

Bench

CS 30700 Design Document

Team 23

Estelle Yu

Jefferson Chandra

Joy Yu

Leo Lee

Shatakshi Singh

Index

- Purpose
 - Functional Requirements 3
 - Non Functional Requirements 5
- Design Outline
 - High Level Overview 7
 - Sequence of Events Overview 8
- Design Issues
 - Functional Issues 9
 - Non Functional Issues 11
- Design Details
 - Class Design 13
 - Sequence Diagram 17
 - Navigation Flow Map 20
 - UI Mockup 22

Purpose

Existing social networking applications are not optimal for expanding the horizons and networking of professional and leisure athletes in terms of the activities they participate in. Here comes the demand for a new social networking platform designed for sports enthusiasts to connect with each other.

The purpose of our project is to develop a web application aiming to find matches in an accurate and efficient manner by taking various factors into account, including sports type, user availability, and skill level. Our goal is to provide users with a platform that can be used to help sports enthusiasts find other people who share the same sporting interests and preferences.

Functional Requirements

1. User Account

As a user,

- a. I would like to be advised to visit the welcome page of the web-app.
- b. I would like to be able to create a Bench account.
- c. I would like to be able to log in with my existing Bench account
- d. I would like to have a settings panel.
- e. I would like to be able to manage my Bench account.
- f. I would like to be able to deactivate/delete my account.
- g. I would like to be able to find my username using my email address.
- h. I would like to be able to reset my password if I forget it.
- i. I would like to upload a local file as my profile picture.
- j. I would like to be able to view and review my profile.
- k. I would like to be able to edit my profile information.
- l. I would like to be able to set my status in my profile.
- m. I would like to change the color theme of the page.
- n. I would like to be able to access other user's social media accounts that are public and listed in their profile.

2. Filtering and Sorting System

As a user,

- a. I would like to see the main matching page with a navigation bar.
- b. I would like to view the set of the matched and sorted users on the matching page.
- c. I would like to be able to see a dashboard that shows the user's profile information.
- d. I would like to be able to delete a user in the current matching set.

- e. I would like to be able to match with a new user.
 - f. I would like to be able to access other user's social media accounts that are public and listed in their profile.
 - g. I would like to search for a user by name.
 - h. I would like to have a filter panel and be able to apply it.
 - i. I would like to see users matching according to the highest ratings.
 - j. I would like to block another user from appearing on my matching page.
 - k. I would like to see the last time that the user logged in on the user's dashboard.
3. Calendar
- As a user,
- a. I would like to be able to view my calendar.
 - b. I would like to be able to easily navigate through my calendar.
 - c. I would like to edit/personalize my calendar.
 - d. I would like to add an event scheduled with a person I matched with into the Bench calendar.
 - e. I would like to export my event calendar (if time allows).
4. Real-time Notifications
- As a user,
- a. I would like to get mail to my registered email account regarding my notifications.
 - b. I would like to be able to receive a notification when there is a new incoming chat.
 - c. I would like to turn off my notifications.
5. Review System
- As a user,
- a. I would like to be able to show/see other user's ratings/reviews for me.
 - b. I would like to be able to show/see other user's ratings/reviews.
 - c. I would like to give other users reviews, only if I have given a rating.
 - d. I would like to be able to show/see the average of rating for a user.
 - e. I would like to be able to set myself anonymous when giving another user reviews.
 - f. I would like to be able to edit/delete a review I gave for someone.
 - g. I would like to be able to go to the user that is not anonymous that has given a review to my profile or another user.
6. Report giving
- As a user,
- a. I would like to be able to report a given review.
 - b. I would like to be able to report a suspicious person in the chat box.
7. Messaging

As a user,

- a. I would like to be able to easily navigate through the message panel.
- b. I would like to be able to send/request messages with a person that I was matched with.
- c. I would like to be able to reply to an incoming message from a matched user.
- d. I would like to be able to show/manage all the chat history.
- e. I would like to be able to share pictures in the chat with a matched person.
- f. I would like to be able to accept/deny an incoming chat request.
- g. I would like to be able to block a user's incoming messages.
- h. I would like to be able to share videos in the chat room (if time allows).
- i. I would like to request another user review/rating from other people (if time allows).

Non Functional Requirements

1. Performance

- a. I would like the application to run smoothly without crashing.
- b. I would like the application to be launched in 5 seconds.
- c. I would like the application to support 10000 users.

2. Server

- a. I would like the server to support real time server client communication.
- b. I would like the server to be able store users data to a database

3. Appearance

As a developer,

- a. I would like to create a user-friendly and pleasing user interface.
- b. (If time allows) I would like to allow users to customize their user interface

4. Security

As a developer,

- a. I would like to let the users choose whether to enable access to their personal information (e.g. specific location and other social media account) or not.
- b. I would like the application to have some algorithms for rating, reporting and blocking users to avoid bad behaviors of some users.
- c. (If time allows) I would like to protect the user's password and personal information being stored in the database.

- d. (If time allows) I would like to limit one account per individual to avoid users from abusing the application.

5. Usability

As a developer,

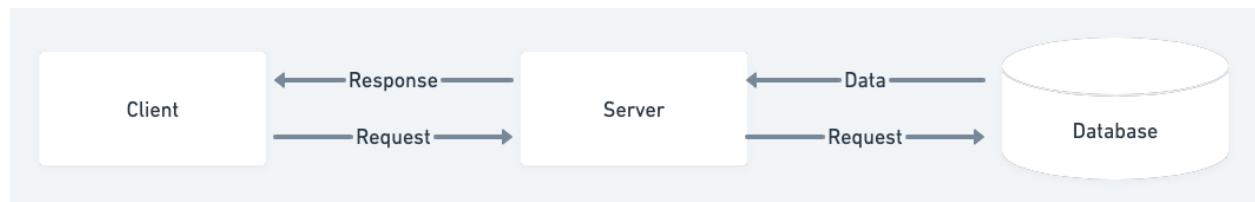
- a. I would like the application to be friendly, helpful and self-explainable.
- b. I would like the application to be able to run on web browsers.

Designing Outline

High Level Overview

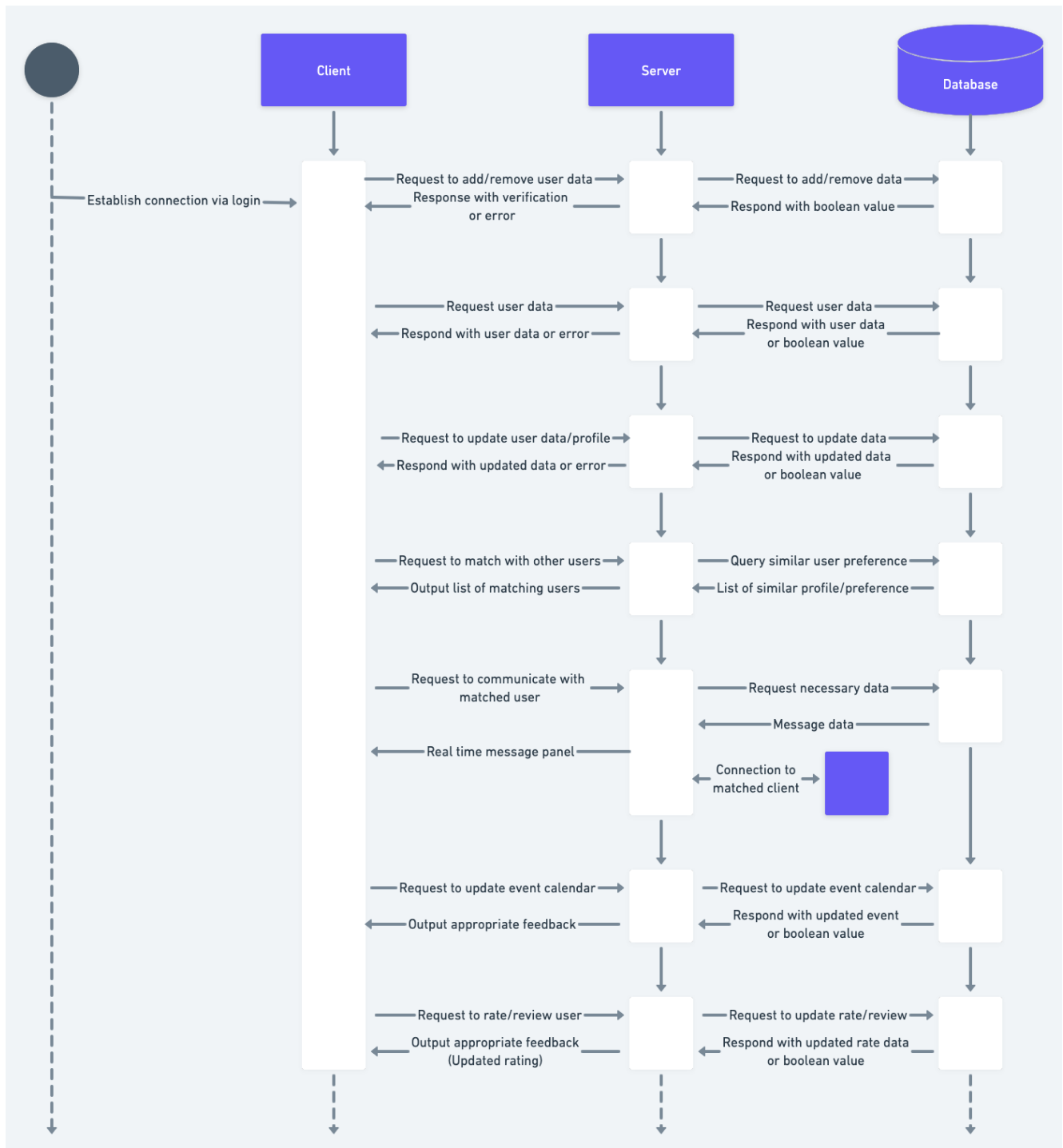
This project is aiming to be a social networking web application that encourages sports enthusiasts to connect with other users with similar sporting interests. It will allow users to interact with each other, as well as holding a potential sporting event together. Bench will utilize a client-server concurrency model, where the server will accept and appropriately react to any incoming client-side request, such as accessing/retrieving data from the database.

1. Client
 - a. Web application clients, serving on each user's device, will be the main source of interaction with our server.
 - b. Client will be able to send requests to the server.
 - c. Client will retrieve data from the server, display it as a user friendly UI.
2. Server
 - a. Server will solitary process all the incoming requests from the client.
 - b. Server will have access to information in the database, it will be able to retrieve and prepare the data when requested.
 - c. Server will be able to interact/send responses to an appropriate client.
3. Database
 - a. Database will store all data that is associated with the web application.
 - b. It will receive a request from the server and output proper information/data.



Sequence of Event Overview

Below model is a prototypical sequence of interaction between client, server, and database. Upon user login, the client will establish a connection between the server. Afterward, users will have the authority to send various types of requests to the server. Once the server receives a request, it will first verify the incoming request, to prevent any malicious inputs from cascading into the database. Then, it will send a query to the pre-entrenched database and retrieve the desired data. Finally, it manages the client request by returning the appropriate data back to the client's request.



Design Issues

Functional Issues:

1. Complexity of user credentials. What is the required information to initiate an account?

1. No credentials.

2. Username and Password.

- 3. Username, Password, and Email.**

4. Username, Password, Email, and optional address.

Choice : 3

Username and password are required to distinguish each user, and it ensures the security of each account. Recording an email can be advantageous to both the user and the web application; on the user's end, it provides an extra layer of protection and the ability to retrieve username or password when lost. On the application end, it prevents users from initiating multiple accounts and offers notifications to users. Although address information is a crucial aspect of the application matching process, it could potentially scare away users when they are attempting to sign up for the application.

2. Matching profile configuration. How to retrieve user preference/interest in sports?

1. Selection from list of defined/fixed categories.

2. Selection from list of defined, but customizable categories.

- 3. Selection from list of defined, but customizable categories, addition of optional user description.**

4. Fully customizable user description page.

Choice : 3

List of defined categories will aid users to make choices, as well as simplifying the process of using the application. In addition, it will allow developers to make accurate predictions when matching users. Having customizable categories will offer more flexibility and respect choices that are made by users; furthermore, it improves the precision of the matching process. Finally, having a user description page allows users to review the matched peer before making a final decision. Letting users have a fully customizable description page may complicate the process of matching, additionally, some users may not be open towards the idea of having to write a personal description in order to get matched with someone.

Therefore, the description page must be optional; finally, choice 3 was selected over other choices.

3. How to collect user address/location

1. Manual entry of user location.

2. Periodically retrieve user location when user is actively using the application.

3. Ceaselessly collect user location data.

Choice: 1

User address is a crucial aspect of the user matching process. However, both ceaselessly or periodically accessing user addresses may cause a couple issues. First, tracking features of the application may lead to a leakage of unwanted user information and ultimately invade an individuals' privacy. In addition, continuously acquiring location data would likely drain the device battery. Combining this on top of using React as a frontend framework is not an optimal choice. By allowing users to manually input their location, it adds an extra layer of security (e.g submitting location of their neighborhood rather than their home address), and also be more flexible with the address preference.

4. User communication

1. Reference of email or other social media platforms.

2. Real time message service.

3. Real time message service and optional links to other social media.

Choice: 3

Allowing users to communicate is a crucial part of the matching process. Since the application is only matching users based on their preferences and location, it is up to users to share the logistics of the actual sporting event. Having a link to another social media platform or email is not the most optimal way of communication, since not all users will have access to other social media (e.g Instagram, discord ... etc) and regularly checking email can be tedious. Having a real time message service will ensure fast and secure communication between users. However, some individuals will prefer chatting over other social media. Therefore, choice 3 was selected.

5. How to store users' addresses?

1. Store users address the way it is.

2. **Convert input addresses into a set of coordinates, then store them in the database.**

Choice: 2

Users will not tolerate a slow rendering application, by storing each address into the database, calculating the distance between each user is going to be very difficult and time consuming. By converting the address into a set of coordinates, the server will be able to calculate the distance much faster. Finally, the application will load faster.

Non-Functional Issues:

1. Frontend framework selection.

1. HTML, CSS, and JS

2. **React**

3. Angular

Choice: 2

Compared to other frameworks, React has more learning sources and tutorials. Furthermore, React also has a faster rendering rate, optimal tools for designing web applications, and offers an easily testable developing environment. Given the timeframe of the project, React is preferred over other frontend frameworks.

2. API for location tracking

1.Distance Matrix API (GoogleMaps)

2.**OpenStreetMap API**

Choice: 2

Since the application is taking a manual entry of address and not tracking the user's device, both Distance Matrix and OpenStreetMap can be used to identify the distance. However, Google Maps APIs are expensive, whereas OpenStreetMap is an open source API. Therefore, choice 2 was selected.

3. What web service should be used?

1.AWS

2.**Firebase**

Choice: 2

Firebase allows developers to simplify the configuration step of development. More importantly, the cost of running a web service via AWS is significantly higher than firebase. Considering the scale of the web application, utilizing firebase is optimal.

4. What database should we use?

1.MySQL

2.Firebase

3.Oracle XE

Choice: 2

Since we are utilizing firebase to host our web service, using firebase as a database comes naturally and will be easy to implement. Furthermore, firebase has a fast bootstrap rate, integrates well with user data, Node.js, and few of our team members have experience with firebase.

5. Backend framework selection.

1.Spring Boot

2.Node.js

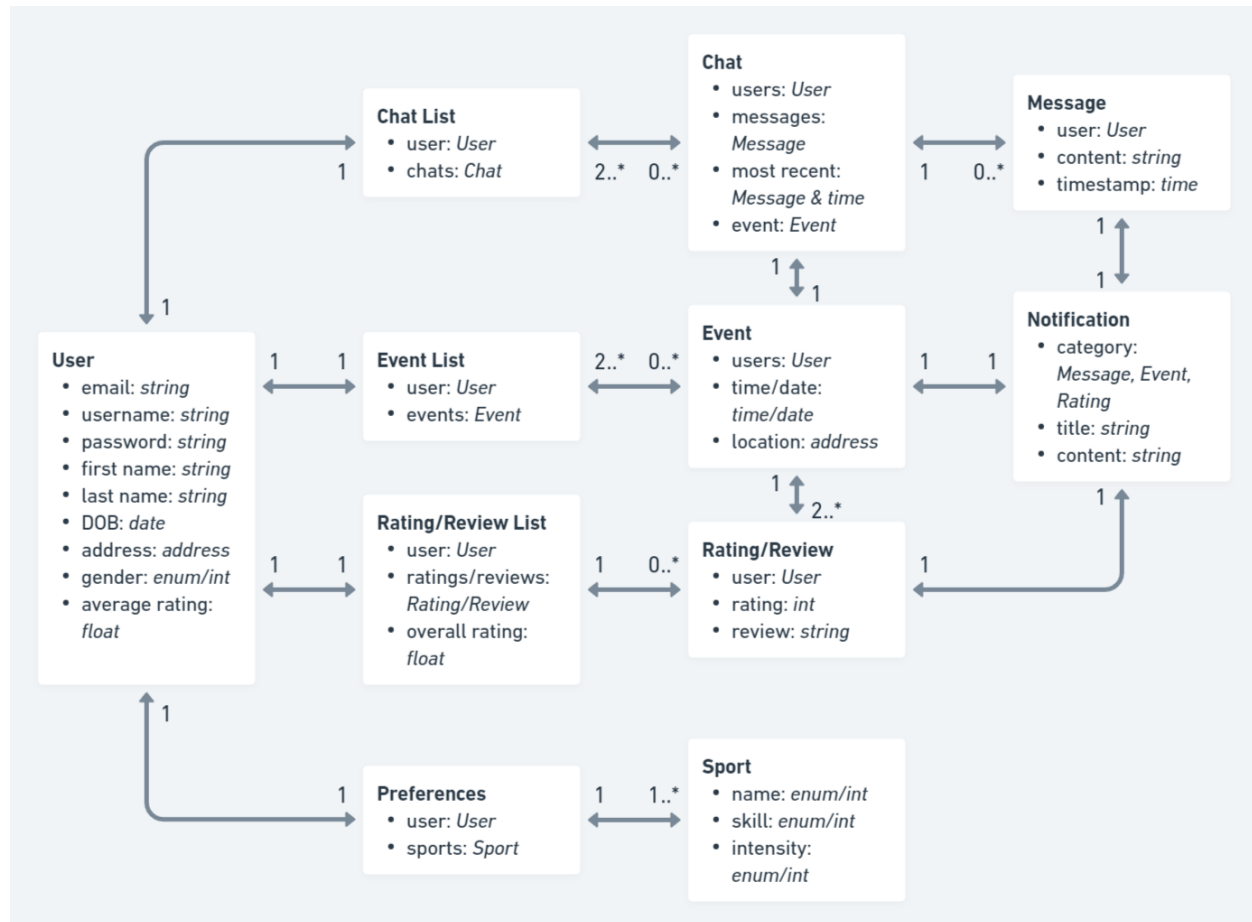
3.Django

Choice: 2

Node.js is compatible with react and firebase. In addition, compared to other frameworks, it has more learning resources and open source communities, and it is very scalable. Furthermore, some team members are already familiar with Node.js as well as javascript coding. Therefore, we picked Node.js over other frameworks.

Design Details

Class Design



Descriptions of Classes and Interaction between Classes

The classes are designed based on the objects in our application. Each class has a list of attributes which is the characteristics owned by each object.

- **User**
 - User object is created when someone signs up in our web-application.
 - Each user will have a unique username.
 - Each user will set their date of birth when they sign up.
 - Each user will have an email assigned and password for login purposes, along with the username that was already created.
 - Each user will fill in their first name, last name, gender, and address when setting up the profile.

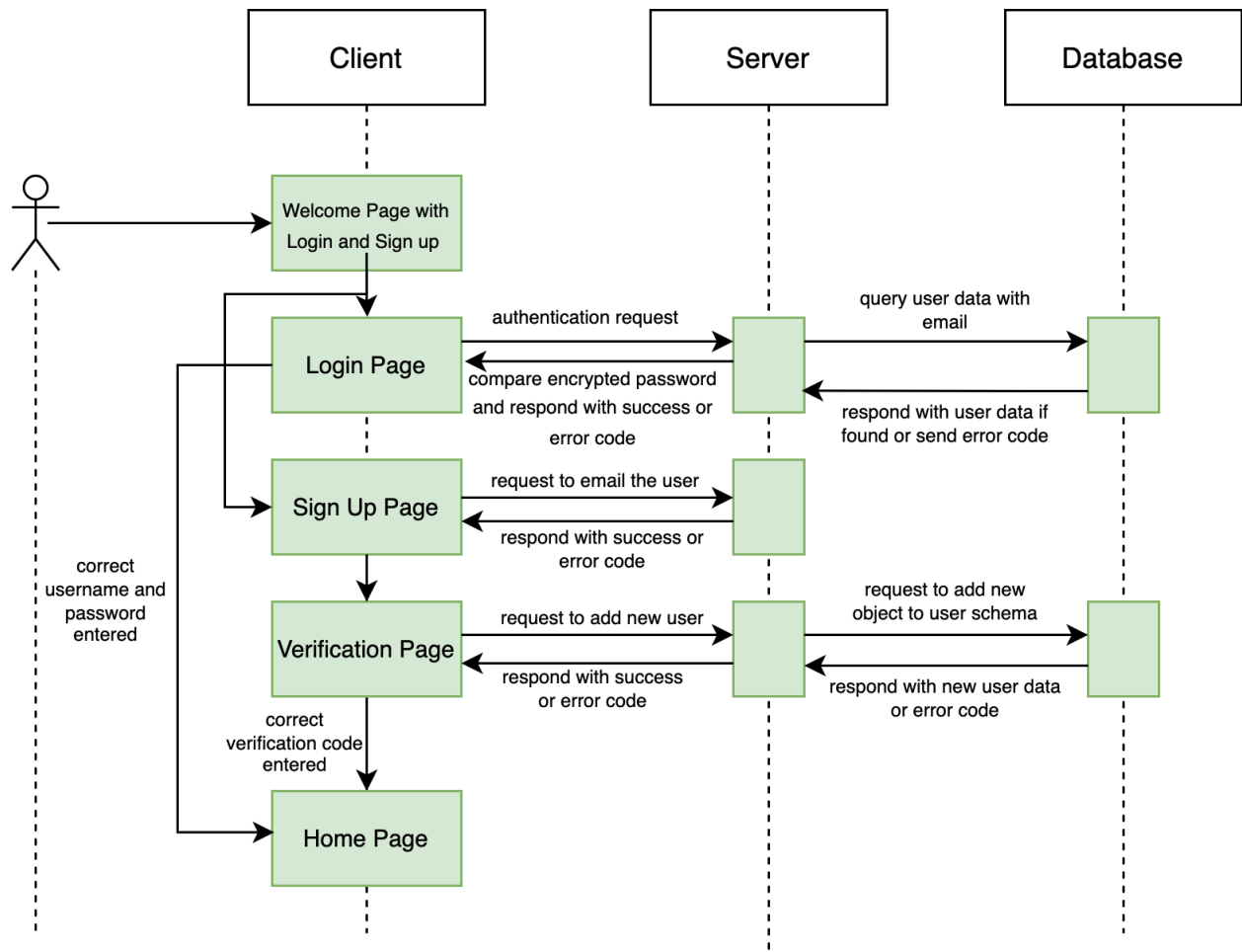
- Each user also states their preferences by choosing at least one sport when setting up their profile
- Each user will have a list of preferences, and each will be categorized by a sport
- Each user will have a chat list that represents the list of chats, which would be empty in the beginning
- Each user will have an event list that represents the list of events, which would be empty in the beginning
- Each user will have a rating/review list that represents the list of ratings and reviews, which would be empty in the beginning
- Chat List
 - A chat list object is made when a user is created
 - Each chat list is only associated with only one user
 - Each chat list has 0 or more number of chat room
 - Each chat list displays only the most recent message element from each chat room in that chat list, along with the user(s) associated with that chat, excluding the user that the chat list is associated to
- Event List
 - A event list object is made when a user is created
 - Each event list is only associated with only one user
 - Each event list has 0 or more number of events
 - Each event list displays the location and time/date of each event in the event list
- Rating/Review List
 - A rating/review list object is made when a user is created
 - Each rating/review list is only associated with only one user
 - Each rating/review list has 0 or more number of rating/review
 - Each rating/review list displays the rating element from each rating/review object along with the user that made
- Preferences
 - Preferences object is made when a user is created
 - Each preferences is only associated with only one user
 - Each preferences has 1 or more number of sports
- Chat
 - Chat is created when a user match with another user

- When a user wants to match with another user, a message is sent from this user to the other user saying they want to match with them
- Each chat will have 1 or more messages
- Each chat holds an element that is the most recent message sent or received
- Each chat holds the most recent message timestamp
- Each chat has an event object that can be edited from the chat
- Event
 - An event object is created when a chat object is created
 - Each event has at least two users associated with it
 - Each event has a location and date and time
 - The default date and time element of each event is the same date and time but one year later
 - The default location element of each event is empty
 - Each event can only be entered a valid date and time
 - Each event can have one or no location or address
 - Each event will send a notification to every user associated to the event on the day of the event
 - Each event can be edited by every user associated with it, but can has to be confirmed by every user
 - Each event will close when it passes the time/date
- Rating/Review
 - A rating/review object is created when an event closes, every user associated to that event gets one notification for rating/review for every other user
 - All rating/review for one user can be viewed from the rating/review lists of that user
 - Each rating/review has a user field to show who made the rating/review
 - Each rating/review will have a rating field that will be represented by an integer from 1 to 5
 - Each rating/review will have a string content that is available but not required to be filled by the user making the rating/review
 - Each rating/review can only be created from one user to another if the users had an event together
- Sport
 - A sport object is created when a user adds another sport for their preference

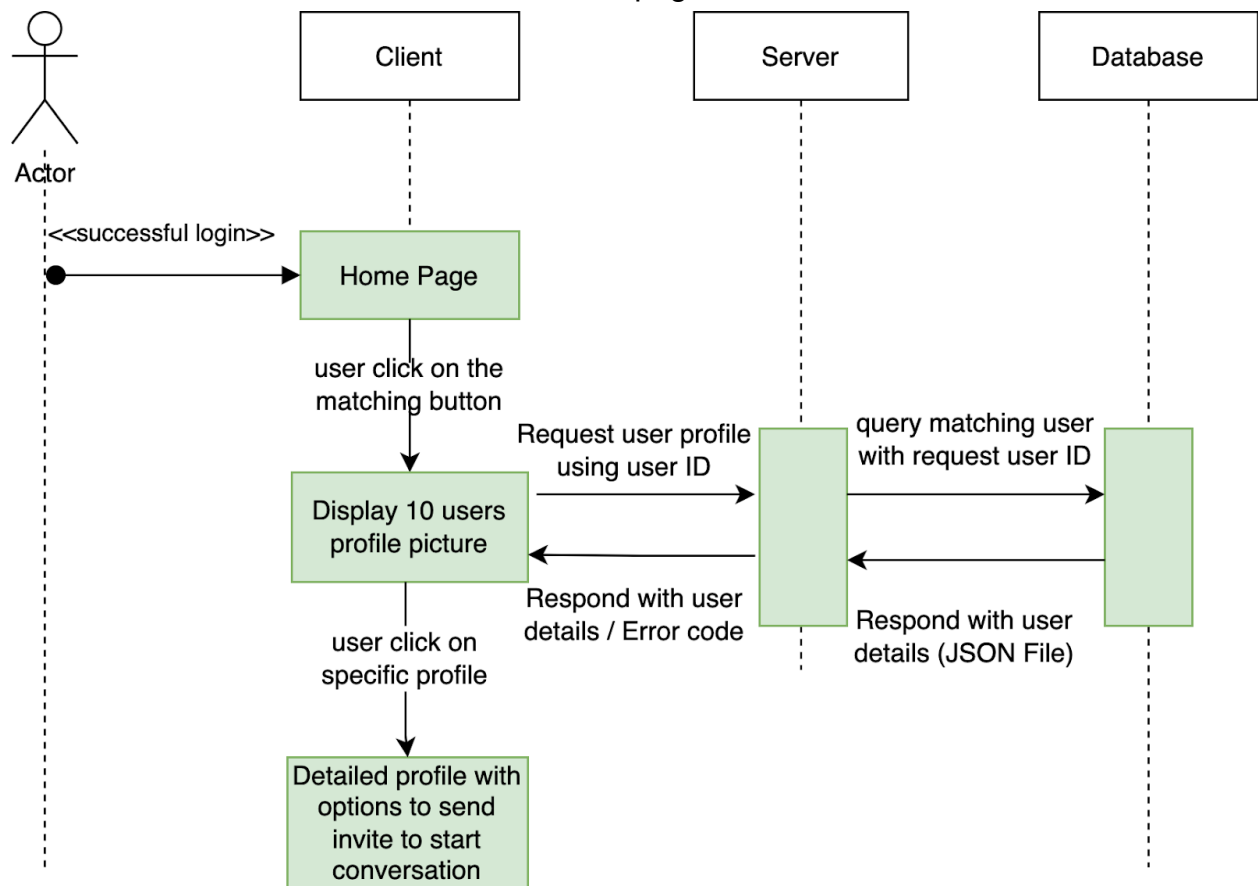
- Each sport has a sport enum field to represent all of the sports that the user have not indicated that they are interested in yet
- A user can only choose one sport field from a sport object
- Each sport has a skill field that is represented by an enum or an integer
- Each sport has an intensity field that is represented by an enum or an integer
- Message
 - A message object is created when a user sends or receives a message to another user in a chat
 - Each message that is received has a user field to show who the message is from
 - Each message that is sent does not have a user field
 - Each message has a content represented by a string
 - Each message has a timestamp
- Notification
 - A notification object is created when there is a new message from a chat of a user from another user, a new rating/review for that user from another user, when an event that the user is associated with ends, including request of rating/review after that event closes
 - Each notification has a title that will be represented by a string
 - Each notification has a content that will be represented by a string

Sequence Diagrams

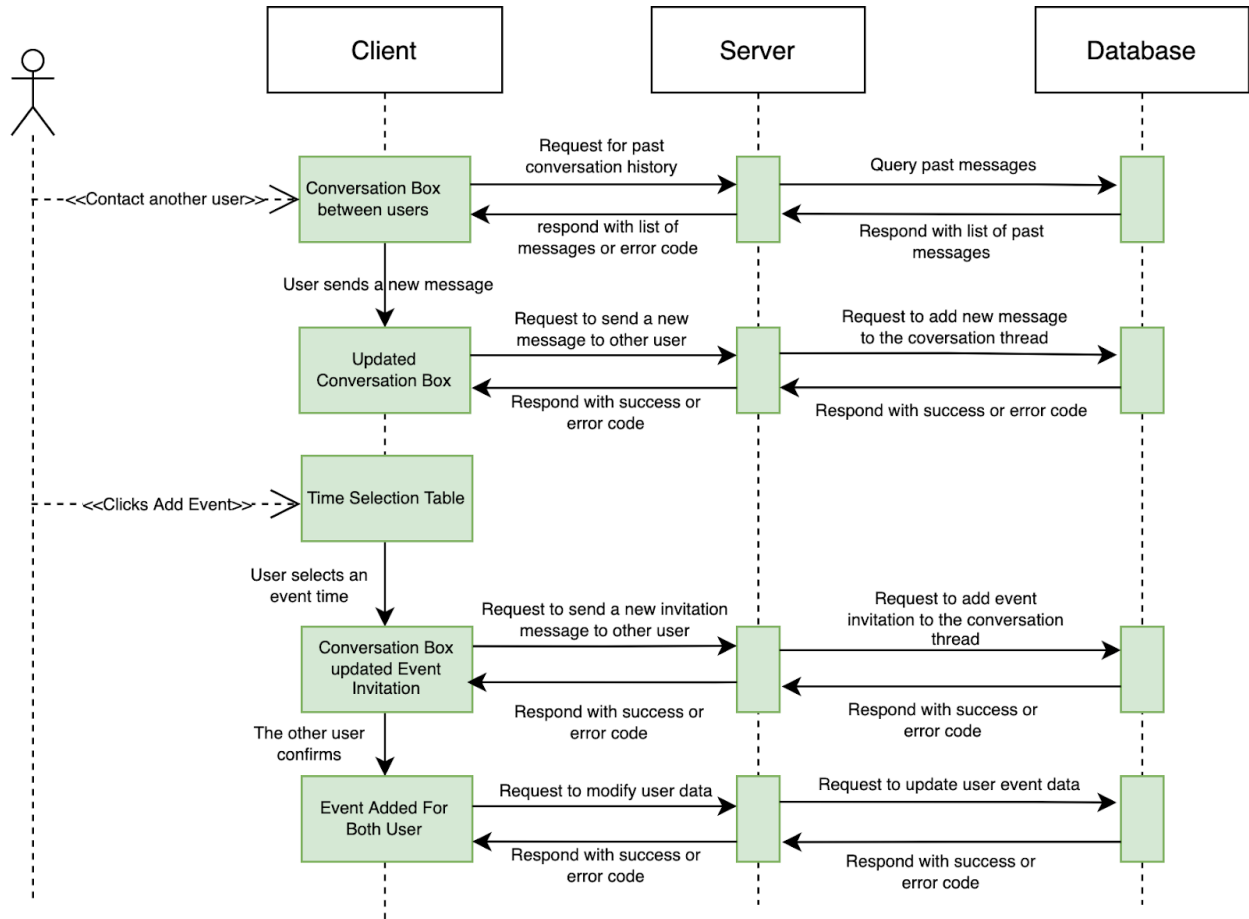
1. When the user first opens the web-app



2. When the user is on the home page and wants to match with other users



3. When the user wants to communicate with the matching user

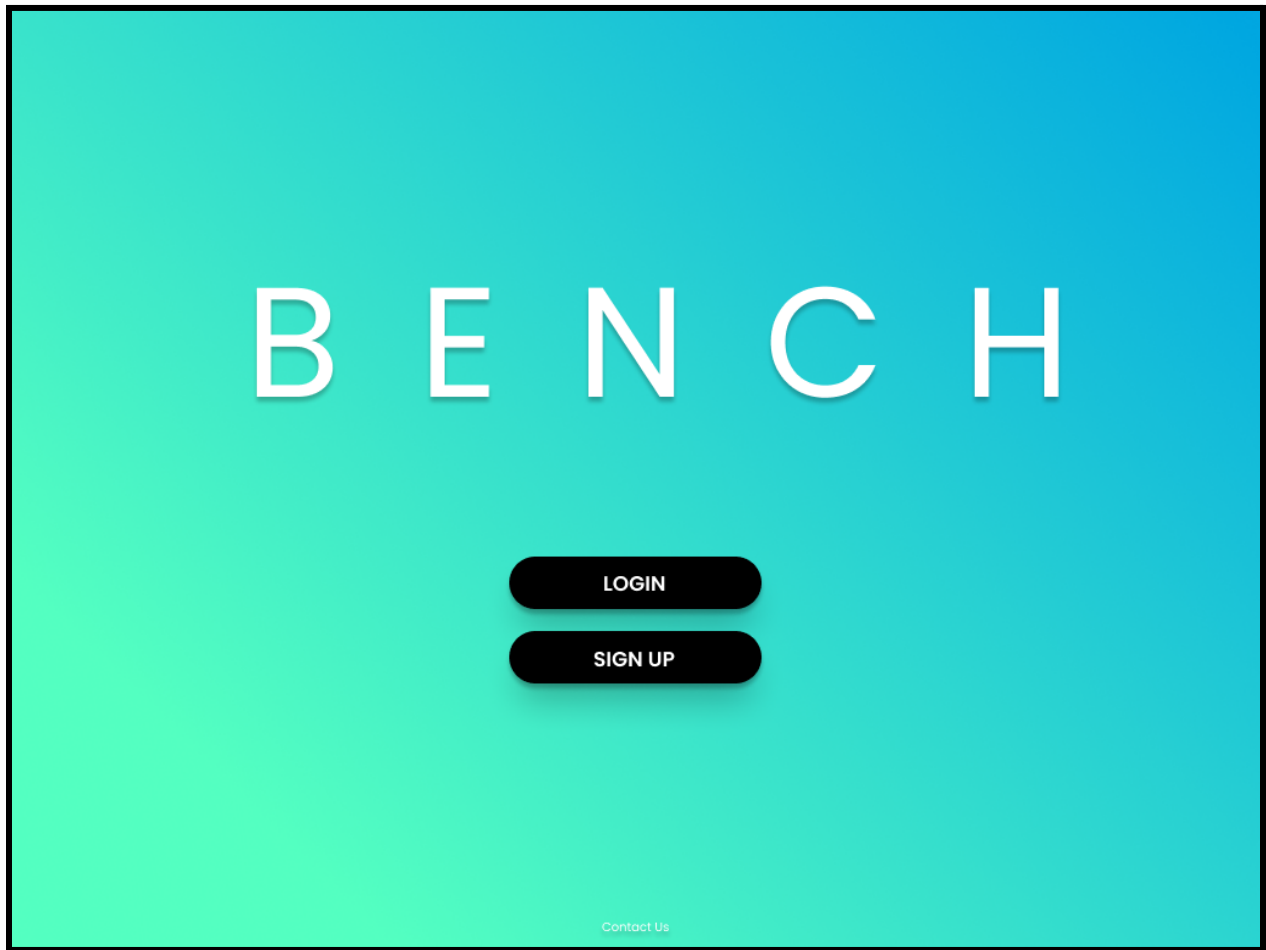


Navigation Flow Map

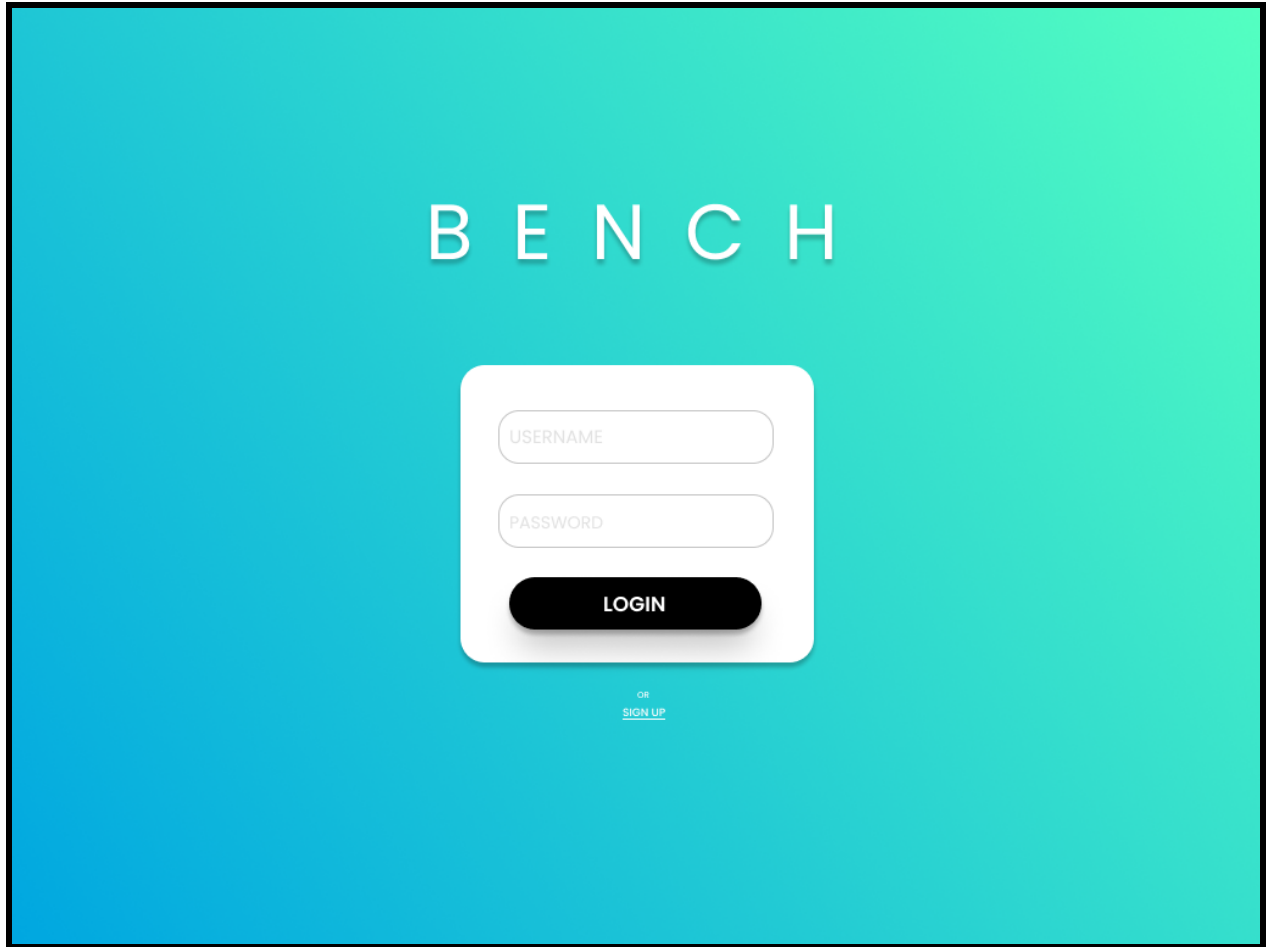
The design of our navigation emphasizes ease of use and a pleasing customer experience. We encourage our customers to log into the homepage with their account. If they do not have an account , they must enter their user information to sign up for one. The homepage contains the navigation bar, a calendar of upcoming scheduled events, and chat notifications. The navigation bar on every page contains all the accessible pages in the web application that includes the home page, the profile page, the message page, and the matching page. The matching page contains the search bar at the top which allows the user to find a specific user by their name and a filter panel to help the user find sport buddies of their best match. The dashboard that shows up every time a user profile picture is clicked allows the user to get in touch with the related user easily.

Diagram shows the ease of use. All main pages after login are accessible from the navigation bar at the top. It also gives a pleasing user experience, as the functionalities cover many aspects of the user demand even in minor details like viewing reviews and reporting negative contents.

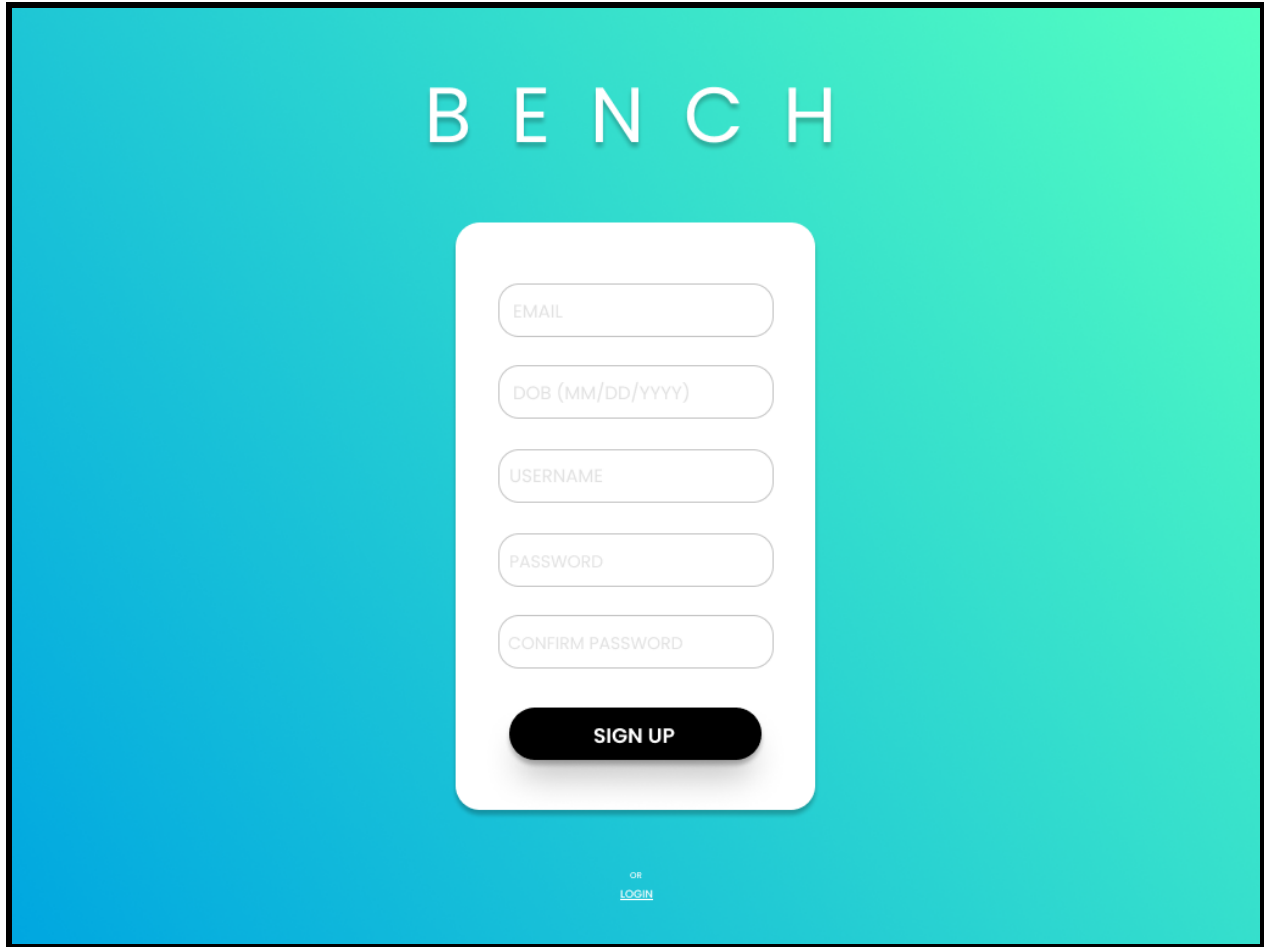
UI Mockup



This is the entrance page. There is a button for login and signing up respectively. If the user needs any help, they could click the button “contact us”.




This is the login page that requires the user to enter their username and password. The login button sends information to the server to allow the user to enter their account. There is a sign up button at the bottom to allow the user to go to the sign-up page.



The image shows a sign-up page for a service named "BENCH". The page has a teal-to-blue gradient background. At the top, the word "BENCH" is displayed in large, white, spaced-out capital letters. Below this, a white rounded rectangle contains a sign-up form. The form consists of five input fields: "EMAIL", "DOB (MM/DD/YYYY)", "USERNAME", "PASSWORD", and "CONFIRM PASSWORD". Each field has a light gray border and placeholder text. Below these fields is a black rounded button with the text "SIGN UP" in white. At the bottom of the white rectangle, the word "OR" is centered above a blue link labeled "LOGIN".

This is the sign-up page that requires the user to enter their email and date of birth and compose their username and password. The sign up button sends the information to the server to check duplication.

[Home](#) [Events](#) [Matching](#) **BENCH** 

SET UP YOUR PROFILE

Personal Information

Address

+ Add more

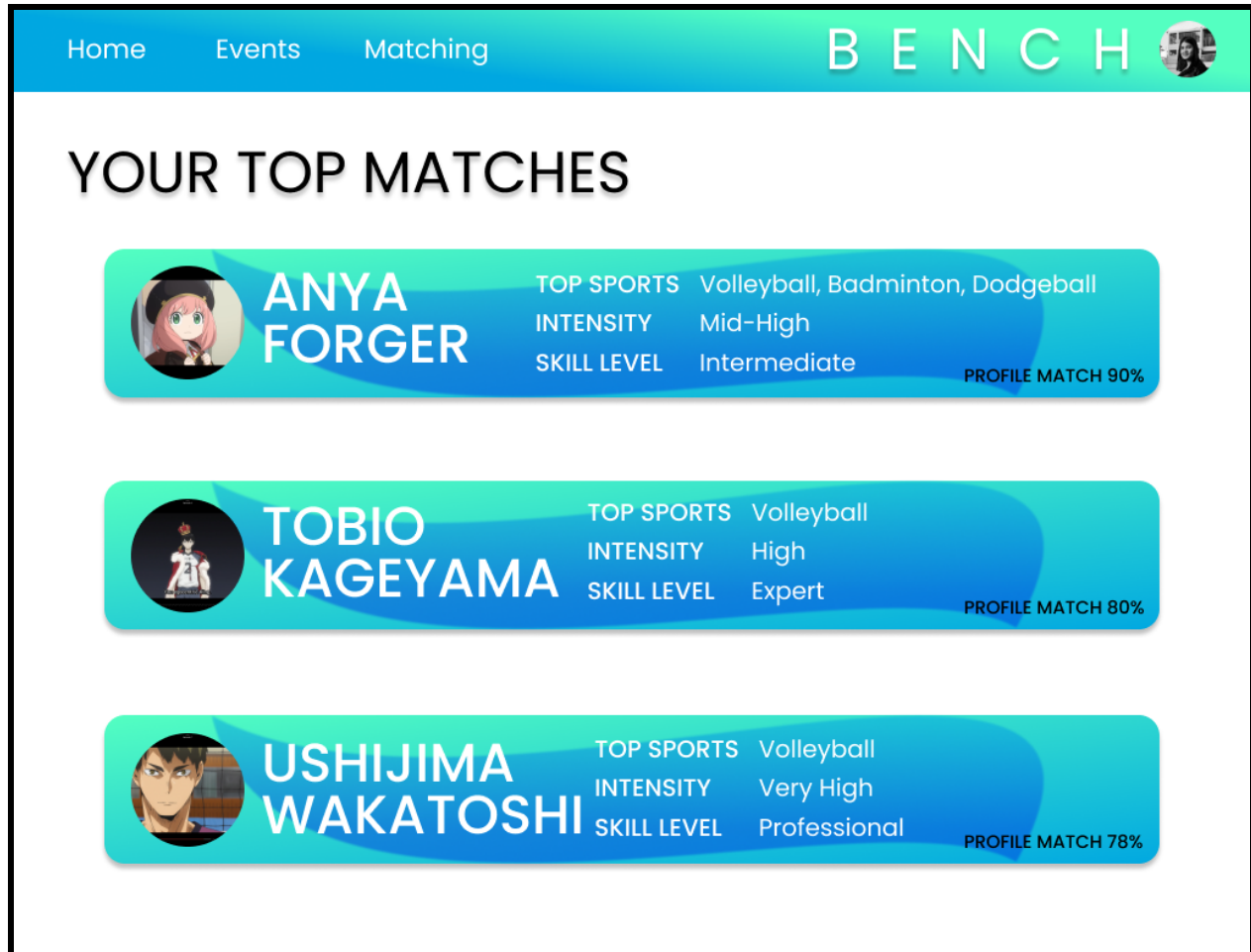
Sports

+ Add more

This is the profile page where the user could set up or edit their account information, including the personal information, address, the areas of sports of their interest and so on. The buttons “+ Add more” allow the user to expand certain information as needed. The navigation bar on top provides the users an easy way to navigate to other pages.



This is the event page where you could see the latest event that you've scheduled for on Bench. Clicking on each event panel will give you the details of that specific event. The events button on the bottom right allows the user to edit or compose events. The navigation bar on top allows the user to easily navigate to the home page, the matching page and the profile page.



This is the matching page where the user could view a list of user profiles that meet the conditions that the user requires and select potential sport buddies from them. Clicking on each profile card of a user would allow the user to view the related user's profile page with more of their information. The page could be scrolled down to show more matched user profile cards. The list of user profiles are displayed in the order of the highest match to the lowest match and are also sorted with the users' ratings.