CS3354 Software Engineering
Final Project Deliverable 1

Gym Rats

Poorna Bharanikumar, Yvonne Hsiao, Mollik Rahman, Naufal Sajjad, Muhilan Selvaa, Temi Badmus, Jacy Grossman

1. Final Project Draft Description

Group members: Poorna Bharanikumar, Yvonne Hsiao, Mollik Rahman, Naufal Sajjad, Muhilan Selvaa, Temi Badmus, Jacy Grossman

What are we doing: Fitness App/website

Description:
Many people do workouts that aren't properly balanced leading to injury and not working out the whole body. They also find it hard to be consistent when traveling or following online workout plans that use equipment they don't access too. Therefore, an app that makes custom workouts based on users' goals and what equipment they have access to. The app continuously changes and adapts based on users progress by tracking your workouts.

Motivation:
Our motivation as fitness aficionados, an app that provides a variety of utility would be greatly appreciated, as such a comprehensive app does not exist on the market.

Task Delegation:
Poorna - I will do anything related to our backend including anything related to the actual software implemented in a project of this kind. Additionally I will help with the research and analysis of our product to ensure the data we collect is reflective of our goals with our app.

Naufal - I would like to work on UIUX and Frontend design, while helping with market research, requirement analysis, and ensuring the app meets certain standards.

Mollik - I will be doing anything related to the research needed to develop a fitness app that meets the suggested requirements. I can also help anywhere else that is needed in the development of this project.

Temi - I will be doing anything related to research and implementation of fitness, muscle groups, and workouts based on the user. I will also assist in backend design for the app.

Yvonne - I will be doing anything related to software demo (UI/UX) as well as research includes user research and competitive research.

Muhilan - I will be working on diagram creation as well as doing the J Unit unit testing for this app. I will also assist in cost analysis and planning.

Jacy - I would like to contribute to the front-end and/or back-end development, wherever additional personnel would be most helpful. I can also assist with requirement analysis.

Scholar Paper: Maybe, still figuring out commitment.

Feedback Received: Great project topic with a fringe benefit, as it will help with maintaining a healthy lifestyle. In the final report, please make sure to include comparison with similar applications -if any- and make sure that you differentiate your design from those and explicitly specify how. Please share this feedback with your group members. You are good to go. Have fun with the project and hope everyone enjoys the collaboration.

Addressing feedback: The feedback we received was to look into similar applications, compare them, and differentiate our design while explaining how it is different. In order to do this, we are looking into similar workout and diet apps such as Strong and Sworkit, in order to see what features they provide. We will then add new features or combine features from multiple apps in ways that have not been done before in order to create a more comprehensive app. Firstly, the Strong app only contains certain preset workouts/routines, which makes our app unique, since we are generating a workout for the user based on available equipment and preferences. Next, the Sworkit app is a really great app which provides a variety of plans, however, it only provides workout plans for at home workouts. This problem is solved in our app, which incorporates body weight and resistance training with weights into the workout. Lastly, many nutrition apps, such as MyFitnessPal, do not include workout plans, so our app encompasses workouts combined with nutritional information. Currently, there is not an app as comprehensive as ours in terms of overall health, as they either specialize in strength training, at home training, or nutrition, making our app a great addition to the category.

Title: The title of our project is Gym Rats. It is a mobile app that creates workout plans based on user preferences and available equipment.

2. Delegation of Tasks

Jacy - Software requirements (functional and non-functional), Architectural design pattern
Naufal - Title, Delegation of tasks, Addressing project feedback, Software process model, 1.3-inviting everyone to the github
Mollik - Title, Delegation of tasks, Addressing project feedback, Software process model, 1.5 commit project_scope file
Temi - Sequence diagram, Class diagram
Muhilan - Sequence diagram, Architectural design pattern
Poorna - Software requirements (functional and non-functional), Use case diagram. 1.4-first commit to the repository, readme file
Yvonne - Use case diagram, Sequence diagram

3. Software Process Model

The software process model employed in our app is the spiral model. The spiral model works well here because of its inherent flexibility and adaptability. The spiral model combines the iterative aspects of prototyping with the controlled nature of the waterfall model. The model is evolutionary, meaning there are a series of releases, like prototypes, throughout the process. This allows for changing requirements to be added and tested thoroughly, increasing the quality of the software. The requirements of our fitness app are subject to change based on user feedback, emerging trends, and technological advancements. The spiral model's iterative nature allows for continuous refinement and enhancement of the app throughout its lifecycle. It allows us to gradually incorporate new features, accommodate evolving user preferences, and address any issues. The model also emphasizes risk management, which is important for maintaining data security and user privacy in our app. By iterating through planning, design, build, and test phases while assessing risks, the spiral model ensures that our fitness app can evolve with the ever-changing health and fitness scene. Overall, the spiral model is the model that best fits our app because it offers great flexibility to changing requirements and plenty of risk management throughout the development process, which are important for mobile applications.

4. Software Requirements

Functional Requirements:

1. User Registration and Profile Management:

Users should be able to create an account and provide essential information such as age, gender, fitness level, and fitness goals.

Users should have the option to edit and update their profiles as needed.

2. Personalized Assessment:

The app should conduct an initial assessment of the user's fitness level, taking into account factors like BMI, medical conditions, and dietary restrictions.

3. Workout Customization:

Users should be able to select their preferred workout type (e.g., strength training, cardio, flexibility) and specify their target areas (e.g., upper body, lower body, core).

The app should recommend workouts tailored to the user's preferences and fitness goals.

4. Progress Tracking:

Users should be able to track their progress over time, including weight, measurements, and workout performance.

The app should use this data to adapt and adjust the workout plan accordingly.

5. AI Workout Recommendations:

The AI should suggest workout routines based on the user's goals, fitness level, and progress.

It should consider factors like exercise intensity, frequency, and duration.

6. Exercise Library:

The app should provide a comprehensive library of exercises with video demonstrations, instructions, and variations.

Users should be able to filter and search for exercises based on their preferences and equipment availability.

7. Nutritional Guidance:

Provide dietary recommendations and meal planning based on the user's goals (e.g., weight loss, muscle gain, maintenance).

Allow users to log their daily food intake for tracking purposes.

Non-Functional Requirements:

Product Requirements

Usability Requirement:

The software shall have user interface elements that are clear and explicit.

Efficiency Requirements

Performance Requirement:

The software shall have a two second maximum response time.

The software shall support a minimum of 250,000 concurrent users.

Space Requirement:

The software shall take up no more than 150MB on mobile devices.

Dependability Requirement:

The server shall be available to clients 99.5% of the time.

Security Requirement:

The software shall not expose personally identifiable information to unauthorized parties.

Organizational Requirements

Environmental Requirement:

The software shall maintain a unique directory to avoid interference with other software.

Operational Requirement:

The client shall run on Android.

The client shall run on iOS.

The client shall run on WearOS.

Development Requirement:

Development of the software shall take no more than two years.

External Requirements

Regulatory Requirement:

Parts of the software that handle transactions shall be PCI compliant.

Ethical Requirement:

The software shall not recommend unsafe exercises.

Legislative Requirements

Accounting Requirement:

The software shall be compliant with the Sarbanes-Oxley Act.

Safety/Security Requirement:

The software shall not offer features subject to legislative safety requirements.

5. Use Case Diagrams

1. End User
    a. Edit Dietary restriction
    b. Edit Medical Conditions
    c. Create Weekly Availability
    d. Create Body Goals
    e. Edit Current Weight
    f. Edit Current Height
    g. Edit Available Equipment
2. Dietitian/nutritionist
    a. Provide Dietary Advice
    b. Create Meal Plans
    c. Review Food Logs
    d. Approve Dietary Content
3. Trainer
    a. Design Workouts
    b. Post Training Videos
    c. Host Live Classes
    d. Answer Fitness Queries
    e. Track User Progress
4. System Administrator
    a. Manage All Users
    b. Manage All Trainers
    c. Manage All Dietitians
    d. Edit System Updates
    e. Create AI Learning Model Updates

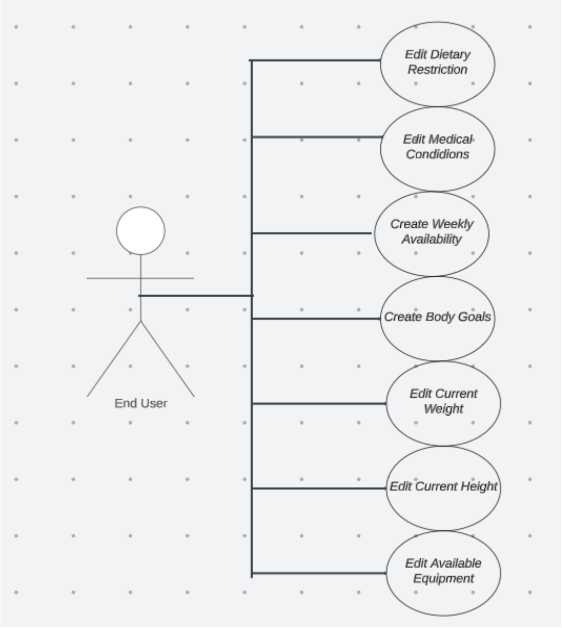

Figure 5.1 Use case diagram for end user

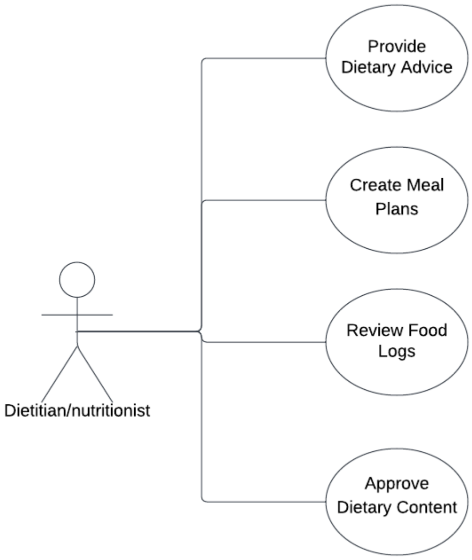

Figure 5.2 Use case diagram for dietitian/nutritionist

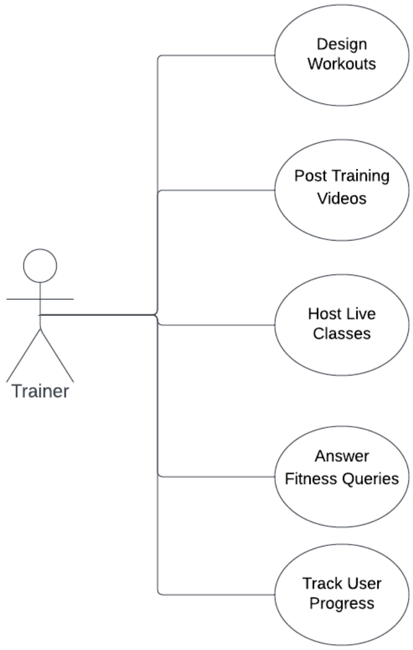

Figure 5.3 Use case diagram for trainer

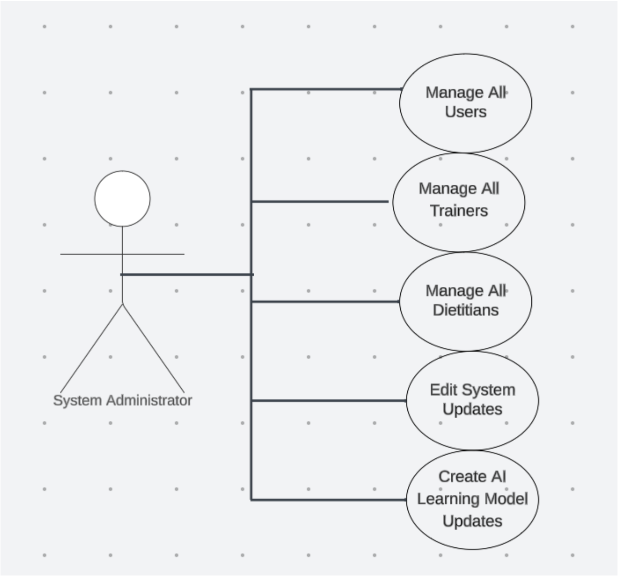

Figure 5.4 Use case diagram for System Administrator

6. Sequence diagrams

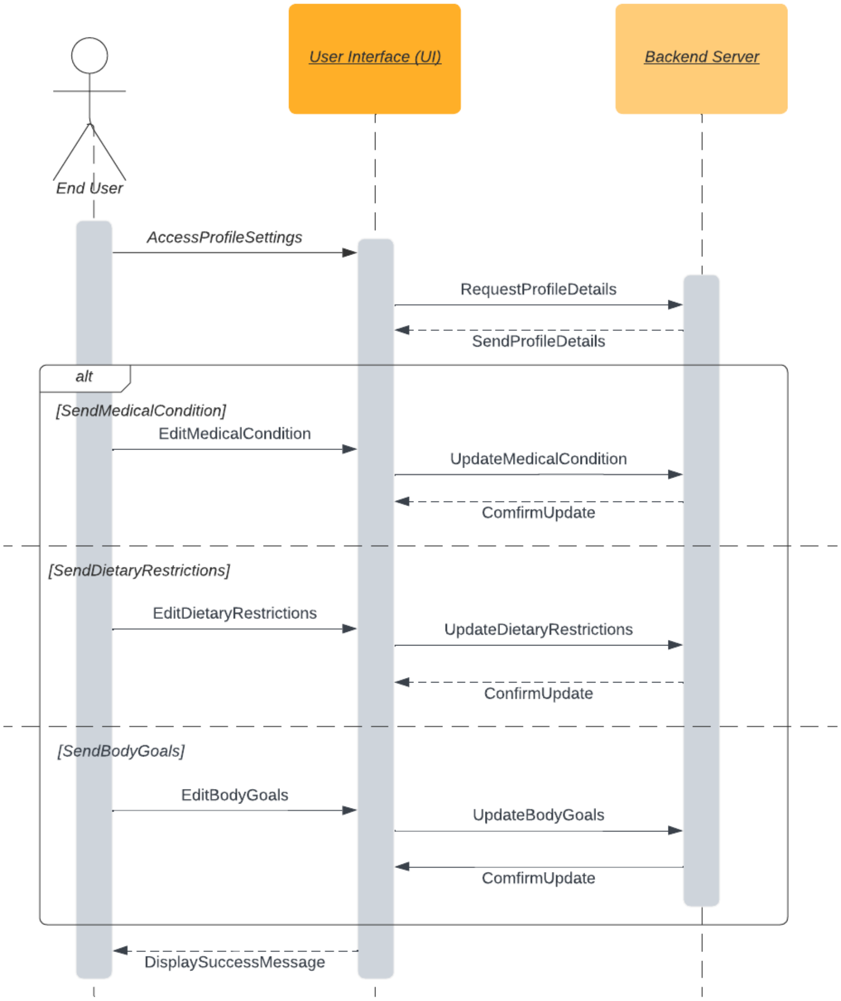

Figure 6.1 Sequence diagram for end user role

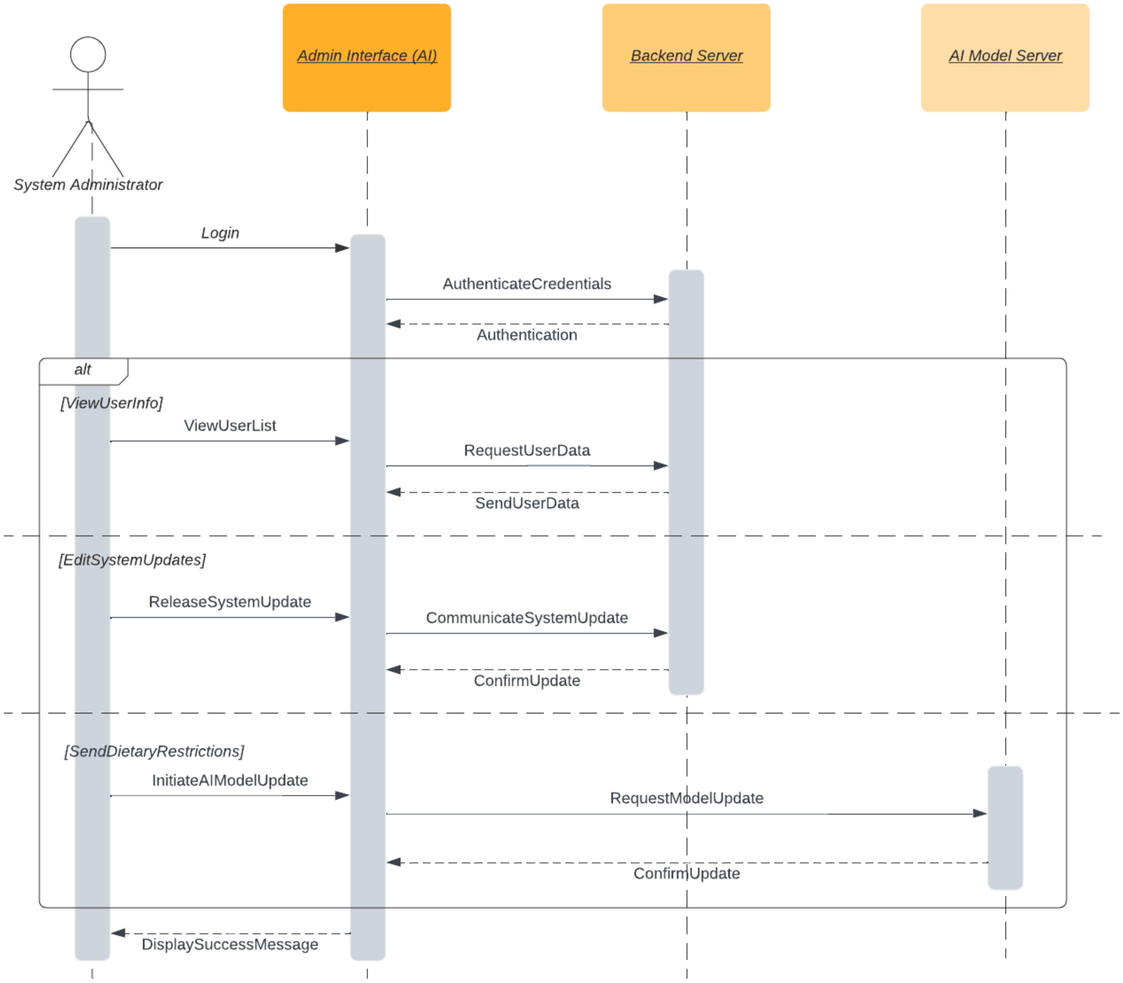

Figure 6.2 Sequence diagram for system administrator role

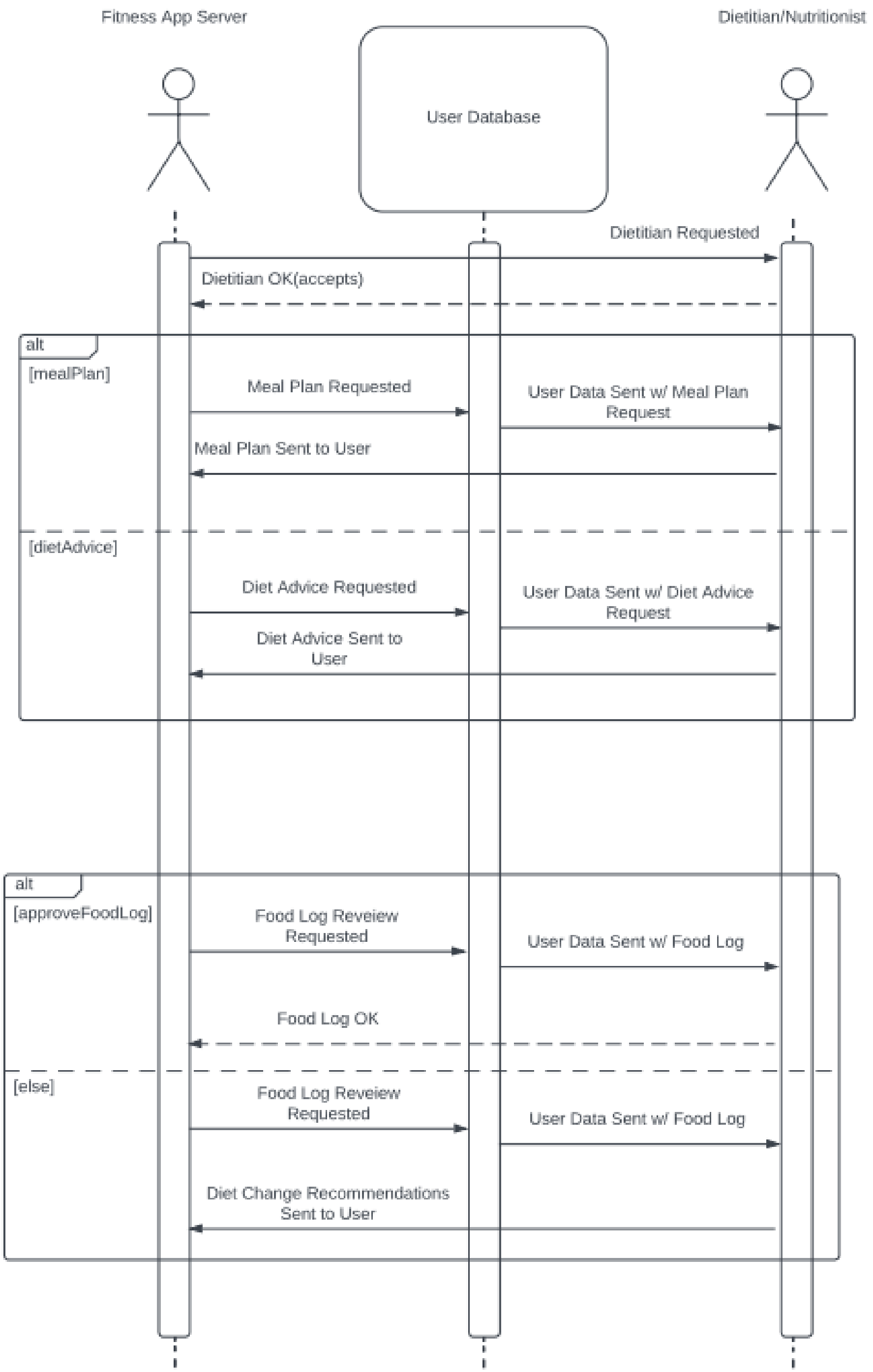

Figure 6.3 Sequence diagram for dietitian role

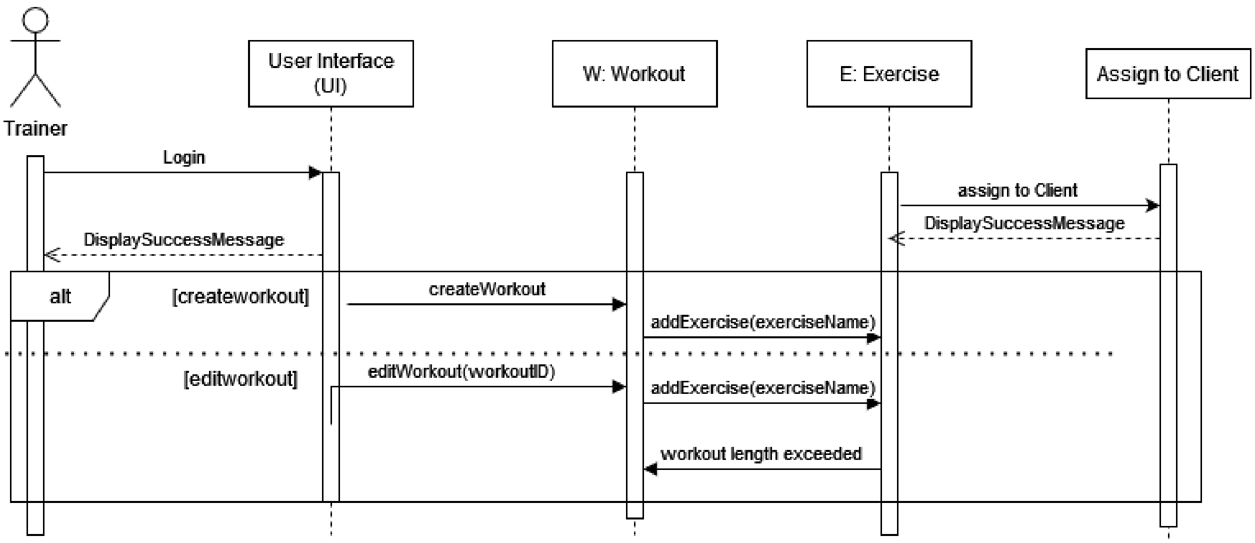

Figure 6.4 Sequence diagram for trainer role

7. UML Class Diagram

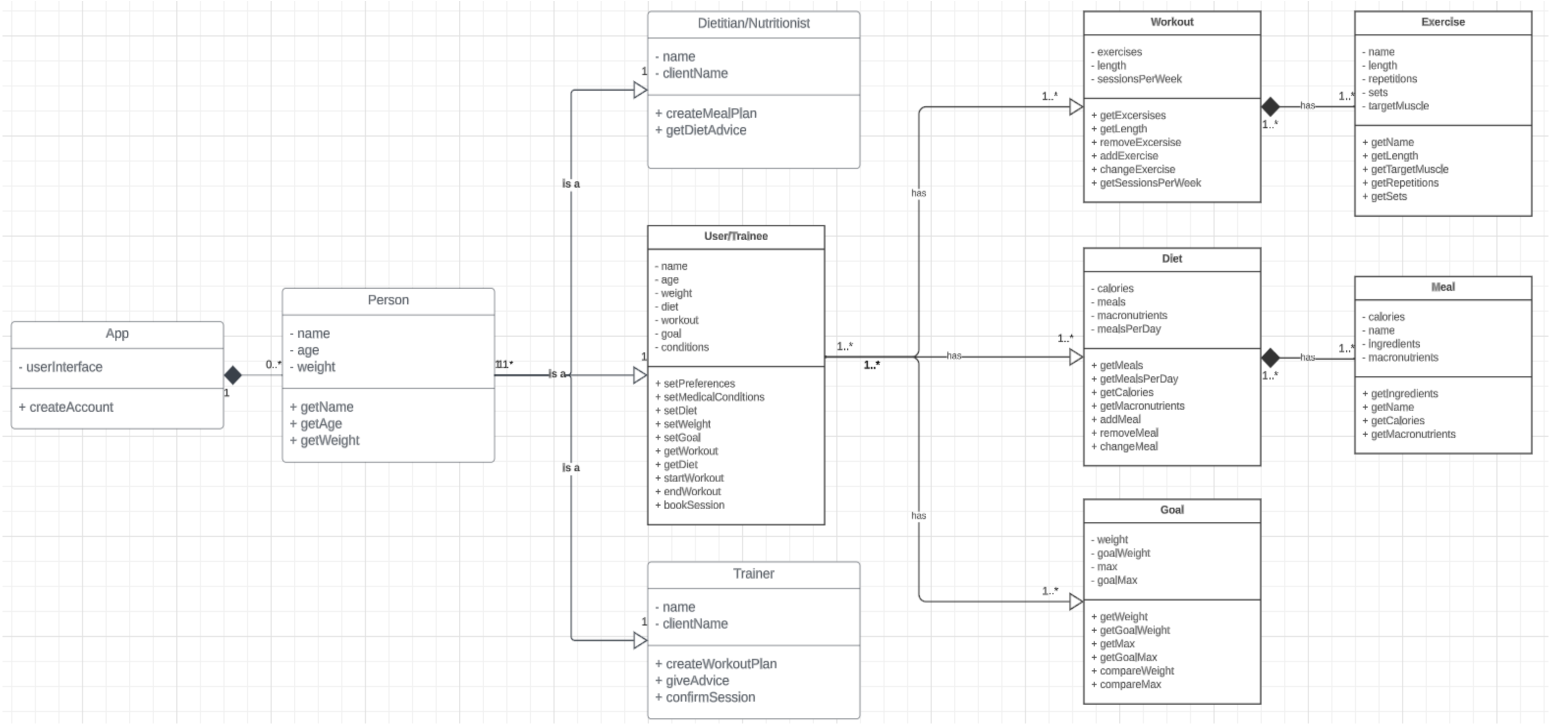

Figure 7.1 UML class diagram of the app

Link:
https://lucid.app/lucidchart/242f8d02-21a0-4140-8f90-69635128cf09/edit?viewport_loc=-2304%2C-1404%2C3025%2C1460%2C0_0&invitationId=inv_1f263d46-e7b0-400d-b08c-af79d8f86176

7. *Architectural Design*

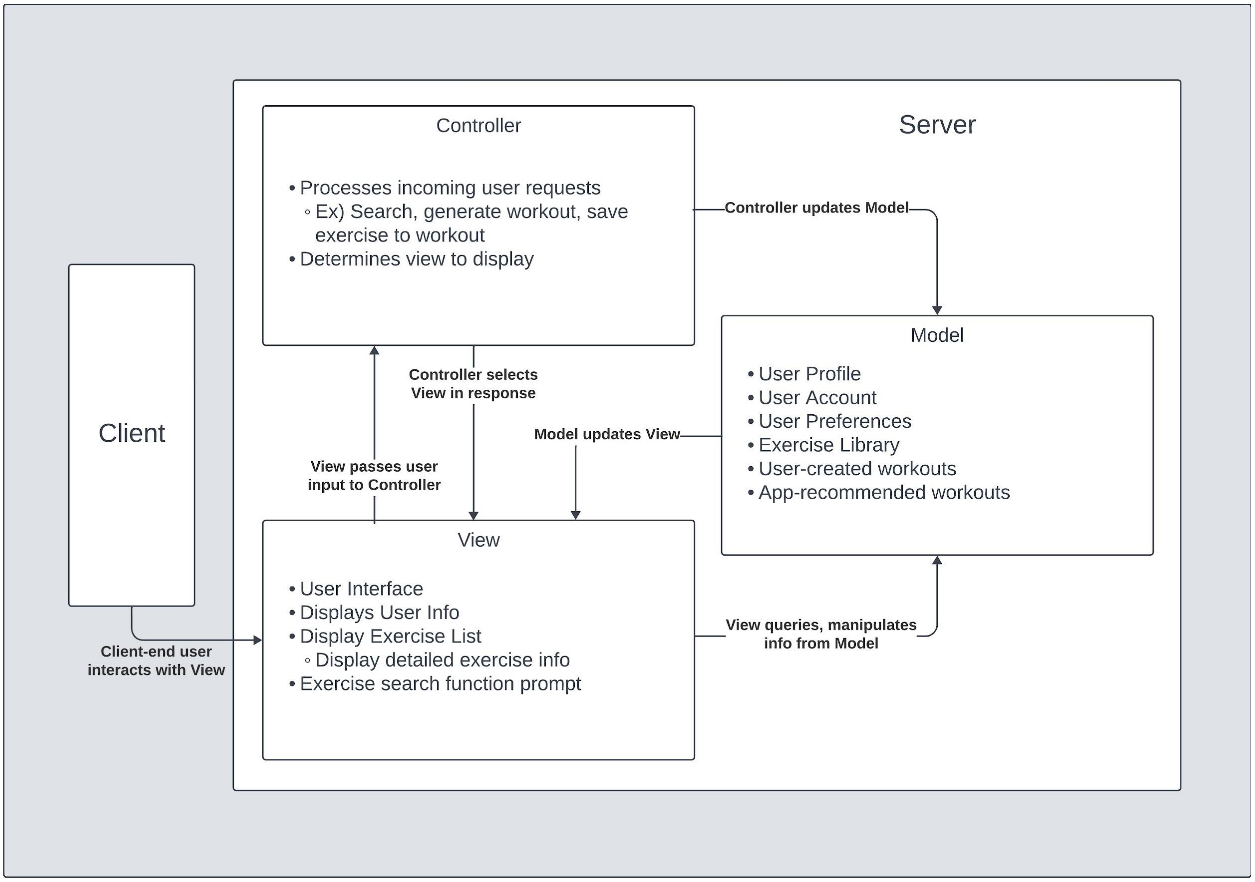
Server
Controller
• Processes incoming user requests
◦ Ex) Search, generate workout, save exercise to workout
• Determines view to display
Controller updates Model
Model
• User Profile
• User Account
• User Preferences
• Exercise Library
• User-created workouts
• App-recommended workouts
Controller selects View in response
Model updates View
View passes user input to Controller
Client
View
• User Interface
• Displays User Info
• Display Exercise List
◦ Display detailed exercise info
• Exercise search function prompt
View queries, manipulates info from Model
Client-end user interacts with View

Figure 7.1 Architectural diagram of the app using MVC pattern

The Model-View-Controller pattern is the best fit for our application because it allows us to process and show the user's workout data in various ways easily. The main function of our app is to process the user's workout data to generate optimal workouts. The model does the processing and generation, the controller decides how to display workouts and suggestions and takes in user customization, and the view displays the workout and all the associated details.

8. Github Repo:

https://github.com/NSPolloWaffle/3354-GymRats