Software Implementation and Testing Document

For

Group 26

Version 1.0

Authors:
Kyla Henry
Brian Ho
Beck Boyd
Holly Lapini
Flora Charles

Red = increment 1, Blue = increment 2, green = increment 3

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

The languages we used were react native for the frontend of the application on expo and java for the backend also using AWS.

The languages we used were Swift for the front end then Python and Json for the backend. We wanted our initial application to be compatible with web, ios and android but soon found that would create problems. So, we decided to stick to the versatile language of Swift and create an IOS application.

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

We used Xcode to begin creating the frontend and UI/UX of the application and AWS for the backend connection and set up.

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

We used a simulator to test the functionality of the application. For example, when testing the navigation tabs, we created different presences of the pages to check that they would display.

Using the simulator, we began to test the interface of the apps; specifically making sure the buttons directed to the correct pages and that they loaded in the manner they were supposed to.

Finally, using the simulator, we ensured that the overall operations of the app were exactly how we wanted them. We had a little trouble connecting the backend in AWS to the frontend so we made sure that the correct data and information would load based on the users input.

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

Using the simulator, we tested the usability of the app. Specifically, we tested how well the app moved from one page to another. We also timed how quickly the app displayed the books information and pages and made necessary adjustments to ensure runtime doesn't take long if it does.

The maintainability of the app was tested using comments and code reviews to ensure that as we continued building the application and, in the future, if something needed to be changed or the code needed to be referenced, readability and quick comprehension was acquirable.

The portability of the app was made so that the features can easily be transferred into a web or android application to better connect with more users on different platforms. Ensuring that the application can easily be moved and replicated was part of the reason for consistent and basic variable names and comments throughout the code base.

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

We made sure that the readability of the code was consistent; using constant variable names and the different tabs followed the same structure when needed. We also had another person in the group that was working on a similar portion of the code inspecting the other persons code to insure they weren't missing anything.

For this increment, we had a technical review where we went through the documentation of the code to ensure that it aligned with what features we wanted to include and analysis of the code to ensure that we are not building technical debt.

For this increment we also held another code inspection and walkthrough to ensure that the code base was where we wanted it with the features, we are able to implement in the time allotted.