**Software Design Document**

Mobile Application Development

12184026

Kai Qiu

**Table of contents**

1 Introduction 1

1.1 Application Vision 1

1.2 Scope 1

1.3 Document Version History 1

2 Functionality Overview 2

2.1 Feature Summary 2

2.2 Application Flow Diagram 3

3 Design 3

3.1 Overall Components overview 3

3.2 Mockups 4

3.3 Used Application Programming Interfaces 8

4 Application Structure 8

4.1 Packages 8

4.2 Activities 9

4.3 Fragments 9

5 Data Structure 10

6 Testing 11

7 Conclusion 11

8 References 11

# 1 Introduction

Information technology has provided many efficient ways to running modern business management. At the point of cargo management, up to now, information system has been developed perfectly. When a warehouse keeper tries to add in or take out the cargo from warehouse, the cargo management system can keep pace with it accurately and efficiently. However, many of these systems are generally large and inflexible for small business (i.e. a small store) to operating. In addition, the prevalence rate of mobile device is increasing every year and the number of mobile application developments is ascending. Therefore, design a cargo management system base on small business is necessary. This application focus on helping small business owner viewing their storage, warehousing cargoes, delivering cargoes and check their storage.

## 1.1 Application Vision

This application is a small-size software, it only takes 3.3 MB of device storage capacity. It meets the exact needs of small business owner. They can easily access to the database of this application even on their way to home. Free download, easy to master and open source.

## 1.2 Scope

This Software Design Document is for a system face to small business owner. It demonstrates the possibility for small business owner to run cargo management on their mobile devices. Therefore, the application design mainly contains the basic cargo management functions. These functions can be extended for future customer requirements. This system can be improved to compatible with other pre-existing management system.

## 1.3 Document Version History

26/09/2016:

Finalized the main sidebar design and programming.

Finalized most of the fragment design in MainActivity.

Finalized Login and Register function design and programming.

Implement the function of adding new items in one storage.

11/10/2016:

Implemented Firebase Authentication API on Login and Register function.

Implemented Firebase Realtime Database API as database for the whole application.

Implemented Firebase Storage API as data storage to store and display picture of cargos.

Finalized Barcode scanning function of storage item adding procedure.

Finalized camera using function implementation which allow user to take photo of their cargo and save it to the database storage.

Finalized show cargo details activity, delete cargo function and most of edit cargo details function design and programming.

16/10/2016

Finalized Warehousing and Delivery function programming.

Finalized Storage Overview function programming.

Finalized application testing and fixed bugs.

Finalized all of the comments and formatting the code.

# 2 Functionality Overview

The application provides a virtual storage which contains a list of cargoes. User can add items to the storage as well as edit and delete it. In the warehousing section, user can increase the amount of items manually by inputting the number of items or by scanning the barcode of these items. Each user has their own email as username and password to access to their own data from database.

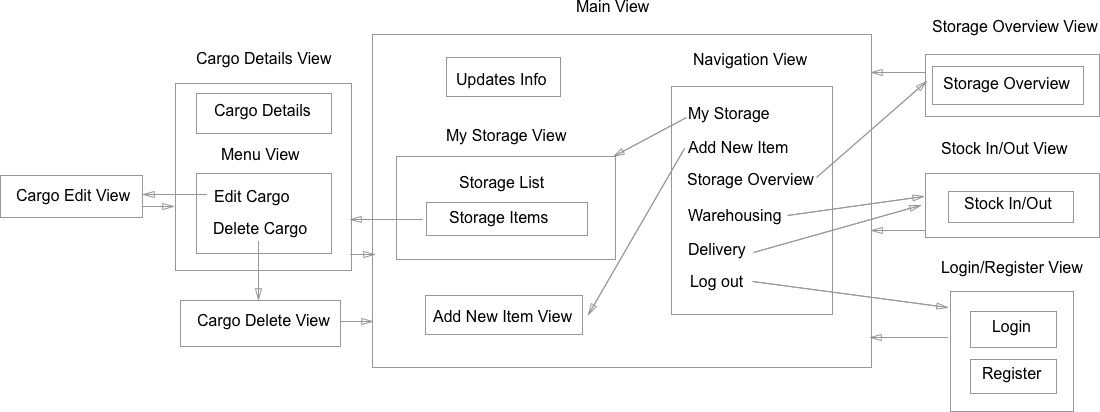
## 2.1 Feature Summary

Login and Register feature: User must provide an email address and password to register into the system. After that, each time when user want to use this application, he/she can submit their email address and password to login. This allows only the owner of current storage can access to the storage data. On the other hand, user can log into the system on any kind of android device as long as he/she provide the correct email address and password. This achieves remote manipulation.

Storage list review feature: When user click on “My Storage” option on the navigation bar, the application would show the whole storage list to the user. This contains the picture, name, price, amount of all the cargos currently in the storage. User can click one of the items on the list to see the details. On the base of previous cargo information, the cargo details view shows the description and barcode of selected cargo. User can edit the cargo information and delete cargo as well. On the main view, user can be able to check the statistics overview of the storage. For example, how many cargos have been stored in the storage or how much money do all my cargos worth.

Storage list management feature: On cargo details view, user can edit cargo information or delete current selected cargo. On the main view, User can be able to add new cargo to storage list. It also provides two functions—warehousing and delivery to manipulate the storage list when some cargos are being added to the storage or being delivered to other place from this storage.

## 2.2 Application Flow Diagram



# 3 Design

3.1 Overall Components overview

The following list shows the packages and the corresponding Java class.

com.mad.easystorage.activity:

1. LoginActivity
2. MainActivity
3. CargoDetailsActivity
4. CargoEditActivity
5. StorageOverviewActivity
6. ScannerActivity
7. StockInOutActivity

com.mad.easystorage.adapter:

MyCargoRecyclerViewAdapter

com.mad.easystorage.application:

MyApplication

com.mad.easystorage.constants:

Constants

com.mad.easystorage.fragment:

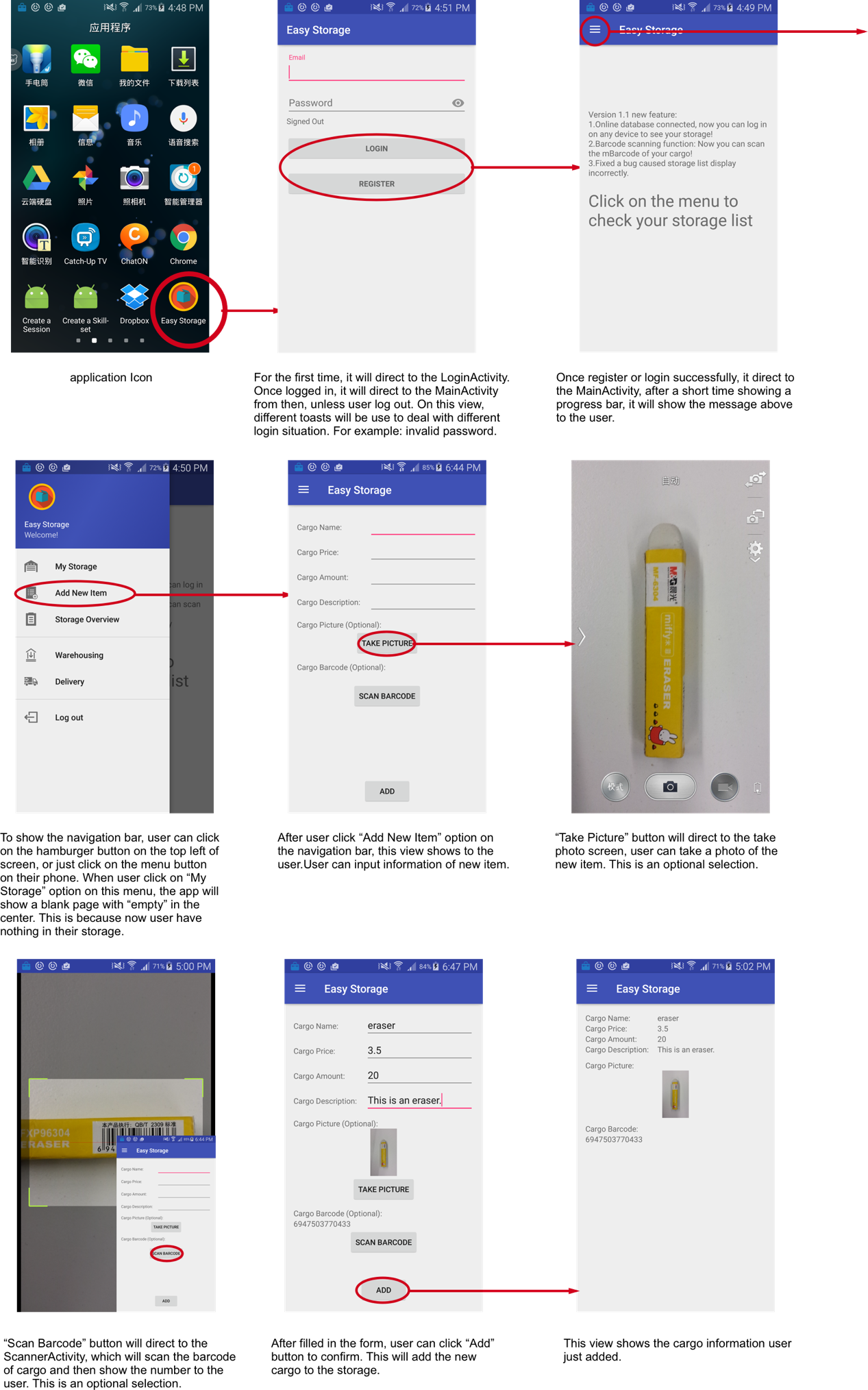
1. AddCargoFragment
2. StorageFragement

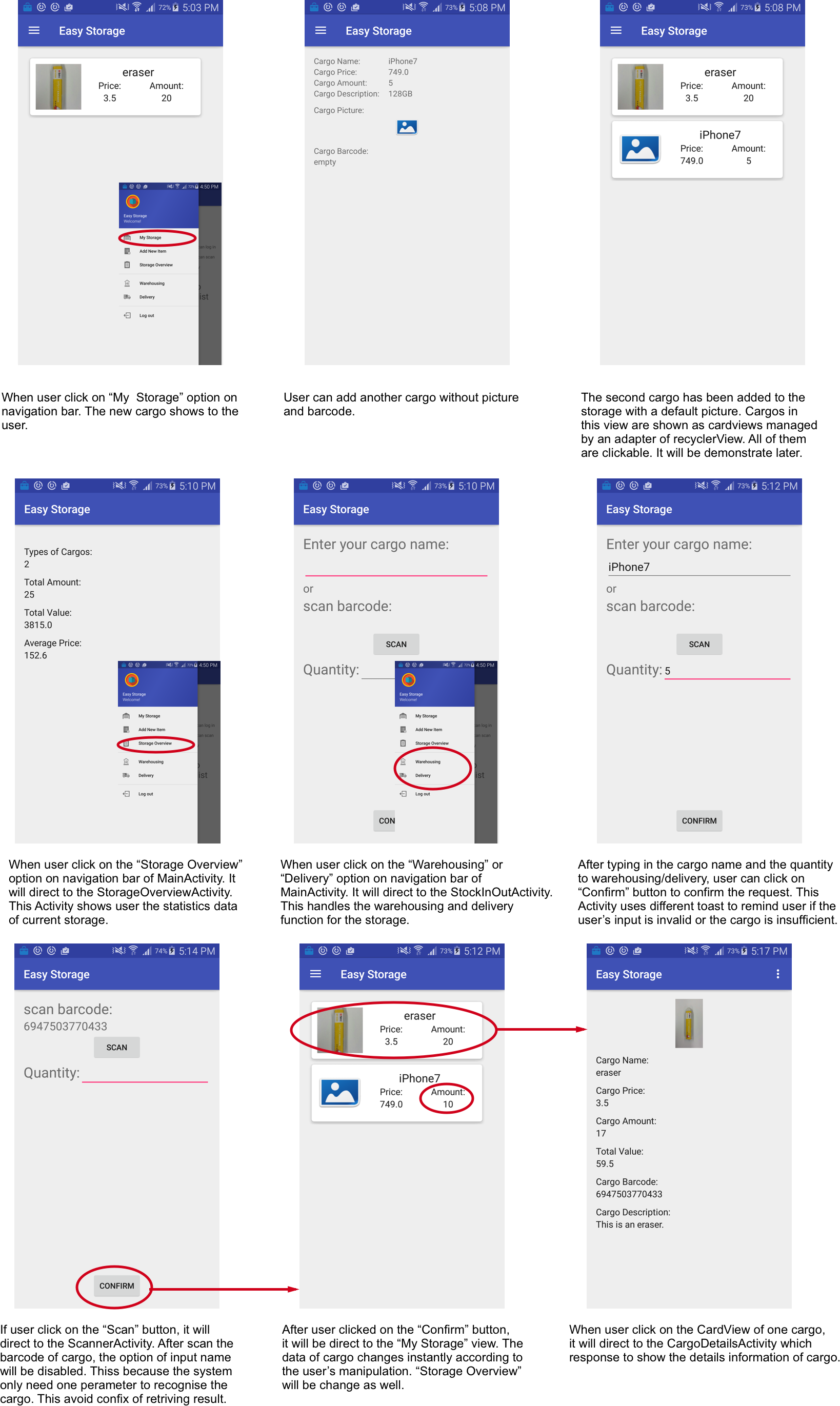
com.mad.easystorage.model:

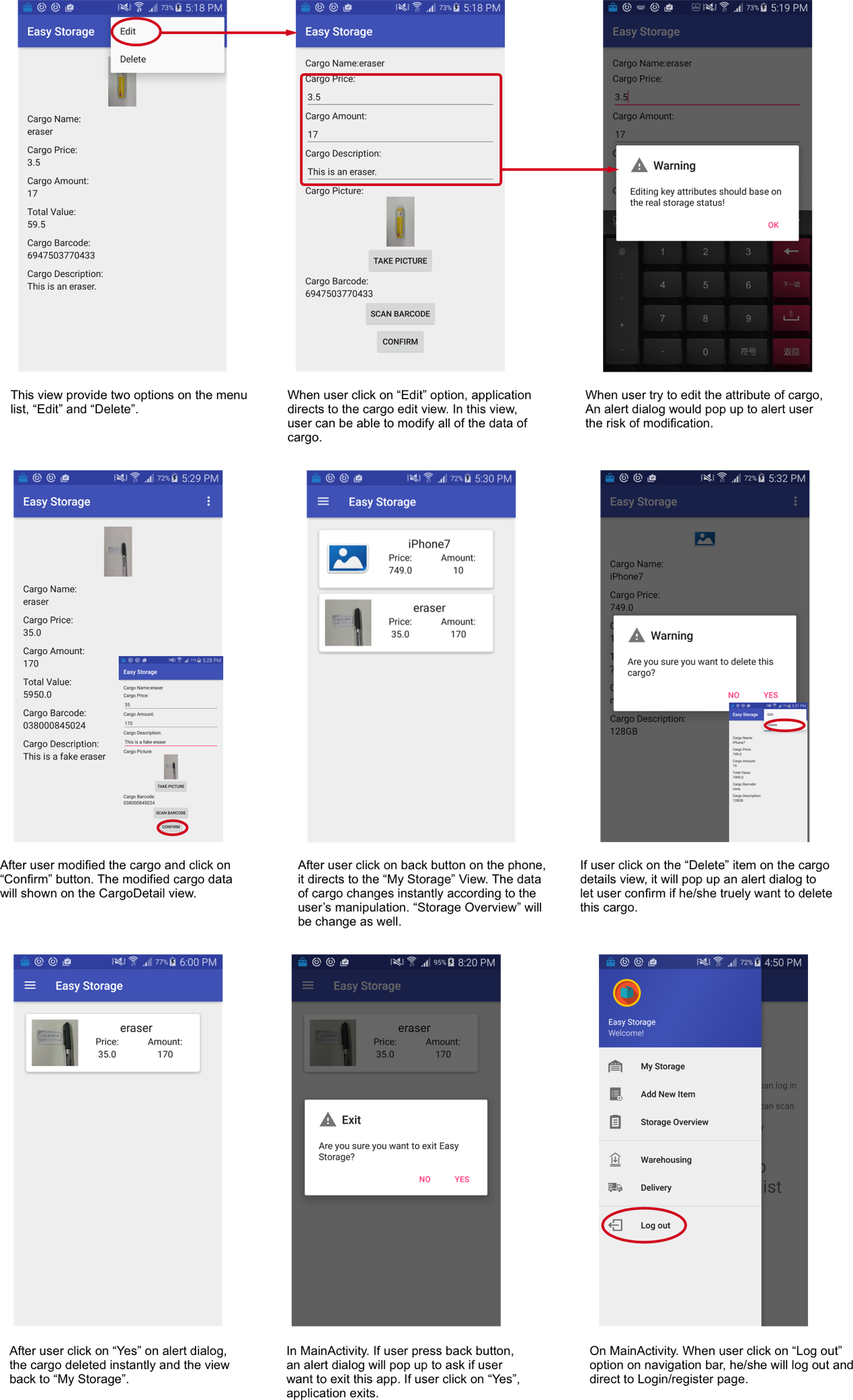
1. Cargo
2. Storage

Besides this, the database and file storage are using Firebase backend: BaaS(Backend-as-a-service)

3.2 Mockups







## 3.3 Used Application Programming Interfaces

com.google.firebase: This includes Firebase Database, Firebase Authentication and Firebase Storage. This API provides convenient database, file storage and authentication management system. It is easy to apply its interface to the software.

com.android.support: This API provides fragment function which can help coordinate the navigation bar item handling. It also provides recyclerview and cardview to display cargo list to user.

me.dm7.barcodescanner: This API provides the barcode scanning function which can efficiently recognize the target barcode and return it to the system.

# 4 Application Structure

## 4.1 Packages

The base package of this application is com.mad.easystorage, it contains other 6 packages:

com.mad.easystorage.activity,

com.mad.easystorage.adapter,

com.mad.easystorage.constants,

com.mad.easystorage.model

com.mad.easystorage.model.fragment

com.mad.easystorage.application.

com.mad.easystorage.activity: All of the activity classes are in this package, it contains LoginActivity, MainActivity, StockInOutActivity, StorageOverviewActivity, CargoDetailsActivity, CargoEditActivity, ScannerActivity.

com.mad.easystorage.adapter: It contains MyCargoRecyclerViewAdapter which support the function of showing the storage list to the user.

com.mad.easystorage.application: It contains MyApplication class which responsible for centralization management of all activities.

com.mad.easystorage.constants: The Constants class is in this package, it contains all of the constants used in the other classes.

com.mad.easystorage.fragment: It mainly use for package the fragment classes, which is used to display the different fragments when the user selected one option in the sidebar list.

com.mad.easystorage.model: All of the models (Storage and Cargo) are in this package.

## 4.2 Activities

LoginActivity: It is used to verify the authentication information of user. It utilizes Google Firebase authentication API. It handles login function, register function and authentication function.

MainActivity: It is the main activity of the whole application. The function in this activity is used for displaying a navigation bar of the main UI which direct the UI flow to the other activities. In addition, it implements the function to display the storage list to the user. User can add new items to the list as well.

CargoDetailsActivity: This activity response to display the detail information of cargo.

CargoEditActivity: This activity makes all of the cargo information can be edited by user.

ScannerActivity: This activity supports user to scan barcode of a specific cargo.

StockInOutActivity: This activity handles all of the warehousing and delivery of one specific cargo.

StorageOverviewActivity: Function in this activity analyses the whole data of cargos in storage and provide statistics overview of user storage.

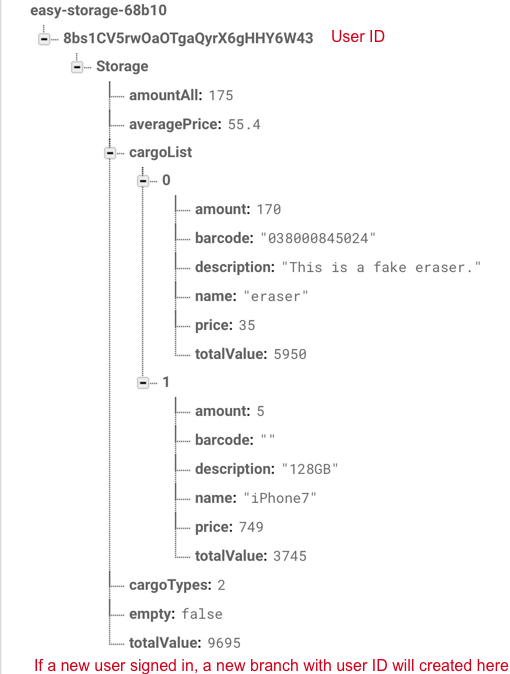
## 4.3 Fragments

AddCargoFragment: A fragment to display a form of empty details of one cargo. Usercan fill in the form and submit to add the information of new cargo to the cargo list. It works when user select “Add New Item” option on the sidebar of main UI.

StorageFragement: A fragment to display a list contains all of the cargo details in the storage of user. It works when user select “My Storage” option on the sidebar of main UI.

# 5 Data Structure

Since this application use Firebase database API as database. All data stored in a Jason tree structure. The diagram below is an example shows the structure specifically.



Cargo pictures stored in Firebase Storage. If one picture of cargo created, it will be upload to this directory: “gs://easy-storage-68b10.appspot.com/images/<UserID>/” as a jpg file with its name. “gs://easy-storage-68b10.appspot.com” is the storage reference of Firebase Storage. For example, if a user with ID “somebody123” uploaded a cargo image called “cargo456”, the full path of file would be “gs://easy-storage-68b10.appspot.com/images/somebody123/cargo456.jpg”. Images here can be downloaded, deleted, changed as well.

# 6 Testing

For the testing, this application must be installed on an android device with SDK version higher than 15. Network connection is essential.

The diagram in [**3.2 Mockups**](#Mockups)is an example of testing case as well.

# 7 Conclusion

The functions of this application is basically done. All of them works very well. One shortcoming is lack of status bar using. One imagine idea is to setup an alarm system. When the cargo is almost run out, it will alarm the user timely.

The easy part is to setup the different UI component such as TextView, Button etc. The hard part is to implement the Firebase API and fragment API.

# 8 References

1. Lynda tutorial: Google Firebase for Android: First Look

https://www.lynda.com/course-tutorials/Google-Firebase-Android-First-Look/501162-2.html

1. Lynda tutorial: Building Flexible Android Apps with the Fragments API

https://www.lynda.com/Android-tutorials/Building-Flexible-Android-Apps-Fragments-API-Revision/487934-2.html

1. Google Firebase Documentation

https://firebase.google.com/docs/android/setup