**Software Implementation and Testing Document**

**For**

**Group 2**

Version 3.0

**Authors**:

Sydney McGinnis

Sophie Pavia

Henry Gilbert

Alejandro Serrano

Jack Garthwaite

# 

# **1.** **Programming Languages (5 points)**

Currently the majority of our project is built using Swift and in XCode. We chose this platform because it is the easiest way to create an iOS app and Swift is the primary language used for iOS app creation. We are also in the process of integrating a PHP API that connects us to a SQL database. This SQL database will hold the workouts and exercises created by the user. The changes to the database will be reflected ~5 minutes after the changes are made.

# **2.** **Platforms, APIs, Databases, and other technologies used (5 points)**

We currently use XCode as our primary platform. We also have a Sign In with Google API used in the login screen of the app. This API lets the user login if they have a Google account. Before integrating this into the project, we registered our app with Google so that we would have the proper Client ID to link the app and the API. Finally, we are also implementing a self-created API to connect xcode to a database. The API consists of two separate PHP pages that handle the dynamic insertion and displayal of the data. This database will allow users to save their workouts and exercises to the database, to look back on it at a later date. This is done to be able to track progress made in any fitness circumstance.

# **3.** **Execution-based Functional Testing (10 points)**

*Describe how/if you performed functional testing for your project (i.e., tested for the* ***functional requirements*** *listed in your RD).*

In this increment, a lot of our time was done with testing new features and trying new things. We took a unit testing approach where we each individually tested any new features we added to the app. Also, we were pushing and pulling out commits often, and consistently interacting with the app to fix any errors we came across. Also while doing unit testing, if our individual feature did not work correctly, we did not push it to GitHub until it was fully working. We made sure that each of the features we currently have implemented work as best as possible so the user has the best possible experience. We also conducted system testing to ensure that one part of the project did not interfere with data in another sector of the project. Specifically we made sure that by Creating New Workouts, they were correctly populated into the database.

# **4.** **Execution-based Non-Functional Testing (10 points)**

*Describe how/if you performed non-functional testing for your project (i.e., tested for the* ***non-functional requirements*** *listed in your RD).*

We tested the non-functional requirements by closing and opening the app and making sure the user data remained unchanged. We also made sure the database stores all of the user’s previous workouts even when the app is closed. Henry added a security check to not allow unautherized insertion into the database, this was tested and verified. We went through and made sure all of the buttons are linked to a specific view controller and that they switch to the next page correctly.

# **5.** **Non-Execution-based Testing (10 points)**

*Describe how/if you performed non-execution-based testing (such as code reviews/inspections/walkthroughs).*

Each week we met in person or through Zoom to go over the code we had written previously. As new changes were pushed to GitHub we fixed any code that gave us errors. For the most part, we would also go over our changes each week, and show our code to each other. Oftentimes, the code is simple enough to understand but when something complex is implemented, or someone doesn’t understand something, the person responsible for the task walks the others through what was done. One non-execution based test we did heavily was trying to implement charts. Each time Sydney pushed from her version with charts working, on everyone else’s version it showed that there were missing files in GitHub. As we were unfamiliar with charts and pods, we decided debugging this issue took too much time and we did not include it in our final version.