Tab 1

**Research Labs Inventory**

Evan Ross

**mobile application subsystem final report**

REVISION – Final

5 December 2024

Subsystem final report

for

Research Lab Inventory

Prepared by:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Author Date

Approved by:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Leader Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

John Lusher, P.E. Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

T/A Date

**Change Record**

| **Rev.** | **Date** | **Originator** | **Approvals** | **Description** |
| --- | --- | --- | --- | --- |
| **1** | 12/05/2024 | Evan Ross |  | Final Release |

**Table of Contents**

[**1.Introduction 7**](#_tyjcwt)

[**2. Tools and Applications Used 7**](#_6aq5cryvsr0o)

[2.1 Android and Android Studio 7](#_lnxbz9)

[2.2 Flutter 7](#_lnxbz9)

[2.3 SQFlite 8](#_lnxbz9)

[**3. Design and UI 8**](#_6aq5cryvsr0o)

[3.1 Login Page (Figure 1) 8](#_lnxbz9)

[3.2 Register Page (Figure 1) 8](#_4zja0wartlz4)

[3.3 Home/Search Page (Figure 1) 8](#_lnxbz9)

[3.4 View Items Page (Figure 2) 9](#_lnxbz9)

[3.5 Check In/Out page (Figure 2) 10](#_lnxbz9)

[3.6 FAQ page (Figure 2) 10](#_lnxbz9)

[3.7 Developer Page (Figure 3) 11](#_lnxbz9)

[**4. Implementation and Operation 12**](#_lv5gggoin3l7)

[4.1 Database 12](#_lnxbz9)

[4.2 User interface Display 13](#_lnxbz9)

[4.2 User Interface Functions 13](#_lnxbz9)

[**5. Validation 13**](#_ccuorl1nfd0j)

[5.1 User Interface Functionality 13](#_lnxbz9)

[5.2 Database Functionality (Users) 13](#_lnxbz9)

[5.3 Database Functionality (Items) 14](#_lnxbz9)

[5.4 Check Out/In (Checkout List) 15](#_lnxbz9)

[**6. Challenges 16**](#_blbzbzhc591y)

[**7. Conclusion 16**](#_jt6wcvnmg28r)

[**8. Future Plans and Implementation 16**](#_97fstopbl58t)

[**9. System Requirements 17**](#_97fstopbl58t)

[**10. References 17**](#_c6gv0yit1r3u)

**List of Tables**

No table of figures entries found.

**List of Figures**

[**Figure 1. Login Page, Register Page, and Home/Search Page**](https://docs.google.com/document/d/1JT85ppRDy_ktOJj15dlnfq1aH-1qnvgt/edit#heading=h.1mrcu09) **9**

[**Figure 2. View Items Page, Check In/Out Page, and FAQ Page**](https://docs.google.com/document/d/1JT85ppRDy_ktOJj15dlnfq1aH-1qnvgt/edit#heading=h.46r0co2) **11**

[**Figure 3. Developer Page**](https://docs.google.com/document/d/1JT85ppRDy_ktOJj15dlnfq1aH-1qnvgt/edit#heading=h.46r0co2) **12**

[**Figure 4. Users Table exported from database as CSV file 14**](https://docs.google.com/document/d/1JT85ppRDy_ktOJj15dlnfq1aH-1qnvgt/edit#heading=h.46r0co2)

[**Figure 5. Items Table exported from database as CSV file**](https://docs.google.com/document/d/1JT85ppRDy_ktOJj15dlnfq1aH-1qnvgt/edit#heading=h.46r0co2) **15**

[**Figure 6. Active Checkout List Table exported from database as CSV file**](https://docs.google.com/document/d/1JT85ppRDy_ktOJj15dlnfq1aH-1qnvgt/edit#heading=h.46r0co2) **16**

# 1.Introduction

The Mobile Application subsystem can be run on an Android phone and is designed to be the mobile version of the LabRat website/application. The goal of this application as a whole is to modernize the management of research laboratories. The mobile application gives the users the ability to check in and check out items on the go. This allows users to make changes to the inventory easily and intuitively right inside the lab. Though this has yet to be integrated, the app will also have a machine learning aspect which will utilize the built in smartphone camera. From the camera users will be able to scan a specific item to check it in or out. Currently all data including users, items, and the active checkout list is stored in a local SQFlite database. This eventually will be integrated into an externally hosted database which will be connected to the web application as well. The entire mobile application is built with the user in mind and is meant to provide an intuitive and simple solution to research lab inventory management.

# 2. Tools and Applications Used

## 2.1 Android and Android Studio

The Mobile application is specifically designed to function on Android phones specifically with the operating system API 16 (Android 4.1) or above. The reason Android was chosen as the operating system is due to its wide variety of development softwares available, as well as its ability to easily publish and access a new app. Android studio was chosen because it is essentially the standard for developing Android applications, as it was made by Android. It also has a wide range of useful plugins such as Flutter.

## 2.2 Flutter

For development of the application, the plugin Flutter was used along with Android Studio. Flutter is a User Interface toolkit or framework that allows developers to develop an application from a single codebase. It uses the language Dart which makes mobile application development intuitive for people who are not as familiar with app development. Flutter was chosen because of its intuitive language, large community, and its “Hot reload” feature which allows users to make changes to the app and see the result in real time.

## 2.3 SQFlite

SQFlite is a plugin for flutter which allows the creation and utilization of a local SQLite database. The database is contained locally in the simulation or on the smartphone. This is strictly for development purposes, next semester we plan to link the externally hosted database to the mobile and web application.

# 3. Design and UI

## 3.1 Login Page (Figure 1)

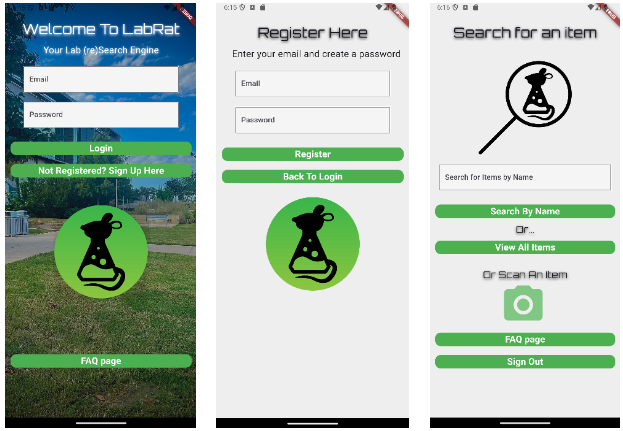
This is the first screen that users see when they open the mobile app. It is meant to greet the users with a visually appealing interface. They will be given the option to enter their email and password to sign in if they already have an account. The password will be obscured for security. If they do not have an account yet there will be a button which brings users to the registration page. There is also a button for the FAQ page located near the bottom of the screen.

## 3.2 Register Page (Figure 1)

This is the page where users can register if they do not already have a username and password. All they do is enter their email address in the email text box, and create a password by entering it in the password box. The entered email address must be valid, for example [something@something.com](mailto:something@something.com). This ensures that a user has an email address that can be used to receive emails such as forgotten password emails, which we intend to implement in the future. After they enter their email address and password they can press the register button which will register their account. Alternatively there is a return to login button. Both buttons will send the user back to the login page.

## 3.3 Home/Search Page (Figure 1)

This is meant to be the main page for the application. It is the page in which users can search for items that they want to check out or check in. They have the option to search for items by name, which will send them to a list of all items with the name that they have searched. They also have the option to “view all items” which will allow users to view all of the items within the laboratory. There is also a camera button that opens the built-in phone camera and takes a picture. At the moment this picture is not used in any way, though next semester it will be passed into the machine learning model to have whatever part is scanned identified. When a part is identified by the model it will automatically send the user to the checkout or check in page for that item. There is also a button for the FAQ page located at the bottom as well as a “sign out’ button which will return the user to the login page.



*Figure 1. Login Page, Register Page, and Home/Search Page*

## 3.4 View Items Page (Figure 2)

This is the page that is the list of the items in the lab. Depending on how the user accesses this page, whether it is via the search bar, “view all items” button, the page will display a different set of items. As mentioned before the view all items button will display all items in the lab while the using the search bar will filter to show only the searched items.

This page is in a table format with columns for “item”, “quantity”, “Availability”, as well as a column for an image of the Item. The image column will display a pre-loaded image of the item based on the name. If there is no image available for a specific item a “no image found” icon will be displayed instead. The name of the item in the “item” column is selectable and will send the user to the respective check out/In page for that specific Item.

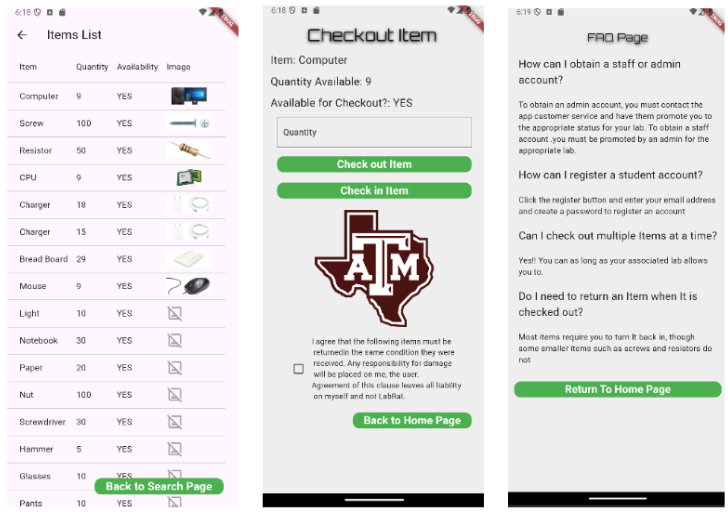
## 3.5 Check In/Out page (Figure 2)

This page is where users can check in or check out the item that they have selected. Displayed at the top of the page is the item name, item quantity, and whether the item is available for checkout. From there users can enter the quantity that they would like to check out or check in. They then can press the respective button to either check out or check in the item. This page also has a button that can return the users to the home/search page.

It is worth mentioning that users are not able to check out more quantity than is available. Users are also not able to check in items that they have not checked out, or more than they have checked out. Users are also not able to return items that do not require return. In order to check out an item, users must first check the acknowledgement box. If the user fails to do any of these things they will be alerted by the system.

## 3.6 FAQ page (Figure 2)

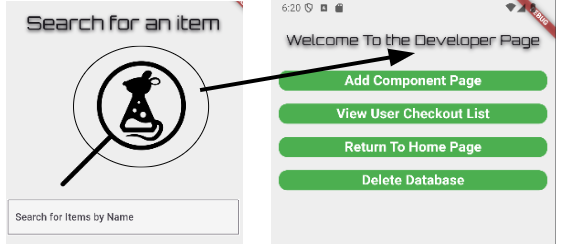
This page serves as the FAQ page where users can go to find answers to commonly asked questions. This page will continue to be expanded upon as new possible questions can come up. If a user has a question not on the FAQ page they can email one of the developers and we can assist them.



*Figure 2. View Items Page, Check In/Out Page, and FAQ Page*

## 3.7 Developer Page (Figure 3)

This page is meant strictly for development and testing purposes, and is not meant to be accessed by the everyday user. The page will not be in the final version of the app. It can be accessed by tapping the image in the middle of the home screen. The page gives the developer the option to add items to the database, view the active checkout list, or delete the database entirely. Deleting the database is useful when wanting to clear all information, or if changes are made to the database that require it to be recreated.



*Figure 3. Developer Page*

# 4. Implementation and Operation

## 4.1 Database

As mentioned before, currently the mobile application temporarily uses a local database for testing and development. Eventually this will disappear as the app is linked to the externally hosted database next semester, but it is important to have a local database to thoroughly test all of the features.

The local database is done using SQFlite plugin which assists with creating a local database in a flutter application. The database is created and maintained in the authentication\_db.dart file. This file also contains any functions that pertain to the database including updating, creating new entries, as well as deleting. There is no actual page for the database.

It is difficult to view the contents of the database directly in Flutter through the command window. To view the database I have been loading the database file (which is stored in the simulated phone) into an sqlite database viewer, which gives the contents of the database as a CSV file, which I can view in excel. Viewing the contents of the database is important for validation and making sure that entries act as intended.

## 4.2 User interface Display

Most code in flutter either operates to display the user interface in a certain way, or change how the user interface acts when interacted with. The root of all pages in flutter is a widget which is the user interface for the page. Within that widget other widgets can be displayed such as columns, text, boxes, or buttons. Widgets are stackable so that one widget can contain others.

Pages can also take in variables which can change the way that the UI is displayed. An example of this is passing the queue to the view items page or passing the selected item to the check out/In page.

## 4.2 User Interface Functions

Widgets in the user interface can be given a function to perform when interacting with in a certain way. For example many of the button widgets in the app perform the function that navigates to another page when tapped. They also can set variables and text controllers as they do in the email and password boxes in the login screen. They can also be given the functionality to interact with the database through created functions.

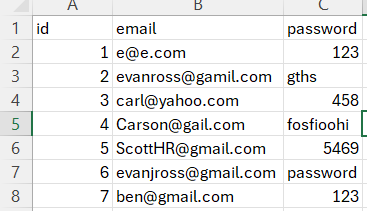
# 5. Validation

## 5.1 User Interface Functionality

The User interface Functionality was tested by using the app as a user normally would. All of the buttons, text boxes, and the camera Icon works as expected. This was originally tested through the Android Studio simulated phone, but it now has also been tested using a physical android phone and performs as expected for both.

## 5.2 Database Functionality (Users)

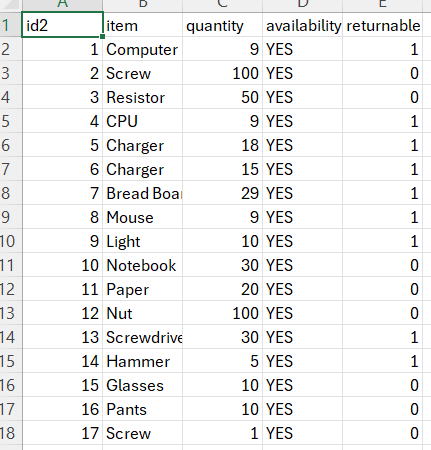
The “Users” table in the database was tested through adding users to the database and then viewing the contents of the database. As more users are registered, these users are added to the database. The Database table is viewed through an SQlite database viewer which exports the tables to a CSV file which can be viewed on excel.



*Figure 4. Users Table exported from database as CSV file*

## 5.3 Database Functionality (Items)

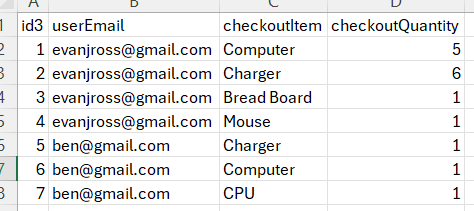
The “Items” table of the database was checked in multiple ways. The first way is the same as the “Users” Table where the contents of the database can be viewed directly through the database viewer. The second way is looking at the items list screen in the user interface of the application. In both of these ways, the items table was tested by adding and removing items from the database. Changing the quantities by checking in and out items was also tested. Everything worked as expected in the “Items” table.



*Figure 5. Items Table exported from database as CSV file*

## 5.4 Check Out/In (Checkout List)

The Check Out/In table of the database was tested in a similar way to the Items table. It can be viewed through the database viewer or it can be viewed through the app interface via the “View User Checkout List” button in the developer page. It was tested by having multiple different users check out multiple different Items to make sure that they are added to the active checkout list. I also tested the ability for users to check back in items to reduce the quantity checked out, or delete the entry all together if the quantity goes to zero. I also tested to make sure that when a user checks out a non returnable item (such as a screw), it is not added to the active checkout list. Everything in this table worked as expected.



*Figure 6. Active Checkout List Table exported from database as CSV file*

# 6. Challenges

The biggest challenge that I faced throughout this project is being unfamiliar with the coding language. Luckily Android Studio and Flutter has a large community and much documentation available online.

# 7. Conclusion

The mobile application has been developed as the mobile version of the LabRat inventory management application, which aims to simplify and modernize the way research labs are managed. The mobile application specifically has the user in mind as it will allow users to walk around the lab and look up or scan various items from inside the lab. The user interface as well as the database functionality was tested and validated to ensure that the application is ready to be linked with the other subsystems.

# 8. Future Plans and Implementation

Currently the mobile application uses a local SQFlite database for testing purposes. In the future we intend to link It to the externally hosted database. This will make the mobile application able to share the tables of information with the web application on one big server.

We also plan to implement the machine learning model into the mobile app. This will be accessed via the camera icon on the home screen of the app. When the built in camera takes a picture of an item it will be passed on to the machine learning model to be identified. When it is identified, the user will be sent to the corresponding check out/in screen for that Item.

# 9. System Requirements

The device used to run the mobile application must use the Android operating system. As stated in the Flutter Documentation, Android API 16 (Android 4.1) or above is required to run the mobile application. Phone must have a functional camera if user wishes to use the machine learning feature.

# 10. References

[1] “SQLite Viewer,” inloop.github.io. https://inloop.github.io/sqlite-viewer/#

[2] Flutter, “Flutter documentation,” docs.flutter.dev, 2024. https://docs.flutter.dev/

[3] “Dart packages,” Dart packages. https://pub.dev/