

# **Sprint 2 Plan - Project Jimmy**

## **Team roles:**

Kushal Lahoti - Scrum Master  
Yash Phatak - Product Owner  
ChuanHsin Wang - Developer  
Wei-Chien Cheng - Developer  
Kuan-Ru Huang - Developer  
Barry Liu - Developer  
Mrunmay Deshmukh - Developer

## **Post sprint client meeting date/time/place:**

Date: 9th October, 2024  
Time: 10:30am - 11:00am  
Place: Zoom Call

## **Sprint goal:**

In this sprint, our goal is to complete seven user stories, which encompass tasks such as UI development for key features, bug fixes, and the creation of testing modules. So far, we have successfully implemented the authentication functionality and user profile setup. Following that, we conducted a demo with the client to present the current progress of the application. The client provided feedback and additional requirements, particularly related to the UI's styling and responsiveness.

We also discussed the client's expectations for the core functionality of the buddy matching feature during the sprint demo call, especially in terms of its design and user experience. Based on this feedback, we will be making the necessary adjustments and adding new features. In this sprint, we aim to revamp the UI design and ensure it follows a mobile-friendly structure. Additionally, we plan to integrate a fitness profile for users, capturing details like activity preferences, location, and gender. We will also work on developing the UI template for the buddy matching feature.

During the last deployment, we identified a bug where user-uploaded profile pictures were not visible on the Heroku production environment. This issue arises because Heroku uses ephemeral file storage, which means uploaded data is erased whenever the server (dyno) restarts. To address this, we will need to implement persistent file storage, either on Heroku or through an external platform like AWS, and integrate it with our current setup.

Finally, there were some tasks left in the previous sprint, creating a small backlog. We plan to address those items in this sprint and ensure that all outstanding tasks are completed by the sprint's end.

**User stories:** A total of 7 user-stories have been added to the sprint plan.

**Feature:** Mobile Responsive User Interface

As a front-end developer,

I want to modify the UI to make it mobile responsive,

So that users can have an optimal viewing and interaction experience across various device sizes, as per the client's requirements.

**Scenario 1:** Ensure Responsive Layout on Mobile Devices with optimized navigation

**Given:** The user accesses the application on a mobile device.

**When:** The screen size is reduced (e.g., below 768px width).

**Then:** The layout should automatically adjust to a mobile-friendly design (e.g., stacked content).

**Scenario 2:** Align Client's Requirements for Mobile UI

**Given:** The client has specific design requirements for the mobile version.

**When:** The developer implements responsive changes.

**Then:** The design should meet the client's criteria for look and feel, such as specific color schemes, spacing, and font styles for mobile devices.

**Feature:** UI Template for Matching with Gym Buddies

As a user,

I want to browse profiles based on workout type, location, and experience level,

So that I can find a gym buddy who fits my preferences.

**Scenario 1:** Display Search Filters

**Given:** The user is on the gym buddy search page,

**When:** The page loads,

**Then:** The user should see filters for workout type, location, and experience level to refine their search.

**Scenario 2:** Swipe to Match/UnMatch

**Given:** The user is viewing gym buddy profiles,

**When:** The user swipes on a profile,

**Then:** The system should record their interest in that profile for potential matching or unmatch it based on swipe direction.

**Feature:** Fix User Profile Picture Visibility Bug in Production

As a developer,

I want to fix the issue causing the user profile picture to disappear in the deployed version on Heroku,

So that users can consistently see their profile picture without it going missing over time.

**Scenario 1:** Display Profile Picture Correctly in Production

**Given:** The user has uploaded a profile picture.

**When:** The user accesses their profile in the deployed version of the application.

**Then:** The profile picture should be displayed correctly on the user's profile page without disappearing.

**Scenario 2:** Implement Persistent Image Storage Solution

**Given:** The application is deployed on Heroku with an ephemeral filesystem.

**When:** The developer looks for solutions to store images.

**Then:** A reliable external image storage solution (such as Amazon S3, Google Cloud Storage, or Cloudinary) should be selected and implemented to ensure profile pictures persist across dyno restarts.

**Scenario 3:** Verify Configuration for External Storage

**Given:** An external storage solution is implemented.

**When:** The application handles profile picture uploads.

**Then:** The images should be correctly stored in the external service, and their links should be properly accessible in the application.

**Feature:** User Fitness Profile Creation and Management

As a user,

I want to create a fitness profile that includes my fitness-related information and preferences,

So that I can connect with workout partners and find activities that suit my interests and availability.

**Scenario 1:** Create a Fitness Profile

**Given:** The user is logged into the application.

**When:** The user navigates to the fitness profile creation page.

**Then:** The user should be able to enter personal information such as age, fitness goals, and experience level.

**Scenario 2:** Specify Preferences for different activities, partner age groups, locations, timings, etc.

**Given:** The user is creating their fitness profile.

**When:** The user selects their preferred activities (e.g., running, cycling, yoga), location, timings, specific age group of partners, etc.

**Then:** The selected preferences should be saved as part of their fitness profile.

**Scenario 3:** Review and Edit Fitness Profile

**Given:** The user has created their fitness profile.

**When:** The user navigates to their profile page.

**Then:** The user should be able to view their fitness information and preferences, and have the option to edit them as needed.

**Feature:** Revamped UI Design for Buddy Finding App

As a UI/UX designer,

I want to create a fresh and engaging UI for the buddy-finding app,

So that users have an enjoyable and intuitive experience while connecting with workout partners.

**Scenario 1:** Design new UI elements with refined styling and vibrant color scheme.

**Given:** The app's UI is being updated,

**When:** A new color scheme is implemented with refined navigation and styling elements,

**Then:** The UI should be visually appealing, responsive and user-friendly

**Scenario 2: Gather User Feedback**

**Given:** The new UI has been implemented,

**When:** Users interact with the app,

**Then:** Feedback should be collected to identify areas for improvement.

**Feature:** Implement of mock testing framework for Authentication Modules

As a developer,

I want to set up tests for third-party authentication and user session management,

So that I can validate authentication flows without relying on external services during testing.

**Scenario 1: Mock Third-Party Authentication for Testing**

**Given:** The user triggers a third-party authentication process (e.g., "Login with Google").

**When:** The authentication system is in test mode.

**Then:** A mock response should be used to simulate successful authentication.

**Scenario 2: Validate User Authentication in Tests**

**Given:** The application uses an authentication framework (e.g., Devise) for user sessions.

**When:** Unit tests are executed for actions requiring user login.

**Then:** A test helper should simulate a logged-in user without invoking the actual authentication process.

**Scenario 3: Handle Failed Authentication in Tests**

**Given:** A third-party authentication process is initiated.

**When:** A mock failure response is returned (e.g., invalid credentials).

**Then:** The application should correctly handle the failure and provide appropriate feedback in the test environment.

**Scenario 4: Test User Session Persistence for Authenticated Users**

**Given:** A user is authenticated via a third-party service or internal authentication.

**When:** The user navigates to restricted areas of the application.

**Then:** The user's session should persist, allowing access to those areas.

**Feature:** Test Integration for Dashboard and User Profile Management

As a developer,

I want to integrate the mock testing framework for dashboard and profile management pages,

So that I can validate user access and functionality in these areas without depending on live authentication during testing.

**Scenario 1:** Test Access to Dashboard and User profile management page for Authenticated Users

**Given:** A user is authenticated via a third-party service or internal authentication.

**When:** The user tries to access the dashboard and the User profile management page.

**Then:** The mock testing framework should validate that the user has access and display the dashboard and the user profile management page content.

**Scenario 2:** Test Access Denial for Unauthenticated Users on Dashboard and User profile management page

**Given:** A user is not logged in or has failed authentication.

**When:** The user tries to access the dashboard and User profile management page.

**Then:** The application should prevent access and redirect the user to the login page, as simulated in the test environment.

**Scenario 3:** Test Profile Update Functionality

**Given:** A user is authenticated and accessing the profile management page.

**When:** The user updates profile information.

**Then:** The mock testing framework should simulate a successful update of the user's profile and reflect the changes.

**Scenario 4:** Test Profile Button Navigation from Dashboard to Profile Management

**Given:** A user is authenticated and viewing the dashboard.

**When:** The user clicks the profile button on the dashboard.

**Then:** The mock testing framework should simulate a successful redirection to the profile management page.

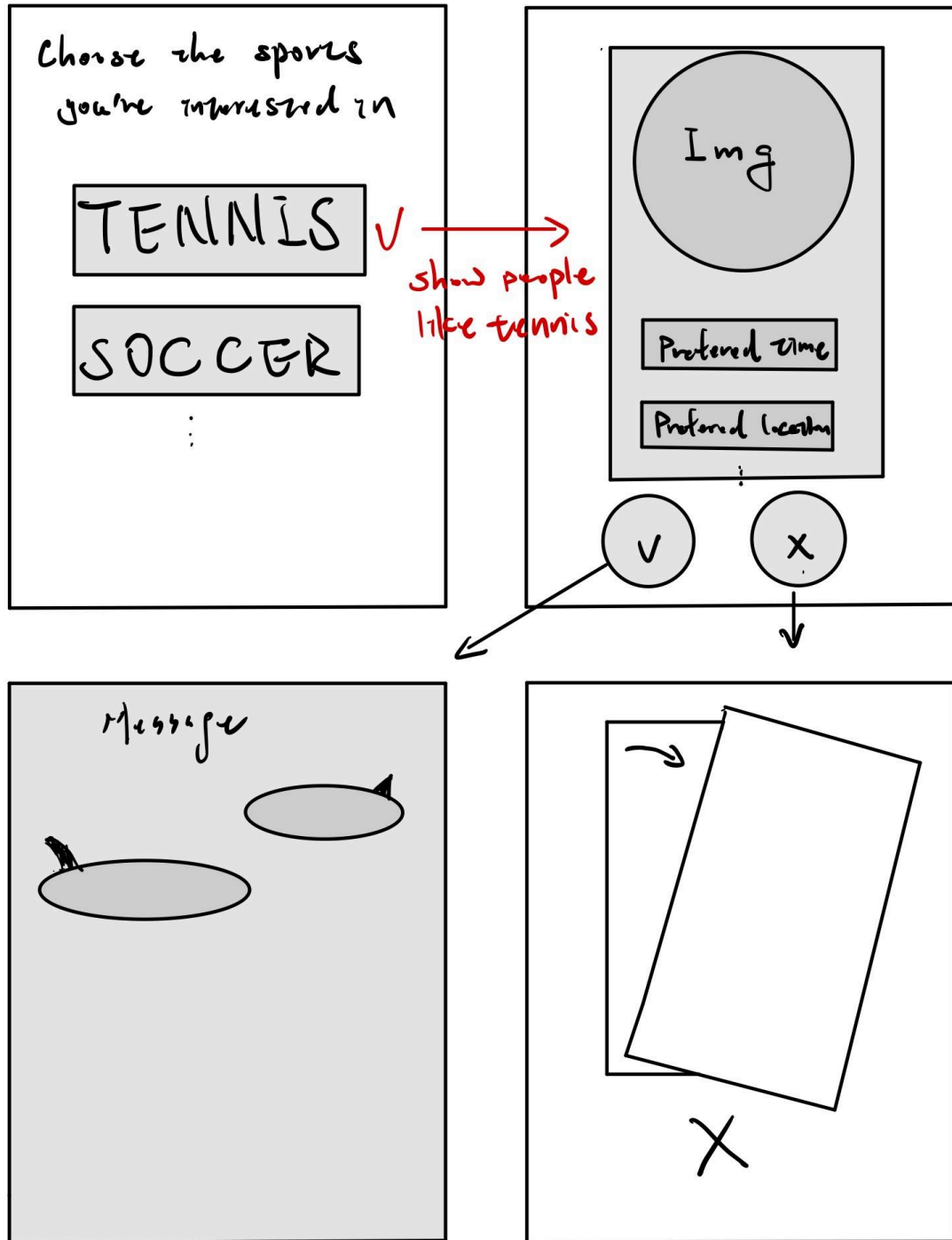
**Scenario 5:** Test Profile Information Display on Profile Management Page

**Given:** A user is authenticated and navigates to the profile management page.

**When:** The user views their profile on the profile management page.

**Then:** The mock testing framework should verify that the profile management page displays the user's name, age, and other relevant information.

## User interface: Lo-fi UI mockups and storyboards





# fitness profile management

### Fitness Goals

weight loss ⊖

muscle gain ⊖

⊕

---

### Workout type

Tennis ⊖

fitness ⊖

⊕

---

### Buddy Preference


Gender 


▼


Age range 


20

30









**Which stories were pulled into the Sprint? To whom is each story assigned?**

Stories	Points	Assigned To
Mobile Responsive User Interface	3	Mrunmay Deshmukh
UI Template for Matching with Gym Buddies	4	Barry Liu Kuan-Ru Huang
Fix User Profile Picture Visibility Bug in Production	3	Kushal Lahoti Yash Phatak
User Fitness Profile Creation and Management	4	Wei-Chien Cheng ChuanHsin Wang
Revamped UI Design for Buddy Finding App	4	Kushal Lahoti Yash Phatak
Implement of mock testing framework for Authentication Modules	3	Barry Liu Kuan-Ru Huang
Test Integration for Dashboard and User Profile Management	3	Wei-Chien Cheng ChuanHsin Wang

**What are the tasks and their time estimates?**

- Mobile Responsive User Interface - 1 Week
- UI Template for Matching with Gym Buddies - 1 Week
- Fix User Profile Picture Visibility Bug in Production - 4 Days
- User Fitness Profile Creation and Management - 1 Week
- Revamped UI Design for Buddy Finding App - 1 Week
- Implement of mock testing framework for Authentication Modules - 3 Days
- Test Integration for Dashboard and User Profile Management - 4 Days

**Links to our GitHub Repo, Pivotal Tracker, and Slack workspace under**

- **GitHub Repo** - <https://github.com/tamu-edu-students/jimmy-gym-buddy-finder>
- **Pivotal Tracker** - <https://www.pivotaltracker.com/n/projects/2721606>
- **Slack Workspace** - <https://app.slack.com/client/T07P2NT2ZM1/C07P00FFRGD>
- **Code Climate** - <https://codeclimate.com/github/tamu-edu-students/jimmy-gym-buddy-finder>