

**Foodie ANDROID APPLICATION**

A Project

Presented to the faculty of Computer Engineering



Fall 2020

Abstract of

FOODIE ANDROID APPLICATION

by

Hadi Abdelmajid

There are a number of applications in Android store for Recipes Search but none of them support interface for searching, and sharing recipes all at once. Foodie is an Android application with image based UI for searching, sharing & creating. This app provides flexibility to user to search variety of recipes from available recipes in cloud. This is very handy application, which every user can search cloud for recipes, share recipe with friends and create personal post recipe. This app is time saver providing recipes in few clicks. By introducing title search, Foodie makes finding recipes easy. With recipes being added daily there will always be something new for user to crave. The project has been implemented using Java, Android and Firebase.

# ACKNOWLEDGEMENTS

# We would like to thank Dr.Muhiedinne , our advisor for providing us an opportunity to work on this project, which significantly broadened our knowledge on Android Development. We thank him for continuously providing the feedback, help and support to complete the project successfully.

**In addition, we would like to thank Dr.Rola Kassem for her willingness and supervision to serve us with all what we need to know.**

Contents

[ACKNOWLEDGEMENTS i](#_Toc59639932)

[1-INTRODUCTION 6](#_Toc59639934)

[2.PROJECT REQUIREMENTS 7](#_Toc59639935)

[2.1 User Options 7](#_Toc59639936)

[2.2 Recipe Search 8](#_Toc59639937)

[2.2.1 Search by Title 8](#_Toc59639938)

[Figure 2- Search by Title And Ingredients Flow Diagram 9](#_Toc59639939)

[2.3 Post recipe. 9](#_Toc59639940)

[Figure 3- Post recipe Flow Diagram 10](#_Toc59639941)

[2.3.1 Create Recipe 10](#_Toc59639942)

[3-ANDROID DEVELOPMENT BASICS 11](#_Toc59639943)

[3.1 Four Types of Application Components 11](#_Toc59639944)

[3.1.1 Activities 12](#_Toc59639945)

[Figure 4- Activity Lifecycle[3] 13](#_Toc59639946)

[3.1.2 Services 14](#_Toc59639947)

[Figure 5- Service Lifecycle[4] 16](#_Toc59639948)

[3.1.3 Content Providers 16](#_Toc59639949)

[3.1.4 Broadcast Receivers 17](#_Toc59639950)

[3.3 Running the Application 18](#_Toc59639951)

[Figure 15- Android Device Chooser 19](#_Toc59639952)

[3.4.2. On Emulator 19](#_Toc59639953)

[Figure 16- Android Virtual Device Manager 20](#_Toc59639954)

[Figure 17- Android Virtual Device Edit 21](#_Toc59639955)

[4 FEATURES IMPLEMENTATION 22](#_Toc59639956)

[4.1 Application Main Page 22](#_Toc59639957)

[4.2 Application Sign up page. 23](#_Toc59639958)

[4.3 Reset password 24](#_Toc59639959)

[4.4 Home Page 25](#_Toc59639960)

[4.5 Post Page 26](#_Toc59639961)

[4.6 Profile Page 27](#_Toc59639962)

# 1-INTRODUCTION

Foodie application is a very useful app for people who love to cook and try out new recipes. Foodie application provides user flexibility to search, share recipes from cloud with an additional capability to maintain personal post recipe for creating new recipe.

Our application “Foodie” makes finding recipes easy. Recipes are from everyday cooks and chefs that have perfected the recipe over time. Recipes range from the decadent to the simple for whatever mood user are in. Recipes are from well-known publishers All Recipes, 101 Cookbooks, Closet Cooking and many more. Foodie Application provides trending recipes. Recipes are displayed with recipe image, title, ingredients list and cooking directions. Title based search is also provided to user. Foodie App provide sharing capability. This application is a time saver providing recipes in few clicks. The user is given choice to create personal recipes, where user can create recipe and view it. The interface is clean and simple. It makes use of Android image button capability to display options on home screen with image icons. The user can search recipes, view added favorite recipe list and access personal post recipe all from home screen.

# 2.PROJECT REQUIREMENTS

## 2.1 User Options

Log in

Sign up

Reset Password

Search recipes by title

Add recipes to recently added view

View User’s Profile

Figure 1- Use Case Diagram

Users of Foodie application have following options as shown in Figure 1:

* Users can Sign up using email , name , username , password & phone number
* User can login using his/her’s credential.
* User can reset password via email .
* User can add new recipe with it’s image.
* User can view newly added recipes.
* User can search through recipes by title
* The following section describes each part in detail.

## 2.2 Recipe Search

User can perform search for recipes based on title, rating, trending, categories and ingredients.

### 2.2.1 Search by Title

Recipe displayed with all details

Recipe title+click search

Select recipes search

# Figure 2- Search by Title And Ingredients Flow Diagram

User is provided with search image icon on home screen to search recipe. Clicking the search image icon on home page user is navigated through recipes and search by title. Figure 2 depicts flow diagram for title search option. On search button click, list of recipes are displayed with image, recipe title and many more details.

## 2.3 Post recipe.

User has option to manage post recipe by creating recipe, viewing recipe

Figure 3 shows flow diagram for managing post recipe in application.

### 

Creates recipe by entering title , cooking time , number of servings, preparation time, ingredients and procedure.

Selects gallery option and select image

Clicks on take recipe image button.

Clicks on post icon

Recipe details displayed with cooking time, number of yields, ingredients, procedure and preparation time

Recipe list displayed with recipe title and several details

User selects home icon

# Figure 3- Post recipe Flow Diagram

### 2.3.1 Create Recipe

User can create and manage his personal post recipe on phone, tablet. When user clicks personal post recipe image button on home page, user gets navigated to new screen. This new screen. Template provide fields for recipe title, cooking time, ingredients list for selection and cooking steps. User can create recipe by filling details in template fields and clicking on add foodie button. When user clicks add foodie, new row gets added in database for created recipe. Take recipe image button allows user to save recipe image. User has option to only select image from gallery.

2.5.2 View Recipe

Foodie application save recipes added by user in database providing capability to view saved recipes when required. On the home icon user can view all added recipes.

# 3-ANDROID DEVELOPMENT BASICS

Android is a [Linux-](http://en.wikipedia.org/wiki/Linux)based [operating system](http://en.wikipedia.org/wiki/Mobile_operating_system) designed primarily for [touch screen](http://en.wikipedia.org/wiki/Touchscreen) mobile devices such as [smartphones](http://en.wikipedia.org/wiki/Smartphone) and [tablet computers.](http://en.wikipedia.org/wiki/Tablet_computer) To make things easier for developers,

Google had made Android as [open source](http://en.wikipedia.org/wiki/Open_source) and releases the code under the [Apache License.](http://en.wikipedia.org/wiki/Apache_License) Android has a large community of developers writing applications ("[apps"](http://en.wikipedia.org/wiki/Mobile_app)) that extend the functionality of devices, written primarily in a customized version of the [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) programming language [1].

Foodie application is written in java programming language using Firebase database. The code of Foodie App is compiled using Android SDK tools into an android package, an archive file with an .apk suffix. This .apk file is considered as one application and is used to install the application in all android devices like tablets and phones.

Application components are the essential building blocks of an Android application. These components are loosely coupled by the application manifest file AndroidManifest.xml that describes each component of the application and how they interact [1][2]. There are four different types of application components: activities, services, broadcast receivers and content providers.

## 3.1 Four Types of Application Components

Android provides four application components: activities, services, content providers and broadcast receivers. Below section describes each component in detail.

### 3.1.1 Activities

Activity represents a single screen with a user interface. For example, in this Foodie application there are many activities such as search by title ,post recipe create recipe fragment and so on. One activity in the application is specified as the "main" activity, which is presented to the user when launching the application for the first time. The main activity can then start another fragment in order to perform different actions. Whenever a fragment starts, the previous activity, fragment is stopped, but the system preserves the activity in a stack (the "back stack"). When a new activity starts, it is pushed onto the back stack and takes user focus. This back stack follows LIFO(last in first out) stack mechanism, so, when the user is done with the current activity and presses the back button, it is popped from the stack (and destroyed) and the previous activity resumes.

Activity must be declared in the manifest file to be accessible by the system. Intent filters can also be specified in the [<activity>](http://developer.android.com/guide/topics/manifest/activity-element.html) element to declare how other application components may activate it.

The activity automatically includes an intent filter that declares the activity responds to the "main" action and should be placed in the "launcher" category. The intent filter looks like this:

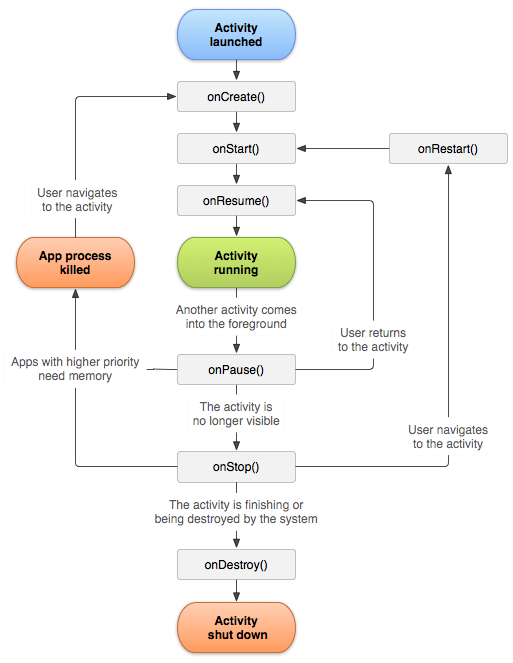
<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>



## Figure 4- Activity Lifecycle[3]

The callback methods for activity are as follows which are shown in Figure 4.

onCreate(): This is called when the activity is first created. This is always followed by onStart().

onStart(): This is called just before the activity becomes visible to the user. This is followed by onResume() if the activity comes to the foreground, or onStop() if it becomes hidden. OnResume(): This is called just before the activity starts interacting with the user. This is always followed by onPause().

onPause(): This is called when the system is about to start resuming another activity. And this is followed either by onResume() if the activity returns back to the front, or by onStop() if it becomes invisible to the user.

onStop(): This is called when the activity is no longer visible to the user. This is followed either by onRestart() if the activity is coming back to interact with the user, or by onDestroy() if this activity is going away onRestart(): This is called after the activity has been stopped, just prior to it being started again. This is always followed by onStart().

OnDestroy(): This is called before the activity is destroyed. This is followed by nothing[1].

### 3.1.2 Services

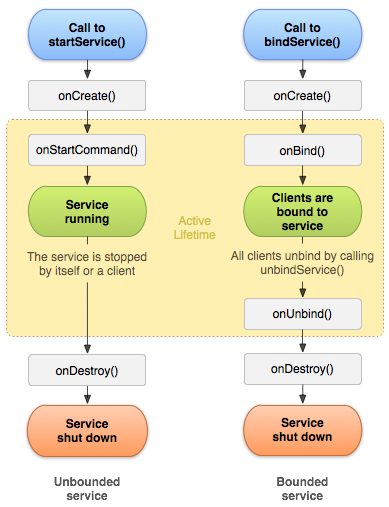
A service is a component which would be running in background without direct interaction with the user. As the service has no user interface it is not bound to the lifecycle of an activity. Services are used mostly when there is a time consuming and long task, like loading an Image, or a File, or download something for the Internet and asynchronous tasks in general.

Service has two forms: Started and Bound. Below section describes in detail:

1. Started: When an activity starts startService(), then a service is "started". A service can run in the background indefinitely once started, even if the component that started it is destroyed. For example, it might download a image over the network.

When the download is completed, the service should stop itself.

1. Bound: When an application component calls bindService(), a service is "bound". A bound service provides a client-server interface that allows components to interact with the service, send requests and get results. Figure 11 shows lifecycle of services in android.



## Figure 5- Service Lifecycle[4]

The callback methods are: onStartCommand(): This method is called when an activity requests for a service to be started by calling startService().

onBind(): This method is called when an activity wants to bind with the service by calling bindService().

onCreate(): This method is called when the service is first created to perform setup of the service before onStartCommand or onBind is called.

onDestroy(): This method is called when service is no longer required and is to be destroyed[1].

### 3.1.3 Content Providers

Content providers are used to provide access to other applications to a shared set of application data i.e content providers provide a mechanism through which the data stored by one Android application can be made accessible to other applications. The data is usually stored in file system or Firebase database which the application can access and through content provider, other applications can query or modify the date depending on the settings of the content provider.

A content provider is implemented as a subclass of [ContentProvider](http://developer.android.com/reference/android/content/ContentProvider.html) and an application accesses the data from a content provider with a [ContentResolver](http://developer.android.com/reference/android/content/ContentResolver.html) client object. In order to insert data into provider,  [ContentResolver.insert()](http://developer.android.com/reference/android/content/ContentResolver.html#insert(android.net.Uri, android.content.ContentValues)) method is used. This method inserts a new row into the provider and returns a content URI for that row. In order to update a row or delete a row [ContentResolver.update()](http://developer.android.com/reference/android/content/ContentResolver.html#update(android.net.Uri, android.content.ContentValues, java.lang.String, java.lang.String[])) and  [ContentResolver.delete()](http://developer.android.com/reference/android/content/ContentResolver.html#delete(android.net.Uri, java.lang.String, java.lang.String[])) methods can be used respectively[1].

### 3.1.4 Broadcast Receivers

A broadcast receiver allows registering for system or application events. When any registered event occurs, receivers for an event will be notified by the Android runtime. System broadcast include screen turning off, the battery is low, or a picture was captured. Applications broadcast would include letting other applications know that some data has been downloaded to the device and is available for them to use. A receiver can be registered via AndroidManifest.xml.

The implementing class for a receiver extends the BroadcastReceiver class. If the event for which the broadcast receiver has registered happens the onReceive() method of the receiver is called by the Android system[5].

#### 3.2 Setup Android Environment

The first step in starting a project would be to download the Android ADT bundle. The bundle provides all features needed to develop the app and also includes a version of the Eclipse IDE with built in ADT(AndroidDeveloper Tools**)** .

Perform the following steps after downloading:

1. Unpack the ZIP file (named adt-bundle-<os\_platform>.zip) and save it to an appropriate location, such as a "Development" directory in our home directory.
2. Then open the adt-bundle-<os\_platform>/eclipse/ directory and launch **eclipse**.
3. In Eclipse, click on the SDK manager and download the latest SDK tools and platforms .

### 3.3 Running the Application

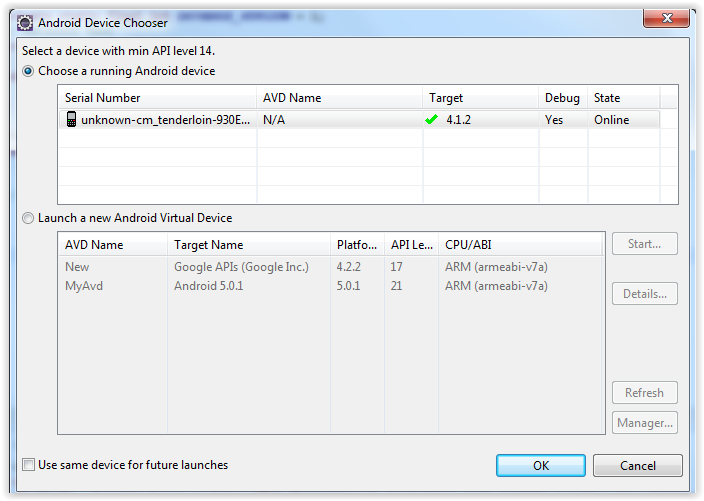
The application can be run either on device or on emulator.

##### 3.3.1. On Device

##### 

There are two ways the application can be run on device:

1. The android device needs to be connected to the laptop with USB cable. On connecting the device, option of downloading USB driver would appear in case the driver is not already installed. Then enable **USB debugging** on the device. This can be found under Settings->Developer options
2. The second way is to run the app from Eclipse:
   * On Eclipse, click on **Run**  from the toolbar.
   * In the **Run as** window that appears, need to select **Android Application** and click **OK**.

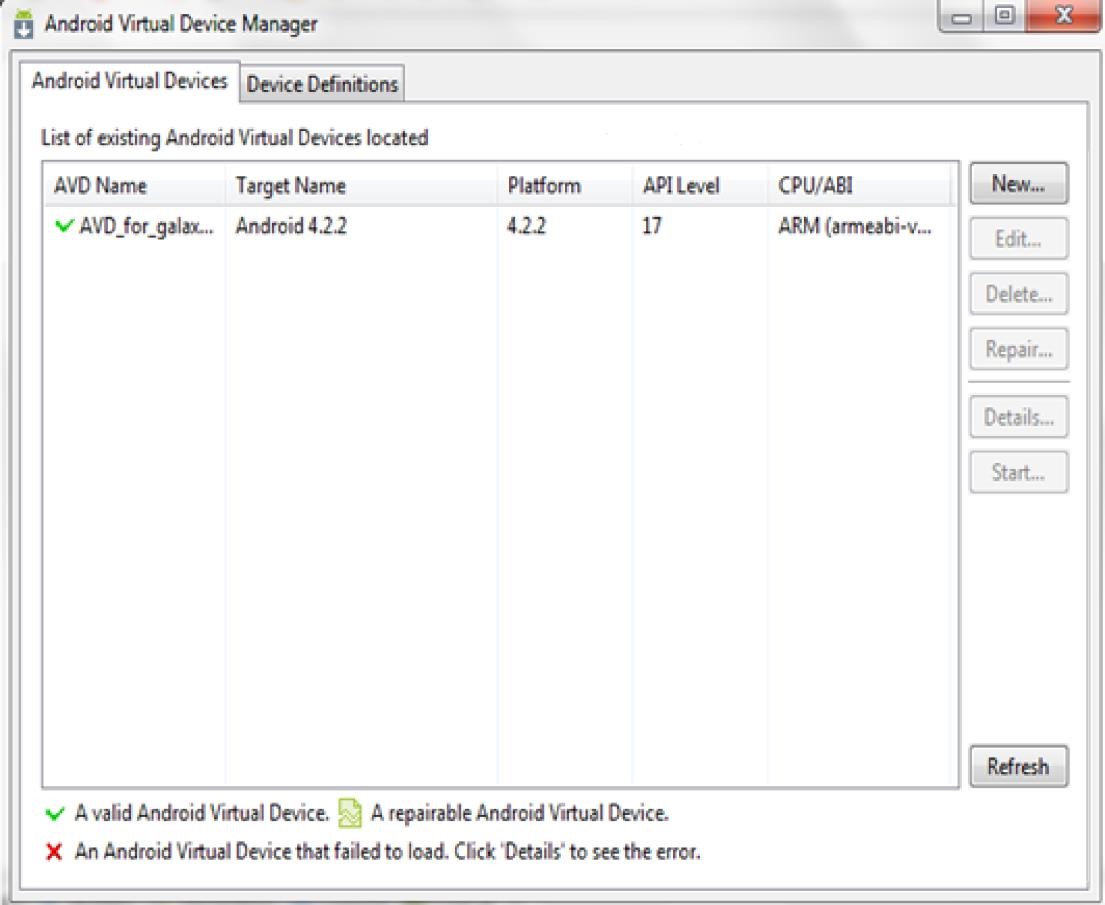


## Figure 15- Android Device Chooser

This will cause eclipse to install the app on the connected device and start it as shown in Figure 15.

### 3.4.2. On Emulator

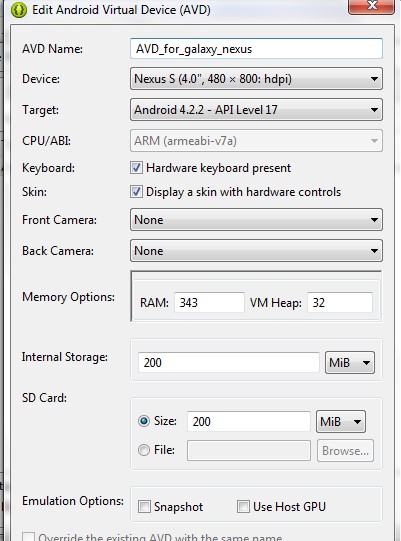
When the application needs to be run on Eclipse, then first step is to set the AVD i.e Android Virtual Device. The AVD is a device configuration for the Android emulator that allows to model different devices.



## Figure 16- Android Virtual Device Manager

In order to create AVD, first need to launch the Android Virtual Device Manager by clicking on the toolbar. The next step is to click on ‘New’ in the Android Virtual Device Manager panel shown in Figure 16.

Next step is to fill in the details for the AVD like name, a platform target, an SD card size as shown in Figure 17



## Figure 17- Android Virtual Device Edit

Click Okay. The new AVD will appear in the Android Virtual Device Manager screen. Select the new AVD and click **Start**. Wait for the emulator to boot up and then unlock the emulator screen.

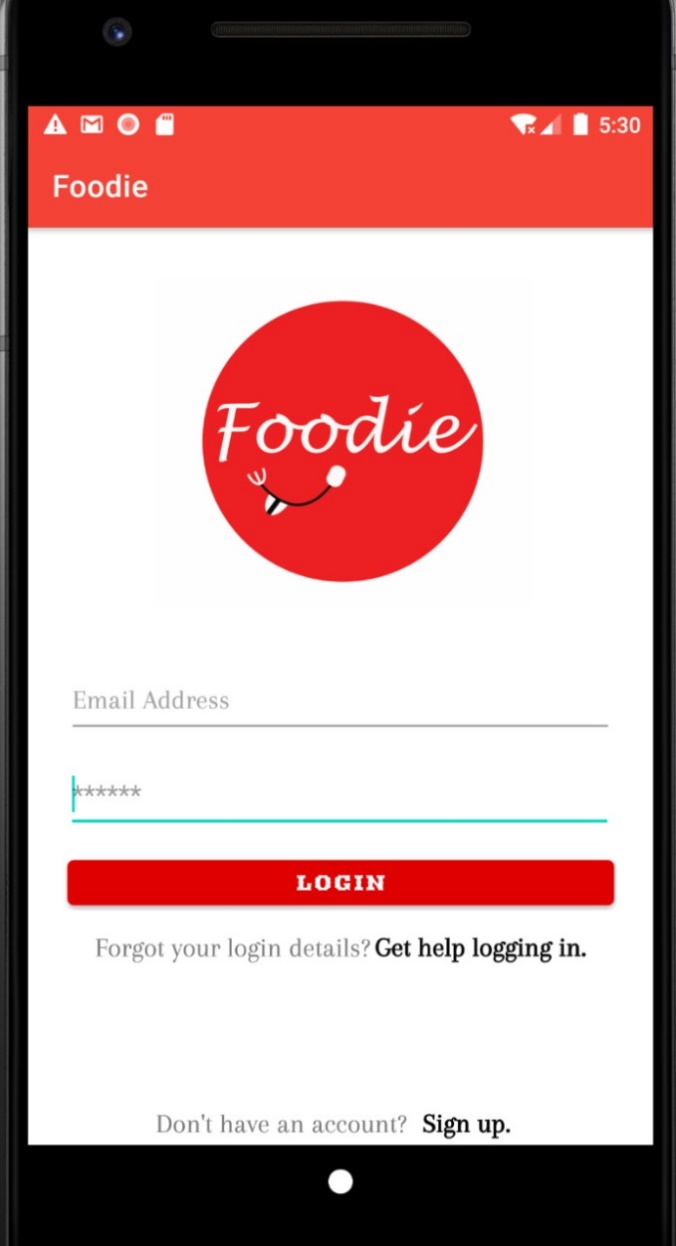
In order to run the app from Eclipse, click **Run**  from the toolbar. And then select **Android Application** and click **OK**. Eclipse will install the app on the AVD and

starts it

# 4 FEATURES IMPLEMENTATION

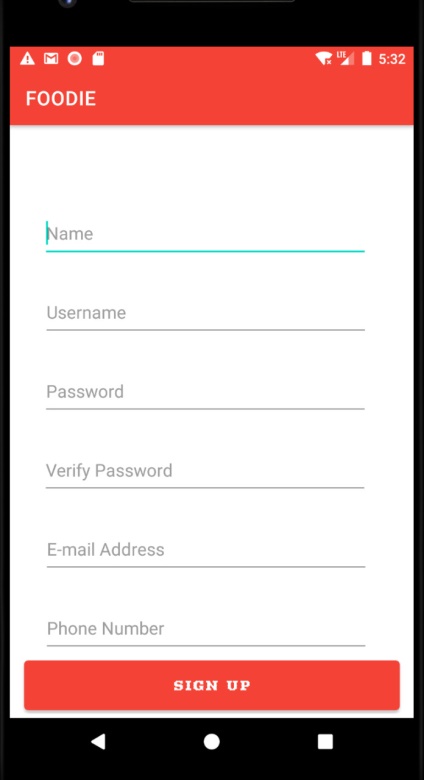
## 4.1 Application Main Page

The application starts by showing the main screen that includes various scenarios for the user to go through whether logging in , signing up or resetting his /her password.



## 4.2 Application Sign up page.

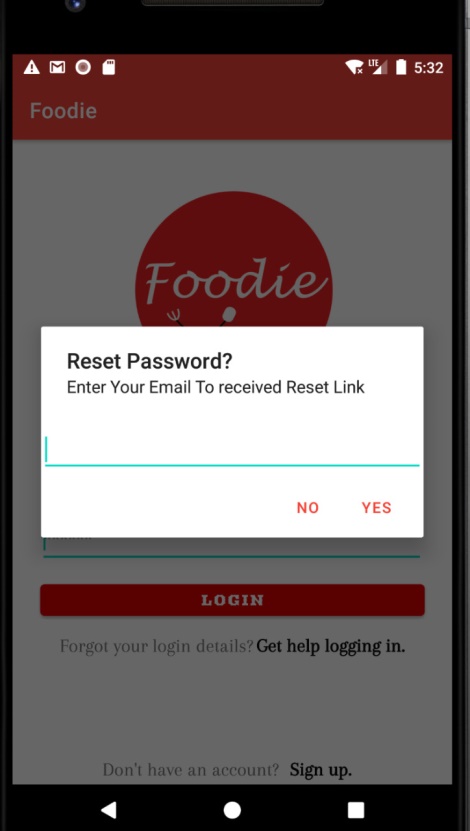
The user can create an account through filling the form below. By pressing signup a user will be created in the authentication section present in firebase. And all data will be stored in the real-time database.



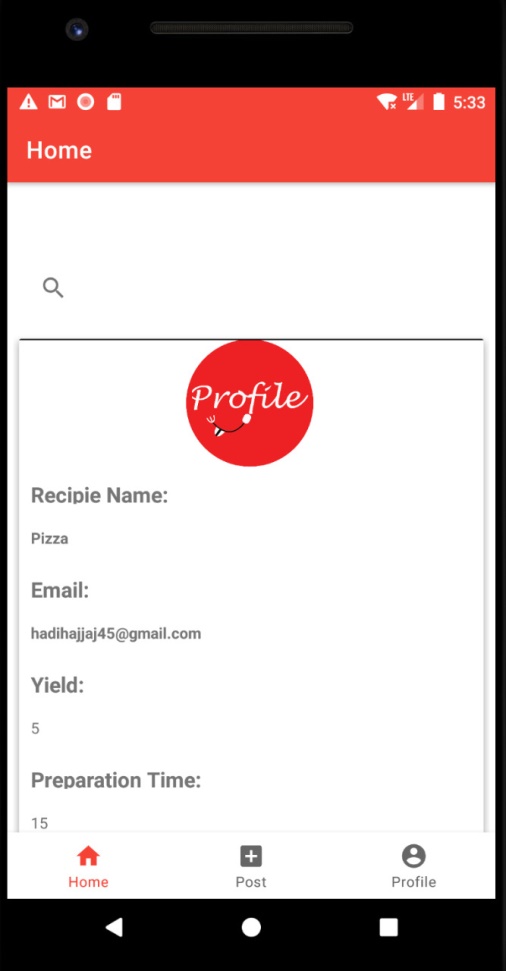
## 4.3 Reset password

In order to reset password. By pressing get help logging in a new alert will appear asking the user if he wants to resent his password via email. By accepting a message will be sent

to the user’s email that includes a reset link sent by firebase

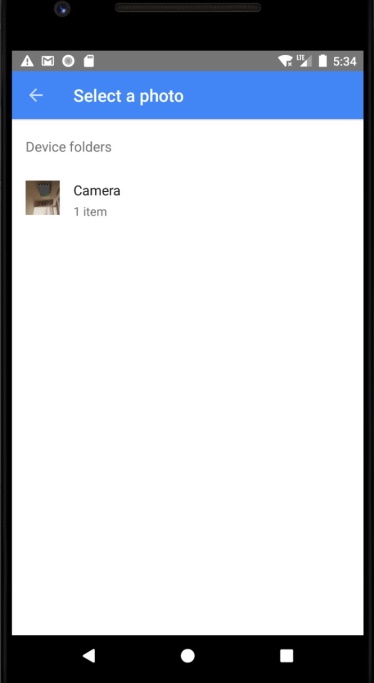


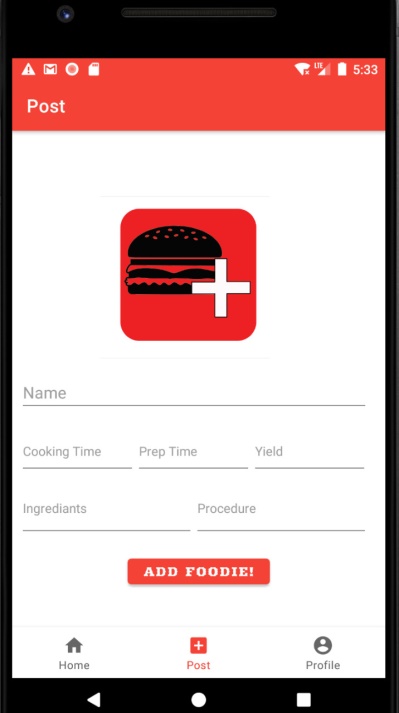
## 4.4 Home Page

Here, in home fragment ,we can surf foodie by searching for all added recipes by name.

## 4.5 Post Page

To post , the user needs to click on post icon to upload the image for his gallery . In addition , the user should fill the fields below in order to add the recipe details to the real timebase.





## 4.6 Profile Page

This is the fragment where the user can logout and see his / her account details.

