Test Plan Document

Version 1

for

Lock & Learn

Prepared by Team Lock & Learn

Leon Zhang (40175616)
William Chong (40176360)
Alice Chen (40176279)
Ryan Kim (40175423)
Georgia Pitic (40175034)
Julie Trinh (40175335)
Hao Yi Liu (40174210)
Anna Hsu (40178711)
Fatema Akther (40177866)

Instructor:	Dr. Rigby, Peter
Course:	SOEN 490
Date:	09/23/2023

Table of contents

1. Introduction

Lock & Learn is a blocking application using React Native and JavaScript frameworks. Thus, Jest is a great testing tool to execute unit and integration testing. For end-to-end testing, Detox is used for the web application and Appium is used for iOS and Android applications.

Purpose

The goal of this software testing document is to explain our testing system to our stakeholders and have a testing documentation reference for the team. It covers any details on units to be tested by answering the questions of what is tested, how it is tested and by whom is tested. Defining and tracking tests can be found in this document as well. Overall, this document serves the purpose of validating if our application meets the requirements set by our stakeholders.

Scope

The tests will cover all stories that are based on the requirements defined by our stakeholders. It will also touch on our code to verify the functionalities of methods and access to each page.

2. Unit Testing

Unit testing is a software development practice which consists of testing the functionality of a unit of the software in isolation to make sure it behaves as intended. The unit of software refers to the smallest testable part of the software: a class, method, or object. This ensures that each unit is robust and reliable and meets its specific requirements.

Jest is used to test unit testing.

3. Integration Testing

Integration testing is a software testing technique where components' behavior is checked when they are integrated together. This testing technique has the purpose of making sure that different parts of our system can be integrated together and that there are no issues when integrating these components together.

Jest is used to test integration testing.

4. End-to-end Testing

End-to-end testing is a software testing technique that verifies that all components integrated together are able to run when a real-world scenario is simulated. The goal of this testing technique is to simulate the user experience of our system from start to finish and make sure the system runs without any issues. This will allow us to find dependencies within our system and validate the system.

Since we are developing Android, iOS and desktop applications, several tools such as Jest and Detox are used to test the end-to-end testing.

5. Acceptance Testing

Acceptance testing is an important part of the development life cycle of any application. This testing has the purpose of validating the requirements, assuring the quality of our product, satisfying our customer's needs, confirming whether the component being developed is ready, and more.

This can be done by adding acceptance criteria to our user stories and making sure that all the acceptance criteria are completed before the issue is marked as closed.

The acceptance criteria can be revised and decided by multiple people on the team, and can even be shared with the stakeholders and product owners. This will allow the whole team to have a written understanding of what needs to be done for each component and will avoid any misunderstandings.

Functional and Non-functional can both be added to the acceptance criteria of a user story, and the user story.

6. Usability Testing

Usability testing is a software testing practice where users are prompted to use the system and are being observed in doing so. This would allow the developers to better understand the user experience, the struggles our users could face when using our system, and things we could ameliorate to make our system more user-friendly.

User feedback is the primary source of data in usability testing and can be used to our advantage. Users would be prompted to think out loud when using our system so we can understand the user's decision-making process and thoughts while using our app.

Task-based testing can also be done as part of usability testing. This would consist of asking the user to complete a task and observing how long it takes and how complicated it is for the user to do that task. User satisfaction can also be measured by asking the user to rate their experience using our system.

Meeting Stakeholder Requirements

We can also see if we are meeting our stakeholder's requirements with usability testing simply by asking our stakeholders to use the system. We can take their feedback and make the necessary changes to our system to make sure that everything is meeting their requirements.

Meeting Target Consumer Expectations

Usability testing can also be used to make sure that we are meeting our target consumers' expectations. This can be done by asking some users within our target audience group to use the application and observing them while they are doing so. We can also ask for their feedback afterwards, and ask them to rate their experience.

7. Testing criteria for Iteration 1 & 2

In the first iteration, we are focusing on two main features: Sign In/ Sign Up and Uploading documents. Here are the testing criteria for these main features:

Sign Up:

1. The following text boxes should appear on the page: Email, First Name, Last Name, Password, Confirm Password.
2. Also, the "Birth Date" drop-downs should appear and the "Select your account type" radio button choices ("parent" and "teacher") should appear.
3. Users can enter data in text boxes, and select data from the drop-down and radio button.
4. Once data is entered throughout the page, the "Sign Up" button, which should also appear on the page, should be clicked.
5. Once it is clicked, the account is created and the user will appear in the database of the account.

The User will also be redirected to a confirmation page.

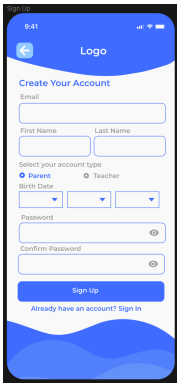

Figure 1: Visual representation of Sign Up page

Sign In:

1. The following text boxes should appear: “username or email”, and “password”.
2. Once the user enters text in both of these text boxes, he should click on the “Login” button which should also be on the page.
3. Then, if the user’s information is correct, he will be logged in and taken to a different system page.

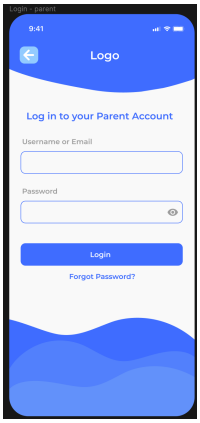

Figure 2: Visual representation of Sign In Page

Uploading:

1. The user should be able to drag & drop or click on the uploading square and select a document to upload. The file uploaded should be in the right format (pdf, txt, docs).
2. If the file is in the wrong format, an error message will appear.

3. The user can remove the file if he realizes he has uploaded the wrong one, or click on the “upload” button.
4. If he clicks on the “upload” button, the file should be stored in the proper database and the user should be redirected to a confirmation page.

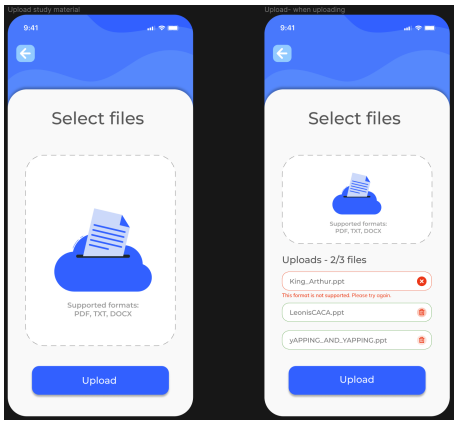

Figure 3: Visual representation of Uploading Document pages