**Software Requirements and Design Document**

**For**

**Group 2**

Version 1.0

**Authors**:

Sydney McGinnis

Sophie Pavia

Henry Gilbert

Alejandro Serrano

Jack Garthwaite

# 

# **1.** **Overview (5 points) - Sophie**

*Give a general overview of the system in 1-2 paragraphs (similar to the one in the project proposal).*

*Our project is a fitness app to track workouts and weights. The user can input their workout and have the app give basic suggestions on progression. Currently our program has a method for users to input their workouts and log exercises. We also have a workout page that lists different workouts. Our next step is to implement a database to store these workouts etc. Our main layout of the app is Workouts (list of workouts for user to choose from), Home (has button to profile page and sign out button), and More (includes buttons to calculators for BMI, inserting workout, inserting exercise, etc). The feature to sign in with google has also been implemented since our last submission. We plan to add on to our three landing pages from here.*

# **2.** **Functional Requirements (10 points)**

*List the* ***functional requirements*** *in sentences identified by numbers and for each requirement state if it is of high, medium, or low priority. Each functional requirement is something that the system shall do. Include all the details required such that there can be no misinterpretations of the requirements when read. Be very specific about what the system needs to do (not how, just what). You may provide a brief design rationale for any requirement which you feel requires explanation for how and/or why the requirement was derived.*

1. Ability to input weights and workout sets. (high)
2. Ability to log data such as workouts and user provided data. (high)
3. Track user’s progress using a graph. (low)
4. Login with Google (medium)
5. Motivate the user while working out through motivational quotes. (low)
6. Create custom workouts. (medium)
7. Make users new to fitness feel comfortable. (high)
8. iPad OS support (medium)
9. Calculate the user’s BMI, calories to be burned, and one max reps. (medium)
10. Ability for user to Upload Profile Picture on profile page (medium)
11. Ability to store user information in database (high)
12. Ability to pool workouts from database (medium)

# **3.** **Non-functional Requirements (10 points)**

*List the* ***non-functional requirements*** *of the system (any requirement referring to a property of the system, such as security, safety, software quality, performance, reliability, etc.) You may provide a brief rationale for any requirement which you feel requires explanation as to how and/or why the requirement was derived.*

1. The application shall protect personal user data by safely storing all workouts and personal data such as height, weight, BMI, goals, and workout progress (high)
2. Include a user friendly and interactive interface (new users won’t feel intimidated). (high)
3. Supports mobile phone data so the user has the ability to exercise anywhere and anytime that they would like (medium)
4. Progress can be shared/accessed across different platforms as long as the user as long as the user has an account. (iOS and iPad OS) (medium)
5. The application should take less than 5 seconds to switch between screens when the user presses a button. (low)
6. Reliable data management system meaning that the application shall keep a user logged in as long as the application remains downloaded on their iPhone/iPad (high)
7. The application shall remain running and able to use as long as the app is open on the user’s device. (high)
8. The login with Google feature shall safely store the user’s information
9. The profile picture shall be uploaded from user camera roll with permission message from user (medium)
10. Database API should be secure and not accessible by general public (high)
11. User will be able to quickly add exercises to their workout. (High)
12. Application will securely store user information such as workouts and profile information (high)

# **4.** **Use Case Diagram (10 points)**

*This section presents the* ***use case diagram*** *and the* ***textual descriptions*** *of the use cases for the system under development. The use case diagram should contain all the use cases and relationships between them needed to describe the functionality to be developed. If you discover new use cases between two increments, update the diagram for your future increments.*

***Textual descriptions of use cases****: For the first increment, the textual descriptions for the use cases are not required. However, the textual descriptions for all use cases discovered for your system are required for the second and third iterations.*

User:

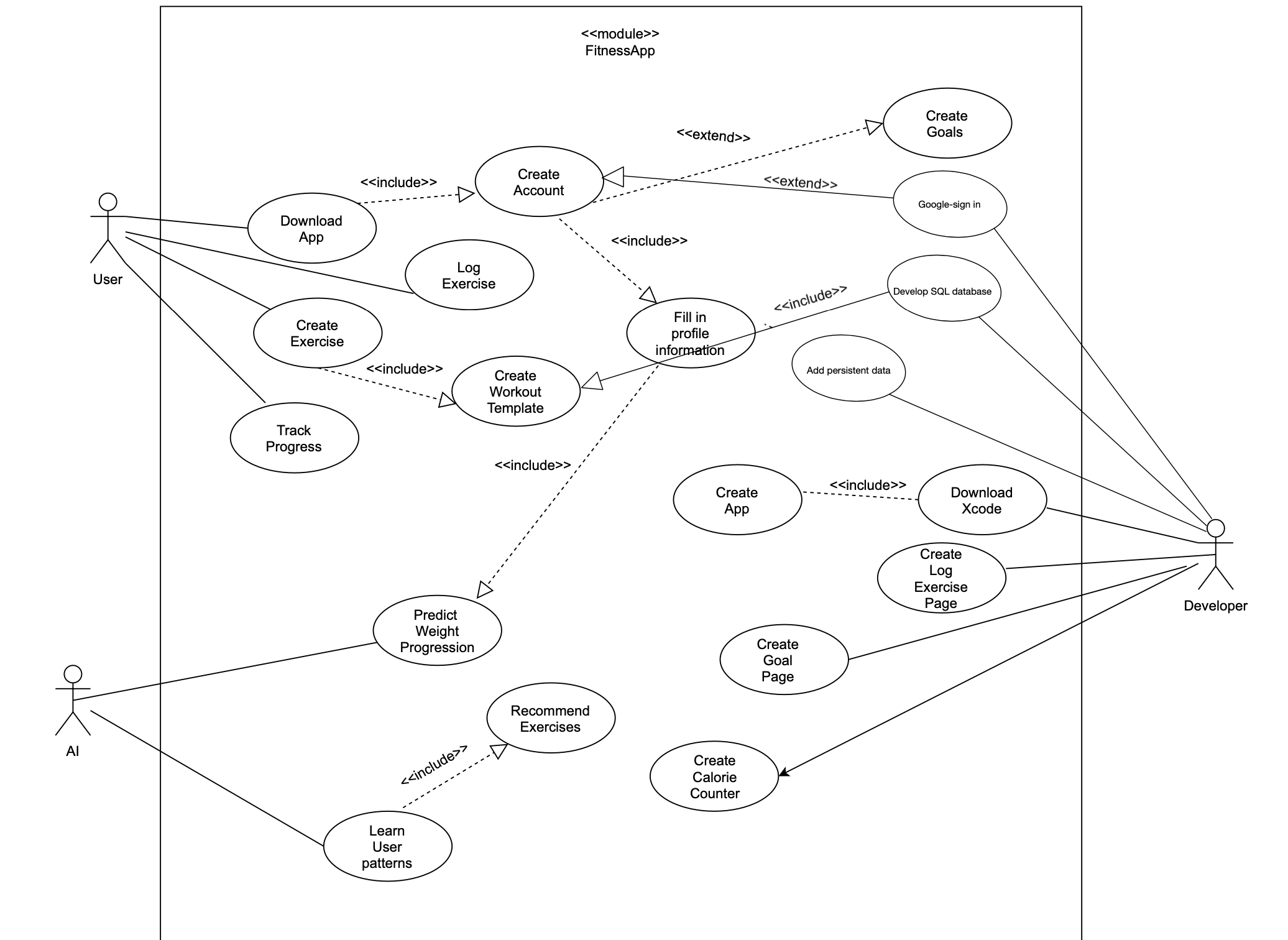
* Download App
  + To interact the user must download from the respective store for their device
* Create Goals
  + User can select goals such as losing weight or trying to build muscle
* Create Exercise
  + The user will be able to insert their own exercises and add them to their own database
* Track Progress
  + Mentioned as the graph in the video the user can insert how they feel after their workout to be able to look back in their progress
* Create Workout Template
  + The user can look through the predefined exercise list or add their own and create a “playlist” of exercises that they can workout through.
* Log Exercise
  + The user will log their exercise and the track progress will reference this data.
* Create Account
  + User has the option to create an account to store the data such as progress and exercises.
* Profile Information
  + User can add a profile picture and have their preferences (from the start pages) saved.

AI:

* Predict Weight Progression
  + Based on the users habits and profile information the AI can help predict how a user may meet their goals
* Recommend Exercises
  + Based on user habits exercises can be recommended
* Learn User Patterns
  + Referencing the exercise log it will learn how the user likes to workout

Developer:

* Create Calorie Counter
  + Counter page that helps people learn how to achieve their goals
* Create Goal Page
  + Page for user to select goals
* Create App/Download Xcode
  + Developer will need to download xcode to create their app
* Create Log Exercise Page
  + Developer creates page for the user to log exercises
* Add Persistent Data
  + Persistent data allows the data to stay in the app ever after a user may close and reset it
* Develop SQL database
  + SQL database allows the user to add their exercises and will keep track of the pre listed exercises then link it within the app
* Google sign-in
  + Developer gives the user the option to sign in using the “sign in with Google” api
* Create BMI Calculator
  + Developer will create a page for helping a user calculate their BMI
* Create Profile Page
  + Developer creates page for user to select their profile picture

**

# **5.** **Class Diagram and/or Sequence Diagrams (15 points)**

*This section presents a high-level overview of the anticipated system architecture using a* ***class******diagram*** *and/or* ***sequence diagrams****.*

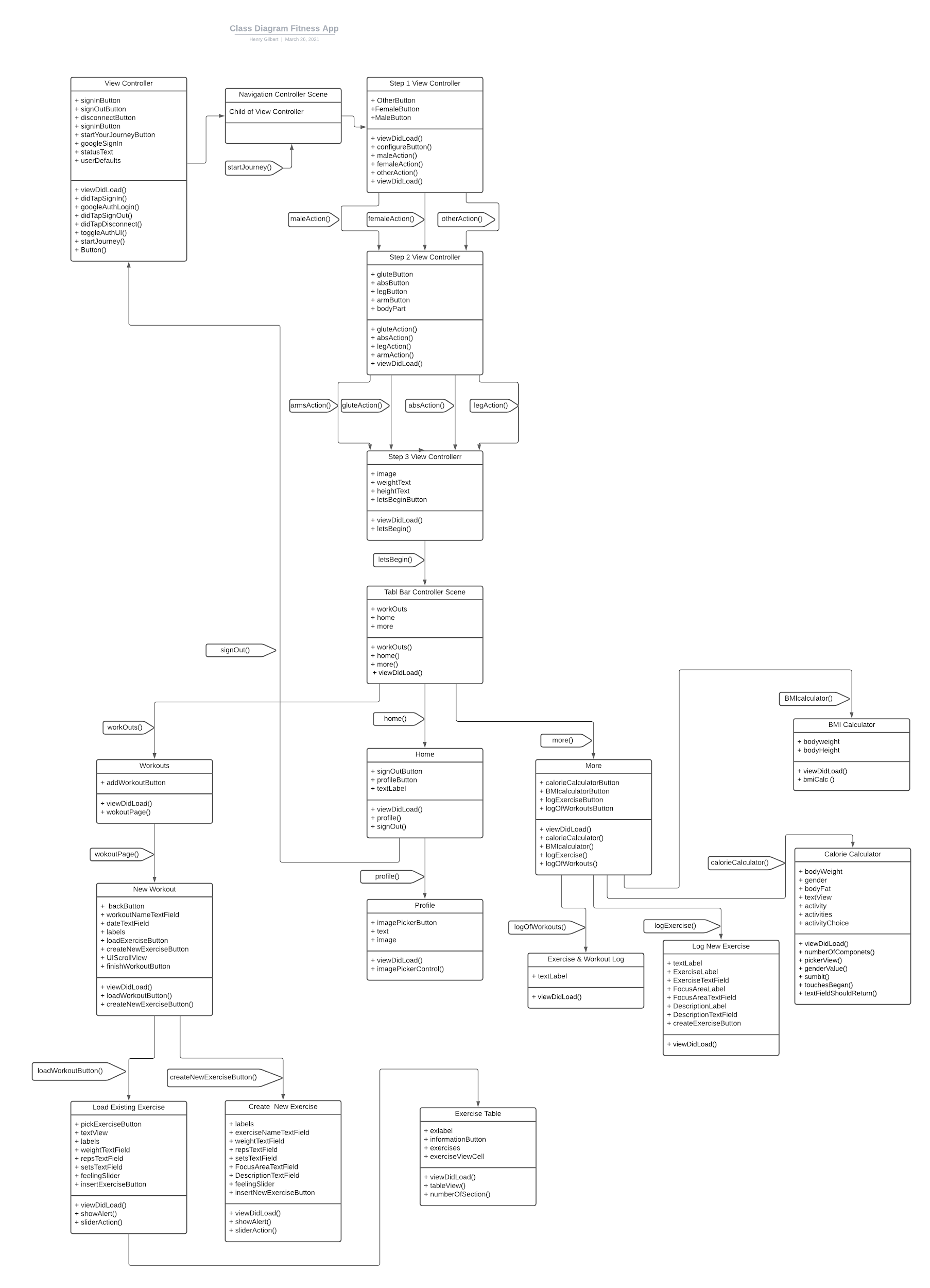
*If the main* ***paradigm*** *used in your project is* ***Object Oriented*** *(i.e., you have classes or something that acts similar to classes in your system), then draw the* ***Class Diagram******of the entire system and Sequence Diagrams for the three (3) most important use cases in your system.***

*If the main* ***paradigm*** *in your system is* ***not Object Oriented*** *(i.e., you* ***do not*** *have classes**or anything similar to classes in your system) then only draw* ***Sequence Diagrams****,* ***but for all the use cases of your system.*** *In this case, we will use a modified version of Sequence Diagrams, where instead of objects, the lifelines will represent the functions in the system involved in the action sequence.*

***Class Diagrams*** *show the* ***fundamental objects/classes*** *that must be modeled with the system to satisfy its requirements and* ***the relationships*** *between them. Each class rectangle on the diagram* ***must also include the attributes and the methods of the class*** *(they can be refined between increments). All the* ***relationships between classes and their multiplicity*** *must be shown on the class diagram.*

*A* ***Sequence Diagram*** *simply depicts* ***interaction******between objects*** *(or* ***functions -*** *in our case - for non-OOP systems) in a sequential order, i.e. the order in which these interactions take place. Sequence diagrams describe how and in what order the objects in a system function.*

*\*\*\*\*\* ZOOM IN TO READ \*\*\*\*\**

**

# **6.** **Operating Environment (5 points)**

*Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.*

The application is developed for the iOS platform for iPhones and iPads, running iOS 14 and above to ensure the user has the most up to date security protections. The operating system we chose is iOS because it is a popular platform, so users will have easy access to the app and ensure they are getting the best user experience. Xcode (the compiler) can be accessed across platforms so the user can have a choice of which device to use the app with. We have currently only tested on the iPhone 11 so as of right now that is what the application currently runs most efficiently on. It is to be determined if the app will correctly support other versions of the iPhone and iPad.

**7.** **Assumptions and Dependencies (5 points)**

*List any assumed factors (as opposed to known facts) that could affect the requirements stated in this document. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project.*

For this application we assumed that uploading pictures from the Photos app would be simple and the picture would constrain the size we program it to. However, as Sophie began integrating it, it became evident that the picture shape is very difficult to conform. With this original assumption in mind, we are not sure if the picture uploaded to the profile page will look how we would like it to. *Another assumption is that we can transition from our application (currently being tested on an iPhone 11) to any other Apple iPhone or iPad. While this is achievable, we are not sure how to format our app so it is portable to different devices yet. If we make changes to the app within an iPhone setting we need to make sure these changes translate to the correct formatting on an iPad too. We are also assuming that we can store all of the information the user inputs into a database.* The team is not exactly sure how this will work but Henry is working hard to figure it out. Another assumption we had was that adding a graph API to the app would be a simple. However, this is not the case and many of the tutorials Sydney watched were difficult to follow. Though we assumed we would have a graph tracking how the user felt after each workout, we might not be able to make this a reality due to difficulty and time constraints. We also assumed that we would let the user create their own workouts. As Alejandro and Jack began working on this, it became clear that having a predetermined list of workouts for the user was much simpler to implement. Therefore, we decided to let the user enter their own “custom” workout but we did not give them the option to “build” their own workout by selecting exercises and creating a custom workout each time.