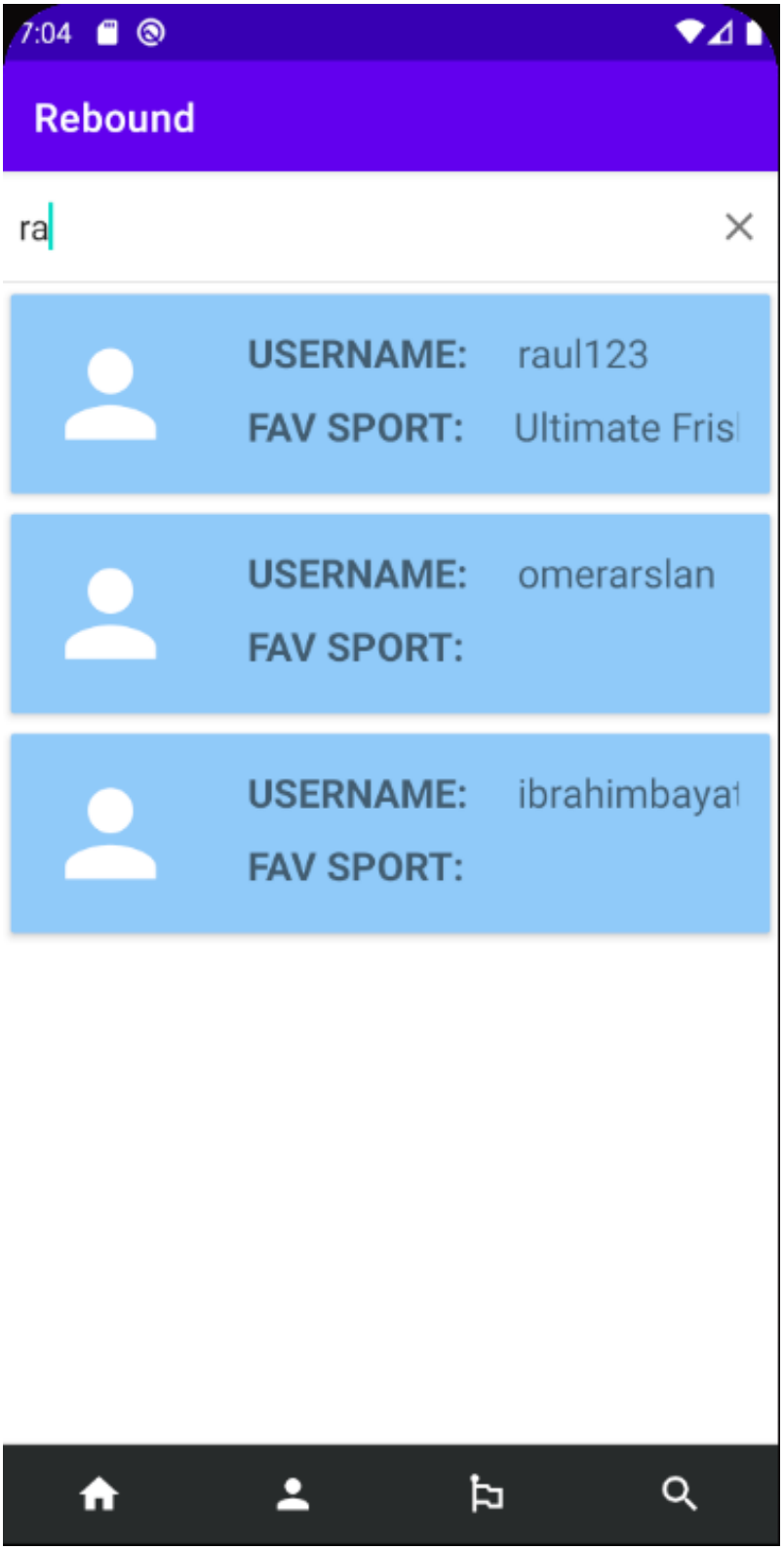
12-) Equipment pages

On these equipment pages users can see the detailed traits of the said equipment. They can click the map icon to see the location of the equipment. They can also post comments to ask questions about the equipment and overall socialize.



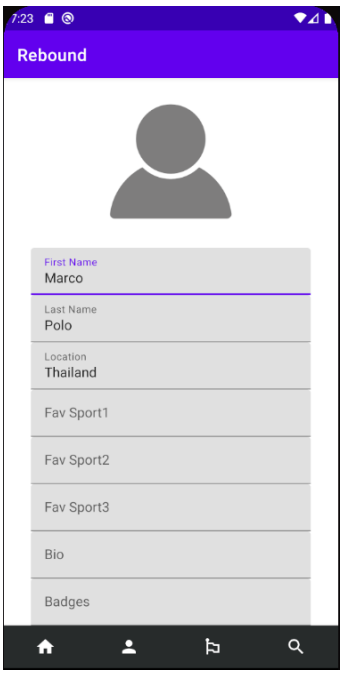
13-) Searching users

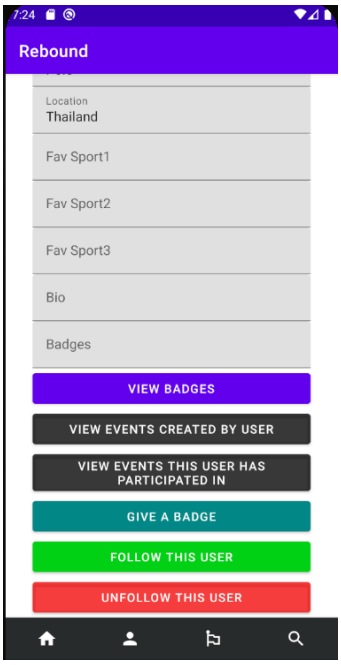
Users can use the search bar on the top to search for specific users. After they have found the users they have been searching for, they can click on them to see their detailed profile page.



14-) User Page

On this page users can see detailed information about another user. They can view user's badges, events created by this user, events this user has participated in by clicking necessary buttons. They can also follow the user by clicking "FOLLOW THIS USER" and unfollow them by clicking "UNFOLLOW THIS USER" . They can give a badge to this user if they have been in an event with them by clicking "GIVE A BADGE" button.





SYSTEM MANUAL

You can access our system manual from:
https://github.com/bounswe/2021SpringGroup2/wiki/System-Manual

Project plan

		Name	Duration	Start	Finish	Predecessors
1		**Infrastructure Setup**	**10 days?**	**3/23/21 8:00 AM**	**4/1/21 5:00 PM**	
2		Discord Server Initialization	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
3		Group First Meeting	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
4		Creating Wiki Pages	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
5		Communication Plan	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
6		Git Study	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
7		Repository Research	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
8		README	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
9		Creating Issue Labels	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
10		Creating First Issues	8 days?	3/23/21 8:00 AM	4/1/21 5:00 PM	
11		**Determining Requirements**	**10 days?**	**3/31/21 8:00 AM**	**4/9/21 5:00 PM**	
12		Glossary	10 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
13		**Functional Requirements**	**10 days?**	**3/31/21 8:00 AM**	**4/9/21 5:00 PM**	
14		User Requirements	10 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
15		System Requirements	10 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
16		**Non-functional Requirements**	**10 days?**	**3/31/21 8:00 AM**	**4/9/21 5:00 PM**	
17		Availability	8 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
18		Deployment	8 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
19		Performance	8 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
20		Privacy	8 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
21		Security	10 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
22		Standards	8 days?	3/31/21 8:00 AM	4/9/21 5:00 PM	
23		**Scenarios and Mock-ups**	**7 days?**	**4/13/21 8:00 AM**	**4/19/21 5:00 PM**	**13;16**
24		**User Personas and Their Stories**	**7 days?**	**4/13/21 8:00 AM**	**4/19/21 5:00 PM**	
25		Person #1 Eren Yiit	7 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
26		Person #2 Ali Topçu	5 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
27		Person #3 Melike Takran	5 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
28		Person #4 Fatih Kerim	5 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
29		Person #5 Zülfikar Tunç	7 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
30		**Scenarios and Mock-ups**	**7 days?**	**4/13/21 8:00 AM**	**4/19/21 5:00 PM**	
31		Scenario #1: Creating an Event	7 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
32		Scenario #2: User Searches for an Event	5 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
33		Scenario #3: User Applies for an Event to ...	5 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
34		Scenario #4: Event Creator Selects from ...	5 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	

Projectplan - page1

		Name	Duration	Start	Finish	Predecessors
35		Scenario #5: Discussion	7 days?	4/13/21 8:00 AM	4/19/21 5:00 PM	
36		**Design Diagrams**	**11 days?**	**4/20/21 8:00 AM**	**4/30/21 5:00 PM**	**24;30**
37		Class Diagram	9 days?	4/20/21 8:00 AM	4/30/21 5:00 PM	
38		Use-Case Diagram	9 days?	4/20/21 8:00 AM	4/30/21 5:00 PM	
39		Sequence Diagram	9 days?	4/20/21 8:00 AM	4/30/21 5:00 PM	
40		Milestone #1	12 days?	4/28/21 8:00 AM	5/13/21 5:00 PM	36
41		Research and development	16 days?	5/18/21 8:00 AM	6/8/21 5:00 PM	
42		Code versioning, Code review	6 days?	5/18/21 8:00 AM	5/25/21 5:00 PM	
43		**Designing the Project Structure**	**8 days?**	**5/25/21 8:00 AM**	**6/1/21 5:00 PM**	**40**
44		Designing Frontend Part	6 days?	5/25/21 8:00 AM	6/1/21 5:00 PM	
45		Designing Android Part	6 days?	5/25/21 8:00 AM	6/1/21 5:00 PM	
46		Designing Backend Part	6 days?	5/25/21 8:00 AM	6/1/21 5:00 PM	
47		**Verification and Validation**	**8 days?**	**6/2/21 8:00 AM**	**6/9/21 5:00 PM**	**43**
48		**Testing**	**8 days?**	**6/2/21 8:00 AM**	**6/9/21 5:00 PM**	
49		Unit Tests	6 days?	6/2/21 8:00 AM	6/9/21 5:00 PM	
50		User Tests	6 days?	6/2/21 8:00 AM	6/9/21 5:00 PM	
51		Debugging	6 days?	6/2/21 8:00 AM	6/9/21 5:00 PM	
52		Configuration Management	6 days?	6/8/21 8:00 AM	6/15/21 5:00 PM	
53		Deployment	6 days?	6/8/21 8:00 AM	6/15/21 5:00 PM	
54		Dockerization	6 days?	6/8/21 8:00 AM	6/15/21 5:00 PM	
55		Lessons Learned	6 days?	6/8/21 8:00 AM	6/15/21 5:00 PM	
56		**Milestone2**	**11 days?**	**6/15/21 8:00 AM**	**6/25/21 5:00 PM**	**47**
57		Executive Summary	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
58		List and Status of Deliverables	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
59		Evaluation of the status of deliverables	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
60		Evaluation of tools and processes	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
61		Summary of Work Done By Each Member	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
62		Project Plan	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
63		RAM	9 days?	6/15/21 8:00 AM	6/25/21 5:00 PM	
64		Review of CMPE352 and the project status	14 days?	10/25/21 8:00 AM	11/11/21 5:00 PM	
65		**Implementation Phase - I**	**21 days?**	**11/12/21 8:00 AM**	**12/2/21 5:00 PM**	**56;64**
66		**Backend**	**21 days?**	**11/12/21 8:00 AM**	**12/2/21 5:00 PM**	
67		Initialization	3 days?	11/12/21 8:00 AM	11/16/21 5:00 PM	
68		Database design	3 days?	11/12/21 8:00 AM	11/16/21 5:00 PM	

		Name	Duration	Start	Finish	Predecessors
69		Register Implementation	3 days?	11/17/21 8:00 AM	11/19/21 5:00 PM	67;68
70		Login implementation	3 days?	11/17/21 8:00 AM	11/19/21 5:00 PM	67;68
71		User Profiles	5 days?	11/20/21 8:00 AM	11/26/21 5:00 PM	67;68
72		Profile Privacy Options	4 days?	11/27/21 8:00 AM	12/2/21 5:00 PM	71
73		**Frontend**	**19 days?**	**11/12/21 8:00 AM**	**11/30/21 5:00 PM**	
74		Initialization	3 days	11/12/21 8:00 AM	11/16/21 5:00 PM	
75		Homepage	5 days?	11/17/21 8:00 AM	11/23/21 5:00 PM	74
76		Navigation Bar	3 days?	11/17/21 8:00 AM	11/19/21 5:00 PM	74
77		Register Page	3 days?	11/24/21 8:00 AM	11/26/21 5:00 PM	75
78		Login Page	2 days?	11/24/21 8:00 AM	11/25/21 5:00 PM	75
79		Profile Page	5 days?	11/24/21 8:00 AM	11/30/21 5:00 PM	75
80		**Android**	**19 days?**	**11/12/21 8:00 AM**	**11/30/21 5:00 PM**	
81		Initialization	3 days?	11/12/21 8:00 AM	11/16/21 5:00 PM	
82		Homepage	5 days	11/17/21 8:00 AM	11/23/21 5:00 PM	81
83		Navigation Bar	2 days	11/17/21 8:00 AM	11/18/21 5:00 PM	81
84		Register Page	3 days	11/24/21 8:00 AM	11/26/21 5:00 PM	82
85		Login Page	3 days	11/24/21 8:00 AM	11/26/21 5:00 PM	82
86		Profile Page	5 days	11/24/21 8:00 AM	11/30/21 5:00 PM	82
87		Test and Review	2 days	12/1/21 8:00 AM	12/2/21 5:00 PM	86
88		Milestone 1 Presentation	1 day?	12/1/21 8:00 AM	12/1/21 5:00 PM	86
89		**Implementation Phase - II**	**37 days?**	**11/30/21 8:00 AM**	**1/5/22 5:00 PM**	
90		**Backend**	**36 days**	**11/30/21 8:00 AM**	**1/4/22 5:00 PM**	
91		Post Creation / View	9 days	11/30/21 8:00 AM	12/10/21 5:00 PM	
92		Event Application	8 days	12/9/21 8:00 AM	12/20/21 5:00 PM	
93		Comments/Answers	9 days	12/17/21 8:00 AM	12/29/21 5:00 PM	
94		Follow/Block Functionalities	7 days	12/26/21 8:00 AM	1/4/22 5:00 PM	
95		**Frontend**	**37 days?**	**11/30/21 8:00 AM**	**1/5/22 5:00 PM**	
96		Post Creation / View	11 days	11/30/21 8:00 AM	12/14/21 5:00 PM	
97		Event Application	11 days	12/11/21 8:00 AM	12/27/21 5:00 PM	
98		Follow/Block Functionalities	1 day?	11/30/21 8:00 AM	11/30/21 5:00 PM	
99		Comments/Answers	11 days	12/22/21 8:00 AM	1/5/22 5:00 PM	
100		**Android**	**37 days?**	**11/30/21 8:00 AM**	**1/5/22 5:00 PM**	
101		Post Creation / View	11 days	11/30/21 8:00 AM	12/14/21 5:00 PM	
102		Event Application	11 days	12/11/21 8:00 AM	12/27/21 5:00 PM	

		Name	Duration	Start	Finish	Predecessors
103		Comments/Answers	11 days	12/22/21 8:00 AM	1/5/22 5:00 PM	
104		Follow/Block Functionalities	1 day?	11/30/21 8:00 AM	11/30/21 5:00 PM	
105		Tests and Review	2 days	1/1/22 8:00 AM	1/4/22 5:00 PM	
106		Milestone 2 Presentation	1 day?	1/3/22 8:00 AM	1/3/22 5:00 PM	
107		**Implementation Phase - III**	**31 days**	**1/4/22 8:00 AM**	**2/3/22 5:00 PM**	
108		**Backend**	**30 days**	**1/4/22 8:00 AM**	**2/2/22 5:00 PM**	
109		Search System	8 days	1/4/22 8:00 AM	1/13/22 5:00 PM	
110		Search filtering/sorting	6 days	1/12/22 8:00 AM	1/19/22 5:00 PM	
111		Recommendation System	7 days	1/18/22 8:00 AM	1/26/22 5:00 PM	
112		Notification System	7 days	1/25/22 8:00 AM	2/2/22 5:00 PM	
113		**Frontend**	**31 days**	**1/4/22 8:00 AM**	**2/3/22 5:00 PM**	
114		Search System	9 days	1/4/22 8:00 AM	1/14/22 5:00 PM	
115		Recommendation System	10 days	1/13/22 8:00 AM	1/26/22 5:00 PM	
116		Notification System	9 days	1/23/22 8:00 AM	2/3/22 5:00 PM	
117		**Android**	**31 days**	**1/4/22 8:00 AM**	**2/3/22 5:00 PM**	
118		Search System	9 days	1/4/22 8:00 AM	1/14/22 5:00 PM	
119		Recommendation System	10 days	1/13/22 8:00 AM	1/26/22 5:00 PM	
120		Notification System	9 days	1/23/22 8:00 AM	2/3/22 5:00 PM	
121		Milestone 3 and Final Presentation	1 day?	2/1/22 8:00 AM	2/1/22 5:00 PM	
122		Test and Finalizations	14 days	2/1/22 8:00 AM	2/18/22 5:00 PM	

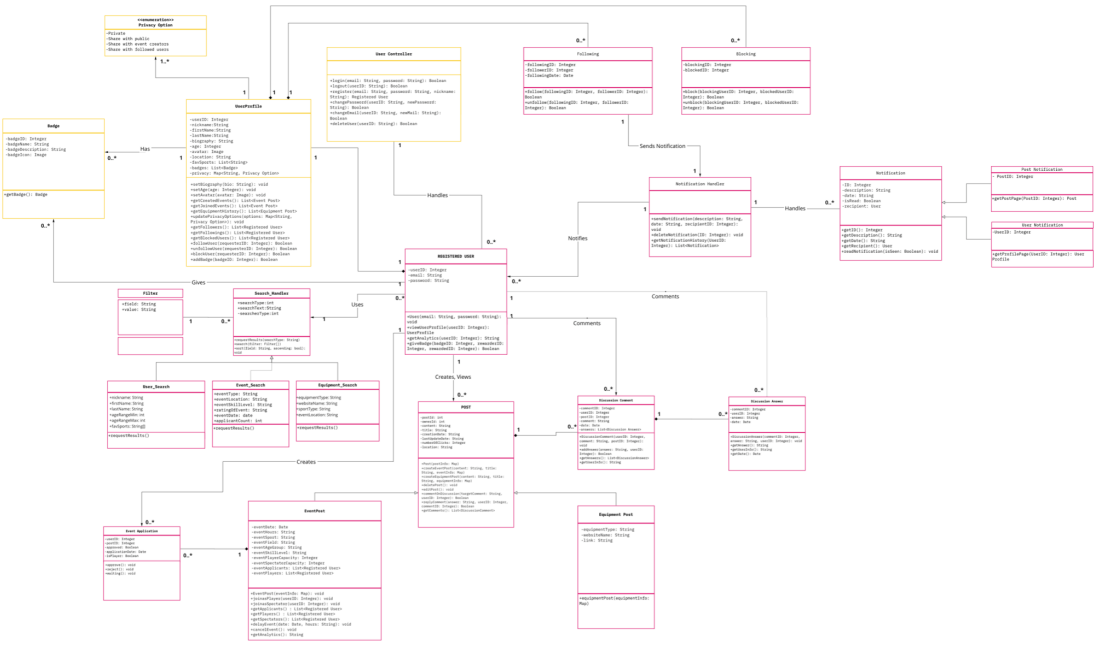
Projectplan - page22

The full project plan can be found in https://github.com/bounswe/2021SpringGroup2/blob/master/deliverables/project_plan_final_final.pod

UML Diagrams

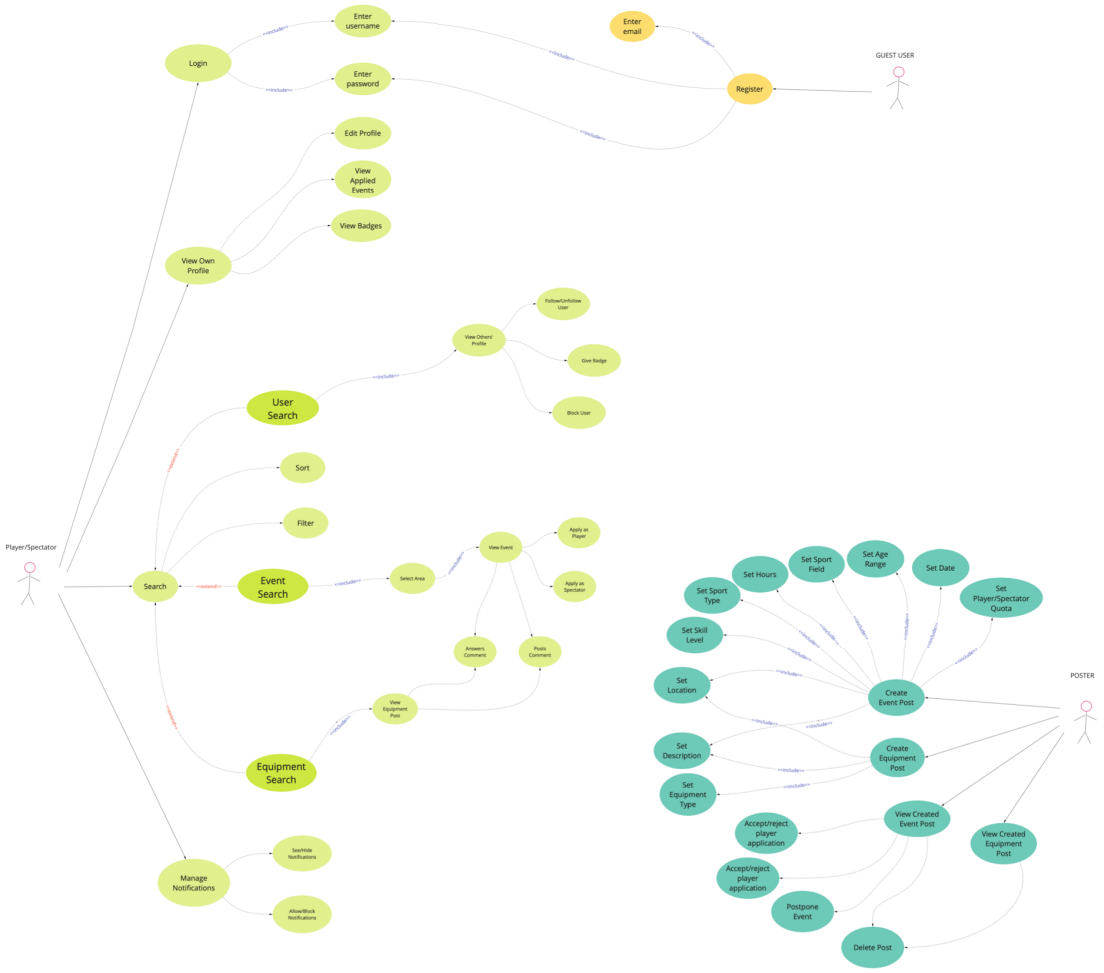
Class Diagram

Click here for high resolution.



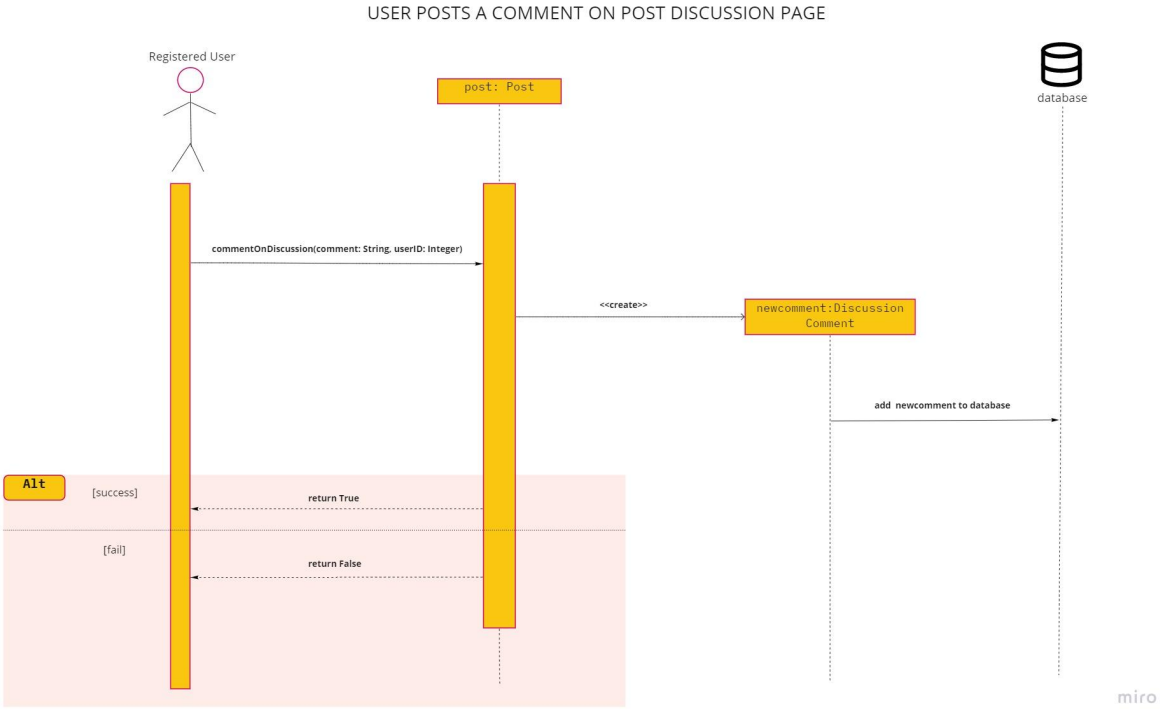
Use Case Diagram

Click here for high resolution.

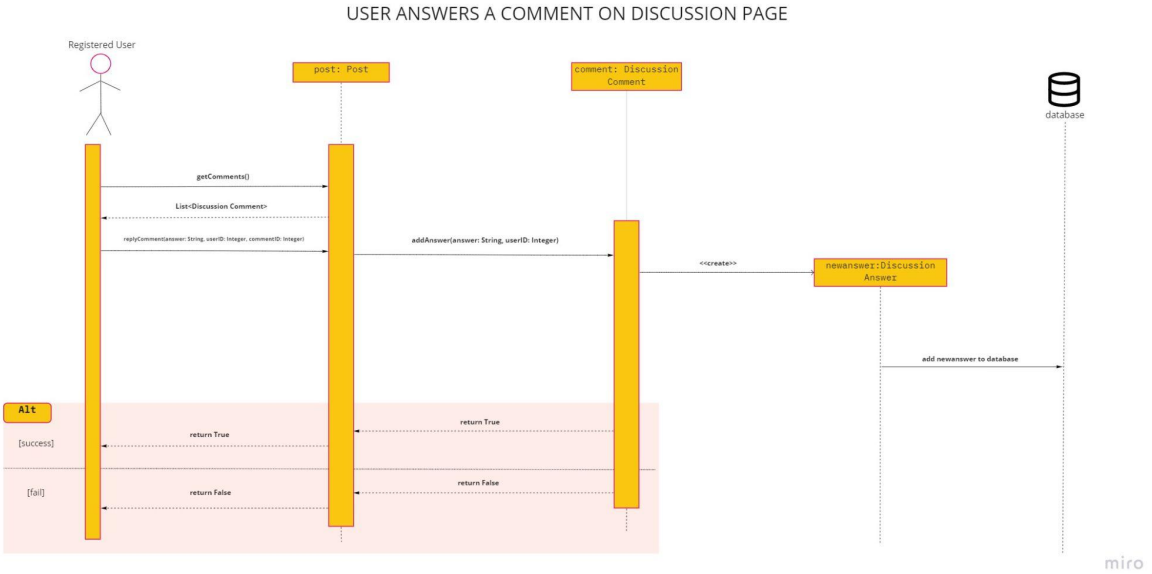


Sequence Diagrams

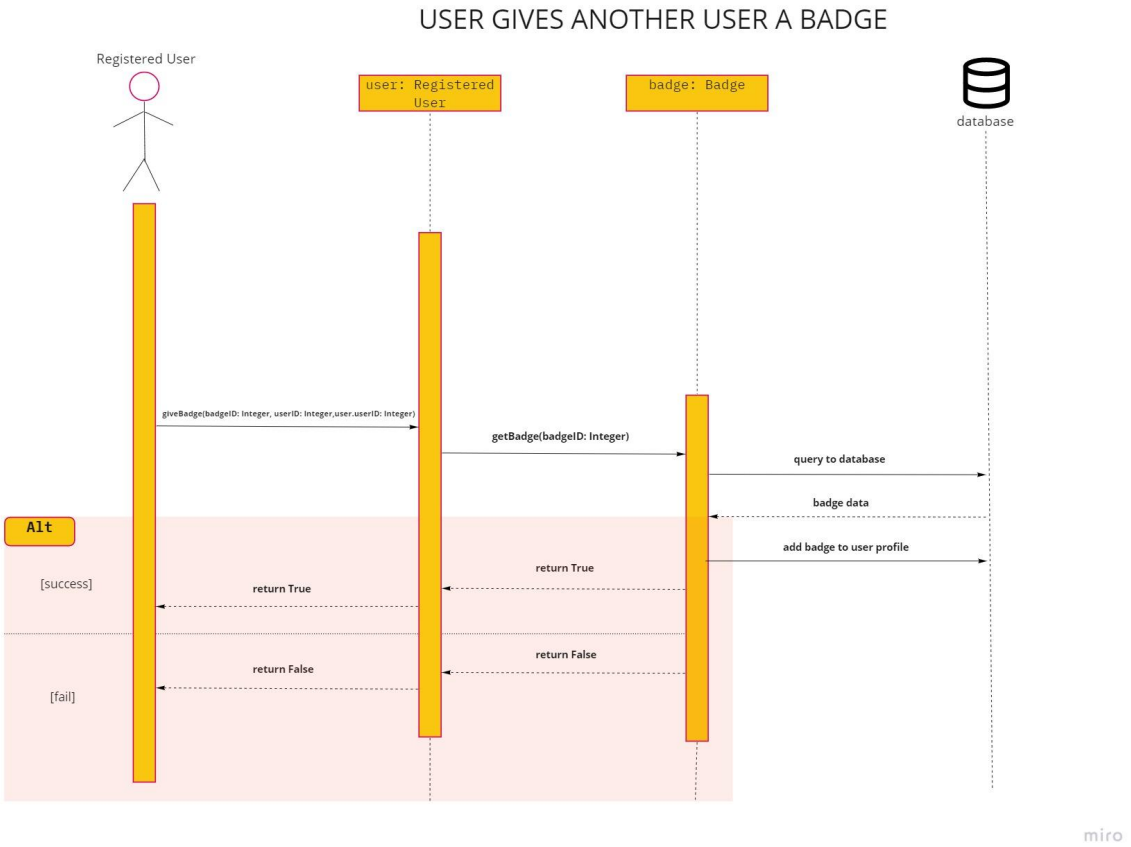
Posting a comment on discussion page



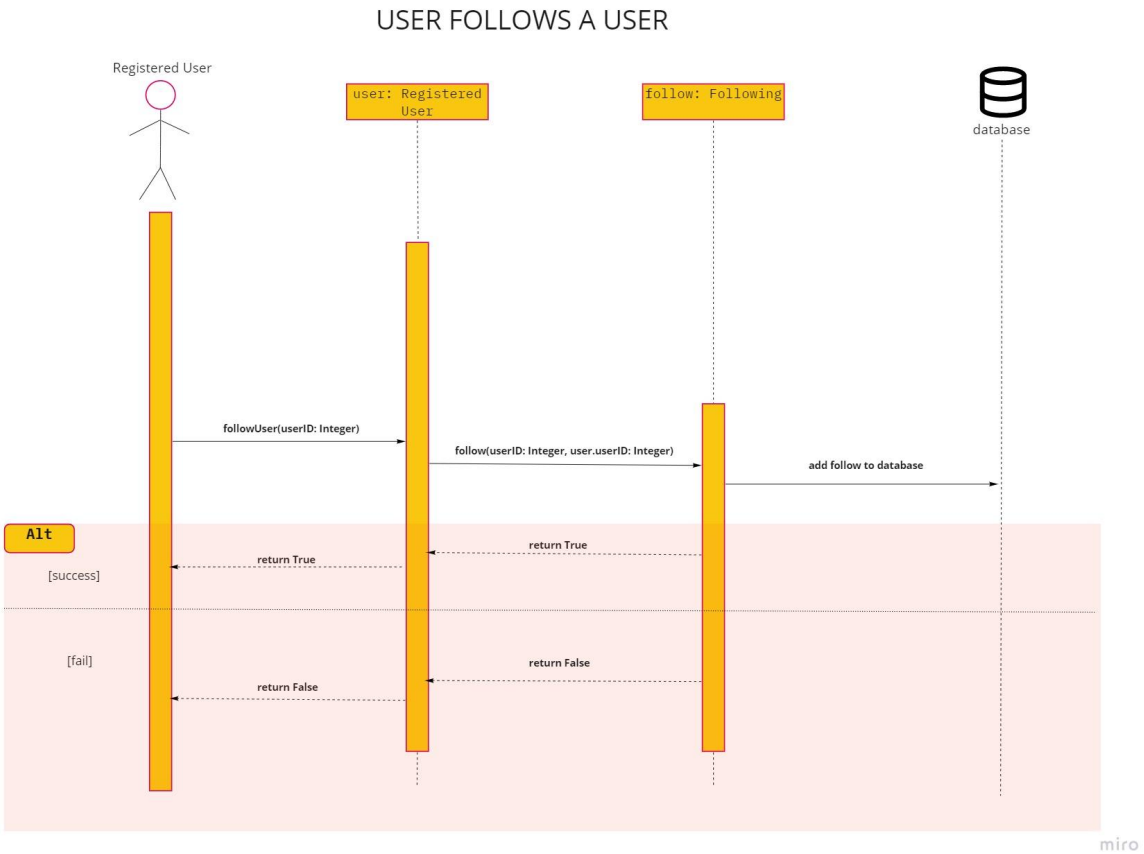
Answering a comment on discussion page



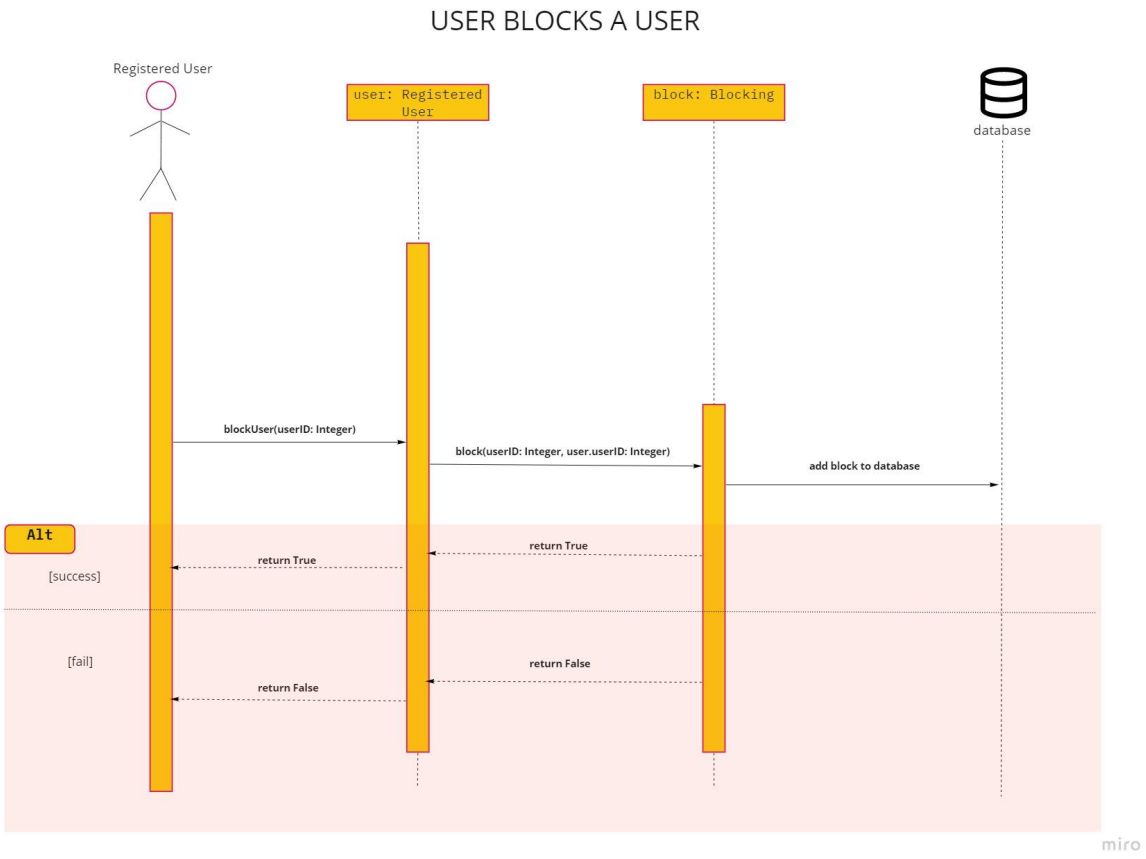
Giving a badge to another user



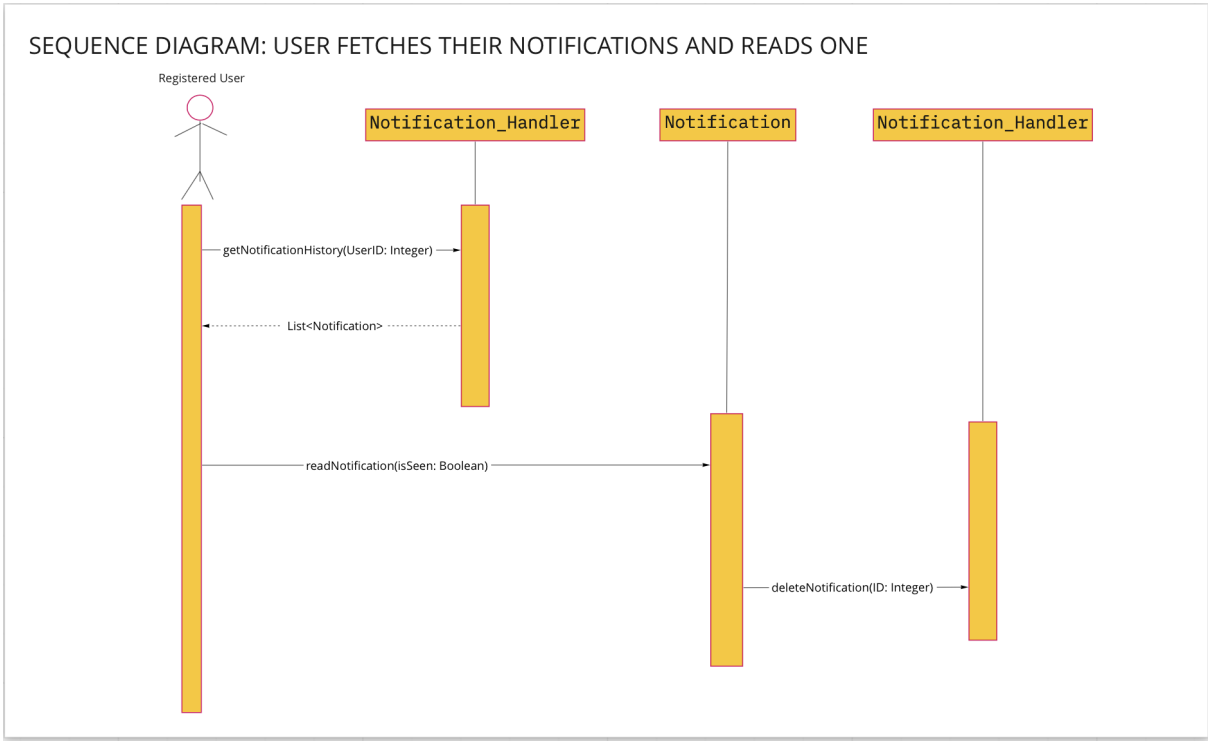
Following a user



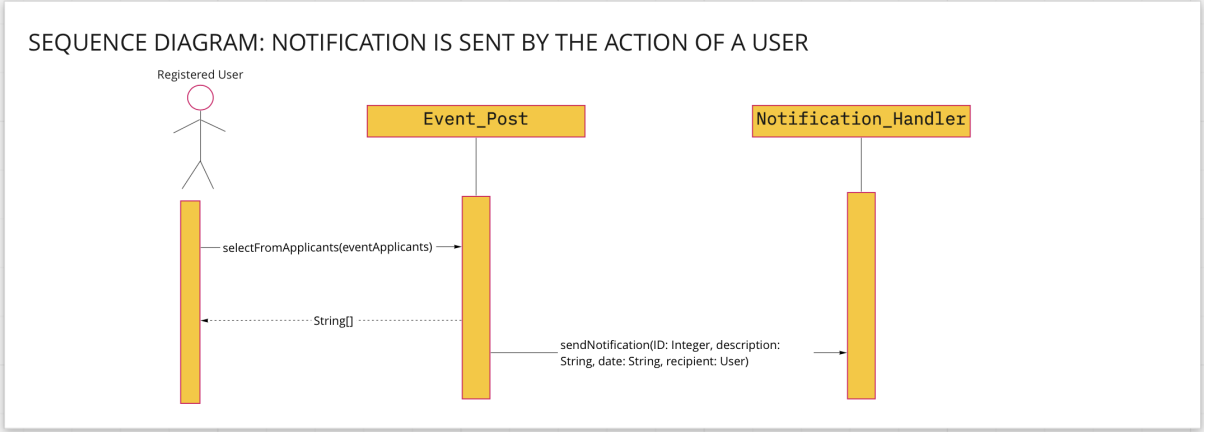
Blocking a user



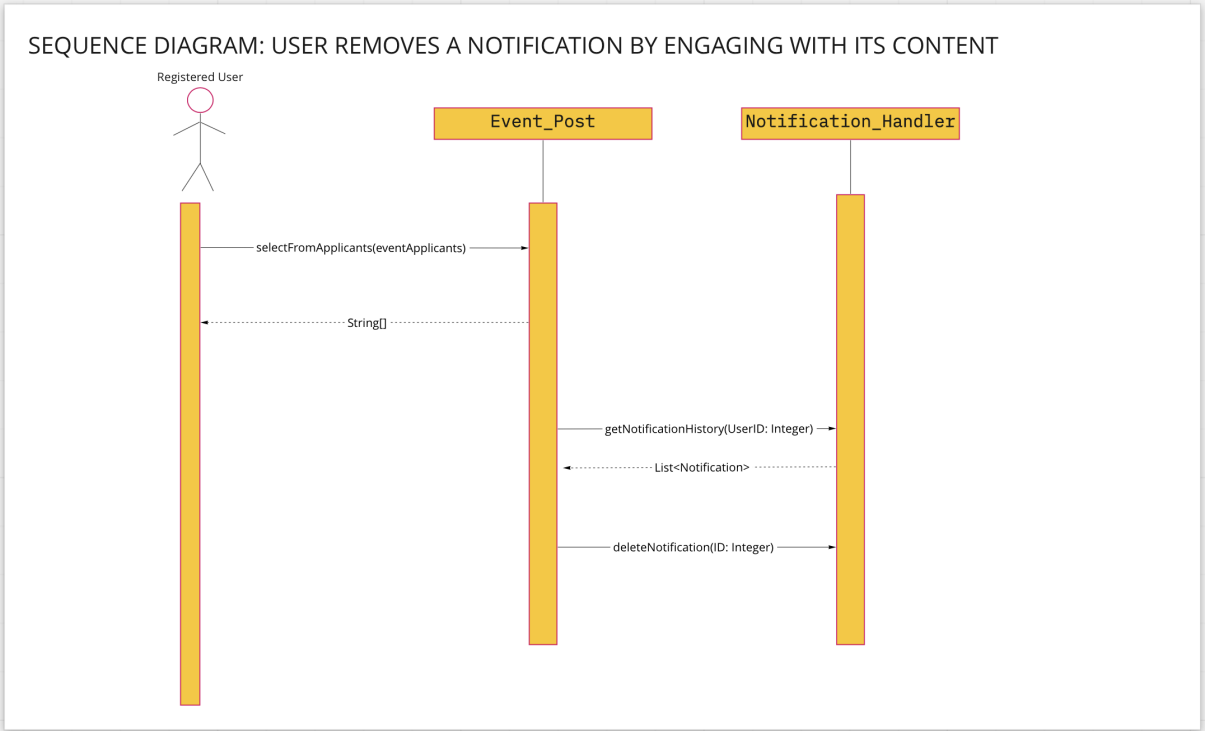
Fetching notifications and reading one



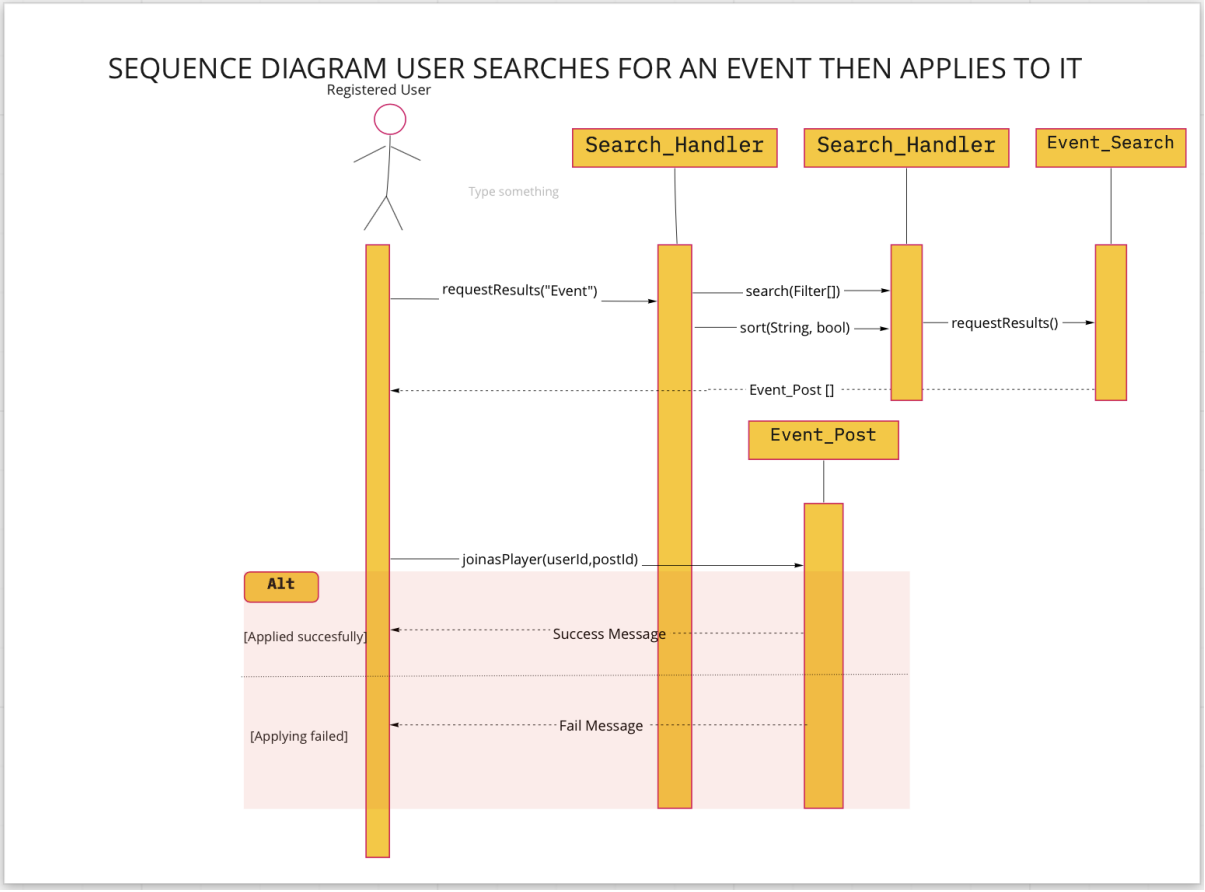
A notification is sent as a result of a user action



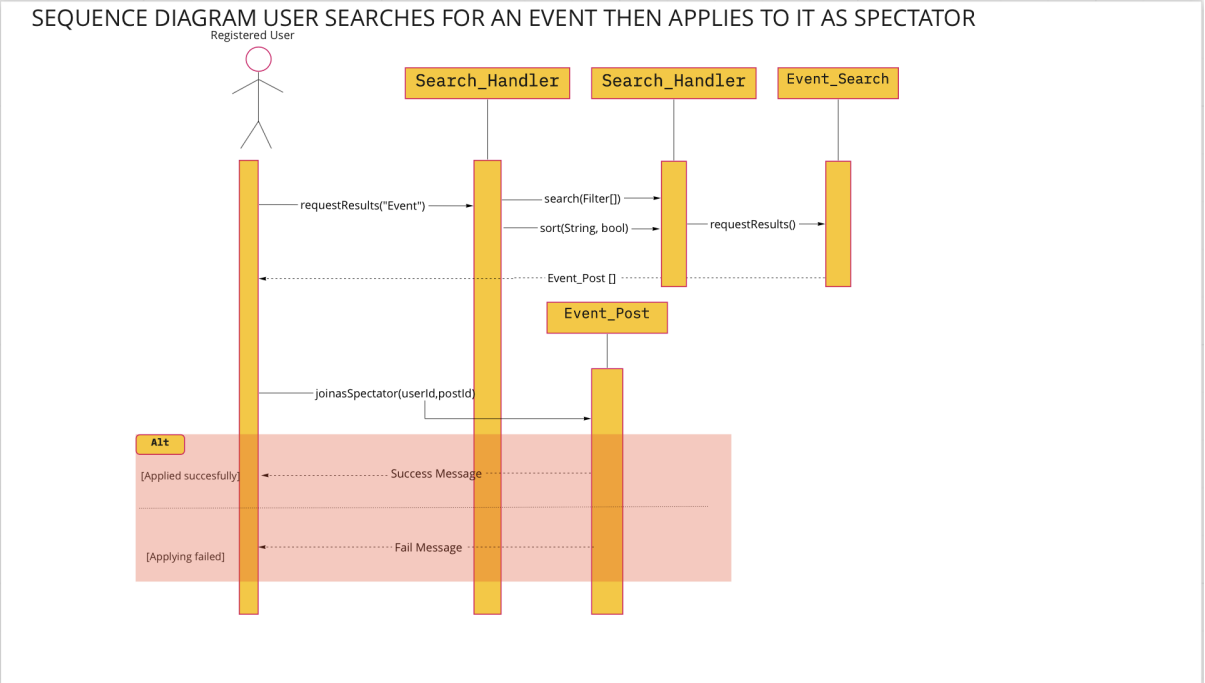
A notification is removed when a user engages with the content of that notification



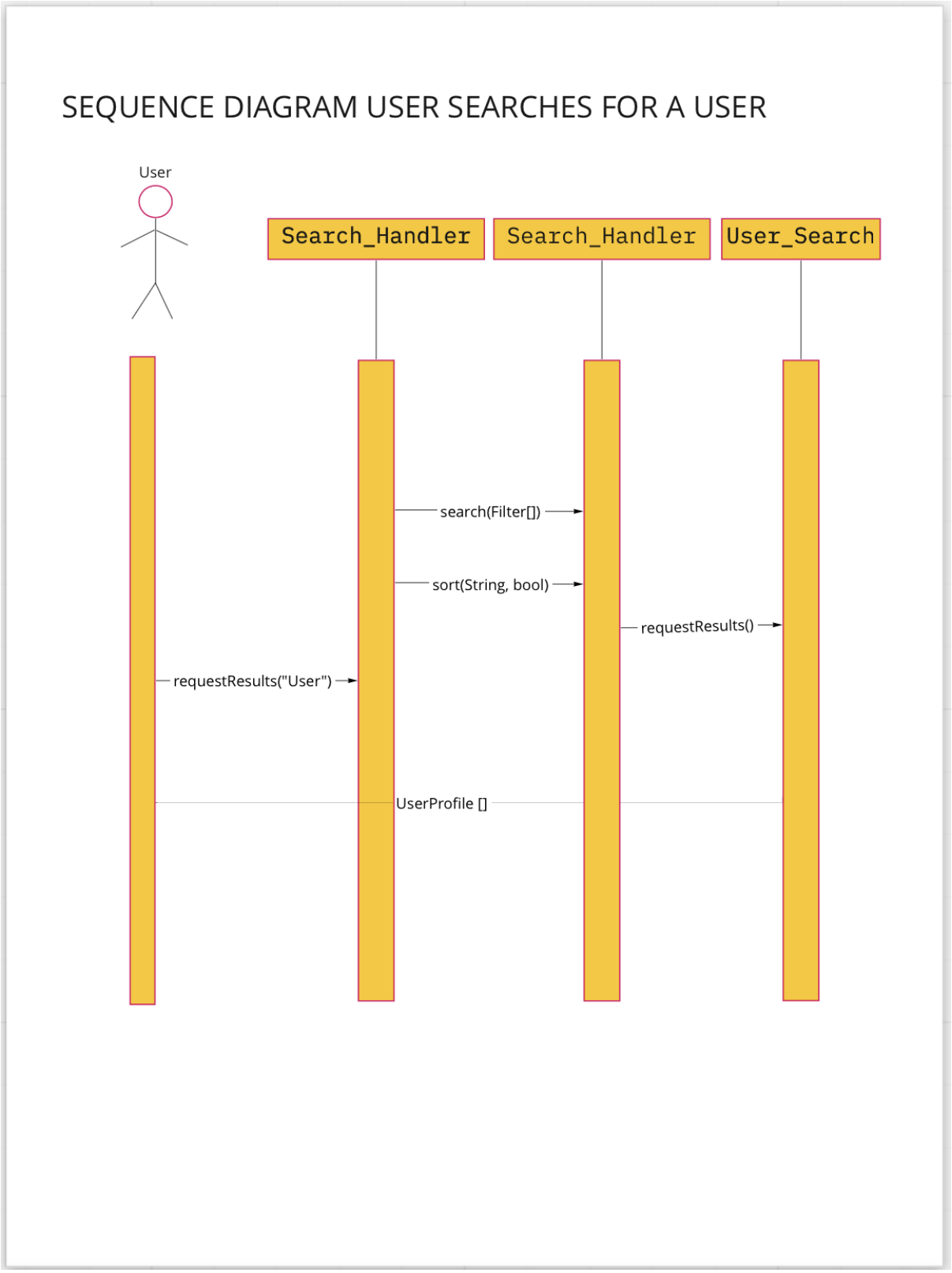
A user finds an event and applies to it



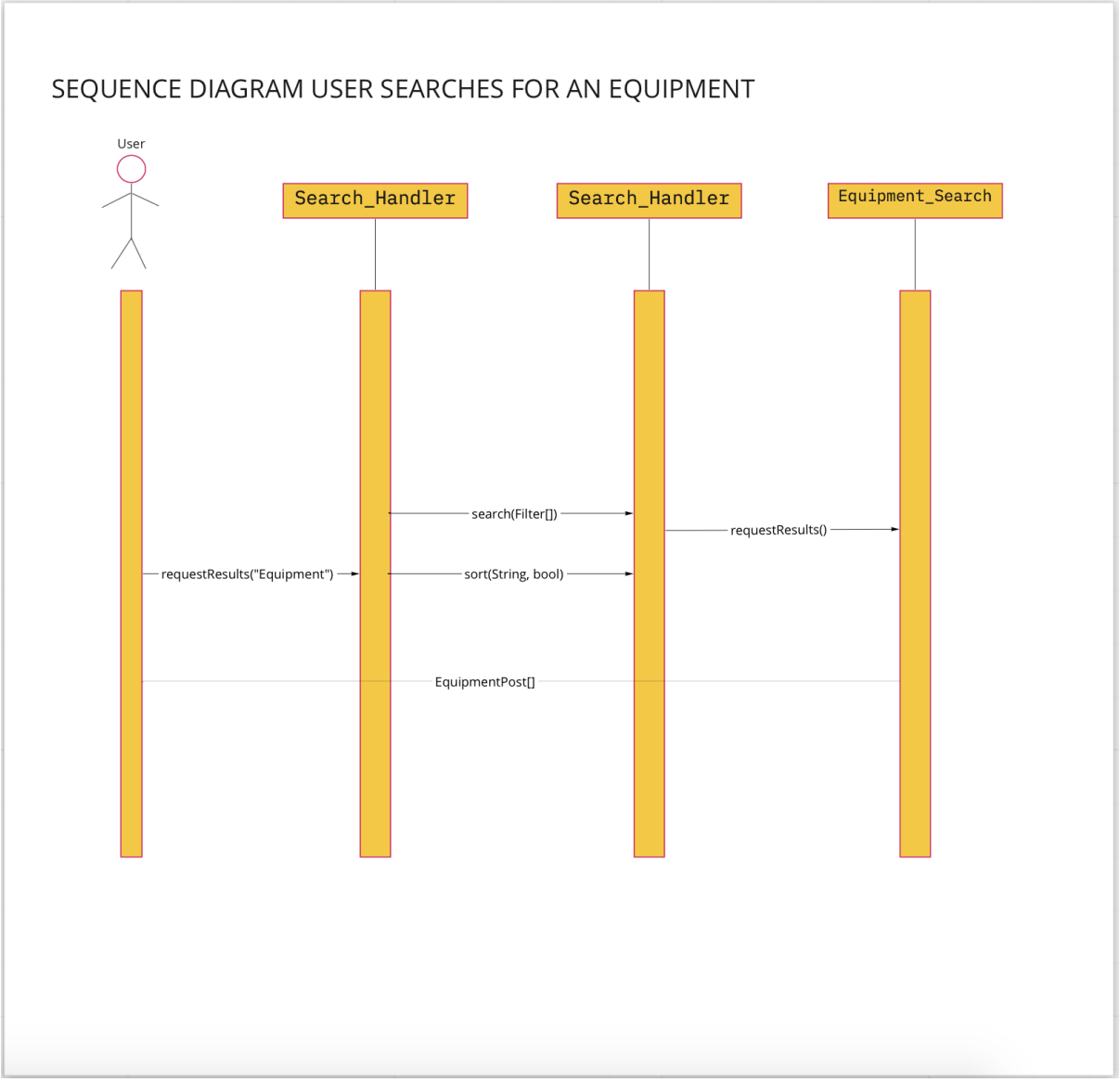
A user finds an event and applies to it as spectator



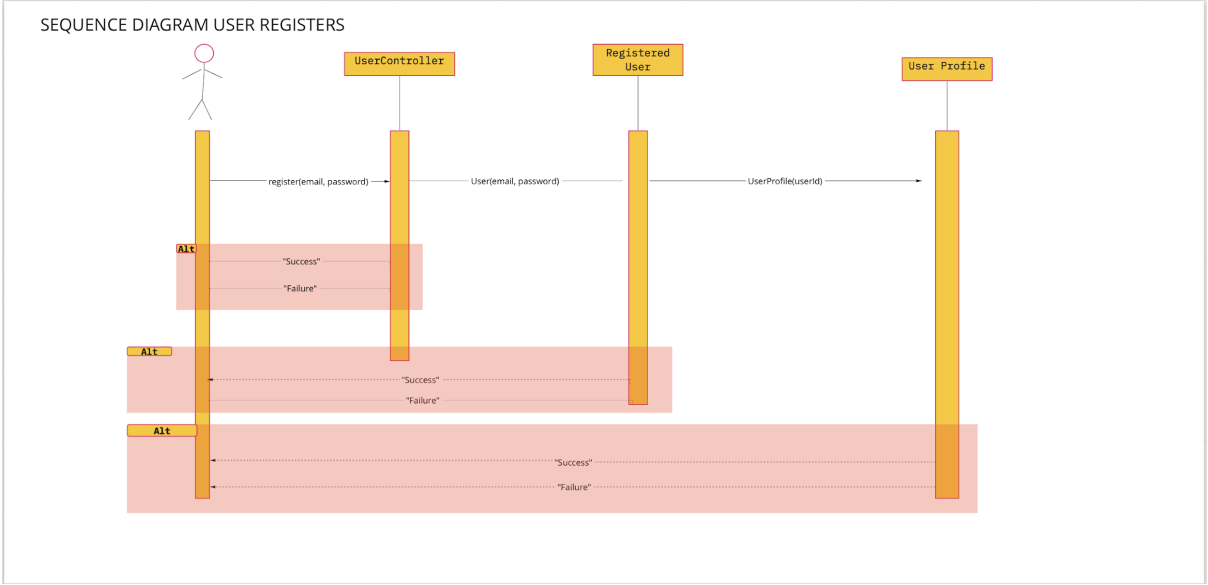
A user searches for a user



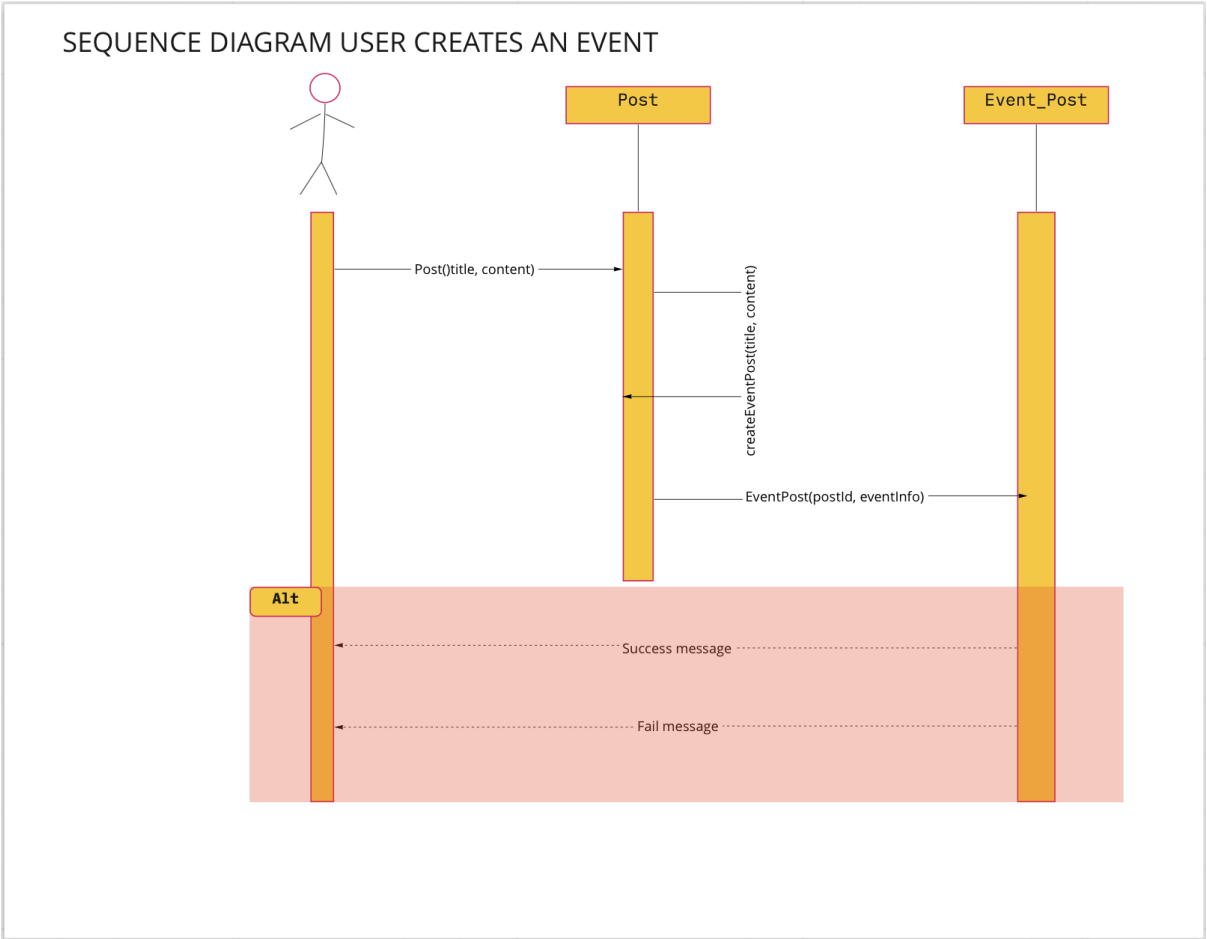
A user searches for an equipment



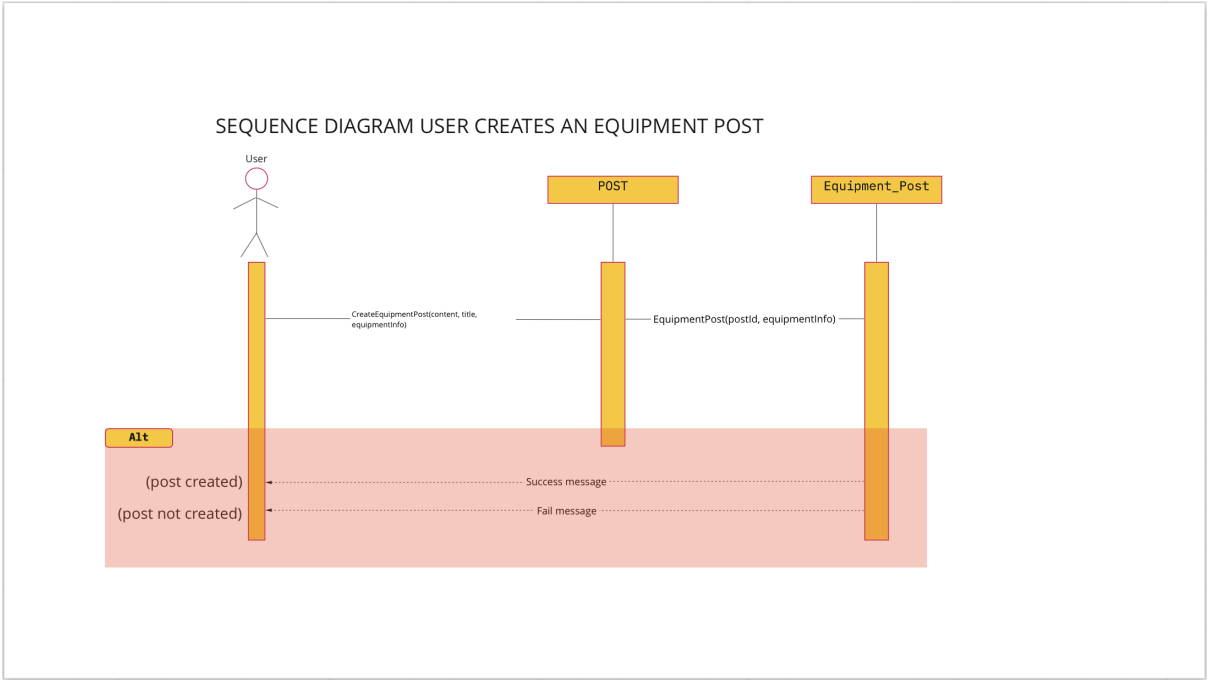
A user registers



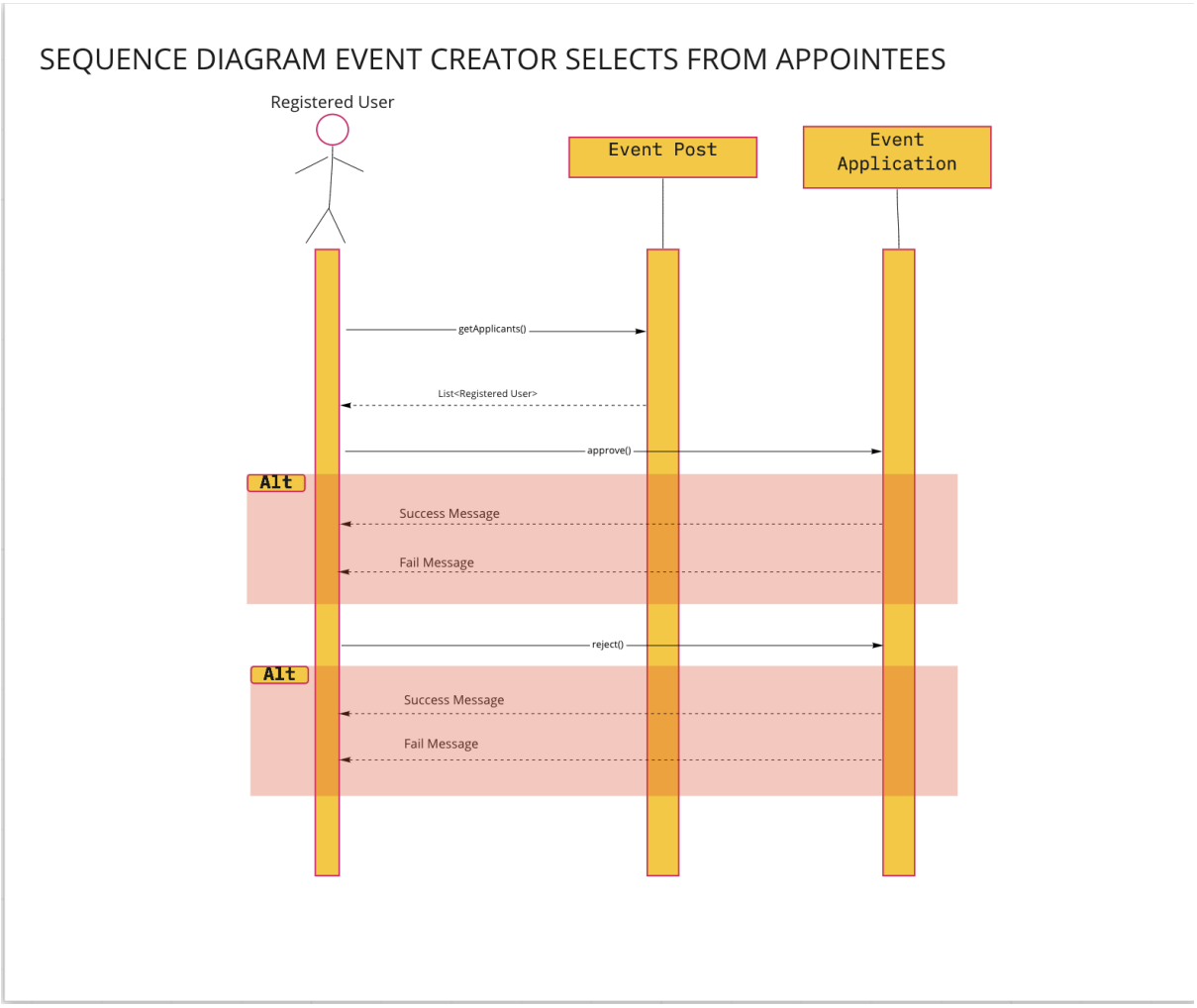
A user creates an event



A user creates an equipment post



An event creator selects from appointees



Scenarios and Mockups

1. Creating an Event

Persona
Eren Yiğit
22 years old
Computer Engineering student at Bogazici University
He is in summer vacation
He spends all day with playing computer games
He is a little interested in football, he can’t play well.

Story
After spending a couple of weeks with online games, Eren realized that his lifestyle is very unhealthy. He never does sport, he is mentally depressed because of the attitudes of other gamers and he decided to give a break to online games. He wants to play football with his friends but he doesn't have enough friends to organize a football match. He decided to use our application to create a football event. He wants players at beginner level because they are not so good players.

Pre-conditions
Eren registered to the system.
Eren signed on to the system.
Eren created his profile and specified his skills and interests.

Goals
Eren wants to create a football event at a stadium in his city.

Actions
Eren login to our website.
He clicks on "Create event" because he and his friends want only four football players.
Eren specified sport type, date, hours and location of the football event.
Eren specified skills, count, age, gender and role of the players because they don't want many players or every type of player.
After some time, Eren has seen some notifications about applications to his events on the website.
Eren analysed applicants and approved them if he wants until the quota is full and the team are ready.

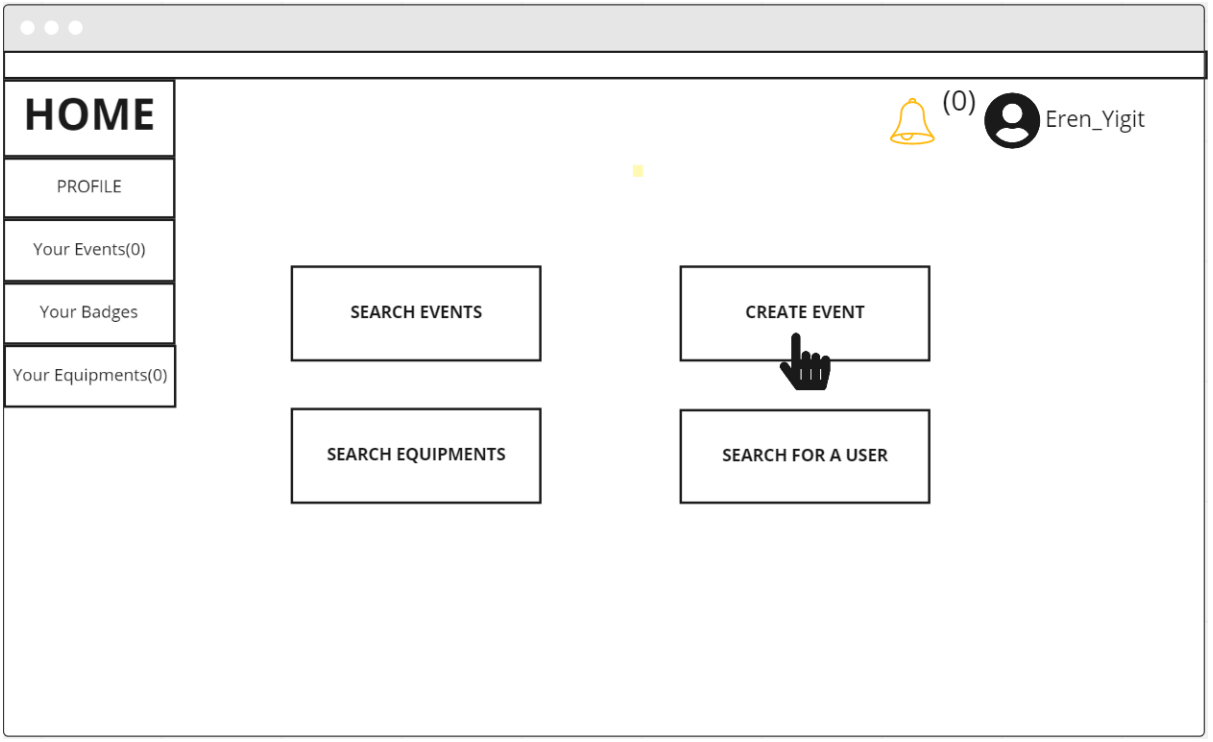
Acceptance Criteria

- (1.2.5.3.) System shall allow only event creators to create events.
- (1.2.5.4.) System shall allow event creators to specify needed number of player, location of event, event time, skills of participants, age of participants.
- (1.1.5.5.) Events with needed users registered shall be signed as full, so other people can filter them.

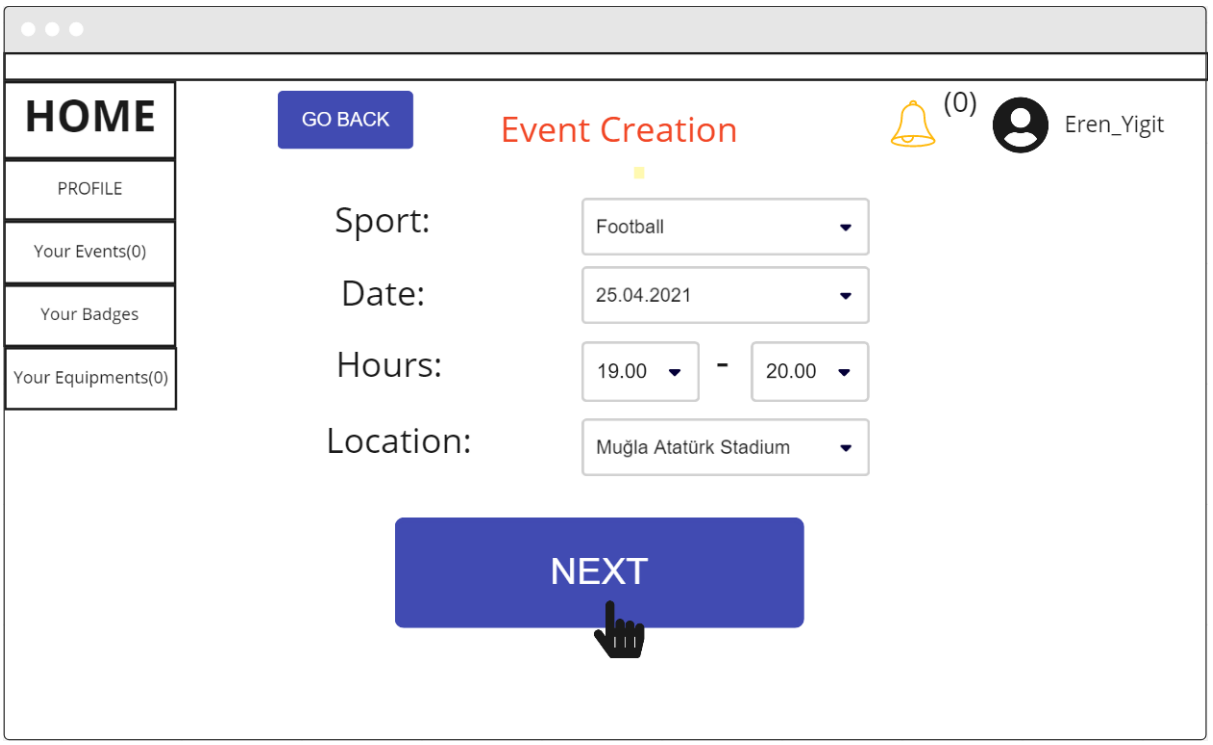
Scenario with Mockups

Web

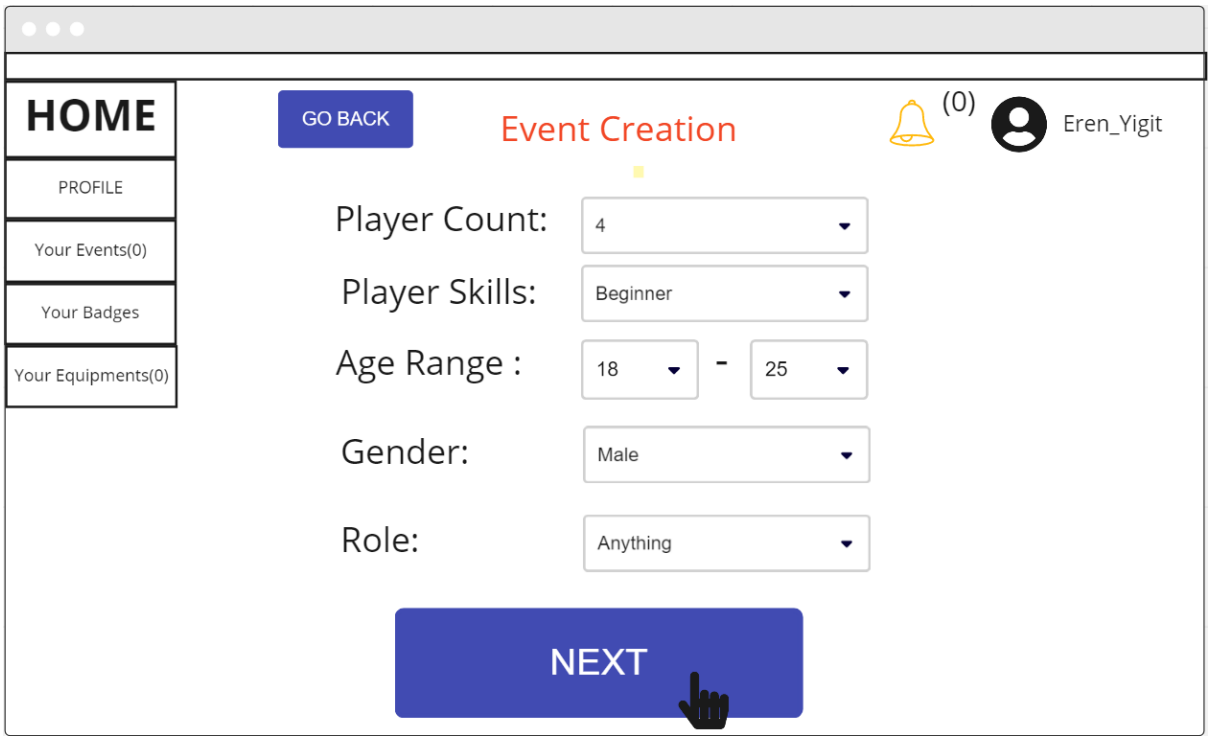
1. Eren clicked on the create event button.



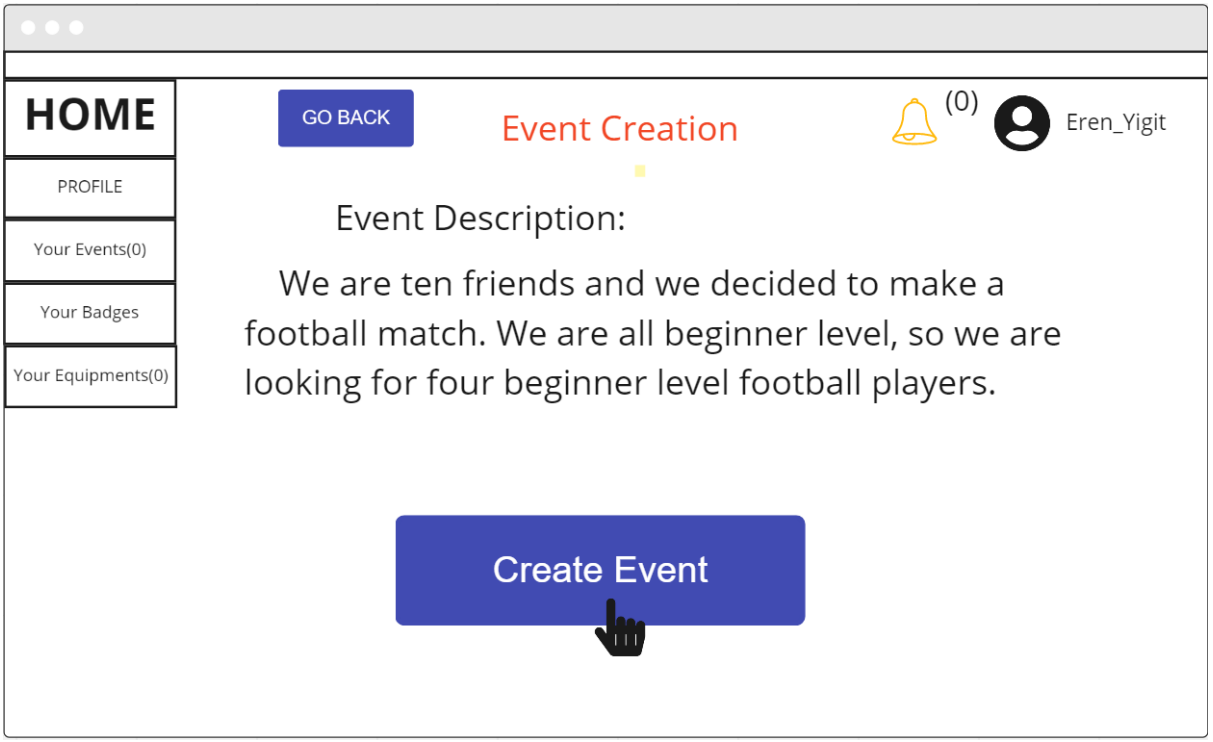
2. Eren specified sport type, date, hours and location.



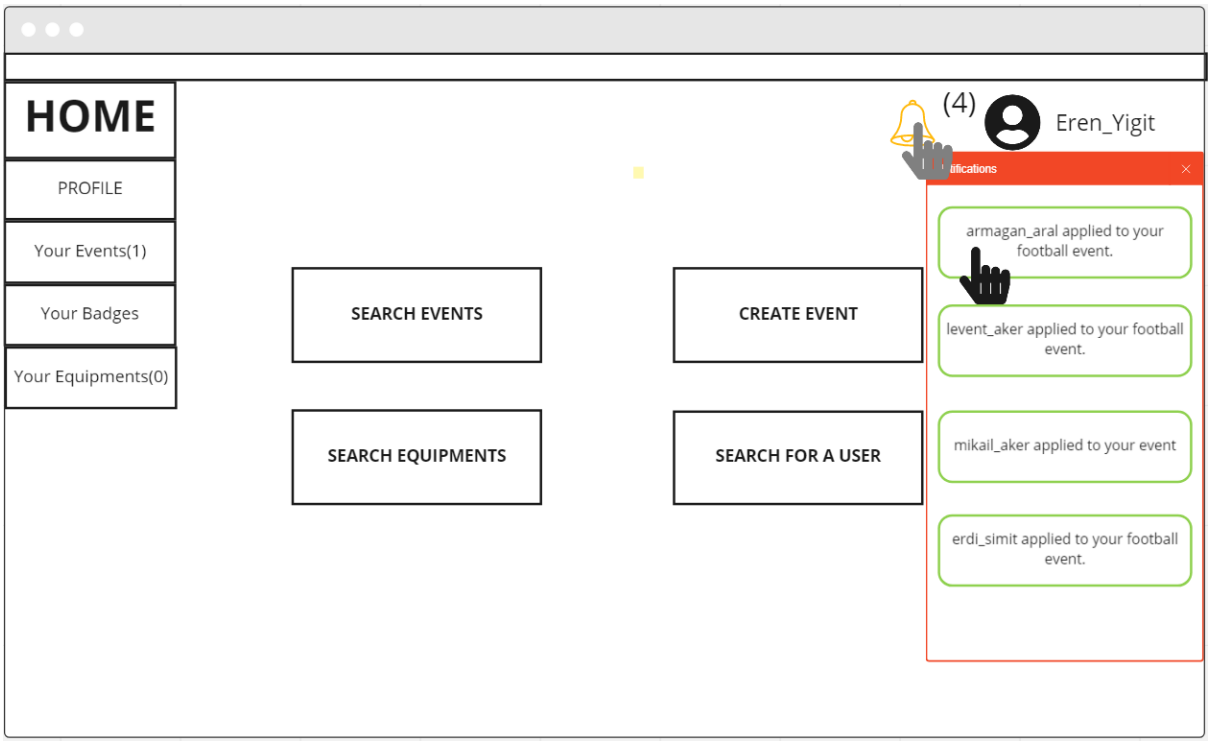
3. Eren specified skills, count, age, gender and role of the players.



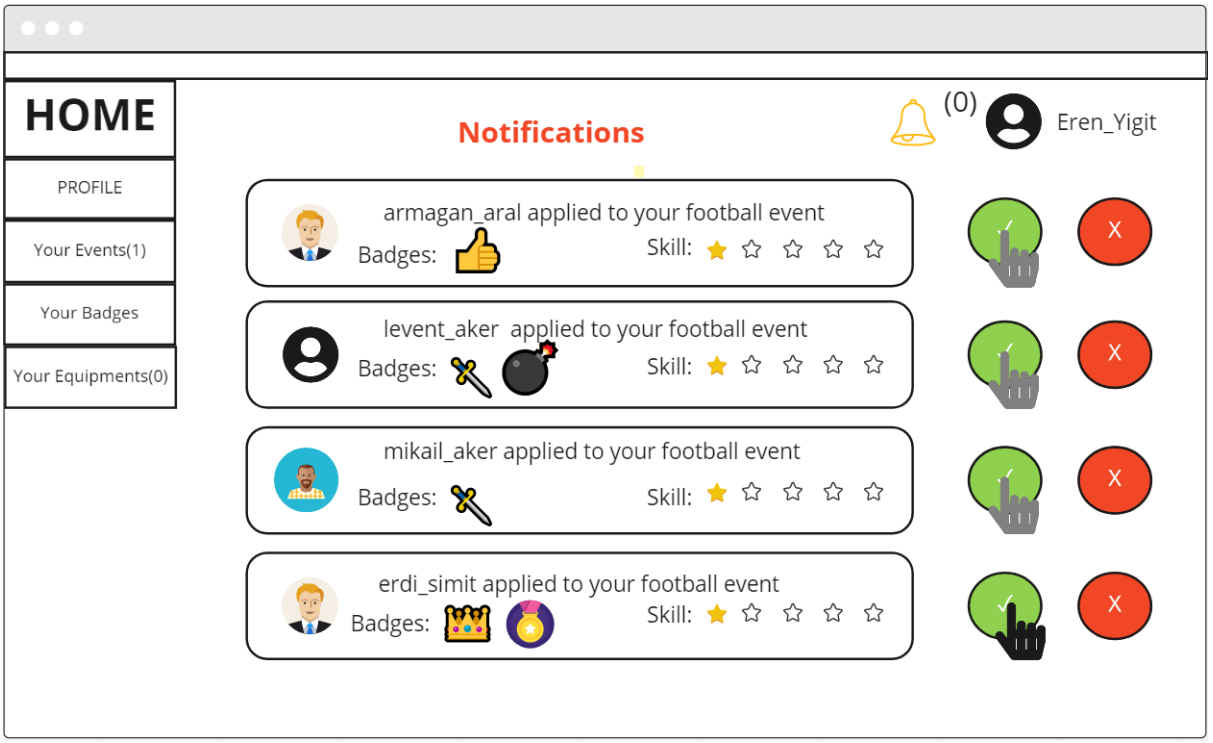
4. Eren described the event and made some comments about it.



5. After some time, Eren received some notifications and checked them.

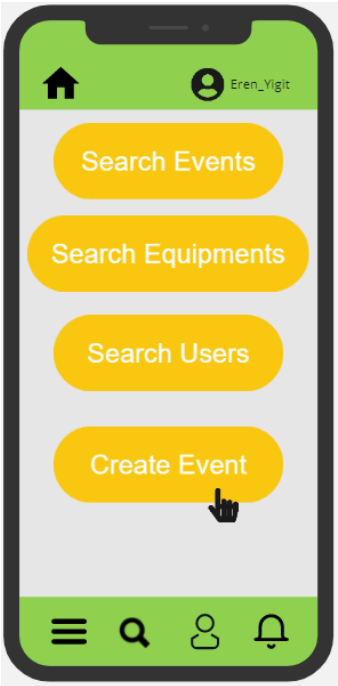


6. Eren analysed applicants and approved them if he wanted.

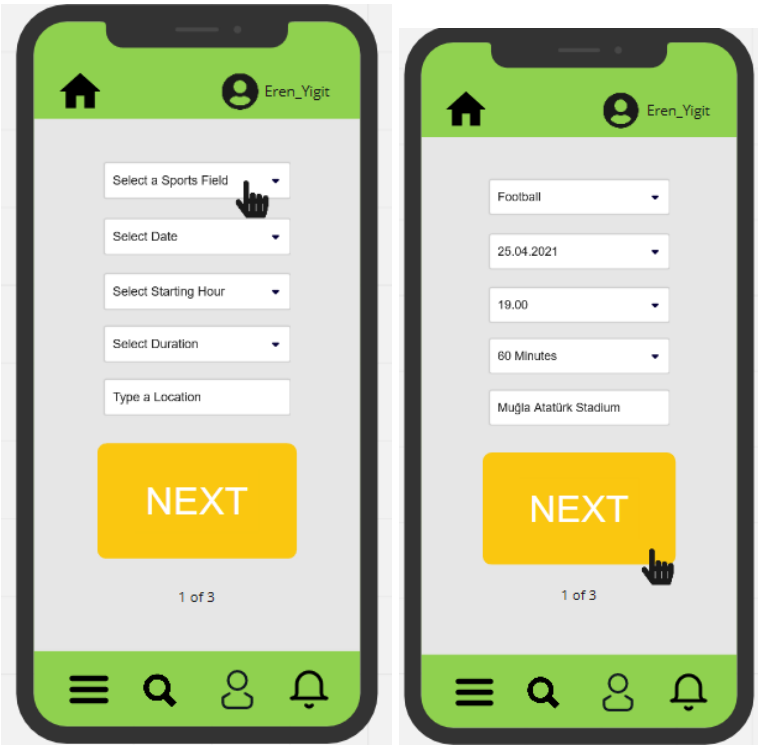


Android App

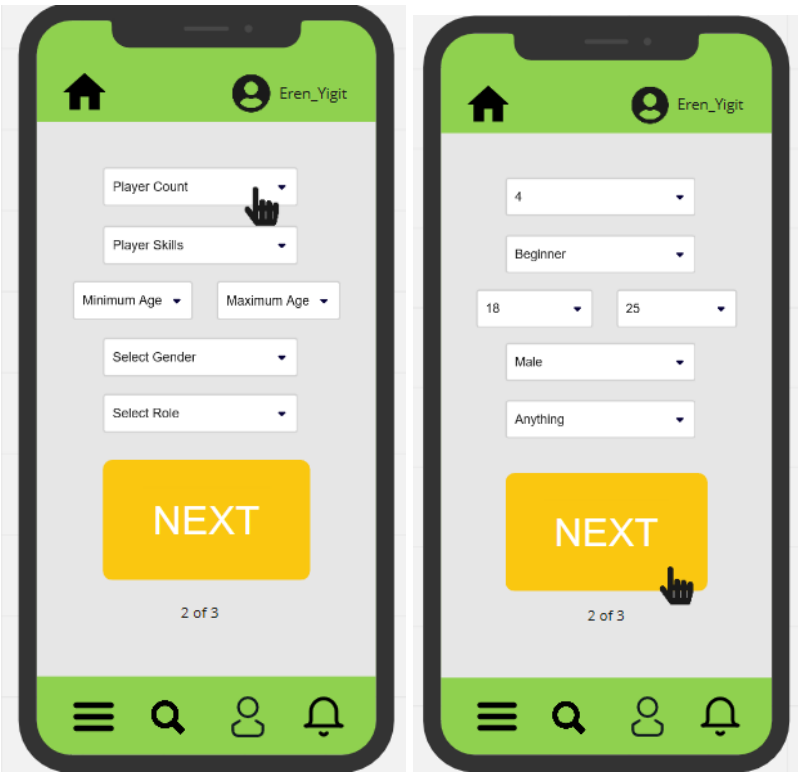
1. Eren clicked on create event button



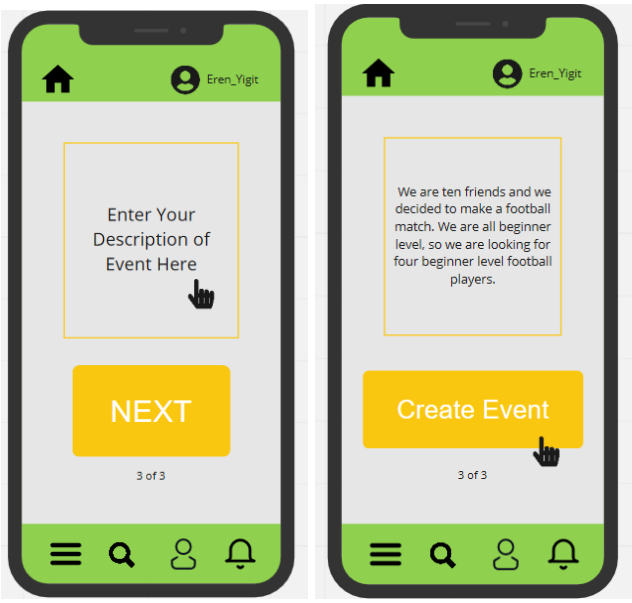
2. Eren specified sport type, date, hours and location



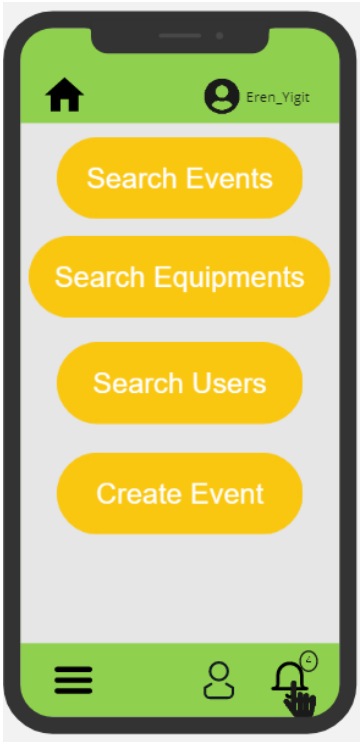
3. Eren specified skills, count, age, gender and role of the players.



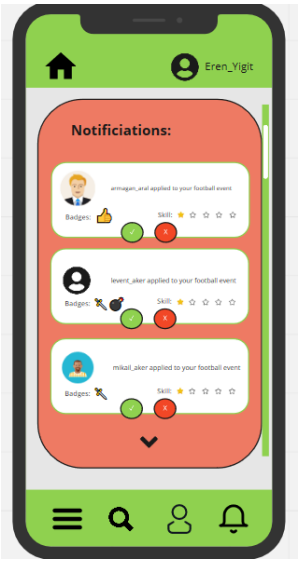
4. Eren described the event and made some comments about it.



5. After some time, Eren received some notifications and checked them.



6. Eren analysed applicants and approved them if he wanted.



2. User Searches for an Event

Persona:

Ali Topçu
40 years old
Father of two little boys
Used to work as an accountant, currently unemployed
Started gaining a little bit of weight
Galatasaray fan
Loves watching and playing football
His main role is defender

Story:
Currently unemployed and bored out of his mind, Ali wants to play football while he's still searching for a job. He thinks playing football also will help him lose some weight. Unfortunately he lost contact with his former football group, he wants to find people who love the sport as much as he does and play football with them.

Pre-conditions:
Ali is already registered as a user.
Ali has already stated his skills and interests in his profile.

Goals:

Ali wants to search for a football game he can play in.

Acceptance Criteria:

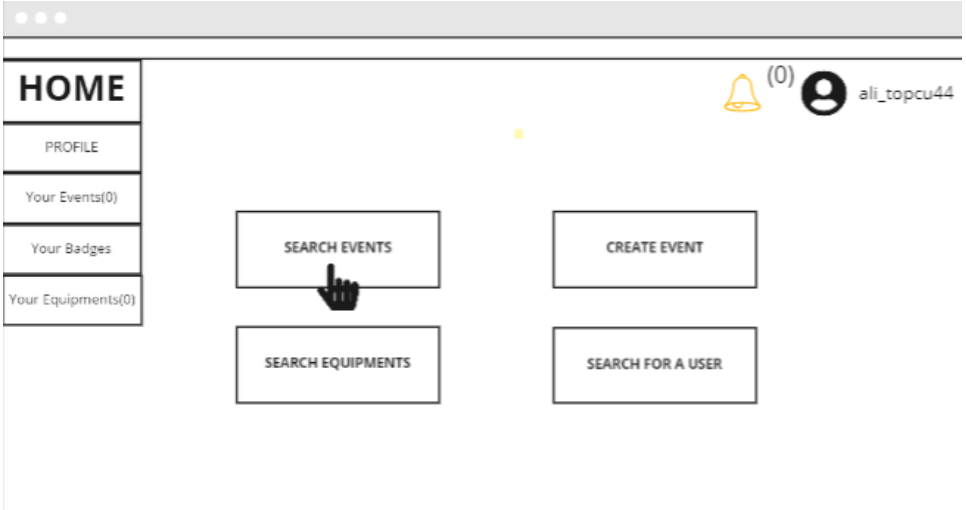
- (1.1.5.1.5) Users shall indicate the type of sport and location in order to filter events.
- (1.2.5.6.) All users shall be allowed to list all events on the events page.
- (1.2.5.7.) Event pages shall display the event creator, event location, sports field, applicants, participants and spectators of the event.
- (1.2.5.9.) System shall allow only registered users to apply to events.
- (1.2.6.1.1.) Events shall be filtered by user according to event type, appropriate age range, location, date, hours, skill levels of attendees and availability (full or player needed).
- (1.2.6.1.2.) Events shall be sorted with respect to name, time and location (e.g. from closer events to far away ones).

Scenario with Mockups:

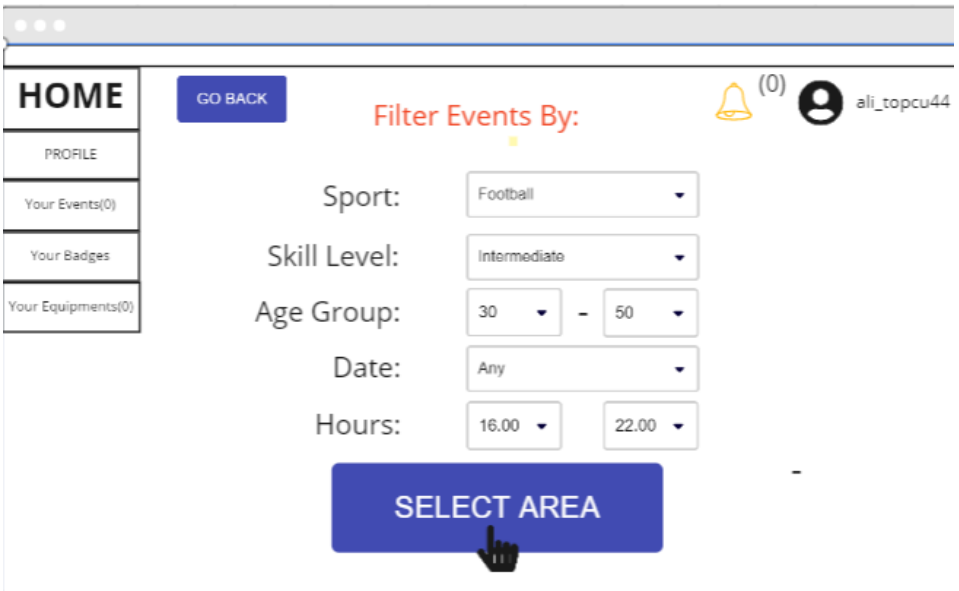
Ali wants to search for a game he can play in, therefore he clicks the "SEARCH EVENT" button.
Ali uses the filtering feature to find a game with the properties he wants.
After selecting the properties, Ali clicks "SELECT AREA" to define the area he wants to search.
He selects a circular area on the map and then clicks "SEARCH EVENTS".
After clicking "SEARCH EVENTS" he can browse the events with the properties he wants.

Web Mockup:

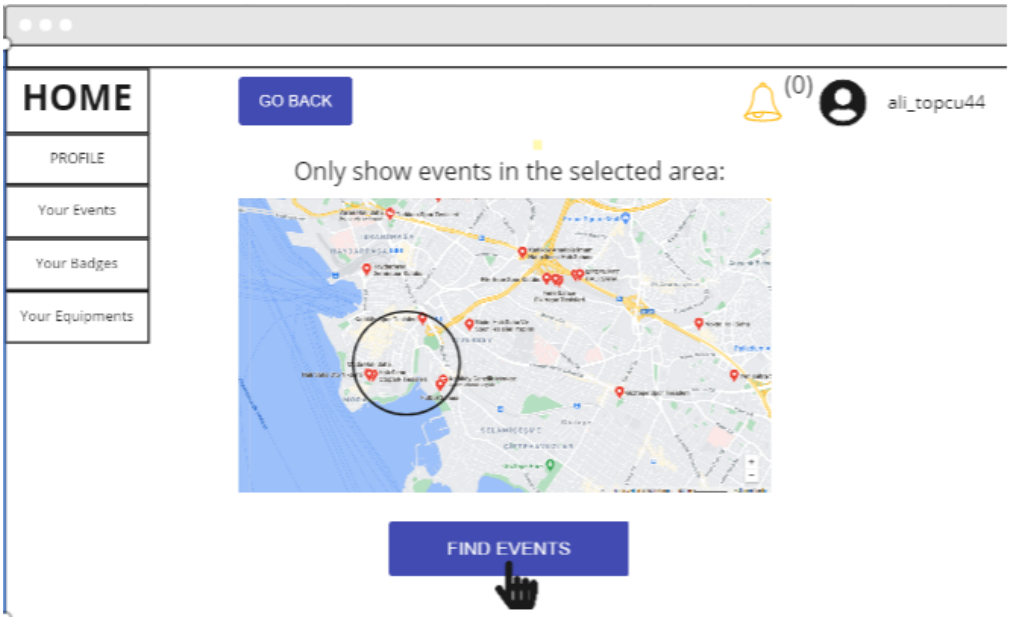
1. Home Page



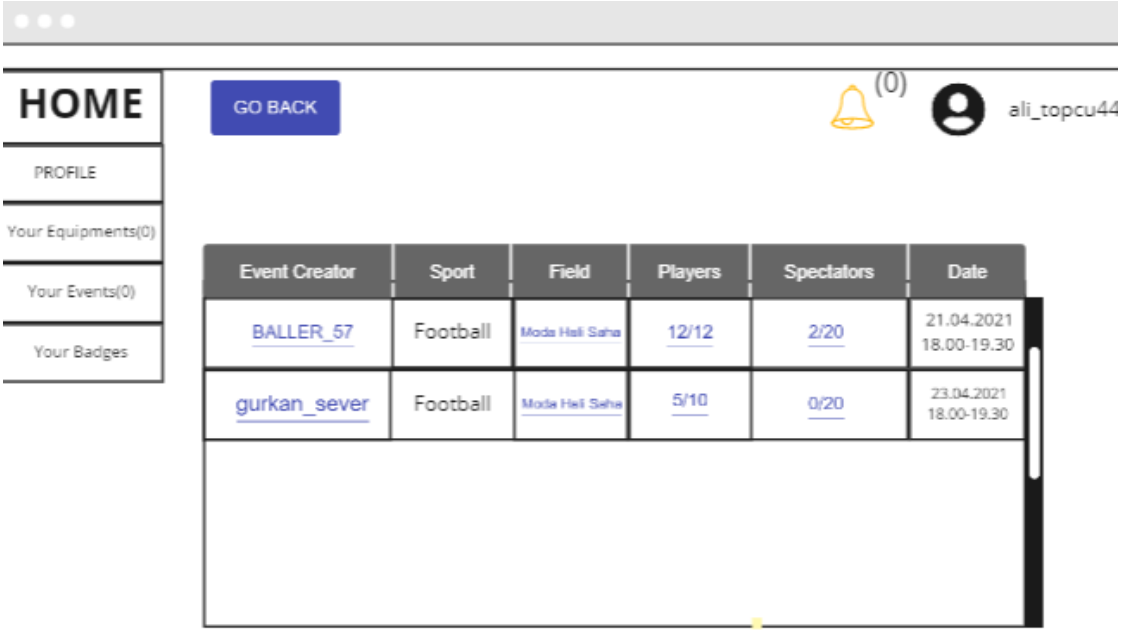
2. Filtering



3. Selecting the Area

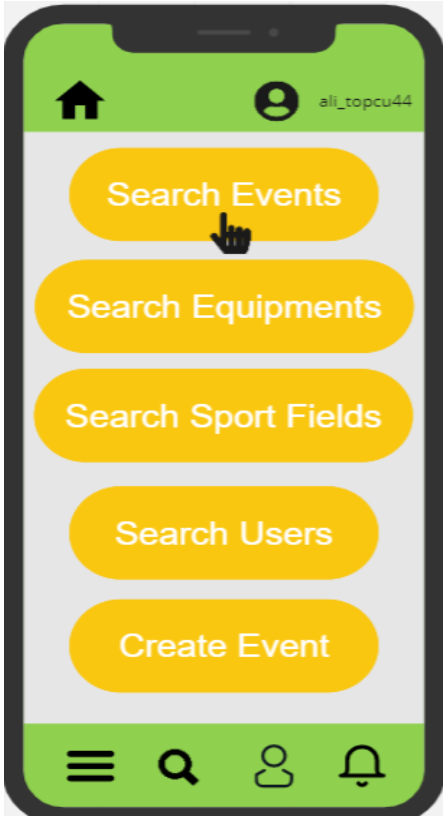


4. Search Results

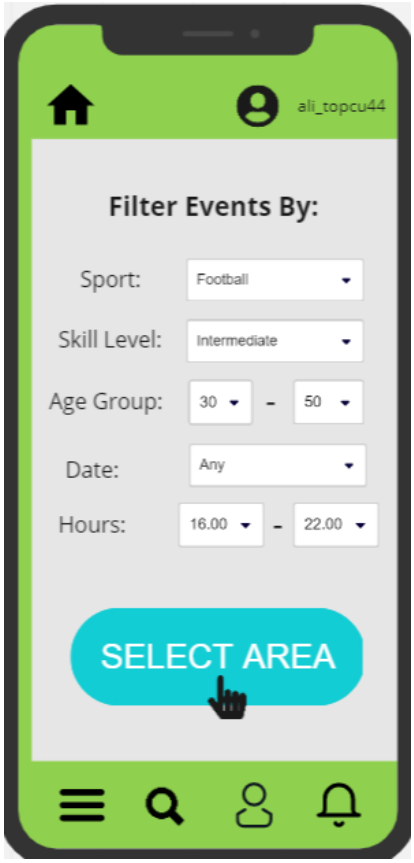


Android Mockup:

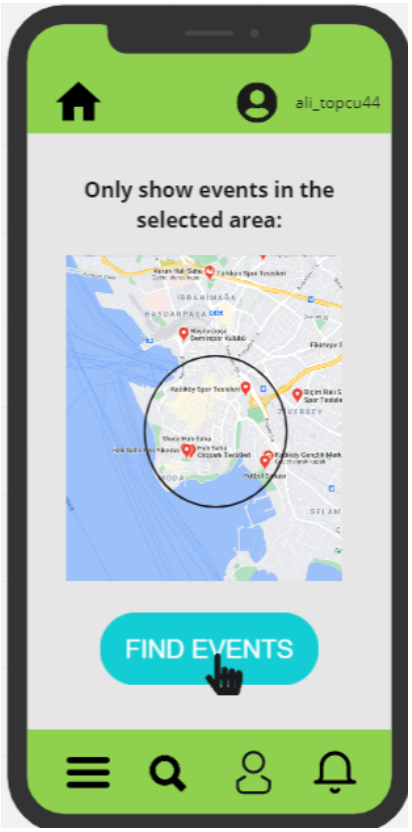
1. Home Page



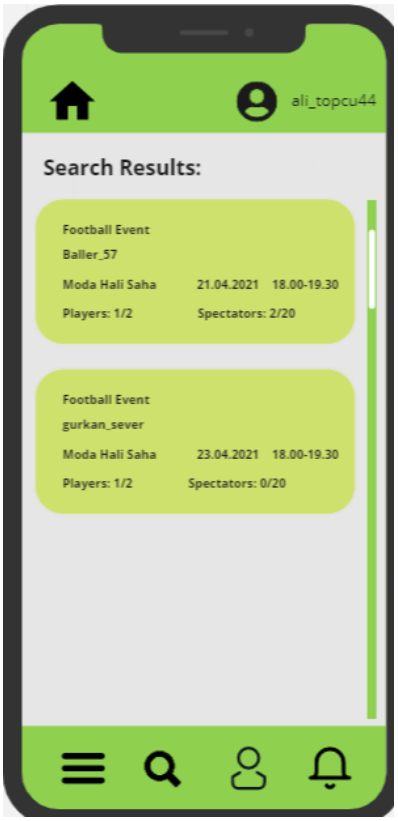
2. Filtering



3. Selecting the Area



4. Search Results



3. Player & Spectator Application

Persona:

- Melike Taşkıran
- 26 years old
- Music lover
- Has one cat and one dog
- Extrovert

Story:

Melike is an assistant brand manager at a leading company. Working remotely for more than one year, she feels unmotivated and bored. She wants to make some changes in her life by playing sports. Since her job allows her to work with a flexible schedule, Melike wants to play sports during the daytime. However she can't play with any of her friends as they are working at that time. She searches for a website that allows matchmaking with other interested people and encounters our platform. She decides to try our platform to make some new friends to play sports with. She wants to minimize the risk of being infected by avoiding team sports. So, she searches for tennis events nearby. Since she is a tennis newbie, she also wants to watch some tennis matches played by advanced players.

Pre-conditions:

- Melike is already registered as a user.
- Melike has already stated her skills and interests in her profile.
- Melike has already logged in to the system and searched for tennis events.

Goals:

- Melike wants to apply to an event to be a player.
- Melike wants to apply to an event to be a spectator.

Acceptance Criteria:

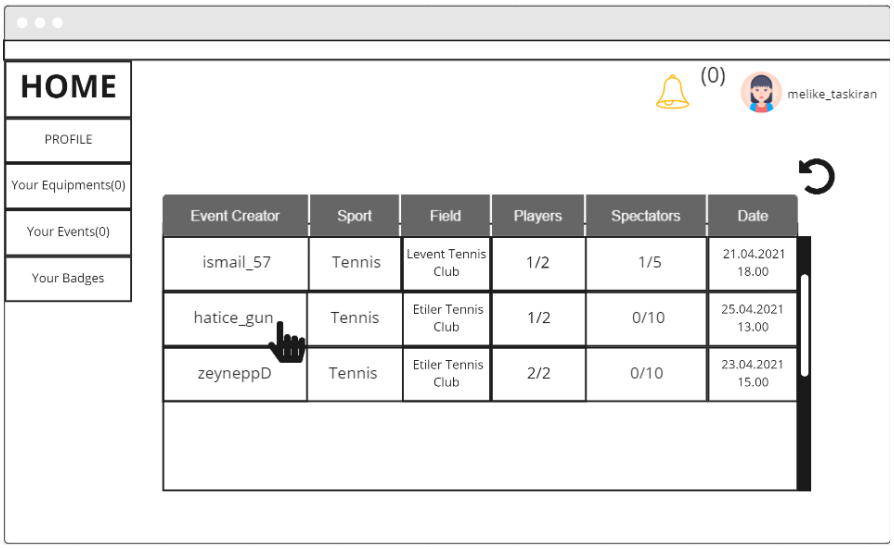
- (1.1.5.4.) All users shall be able to see the event pages that includes event creator, location, participants and badges.
- (1.1.5.5.) All users shall be able to see the profile pages of the event participants.
- (1.1.5.6.) Registered users shall be able to apply to join an event.
- (1.1.7.4.) All participants shall be informed whether he/she is selected.

- (1.2.4.3.) User profile shall display personal information, skills, badges, followers, followings and post history.
- (1.2.10.1.) System shall notify users when their player/spectator applications are approved/rejected.
- (2.4.3.) Users shall be able to choose which of their personal data to be seen publicly.

Scenario with Mockups:

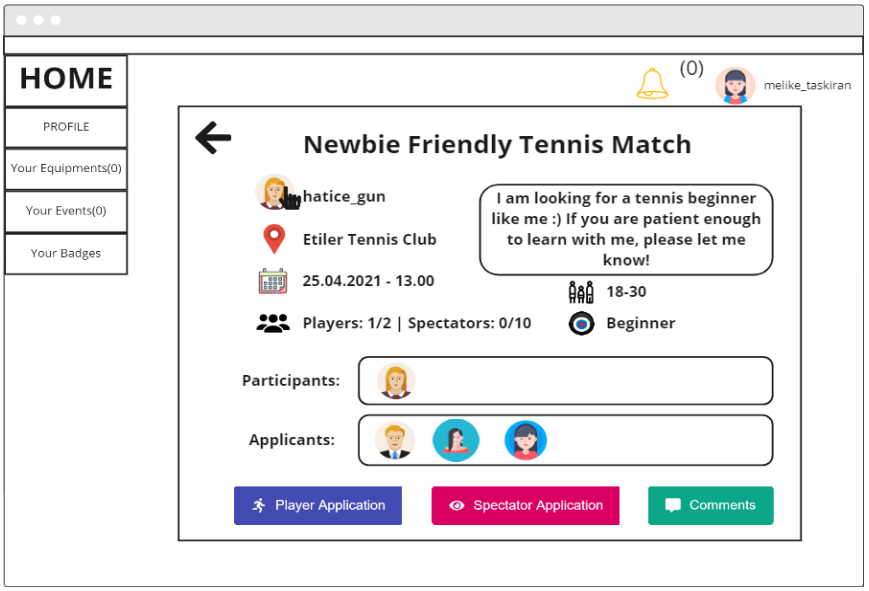
Web Application:

She has already searched for the events she is interested in and she is in the search results page. She likes the date and the location of the second event, so she clicks on the event to get to the event page.

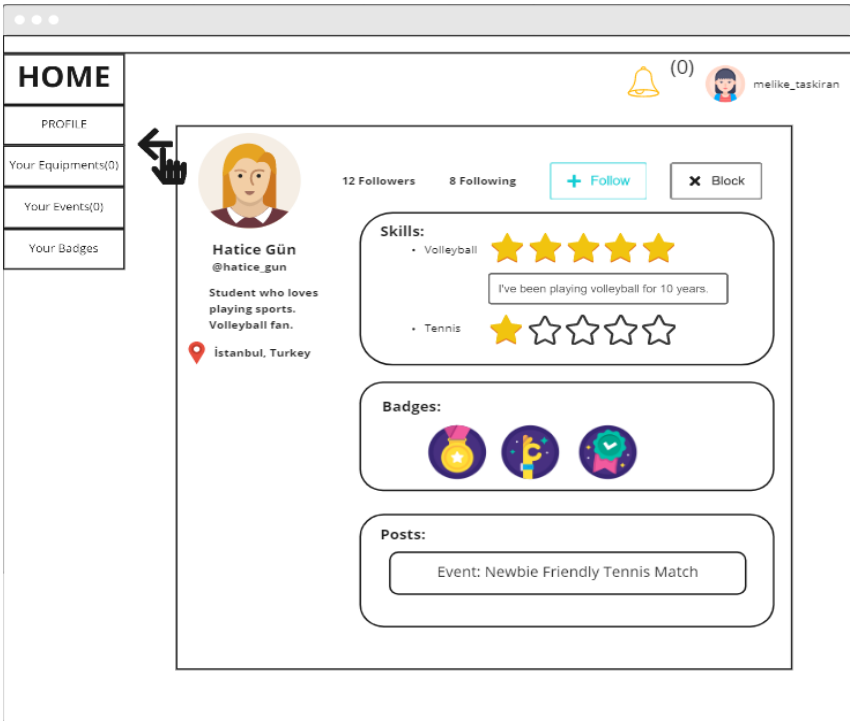


She is happy that the event creator is looking for a beginner player like her. But, she wants to learn more about the event creator before she applies, so she clicks on the

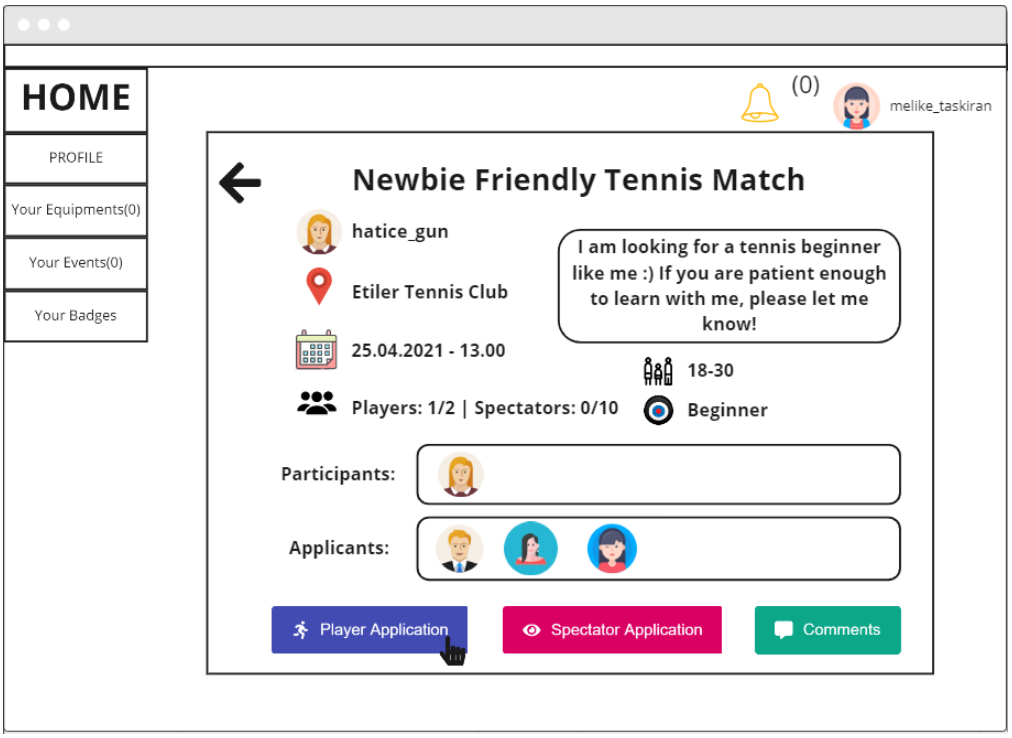
icon of the event creator to get to her profile page.



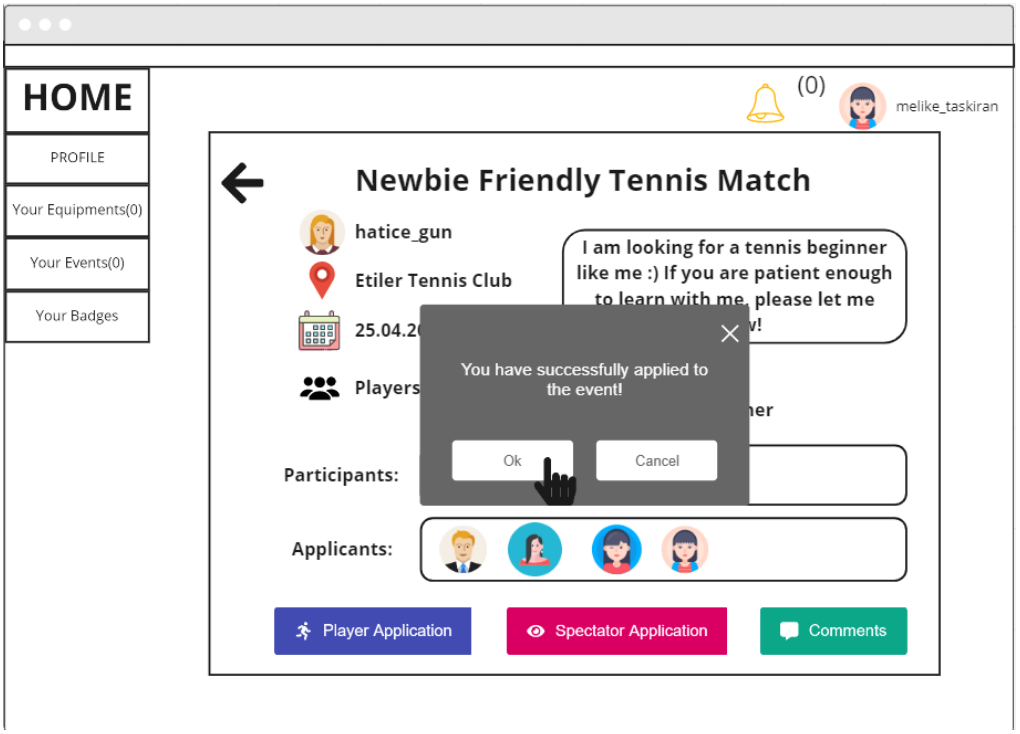
She notices that the event creator decided to hide their age so she can't get an idea from there. However, she also notices that the event creator has some good badges, therefore she thinks that it would be a good idea to apply to her event. She gets back to the event page by clicking the go back button.



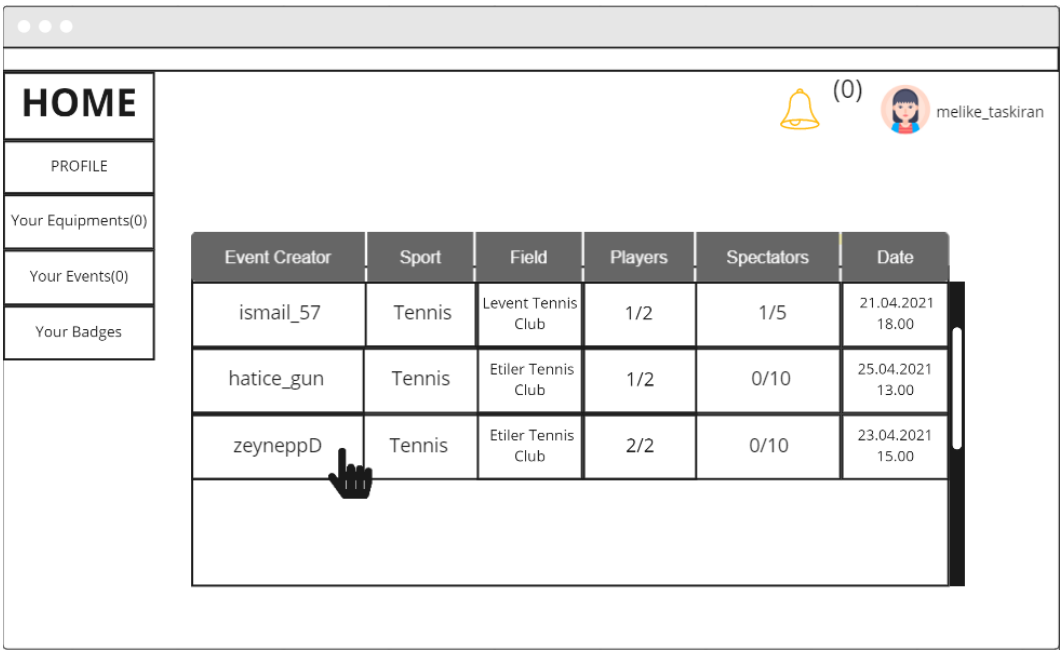
Now that she is in the event page and she is interested, she clicks on the player application button to apply.



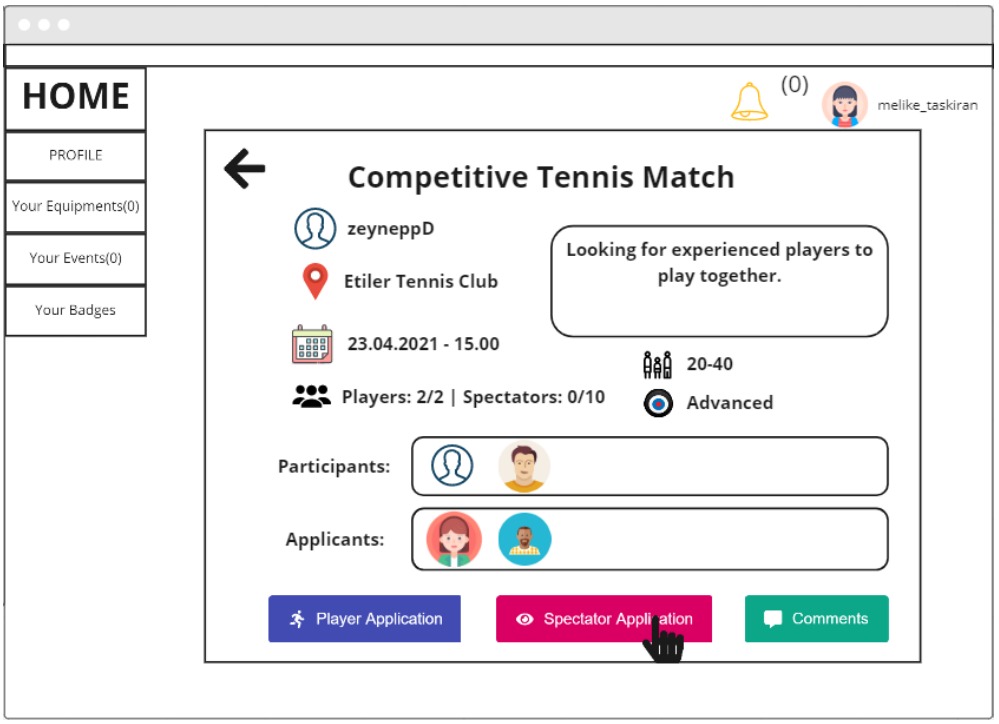
A popup page shows up to inform that she successfully applied to be a player. Also, she is added to the applicants list.



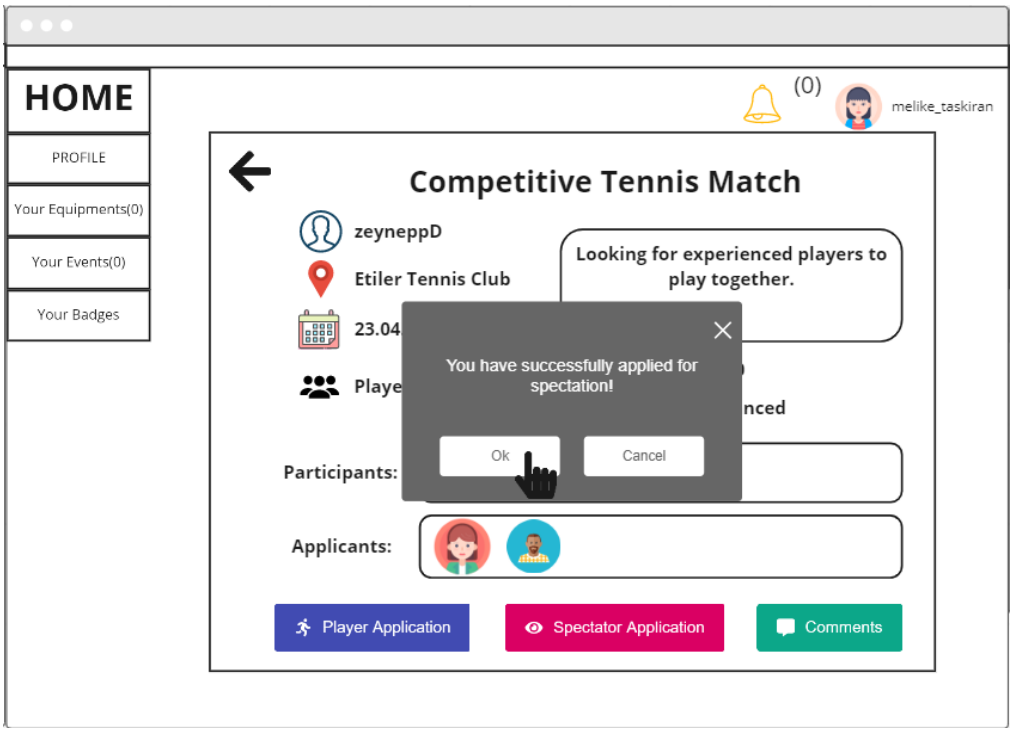
Now, Melike wants to apply to an event to be a spectator and she gets back to the search results. The last event seems reasonable to spectate as it is full. Melike clicks on the event.



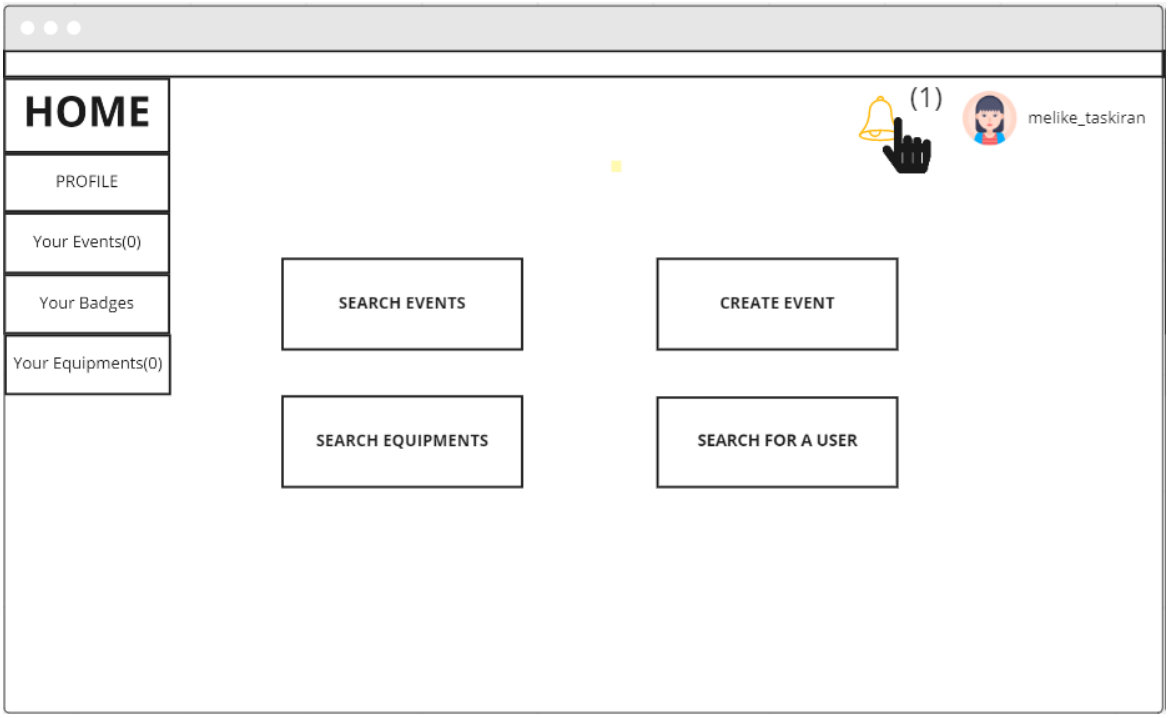
Melike reads the description and sees that the event creator is looking for advanced players. So, she thinks this match could be very useful to observe and clicks the spectator application.



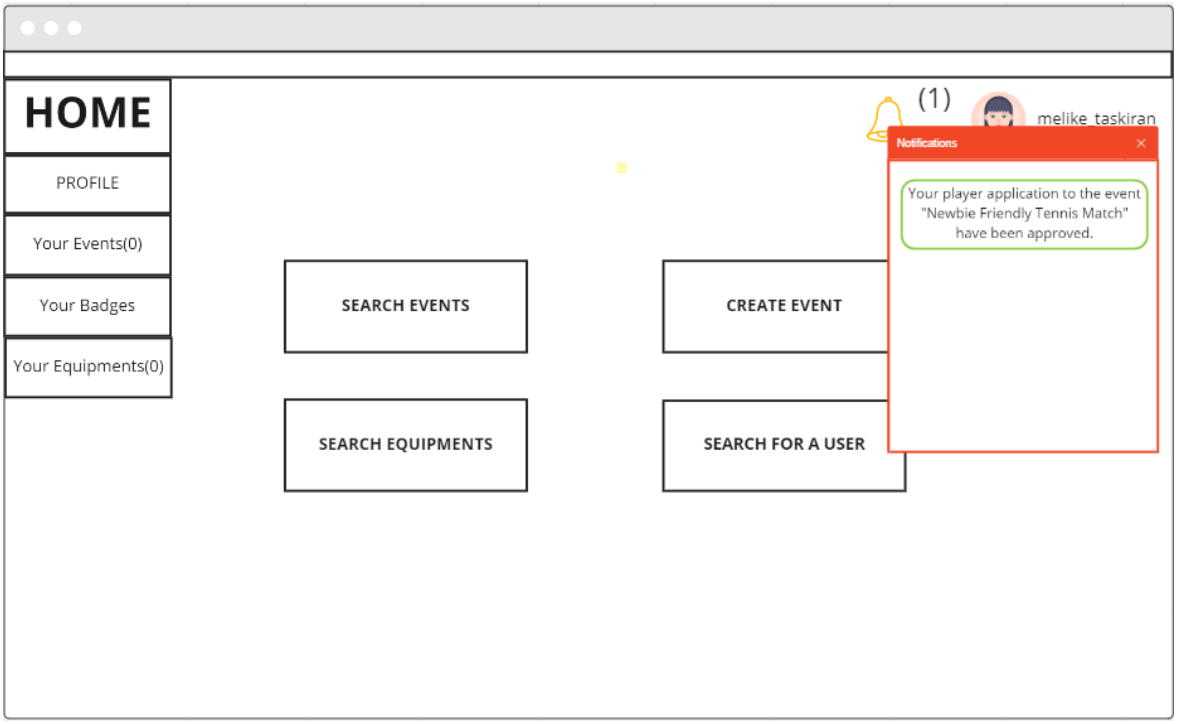
A popup page shows up to inform that she successfully applied to be a spectator.

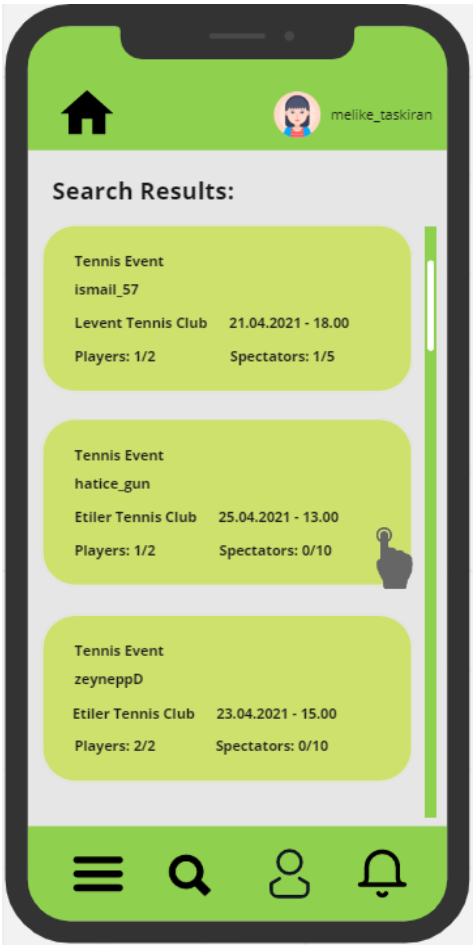


Some time later, she enters the app again and notices that she received a notification. So, she clicks the notification button.



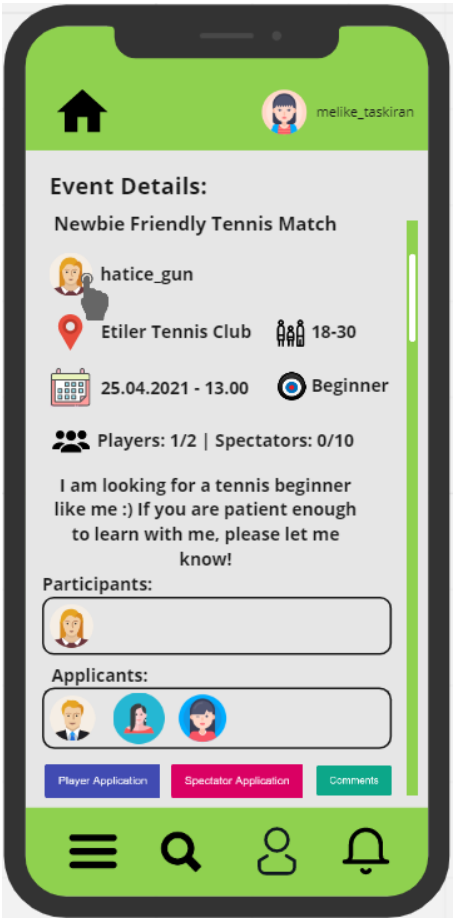
Seeing her player application approved, Melike is now waiting for her tennis match!



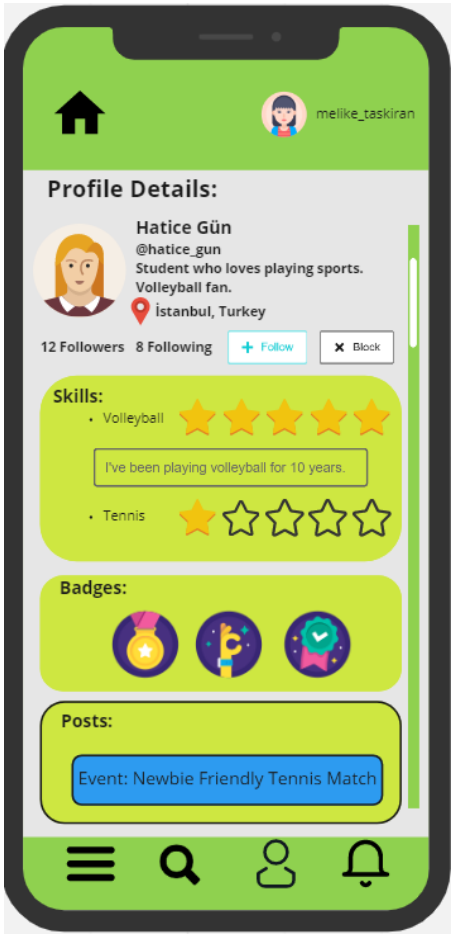


Android Application:

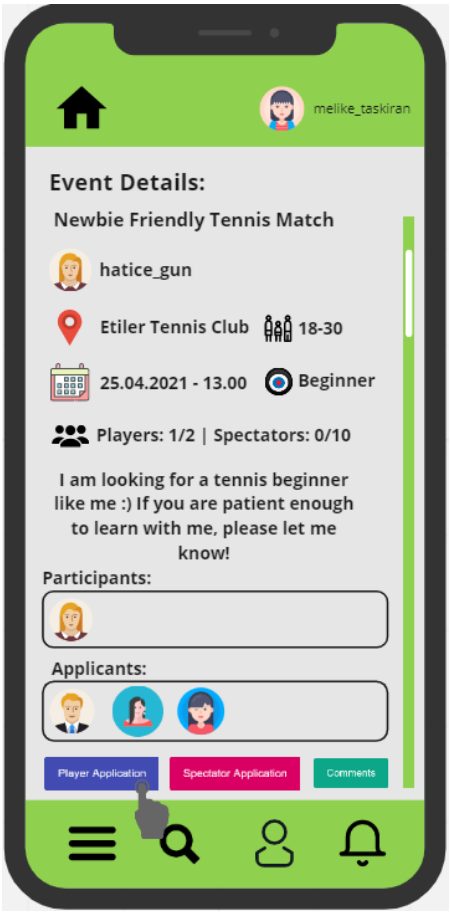
She has already searched for the events she is interested in and she is in the search results page. She likes the date and the location of the second event, so she clicks on the event to get to the event page.



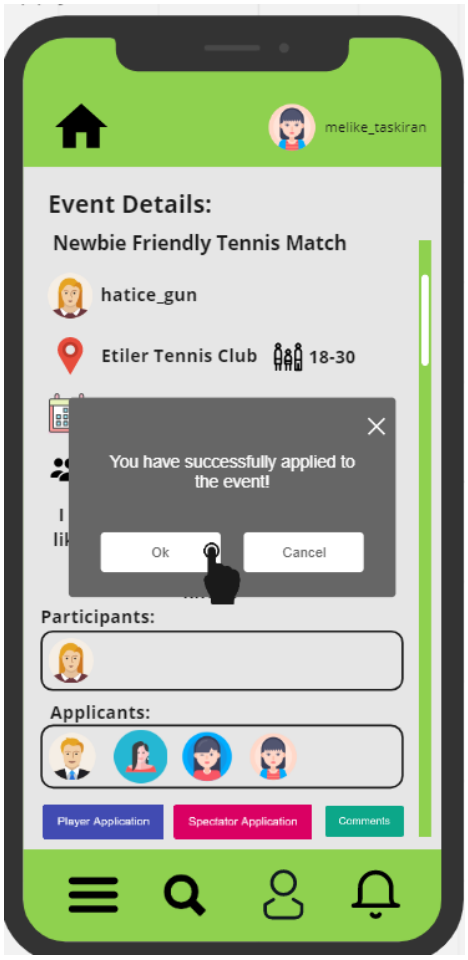
She is happy that the event creator is looking for a beginner player like her. But, she wants to learn more about the event creator before she applies, so she clicks on the icon of the event creator to get to her profile page.



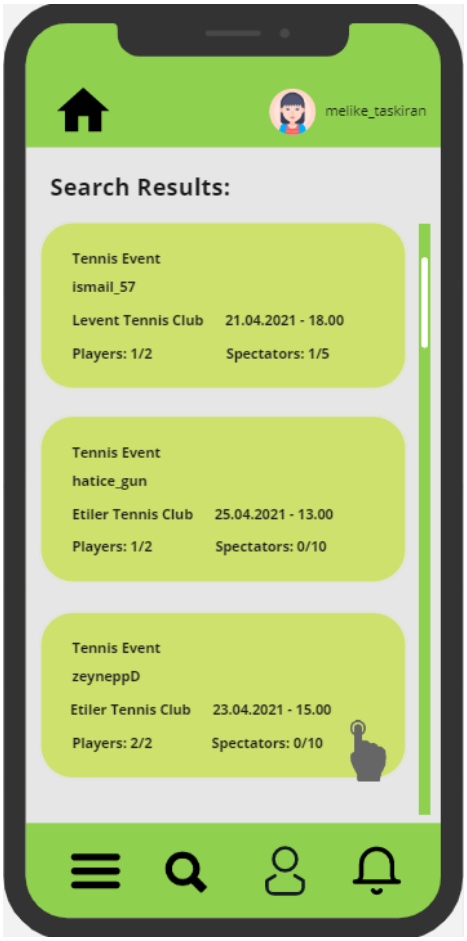
She notices that the event creator has some good badges, therefore she thinks that it would be a good idea to apply to her event. She gets back to the event page.



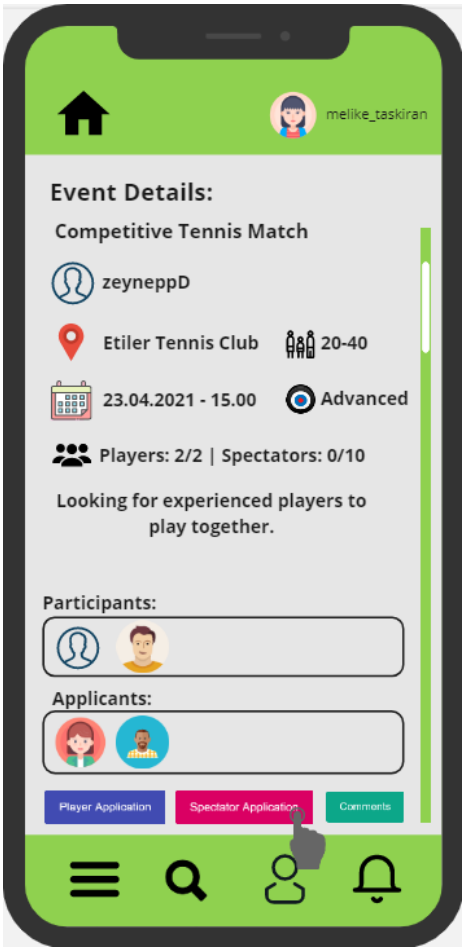
Now that she is in the event page and she is interested, she clicks on the player application button to apply.



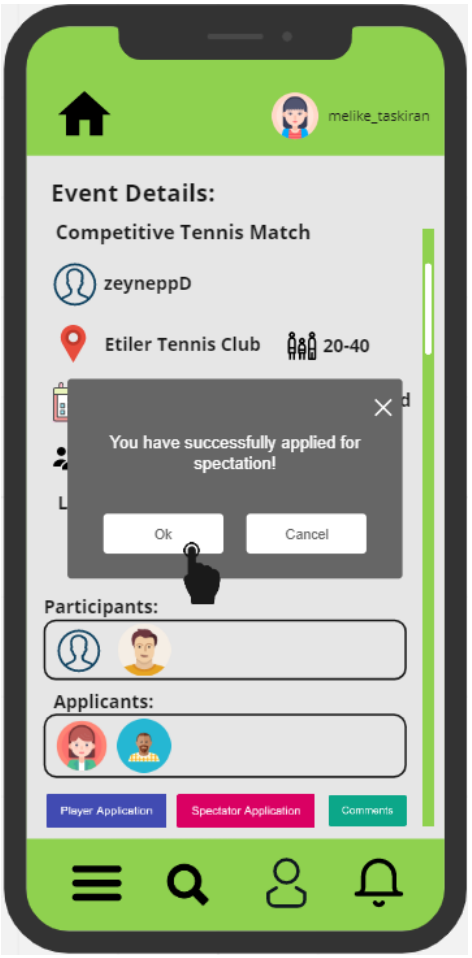
A popup page shows up to inform that she successfully applied to be a player. Also, she is added to the applicants list.



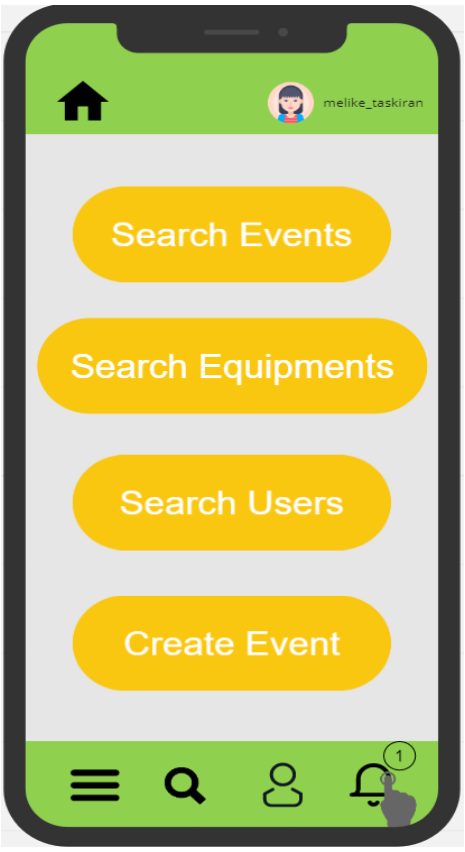
Now, Melike wants to apply to an event to be a spectator and she gets back to the search results. The last event seems reasonable to spectate as it is full. Melike clicks on the event.



Melike reads the description and sees that the event creator is looking for advanced players. So, she thinks this match could be very useful to observe and clicks the spectator application.



A popup page shows up to inform that she successfully applied to be a spectator.



Sometime later, she enters the app again and notices that she received a notification. So, she clicks the notification button.

Seeing her player application approved, Melike is now waiting for her tennis match!

4. Event Creator Selecting from Applicants

Persona

Fatih Kerim

33 years old

Grocery store owner

Loves watching amateur football games

Has a limp on left leg

Avid player of PES and NBA2K console games

Addicted to sports betting

Lives with his mother

Story

Fatih loves watching football and basketball games, both professional and amateur. He was also an amateur football player in his youth, aspiring to move on to higher professional leagues. However, his career ended due to a permanent injury on his left leg caused by a construction machinery accident he had while he was working as a construction worker to support his family. Since then, he cannot play football but loves organizing amateur matches and watching them. He now dreams of being a coach. He hears of our platform from a friend who tells him it might be of interest to him. Without much reading, he registers to the platform and creates a 6v6 football match event at a halı saha. After a day of waiting, he now has enough applications to his event to review and select 12 people.

Pre-conditions

Fatih has registered as a user.
Fatih has signed in to the platform.
Fatih has created his profile, filled in the relevant fields (skills and interests)
Fatih has created an event by specifying relevant properties (sports type, date and time, location, skill level, age, gender, role and number of players)
Fatih has received enough applications to fill the spots in his event

Goals

Fatih wants to create 2 teams of 6 by reviewing the applications he has received and approving 12 of them that he sees fit

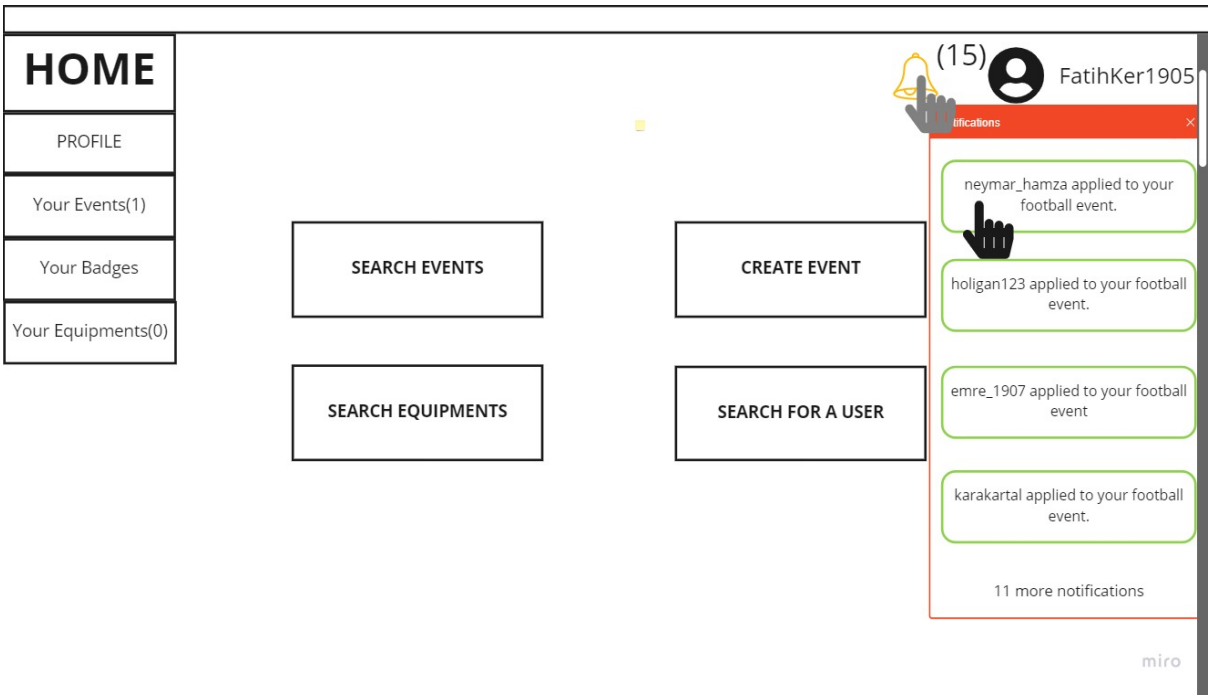
Acceptance Criteria

- (1.1.6.2.) Event creators shall select players among the applicants.
- (1.2.5.8.) System shall allow all users to view the profile pages of the event creator, any of the applicants, participants and the spectators.
- (1.2.13.2.) System shall allow users to see the analytics for players, event creators and sports fields.

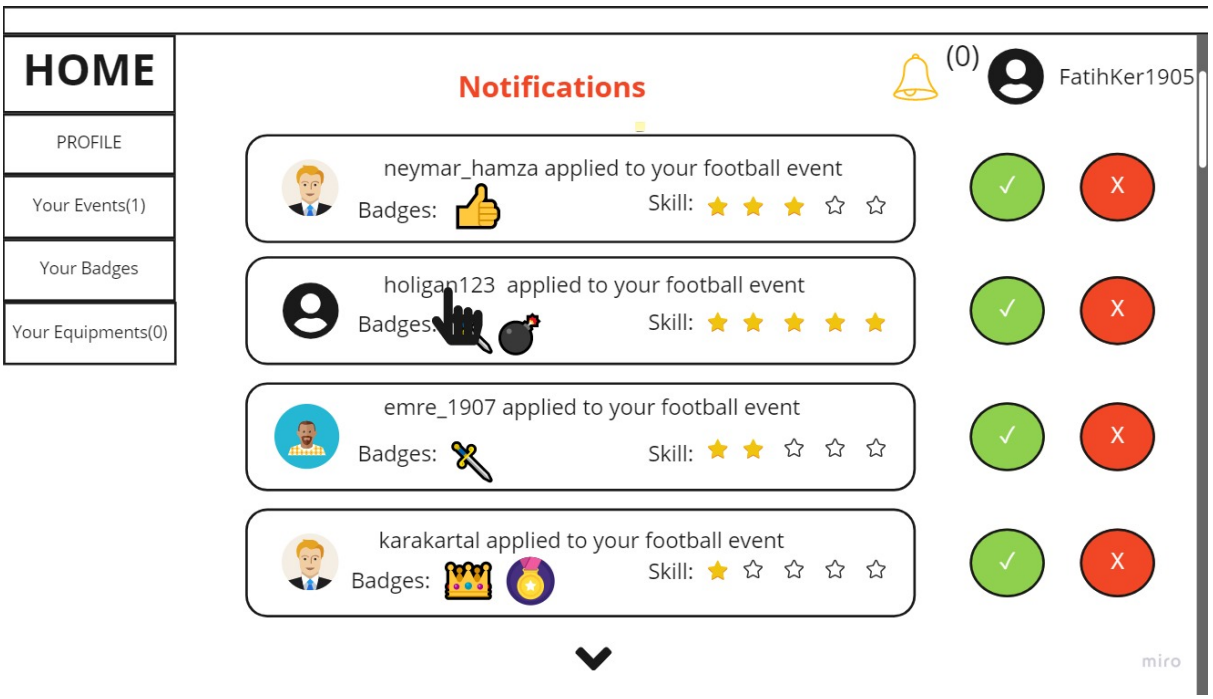
Scenario with Mockups

Web

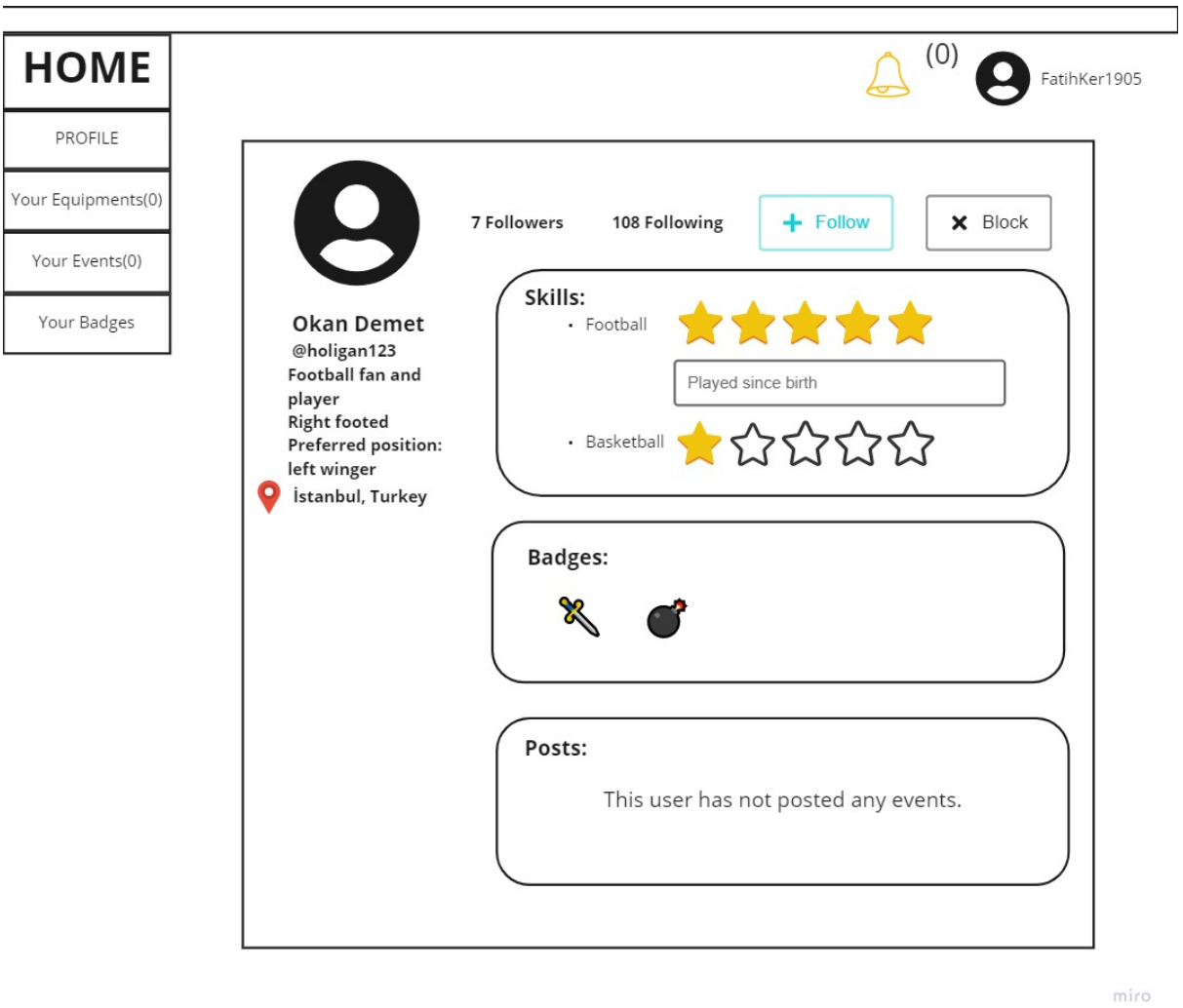
1. Checking application notification from homepage



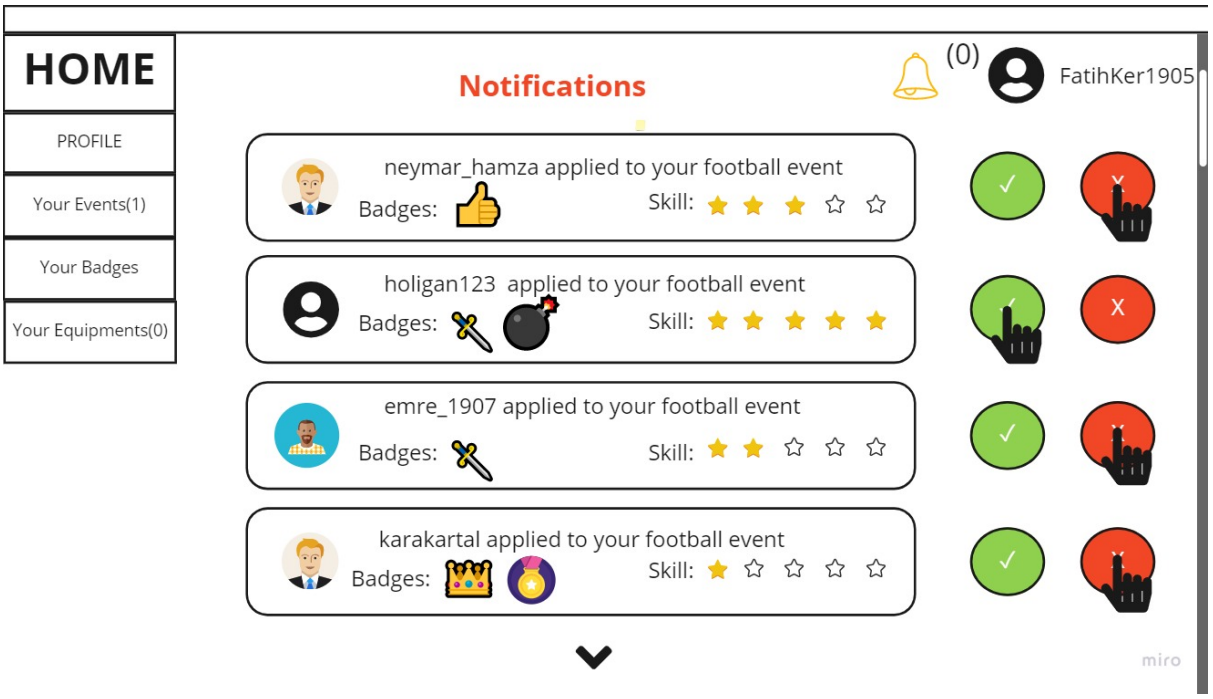
2. Seeing applicants from notifications page and clicking on applicant profile



3. Checking the profile page of applicant

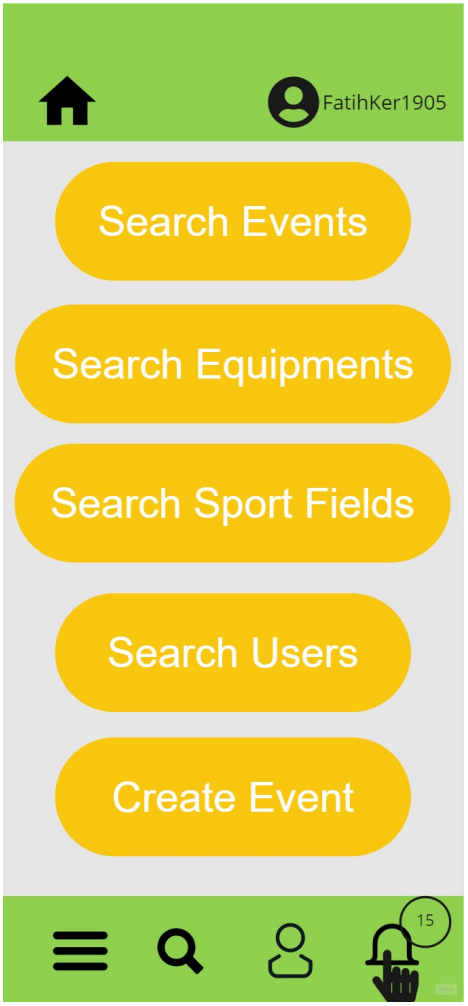


4. Accepting or declining applicants

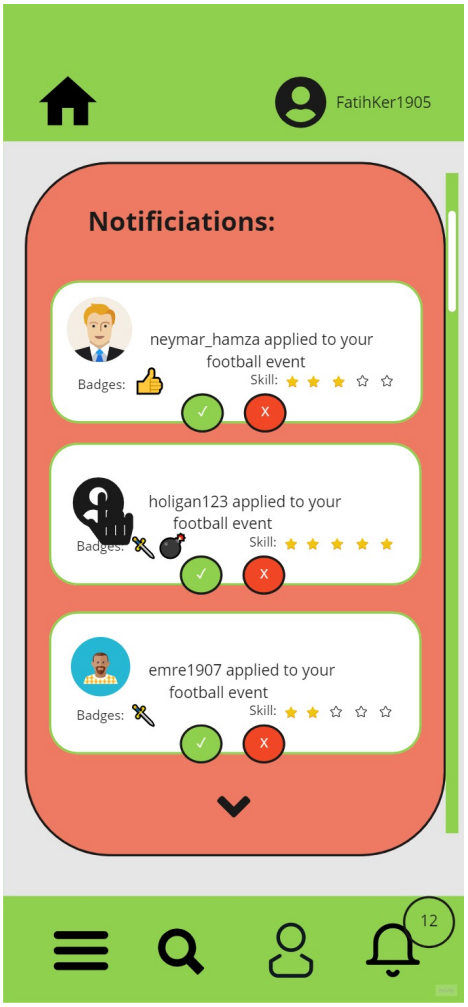


Android

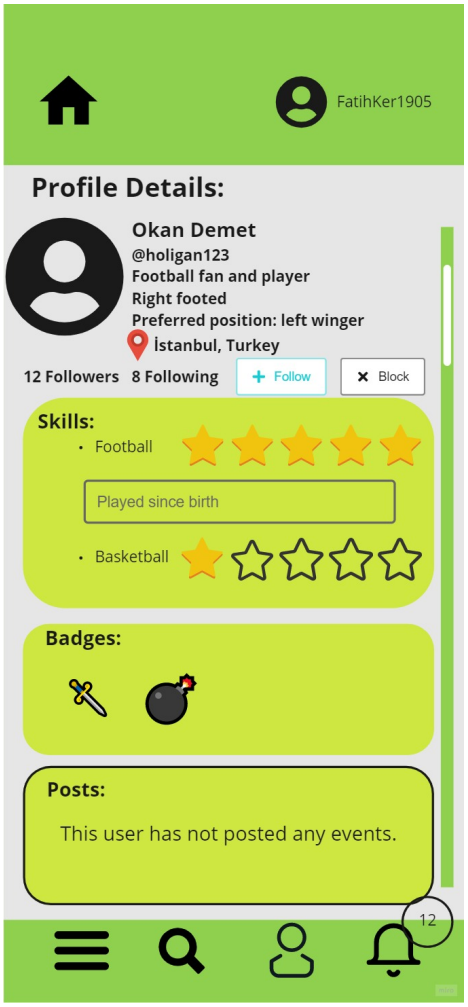
1. Checking application notification from homepage



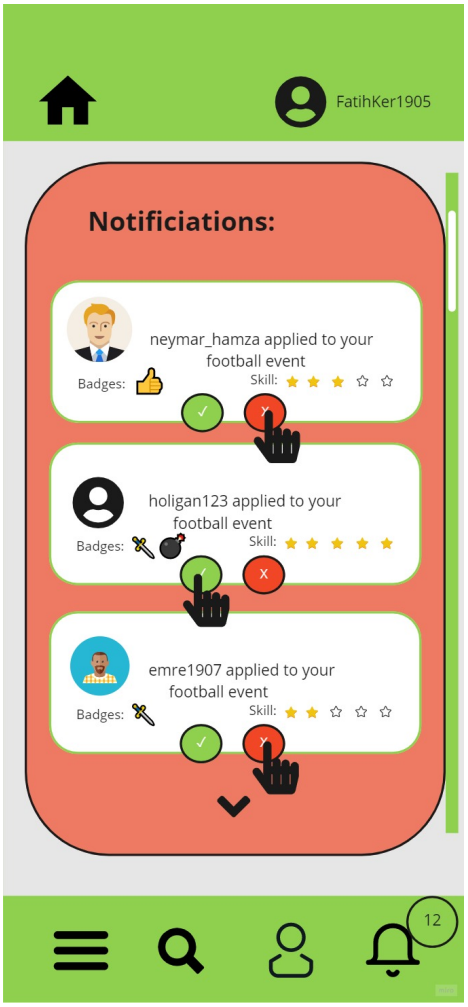
2. Seeing applicants from notifications page and clicking on applicant profile



3. Checking the profile page of applicant



4. Accepting or declining applicants



5. Discussion

Persona:
Zülfikar Tunç
20 years old
University student
Has been staying at home for one year because of the pandemic
Tired of being inactive and wants to do sports
Roger Federer fan
Likes watching tennis and wants to start playing

Story:
Online education made him really bored and he wants to join activities which he can go outside and enjoy. During quarantine, he started watching tennis games and he liked this sport. After the lockdown ended he decided to try playing tennis and found a good event online to join. He will apply for the event but he does not have a tennis racket.

Pre-conditions:

Zülfikar is already registered as a user.

Zülfikar has already found a game and opened the game page.

Goals

Zülfikar wants to ask the event creator if he can borrow a tennis racket.

Acceptance Criteria:

- (1.2.9.1.) System shall allow users to discuss under event or equipment post by writing comments.
- (1.2.9.2.) System shall also allow users to answer/reply discussion comments.
- (1.2.9.3.) Users shall not be able to reply to the answers/replies, thus replies shall be the deepest level of conversations.

Scenario with Mockups:

Zülfikar is on the event's page and he wants to ask a question.

Zülfikar opens the discussion page by clicking the discussion button.

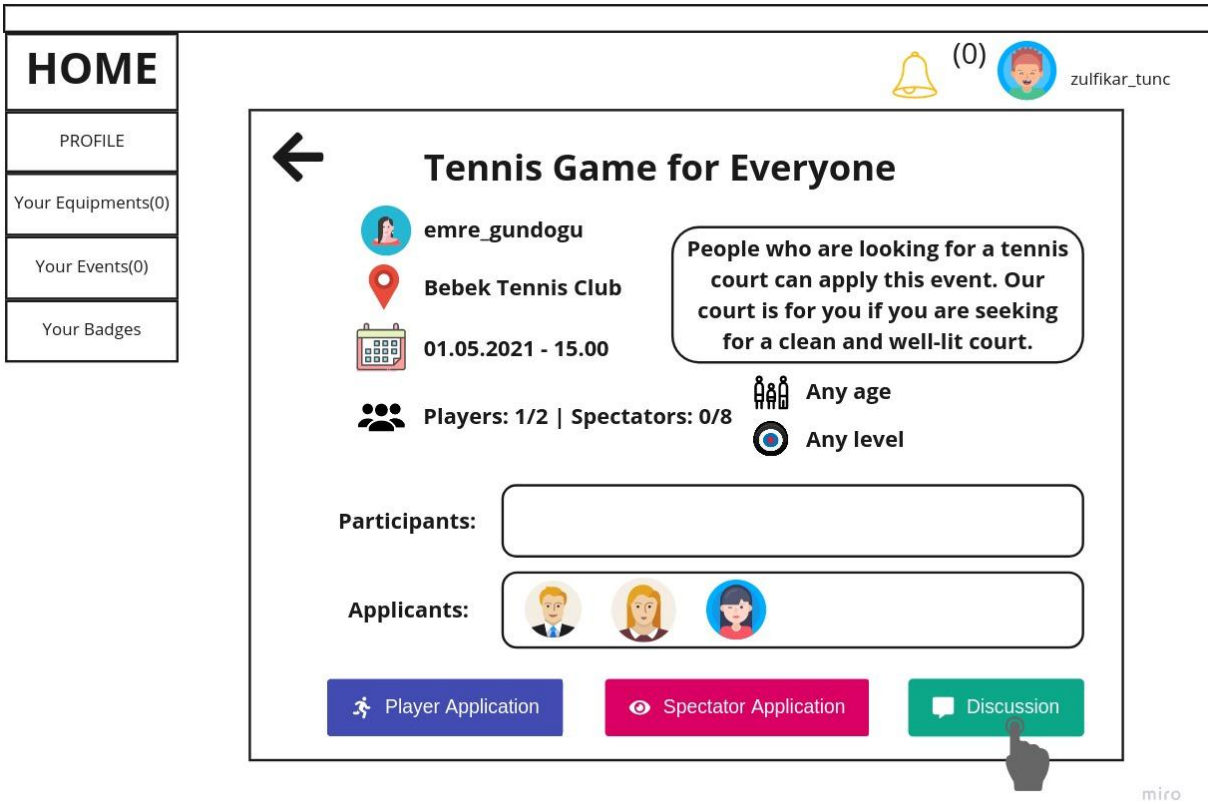
Zülfikar clicks the text field and writes there.

Zülfikar clicks the send button.

The message is posted.

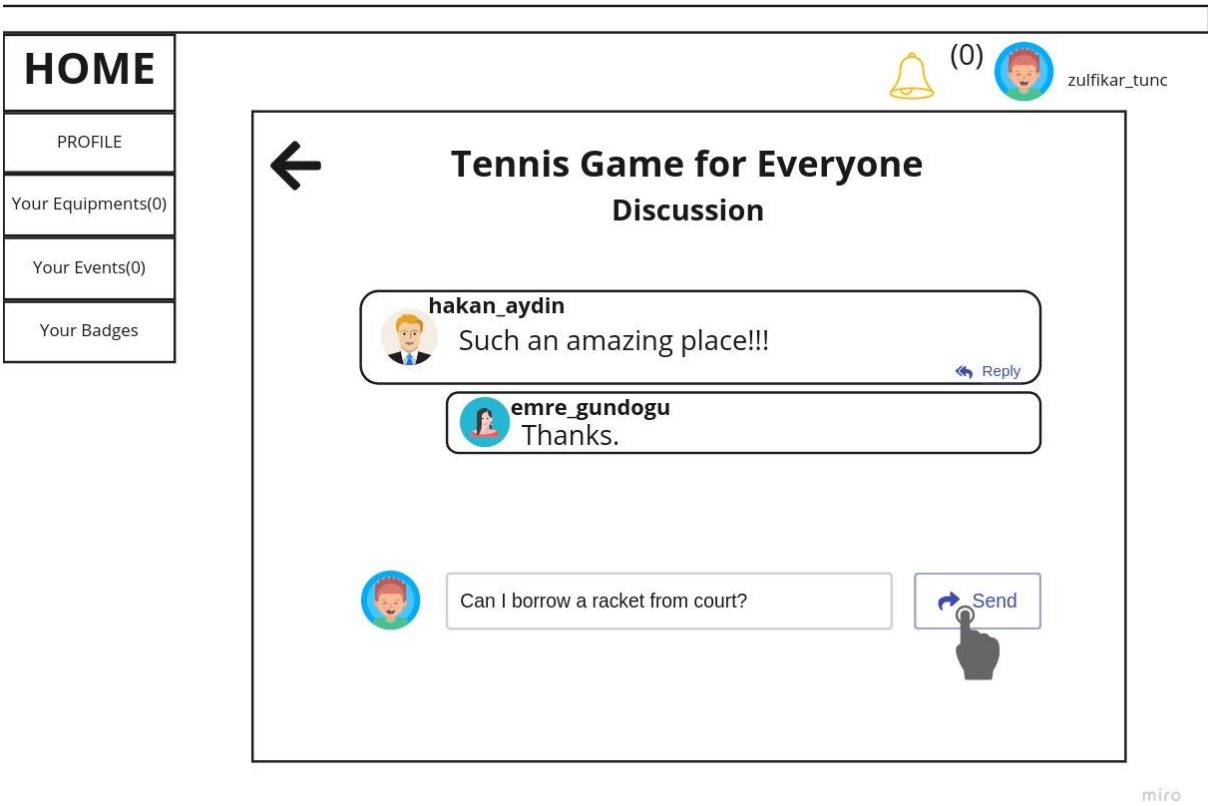
Web Mockup

1. Event Page

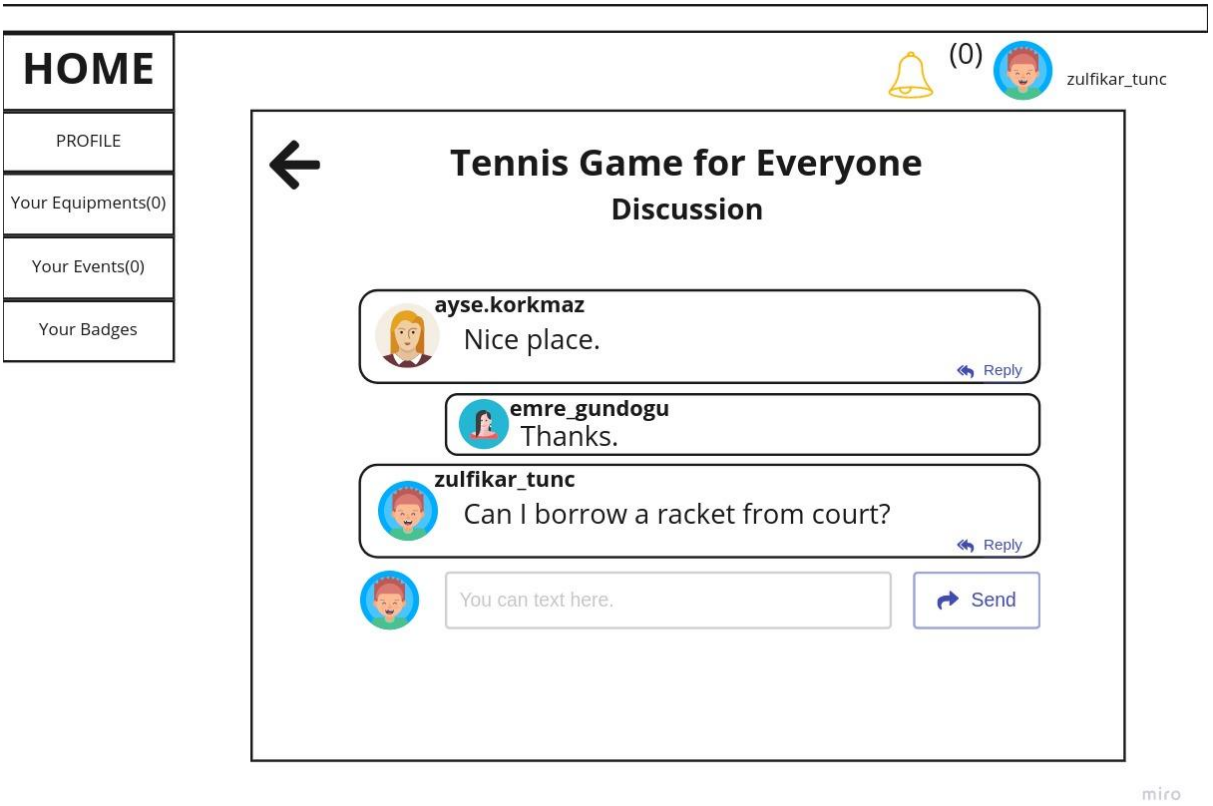


2. Discussion Page

3. Sending Message

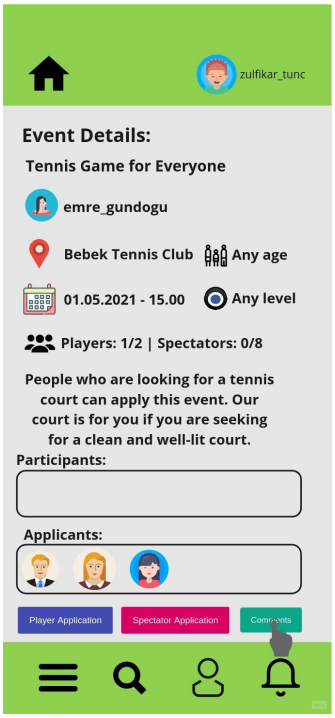


4. Discussion Page (Message is sent.)

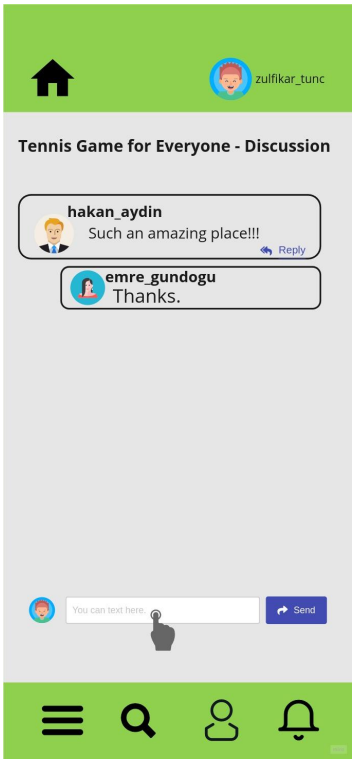


Android Mockup

1. Event Page



2. Discussion Page



3. Sending Message

4. Discussion Page (Message is sent.)