Software Requirements and Design Document

For

Group 3

Version 1.0

Authors:
Andrew Perez-Napan
Cathy Yue
Damon Akins
Joao Valente
Nikolas Hernandez

1. Overview (5 points)

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

FitNest is a workout and nutrition app that is free for everyone. Within the app, users will be able to go through 4 tabs. The homePage tab acts as a quick view of current calories/water intake and goal calories/water intake and the workout for the day. The Workout tab will display workouts and allow the user to create a workout or choose from one of the pre-built plans provided to them. Whether designing a workout or choosing from a pre-built plan, users will have the freedom to customize their workouts and decide if they want it to reoccur on the said day every week.

Within the Nutrition tab, users will be able to input the food they've consumed and input specific measurements such as calories, serving size, carbs, fats, and proteins. Users will then be able to view the meals they've consumed. Lastly, in the Progress tab, users will be rewarded for reaching their goals, such as: Losing a certain amount of weight or hitting a new PR on an exercise. Their progress will be tracked so they can stay motivated and keep going.

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 an explanation for how and/or why the requirement was derived.*

1) The app will fit into four pages (high)
2) At the bottom of the app there is a navigation bar for each page (HomePage, Nutrition, Workout, Progress) (high)
3) HomePage will act as a quick view (high)
3.1) at the top-right of the Homepage, there will be a button for the settings page (medium)
3.2) At the top of HomePage current and goal tracking of caloric and water intake. (high)
3.3) Below that, HomePage will contain today's date and show the workout to perform today. (high)
3.4) At the bottom of HomePage, there will be buttons to increase or lower water intake (high)
4) Nutrition will keep track of all things food and water-based (high)
4.1) Will show current calories and goal calories (high)
4.2) Arrow next to current/goal calories will show current carbs, proteins, and fats eaten
4.3) Will allow the user to add a meal or increase water intake (high)
4.4) When inputting a meal, the user will be able to input: food, serving size, calories, carbs, fats, proteins, etc...
4.5) Meals already inputted will be displayed and show calories, when clicked on will display food that was eaten (high)
5) Workout tab will allow the user to plan workouts for the week (high)
5.1) will show a weekly calendar, where the user can view their workouts (high)
5.2) for each exercise the user will be able to put how many reps they completed for each set (high)
5.3) the user is able to create a personal plan or choose from pre-built plans (high)
5.4) when creating a personal plan user will be able to name the workout (high)
5.5) when creating a personal plan user will insert the exercise they want to do and include the sets, reps, and weight (high)
5.6) When choosing a pre-built plan, the user will decide the weight and day of the week they choose to workout on (high)
5.7) When choosing a pre-built plan, the user can decide if a workout will be recurring every week on the given day
6) Settings will display a profile, where the user can input: name, weight, activity level, height, gain/lose weight, exercise types (medium)

6.1) settings will provide options such as push notifications, determine default amount for water intake, add a goal, etc... (medium)
7) Progress page will reward the user for reaching certain goals (low)
7.1) At the top of the Progress tab will show what medal the user has (low)
7.2) Progress page will display achievements, examples: 10 pounds down, new bench press PR (low)
7.3) chart will display weight loss/gain over the weeks/months (low)
7.4) At the bottom of the Progress goals to accomplish will be listed. (low)

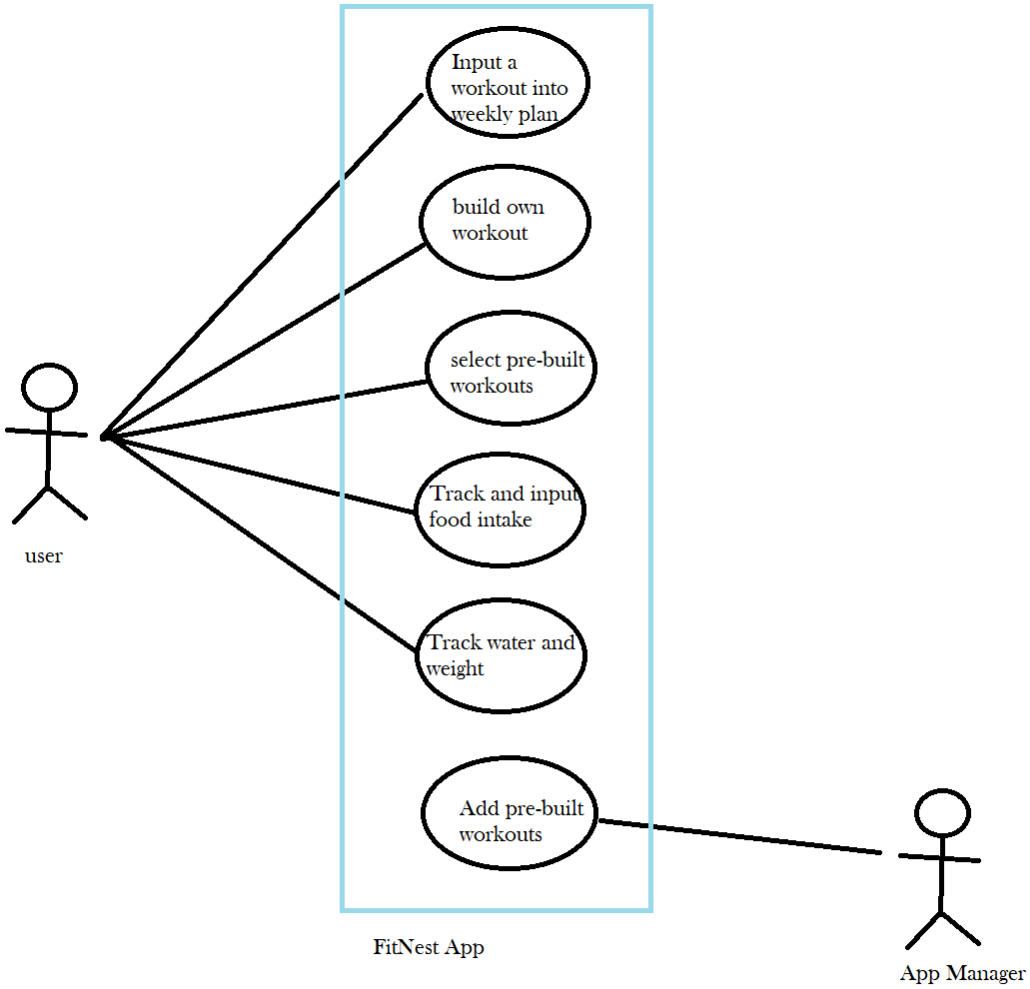
3. Non-functional Requirements (10 points)

*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) When opening the app it shouldn't take longer than 4 seconds for the app to fully load
2) App should be able to handle continuous data
3) App should be responsive to user and save state if interrupted by another app or call/text
4) App should be reliable
5) App should encrypt user data

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.*



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

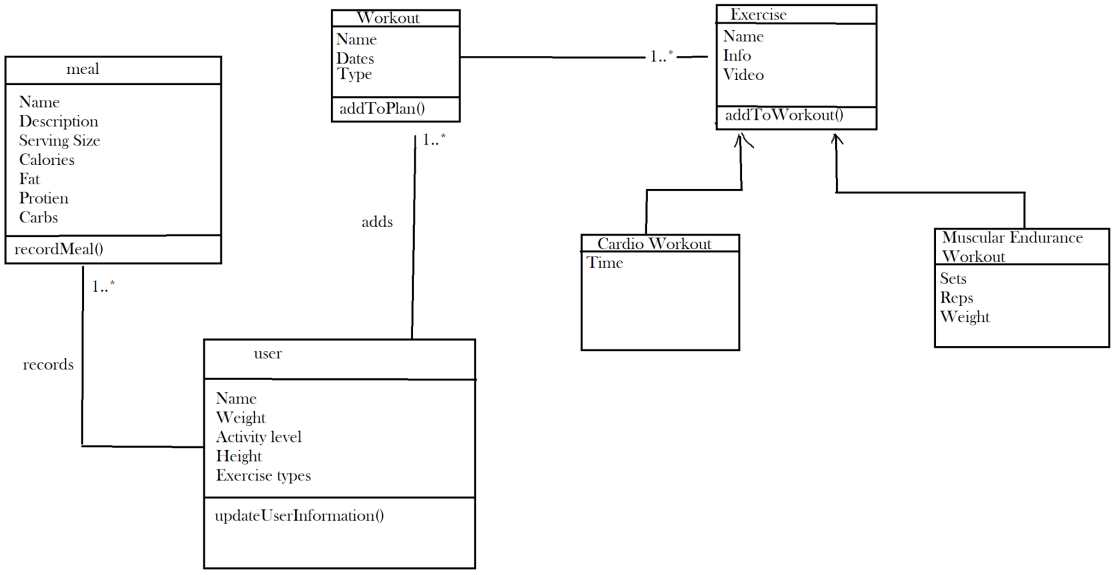
*This section presents a high-level overview of the anticipated system architecture using a **class diagram** and/or **sequence diagrams**.*

*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 the **interaction between objects** (or **functions -** in our case - for non-OOP systems) in 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.*



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 software will operate with the android mobile operating system. This allows the application to work on almost any cellular device or tablet running the Android operating system. We are running Oreo 8.1 which runs in 53% of android devices. The minimum API is 24 and target Api is 30.

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.

We decided to go ahead and use Back4app as our main database. We decided that it's limitations with not having as much documentation as other more popular databases is overshadowed by the user friendliness and ease of use of the platform. We are still working to get the Youtube API to work with our project, however we are closer than ever to get it to work. During development we realized some libraries and dependencies were deprecated, however Android offered alternatives which worked similarly; we will probably keep running into this problem, but learned that android does a good job at offering solutions.