|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Student ID Number**  *(Do not include student name as anonymous marking is implemented)* | Radu Garaz / GAR19472838 | | | | | | |
| **Programme Title** | Computing Technologies Extended Degree (QA) | | | | | | |
| **Module Title** | Mobile App Design and Development | | | | | | |
| **Module Code** *(listed on Moodle and in LTAFP)* | QAC020N251A | | | | | | |
| **Module Convenor** |  | | | | | | |
| **Coursework Title** | Assignment 1 | | | | | | |
| **Academic Declaration:**  *Students are reminded that the electronic copy of their essay may be checked, at any point during their degree, with Turnitin or other plagiarism detection software for plagiarised material.* | | | | | | | |
| **Word Count** |  | | | **Date Submitted** | |  | |
|  |  | | |  | |  | |
|  | | | | | | | |

Table of Contents

[**Project Details** 5](#_Toc92137629)

[Project Name 5](#_Toc92137630)

[Introduction 5](#_Toc92137631)

[Description 5](#_Toc92137632)

[**Functional Requirements** 5](#_Toc92137633)

[User - Functional Requirements 5](#_Toc92137634)

[Critic User - Functional Requirements 6](#_Toc92137635)

[Admin User - Functional Requirements 6](#_Toc92137636)

[**Mobile Application Map** 6](#_Toc92137637)

[**Wireframes** 8](#_Toc92137638)

[Dashboard view 8](#_Toc92137639)

[Main Page Menu View 9](#_Toc92137640)

[Login View 10](#_Toc92137641)

[Register View 11](#_Toc92137642)

[Reset Password View 12](#_Toc92137643)

[Add Restaurant View 13](#_Toc92137644)

[Display Restaurant View 15](#_Toc92137645)

[Restaurant Detail View 16](#_Toc92137646)

[Display Street Food View 17](#_Toc92137647)

[Display User’s Reviews View 18](#_Toc92137648)

[Street Food Detail View 19](#_Toc92137649)

[My Account/Profile View 20](#_Toc92137650)

[Other Profile View 21](#_Toc92137651)

[**Use Case Diagram** 22](#_Toc92137652)

[**Application Design Model** 23](#_Toc92137653)

[Architectural Pattern 23](#_Toc92137654)

[Class Diagram 25](#_Toc92137655)

[Class Diagram Design 25](#_Toc92137656)

[Description of Classes 26](#_Toc92137657)

[**Firebase Design** 27](#_Toc92137658)

[Firebase’s Entity Relationship Diagram 27](#_Toc92137659)

[Authentication 28](#_Toc92137660)

[Cloud Firestore 30](#_Toc92137661)

[Cloud Storage 30](#_Toc92137662)

[**Source Code Snippets** 30](#_Toc92137663)

[SplashActivity Class 31](#_Toc92137664)

[LoginActivity Class 32](#_Toc92137665)

[SignUpActivity Class 34](#_Toc92137666)

[ForgetPasswordActivity Class 37](#_Toc92137667)

[DashboardActivity Class 38](#_Toc92137668)

[AddRestaurantActivity Class 41](#_Toc92137669)

[AddStreetFoodActivity Class 44](#_Toc92137670)

[ChatActivity Class 49](#_Toc92137671)

[RestaurantsFragment Class 51](#_Toc92137672)

[ProfileFragment Class 54](#_Toc92137673)

[AddUserByAdmin Class 59](#_Toc92137674)

[StreeFoodFragment Class 61](#_Toc92137675)

[Resources 64](#_Toc92137676)

[activity\_login.xml 64](#_Toc92137677)

[activity\_signup.xml 66](#_Toc92137678)

[activity\_forget\_password.xml 69](#_Toc92137679)

[Values 71](#_Toc92137680)

[colors.xml 71](#_Toc92137681)

[dimens.xml 71](#_Toc92137682)

[strings.xml 72](#_Toc92137683)

[Manifest.xml Permissions 73](#_Toc92137684)

[Splash View 74](#_Toc92137685)

[Signup View 75](#_Toc92137686)

[Login View 76](#_Toc92137687)

[ForgetPassword View 77](#_Toc92137688)

[Menu View 78](#_Toc92137689)

[DashBoard View 79](#_Toc92137690)

[Restaurant Detail View 80](#_Toc92137691)

[Reviews View 81](#_Toc92137692)

[Add rating View 82](#_Toc92137693)

[Reservation View 83](#_Toc92137694)

[Profile View 84](#_Toc92137695)

[Street Food View 85](#_Toc92137696)

[Add Street Food Stall View 86](#_Toc92137697)

[Chat Channel View 87](#_Toc92137698)

[Chat View 88](#_Toc92137699)

[**Test Documentation** 89](#_Toc92137700)

[Testing- user 89](#_Toc92137701)

[Testing- Critic user 91](#_Toc92137702)

[Testing- Admin user 94](#_Toc92137703)

[**Appendix:** 97](#_Toc92137704)

[GitHub Page of the project: 97](#_Toc92137705)

[https://github.com/tudoraioanei/For-Foodies-By-Foodies.git 97](#_Toc92137706)

[Archived project: 97](#_Toc92137707)

[References 98](#_Toc92137708)

[**Bibliography:** 99](#_Toc92137709)

# **Project Details**

## Project Name

For Foodies by Foodies

## Introduction

One of the assignments of this semester was the preparation of a group project, which will be presented as a mobile application. From the beginning, it seemed like a challenge for all of us, but at the end, the conclusion was that we just came out from our comfort zone to learn something new. As stated in Bassot (2016:15), “practice involves examining ourselves to see our practice more clearly and giving some serious consideration to what we see”, this gives us the opportunity to become more self-aware and to prepare ourselves to cope with different challenges.

## Description

Mecachrome Software Ltd has employed us for creating a new application called” For Foodies by Foodies (FFBF)”. The purpose of the app is to provide a platform to facilitate the user to find out awesome food restaurant and street food in London which provide tasty dishes in normal budget. Also, they can share the reviews and discuss with other users in our global platform.

# **Functional Requirements**

## User - Functional Requirements

* Can sign in, after registration
* Can view restaurants
* Can view restaurant review
* Can read about restaurant
* Can see the reservation table
* Can chat with another user
* Edit its profile
* Can see others profile
* Can’t add a restaurant
* Can’t edit others profile
* Can’t add restaurant review

## Critic User - Functional Requirements

* Can view restaurants
* Can read details of restaurant
* Can add review to the restaurant
* Can login
* Can sign out
* Can recover the password
* Can rate the user review
* Can’t register as critic
* Can’t add the restaurant

## Admin User - Functional Requirements

* Can add new critic and admin
* Can edit its profile
* Can see other profile
* Can add new restaurant
* Can add review
* Can delete review
* Can add street food
* Can chat with another user

# **Mobile Application Map**

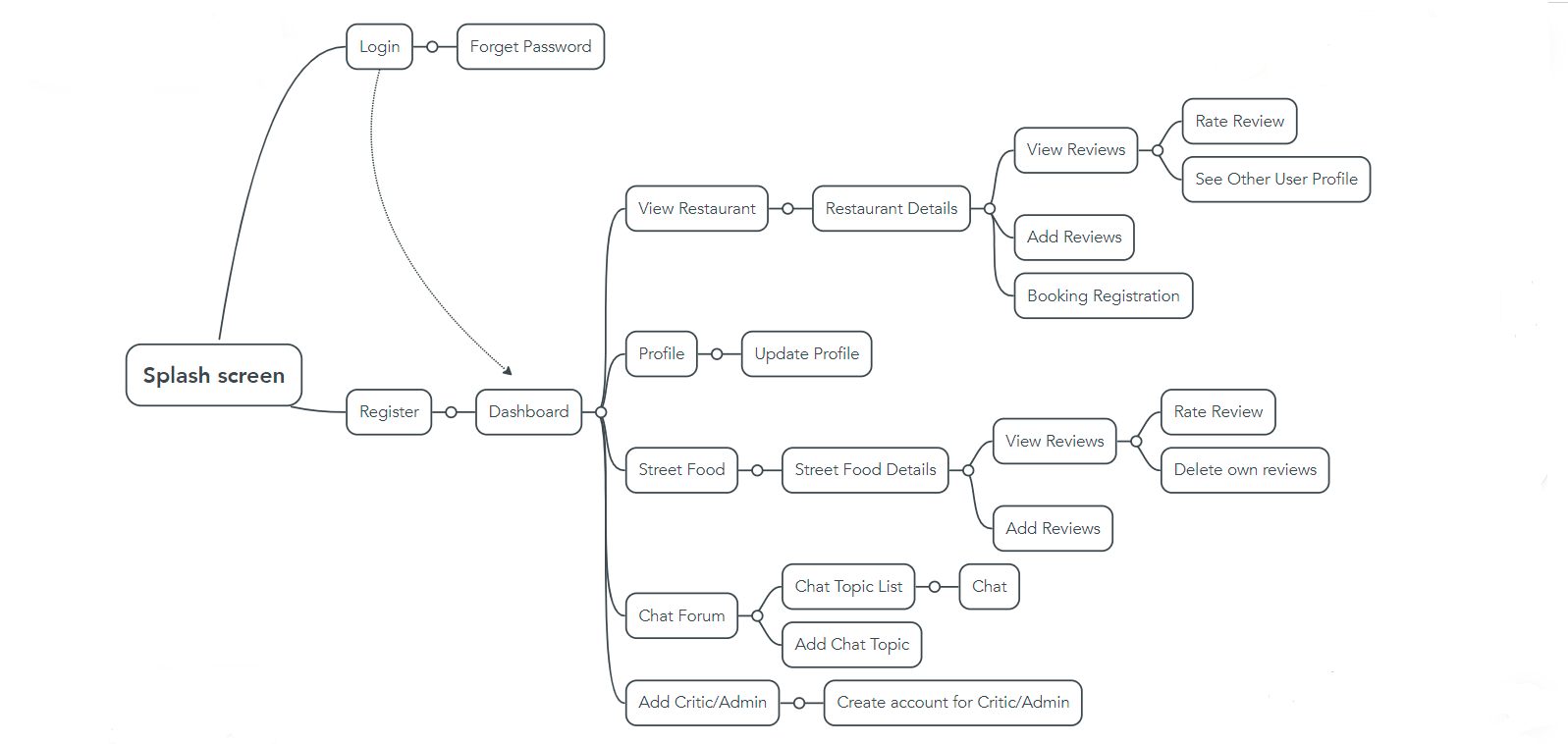


Figure 1

# **Wireframes**

## Dashboard view

Graphical user interface

Description automatically generated

Figure 2

## Main Page Menu View



Figure 3

## 

## Login View



Figure 4

## Register View

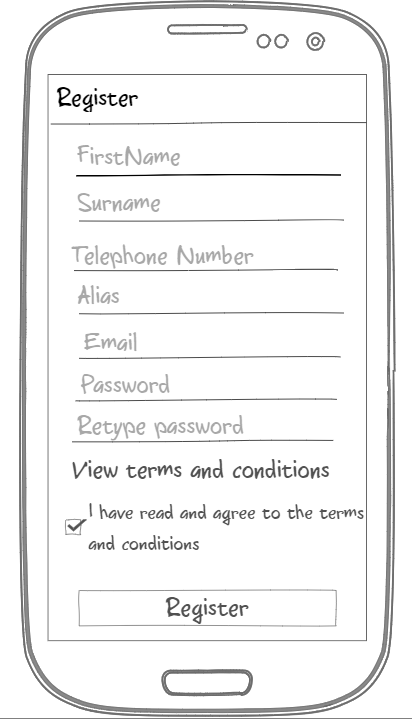


Figure 5

## 

## Reset Password View

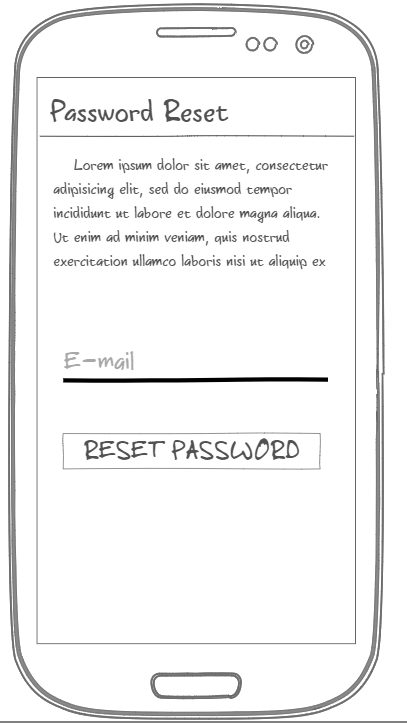


Figure 6

## Add Restaurant View

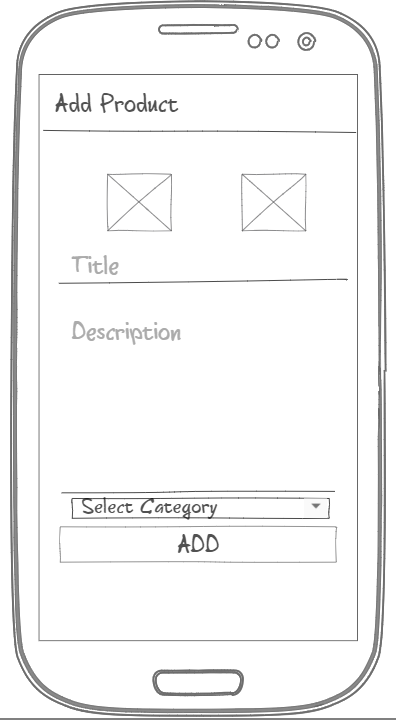
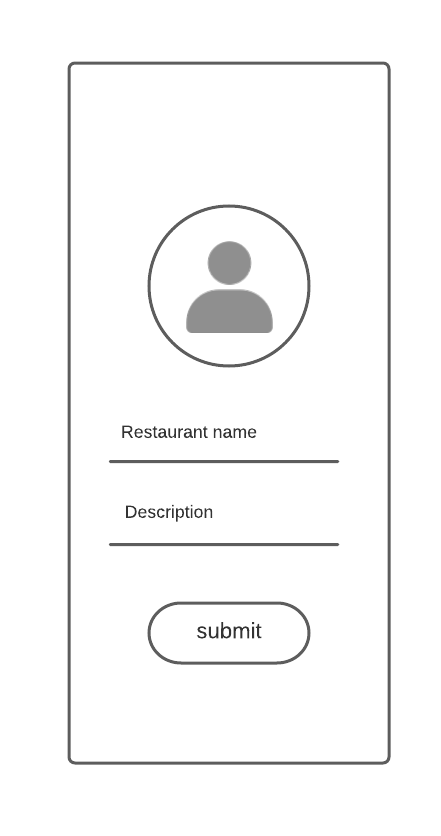


Figure 7



## Display Restaurant View

Graphical user interface

Description automatically generated

Figure 8

## Restaurant Detail View

Text

Description automatically generated

Figure 9

## Display Street Food View

Graphical user interface

Description automatically generated

Figure 10

## 

## Display User’s Reviews View

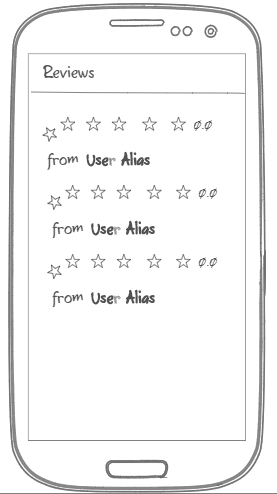


Figure 11

## Street Food Detail View

Graphical user interface, text, application

Description automatically generated

Figure 12

## My Account/Profile View

Graphical user interface

Description automatically generated with low confidence

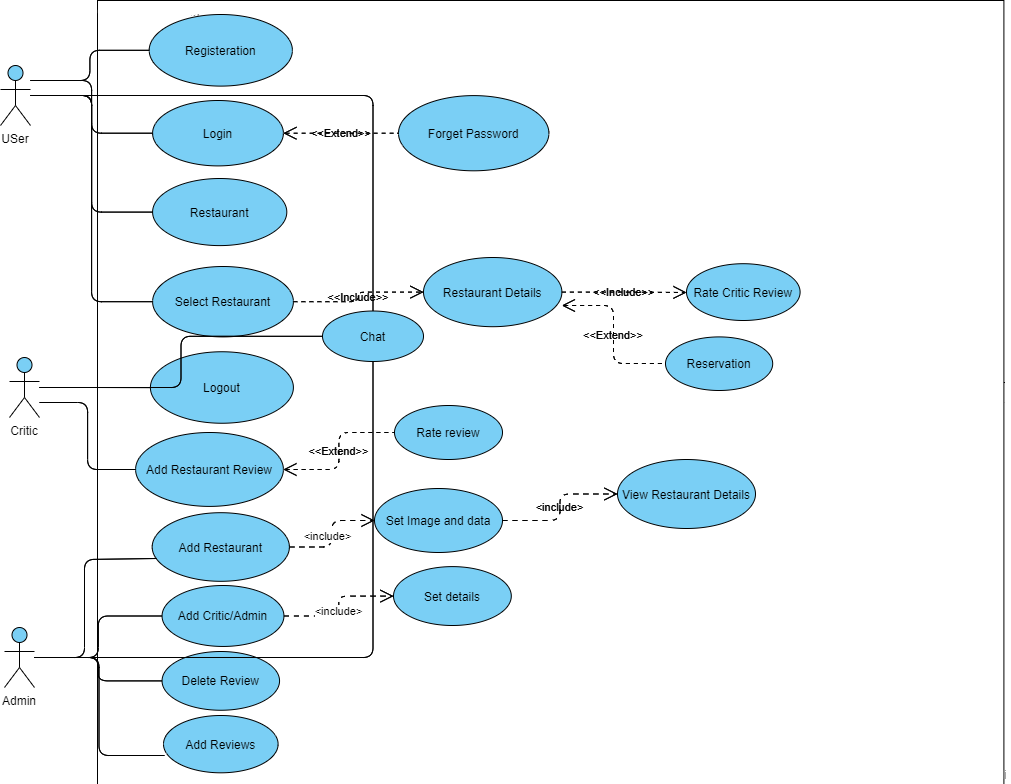
Figure 13

## Other Profile View

Graphical user interface

Description automatically generated with medium confidence

# **Use Case Diagram**



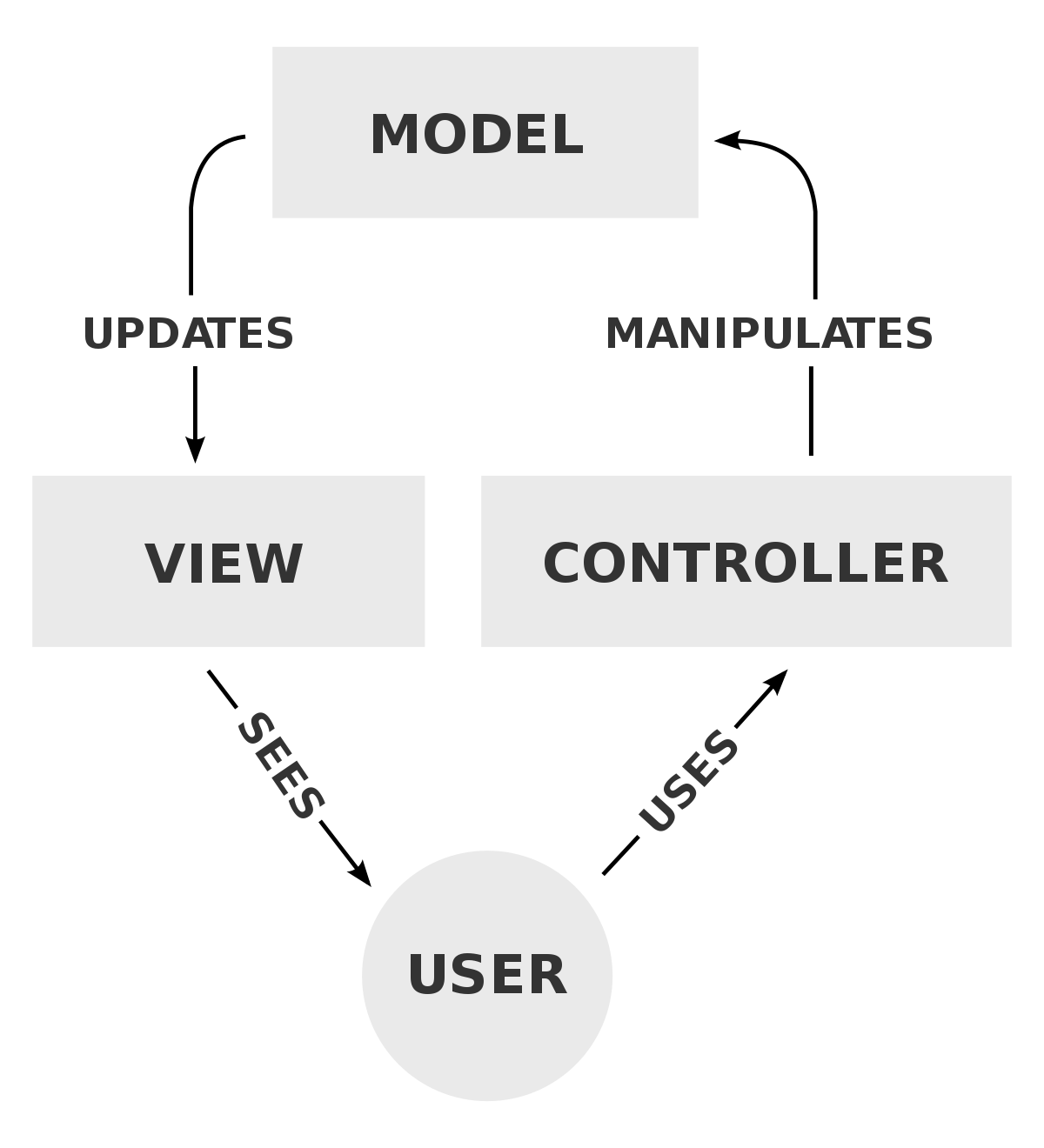
# 

# **Application Design Model**

## Architectural Pattern

We’ve chosen the MVC Model because it’s easy to implement and most of the developers use this architecture. It is used to separate the views from the model and controllers (Google LLC, 2019).

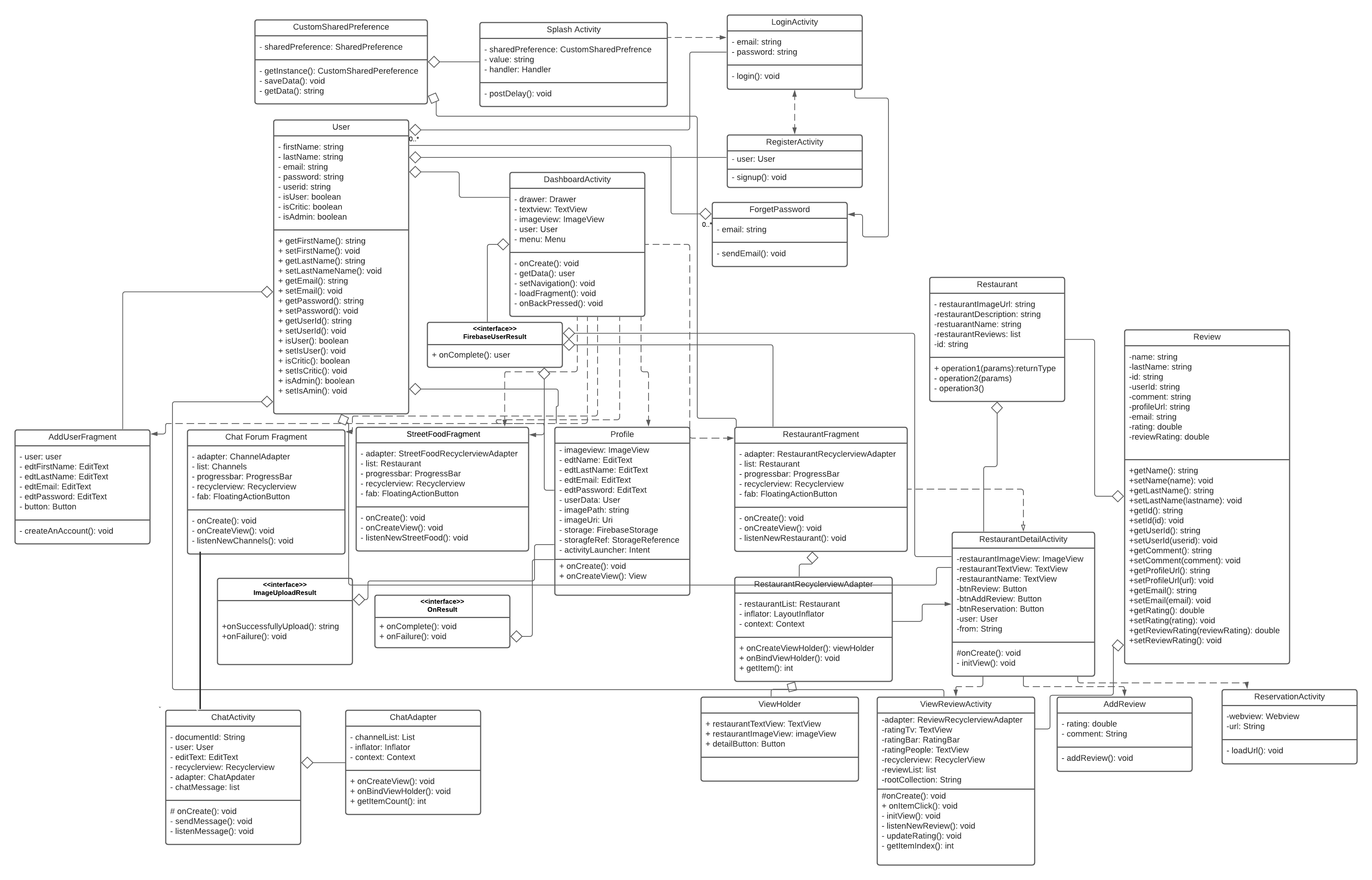
* Model: Besides triggering data changes, the model also contains entity classes used for structuring the data in the application and in the database (Microsoft Corp, 2019).
* View: The layer includes all UI components. When controller gets data from model it will automatically updated the Ui (Microsoft Corp, 2019).
* Controller: This layer is used to make request to model for data and renders it to the required Ui, so that we can see real time data changes like chatting (Google LLC, 2019).



Disclosure: diagrams, pictures or any other images used in this assignment are for educational purposes only.

# Class Diagram

Class Diagram Design



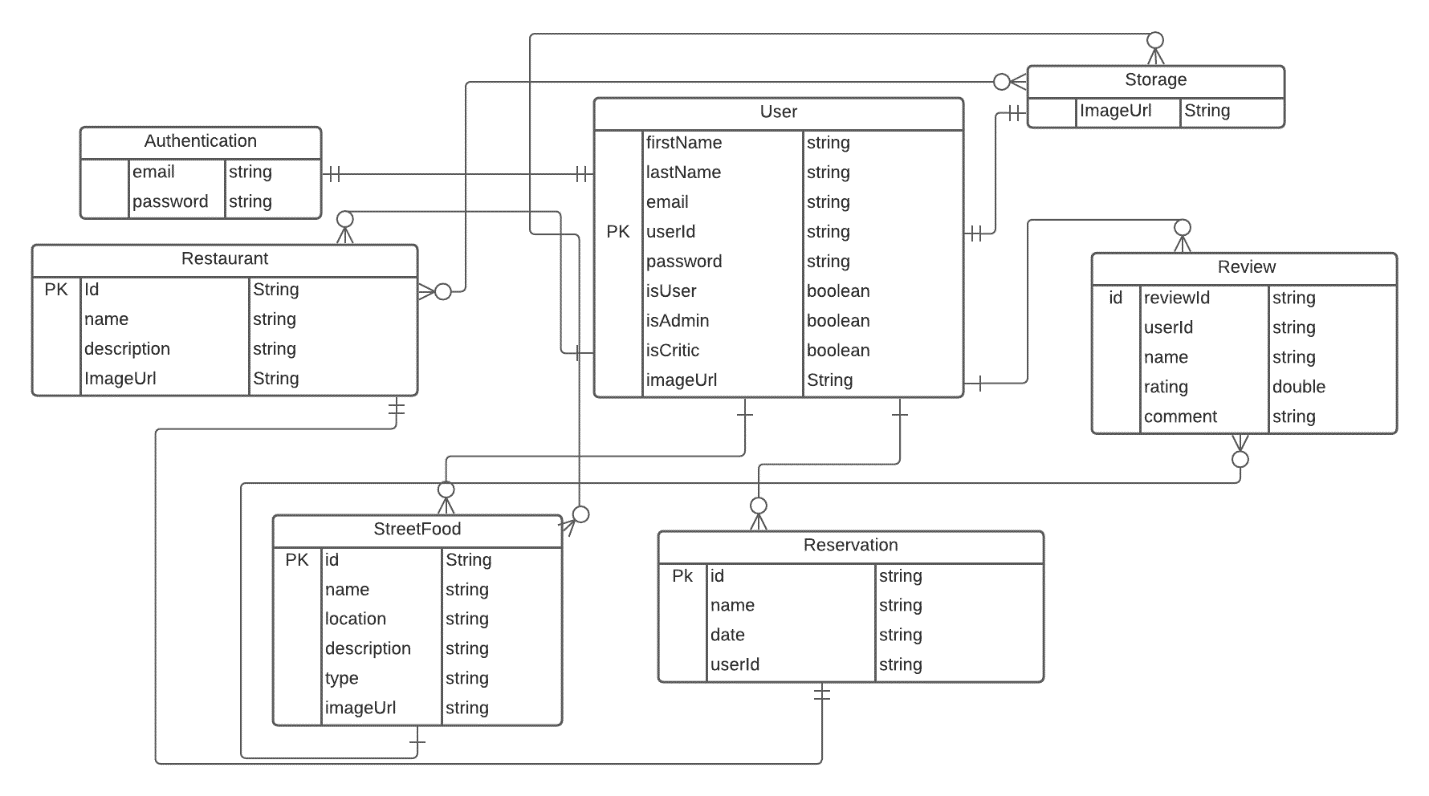
### Description of Classes

|  |  |
| --- | --- |
| **Class Name** | **Description** |
| Splash Activity | It is the launcher activity and show a splash screen (static view) for 3 seconds and move to dashboard if user is already login otherwise navigate to login screen. |
| Login Activity | It is use for Login purpose that require an email and password. |
| Signup Activity | This activity is use for user registration that requires first name, last name, email, and password. |
| Forget Password Activity | Use to send an email to reset their password. |
| Dashboard Activity | This activity is used to set the navigation drawer from which we can replace different fragments on a same activity. |
| Add Restuarant | In this activity only admin can add the restaurant. |
| Add Street Food Activity | This activity is use for adding new street food stalls to database by the user. |
| Other User Activity | This activity shows the other user profile data like profile pic email and name etc. |
| Review Activity | This activity gets all the review to the specific restaurant or street food stall. |
| Add user Fragment | This fragment uses to add new critics and admin by only admin. |
| Chat Forum Fragment | In this fragment here is the list of topics on which user can do discussions. |
| Profile Fragment | Here user can see its profile data and update it. |
| Restaurant Fragment | This fragment is used to show list of restaurants in alphabetic order. |
| Street food Fragment | It is used to Show all street food stall. |

# **Firebase Design**

## Firebase’s Entity Relationship Diagram

Figure 24



## 

## Authentication

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Cloud Firestore

Graphical user interface, text

Description automatically generated

## Cloud Storage

The Firebase Firestorage is used to store the image and video of the products and to provide URI links for each added item (Google LLC, 2019) . Because the Firestorage’s multimedia content is used in parallel with the products the rules are set that anyone can read the content, but only authenticated users can have written access.

# 

# **Source Code Snippets**

## SplashActivity Class

package com.devexpert.forfoodiesbyfoodies;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.os.Handler;  
import android.view.WindowManager;  
  
import com.devexpert.forfoodiesbyfoodies.activities.DashBoardActivity;  
import com.devexpert.forfoodiesbyfoodies.activities.LoginActivity;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
  
public class SplashScreen extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 getWindow().addFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN); //hide the status bar and show the splash screen full  
 setContentView(R.layout.activity\_splash);  
  
 //getting user id from local storage  
 CustomSharedPreference sharedPreference = CustomSharedPreference.getInstance(getApplicationContext());  
 String value = sharedPreference.getData(Constants.userId);  
  
 Handler handler = new Handler();  
 handler.postDelayed(() -> { //running a thread that complete after 3 seconds and move to specifc screen  
 Intent intent;  
 if (value.isEmpty()) { //if user id is null move to login screen  
 intent = new Intent(getApplicationContext(), LoginActivity.class);  
 } else { //otherwise move to dashboard  
 intent = new Intent(getApplicationContext(), DashBoardActivity.class);  
 }  
 startActivity(intent);  
 finish();  
 }, 3000);  
 }  
  
}

## LoginActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.firebase.auth.FirebaseAuth;  
  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
  
public class LoginActivity extends AppCompatActivity {  
  
 private EditText emailField, passwordField;  
 private Button loginButton;  
 private ProgressBar progressBar;  
 private FirebaseAuth auth;  
 private TextView signUpTextView, forgetPasswordTextView;  
 private CustomSharedPreference yourPreference;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login*);  
 initView();  
 yourPreference = CustomSharedPreference.*getInstance*(getApplicationContext());  
 //Get Firebase auth instance  
 auth = FirebaseAuth.*getInstance*();  
  
 loginButton.setOnClickListener(view -> login()); //call login function when click ob login button  
 //navigate to signUp activity  
 signUpTextView.setOnClickListener(view -> startActivity(new Intent(this, SignUpActivity.class)));  
 forgetPasswordTextView.setOnClickListener(view-> startActivity(new Intent(this, ForgetPasswordActivity.class)));  
 }  
  
 //initialize view  
 void initView() {  
 emailField = findViewById(R.id.*emailField\_id*);  
 passwordField = findViewById(R.id.*passwordField\_id*);  
 loginButton = findViewById(R.id.*loginBtn\_id*);  
 progressBar = findViewById(R.id.*progressBar\_id*);  
 signUpTextView = findViewById(R.id.*signUpTextView\_id*);  
 forgetPasswordTextView = findViewById(R.id.*forgetPasswordTextView\_id*);  
 }  
  
 //Function for login  
 void login() {  
 CommonFunctions.*hideKeyboard*(this);  
 String email = emailField.getText().toString().trim();  
 String password = passwordField.getText().toString().trim();  
  
 //check validity before login  
 if (email.isEmpty() || password.isEmpty()) {  
 CommonFunctions.*showToast*("Please fill the data properly.", getApplicationContext());  
 } else {  
 if (CommonFunctions.*isEmailValid*(email)) {  
  
 if (password.length() >= 8) {  
 //login procedure  
 progressBar.setVisibility(View.*VISIBLE*);  
 //authenticate user  
 auth.signInWithEmailAndPassword(email, password)  
 .addOnCompleteListener(LoginActivity.this, task -> {  
 progressBar.setVisibility(View.*GONE*);  
 if (!task.isSuccessful()) {  
 CommonFunctions.*customLog*("Login Error: " + task.getException().getMessage());  
 CommonFunctions.*showToast*("Something went wrong.", getApplicationContext());  
 } else {  
 yourPreference.saveData(Constants.*userId*, task.getResult().getUser().getUid());  
 Intent intent = new Intent(LoginActivity.this, DashBoardActivity.class);  
 startActivity(intent);  
 finish();  
 }  
 });  
  
 } else {  
 CommonFunctions.*showToast*("Password should contain at-least 8 characters.", getApplicationContext());  
 }  
 } else {  
 CommonFunctions.*showToast*("Email is not valid", getApplicationContext());  
 }  
 }  
 }  
  
}

## SignUpActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.models.User;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.firebase.auth.FirebaseAuth;  
  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
  
public class SignUpActivity extends AppCompatActivity {  
  
 private EditText firstNameField, lastNameField, emailField, passwordField;  
 private Button signUpButton;  
 private TextView loginTextView;  
 private ProgressBar progressBar;  
 private FirebaseAuth auth;  
 private CustomSharedPreference sharedPreference ;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_signup*);  
 initView();  
 sharedPreference = CustomSharedPreference.*getInstance*(getApplicationContext());  
 //Get Firebase auth instance  
 auth = FirebaseAuth.*getInstance*();  
  
 signUpButton.setOnClickListener(view -> signUp()); //for signUp  
 loginTextView.setOnClickListener(view -> finish()); // go back to login screen  
 }  
  
 //initializing view  
 void initView() {  
 firstNameField = findViewById(R.id.*firstNameField\_id*);  
 lastNameField = findViewById(R.id.*lastNameField\_id*);  
 emailField = findViewById(R.id.*emailField\_id*);  
 passwordField = findViewById(R.id.*passwordField\_id*);  
 signUpButton = findViewById(R.id.*signUpBtn\_id*);  
 loginTextView = findViewById(R.id.*loginTextView\_id*);  
 progressBar = findViewById(R.id.*progressBar\_id*);  
  
 }  
  
 //Function for signUp  
 void signUp() {  
 String firstName = firstNameField.getText().toString().trim();  
 String lastName = lastNameField.getText().toString().trim();  
 String email = emailField.getText().toString().trim();  
 String password = passwordField.getText().toString().trim();  
 //check validity  
 if (firstName.isEmpty() || lastName.isEmpty() || email.isEmpty() || password.isEmpty()) {  
 CommonFunctions.*showToast*("Please fill the data properly.", getApplicationContext());  
  
 } else {  
 if (CommonFunctions.*isEmailValid*(email)) {  
  
 if (password.length() >= 8) {  
 progressBar.setVisibility(View.*VISIBLE*);  
  
 auth.createUserWithEmailAndPassword(email, password)  
 .addOnCompleteListener(this, task -> {  
 progressBar.setVisibility(View.*GONE*);  
 if (!task.isSuccessful()) {  
 CommonFunctions.*showToast*("Login failed." + task.getException(), getApplicationContext());  
  
 } else {  
 User user = new User(firstName, lastName, email, task.getResult().getUser().getUid(), password, Constants.*defaultImageUrl*,  
 true, false, false);  
 //after signUp save data to firebase  
 FireStore.*addUserToFireStore*(user);  
 //save user uuid in local device  
 sharedPreference.saveData(Constants.*userId*, task.getResult().getUser().getUid());  
 CommonFunctions.*showToast*("Login Successful.", getApplicationContext());  
 //navigate to Dashboard screen  
 startActivity(new Intent(getApplicationContext(), DashBoardActivity.class));  
 finish();  
 }  
 });  
 CommonFunctions.*showToast*("Successfully SignUp", getApplicationContext());  
 } else {  
 CommonFunctions.*showToast*("Password should contain at-least 8 characters.", getApplicationContext());  
 }  
 } else {  
 CommonFunctions.*showToast*("Email is not valid", getApplicationContext());  
 }  
 }  
 }  
}

## ForgetPasswordActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.text.TextUtils;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.Toast;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.google.firebase.auth.FirebaseAuth;  
  
public class ForgetPasswordActivity extends AppCompatActivity {  
 private EditText inputEmail;  
  
 private FirebaseAuth auth;  
  
 private ProgressBar progressBar;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
  
 setContentView(R.layout.activity\_forget\_password);  
  
 inputEmail = findViewById(R.id.email);  
  
 Button btnReset = findViewById(R.id.btn\_reset\_password);  
  
 Button btnBack = findViewById(R.id.btn\_back);  
  
 progressBar = findViewById(R.id.progressBar);  
  
 auth = FirebaseAuth.getInstance();  
  
 btnBack.setOnClickListener(v -> finish());  
  
 btnReset.setOnClickListener(v -> {  
  
 String email = inputEmail.getText().toString().trim();  
  
 if (TextUtils.isEmpty(email)) {  
 Toast.makeText(getApplication(), "Enter your registered email id", Toast.LENGTH\_SHORT).show();  
 return;  
 }  
  
 progressBar.setVisibility(View.VISIBLE);  
  
 auth.sendPasswordResetEmail(email)  
  
 .addOnCompleteListener(task -> {  
 if (task.isSuccessful()) {  
 Toast.makeText(getApplicationContext(), "We have sent you instructions to reset your password!", Toast.LENGTH\_SHORT).show();  
 } else {  
 Toast.makeText(getApplicationContext(), "Failed to send reset email!", Toast.LENGTH\_SHORT).show();  
 }  
  
 progressBar.setVisibility(View.GONE);  
 });  
 });  
 }  
  
}

## DashboardActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.appcompat.app.ActionBarDrawerToggle;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.appcompat.widget.Toolbar;  
import androidx.drawerlayout.widget.DrawerLayout;  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentTransaction;  
  
import android.app.AlertDialog;  
import android.os.Bundle;  
import android.view.Menu;  
import android.view.View;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.fragments.AddUserByAdminFragment;  
import com.devexpert.forfoodiesbyfoodies.fragments.ChatForumFragment;  
import com.devexpert.forfoodiesbyfoodies.fragments.ProfileFragment;  
import com.devexpert.forfoodiesbyfoodies.fragments.RestaurantsFragment;  
import com.devexpert.forfoodiesbyfoodies.fragments.StreetFoodFragment;  
import com.devexpert.forfoodiesbyfoodies.models.User;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.android.material.navigation.NavigationView;  
import com.squareup.picasso.Picasso;  
  
  
public class DashBoardActivity extends AppCompatActivity {  
 private DrawerLayout dLayout;  
 private TextView textViewName;  
 private ImageView imageView;  
 private User userData;  
 private Menu menu;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_dash\_board*);  
  
 //get userId from sharedPreference  
 CustomSharedPreference yourPreference = CustomSharedPreference.*getInstance*(getApplicationContext());  
 String userId = yourPreference.getData(Constants.*userId*);  
  
 //gets user data from firestore  
 FireStore.*getData*(userId, user -> {  
 userData = user;  
 if (user.getFirstName().isEmpty()) {  
 textViewName.setText(R.string.*user\_name*);  
 } else {  
 textViewName.setText(user.getFirstName().concat(" ").concat(user.getLastName()));  
 Picasso.*get*().load(user.getImageUrl()).fit().centerCrop().  
 placeholder(R.drawable.*placeholder\_image*)  
 .error(R.drawable.*error\_image*).into(imageView);  
 if (userData.isAdmin()) {  
 menu.setGroupCheckable(R.id.*second\_group*, true, true);  
 menu.setGroupVisible(R.id.*second\_group*, true);  
 }  
  
 }  
 });  
 //set drawer (left side menu)  
 setNavigationDrawer(); // call method  
 }  
  
 private void setNavigationDrawer() {  
 Toolbar toolbar = findViewById(R.id.*toolbar*);  
 setSupportActionBar(toolbar);  
  
 dLayout = findViewById(R.id.*drawer\_layout*); // initiate a DrawerLayout  
  
 //toggle button in appbar  
 ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, dLayout, toolbar, R.string.*main\_drawer\_open*, R.string.*main\_drawer\_close*);  
 dLayout.addDrawerListener(toggle);  
 toggle.setDrawerIndicatorEnabled(true);  
 toggle.syncState();  
  
 NavigationView navView = findViewById(R.id.*navigation*); // initiate a Navigation View  
 View headerView = navView.getHeaderView(0);  
 textViewName = headerView.findViewById(R.id.*name*);  
 imageView = headerView.findViewById(R.id.*profile\_image*);  
  
 loadFragment(new RestaurantsFragment()); //default fragment is Restaurant fragment  
 menu = navView.getMenu();  
  
  
 // implement setNavigationItemSelectedListener event on NavigationView  
 navView.setNavigationItemSelectedListener(menuItem -> {  
 Fragment frag; // create a Fragment Object  
 int itemId = menuItem.getItemId(); // get selected menu item's id  
 // check selected menu item's id and replace a Fragment Accordingly  
 if (itemId == R.id.*restaurantFragment\_id*) {  
 frag = new RestaurantsFragment();  
 } else if (itemId == R.id.*profileFragment\_id*) {  
 frag = new ProfileFragment();  
 } else if (itemId == R.id.*streetFoodFragment\_id*) {  
 frag = new StreetFoodFragment(userData);  
 } else if (itemId == R.id.*addCritics\_id*) {  
 frag = new AddUserByAdminFragment();  
 } else if (itemId == R.id.*channelFragment\_id*) {  
 frag = new ChatForumFragment();  
 } else {  
 frag = new RestaurantsFragment();  
 }  
 loadFragment(frag);  
 return false;  
 });  
 }  
  
 public void loadFragment(Fragment fragment) {  
 FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();  
 transaction.replace(R.id.*frame*, fragment);  
 transaction.commit();  
 dLayout.closeDrawers(); // close the all open Drawer Views  
  
 }  
  
 @Override  
 public void onBackPressed() {  
 //when you click back on dashboard it will ask you to exist or not  
 AlertDialog.Builder builder = new AlertDialog.Builder(this);  
 builder.setTitle("Alert");  
 builder.setMessage("Do you want to Exist? ");  
 builder.setPositiveButton("Yes", (dialog, id) -> finish());  
 builder.setNegativeButton("No", (dialog, id) -> {  
 });  
 builder.show();  
 }  
}

## AddRestaurantActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.app.Activity;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.interfaces.ImageUploadResult;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
  
import java.io.InputStream;  
  
public class AddRestaurantActivity extends AppCompatActivity {  
 private ImageView imageView;  
 private EditText edtRestaurantName, edtRestaurantDescp;  
 private Button btnSubmit;  
 private String imagePath = "";  
 private Uri imageUri;  
 private ActivityResultLauncher<Intent> someActivityResultLauncher;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_add\_restaurant);  
 initView();  
  
 // when click on image view it open the gallery to select an image  
 imageView.setOnClickListener(view -> {  
 Intent intent = new Intent();  
 intent.setType("image/\*");  
 intent.setAction(Intent.ACTION\_GET\_CONTENT);  
 someActivityResultLauncher.launch(intent);  
 });  
  
 //after selecting image from gallery this will return you the result (image path)  
 someActivityResultLauncher = registerForActivityResult(  
 new ActivityResultContracts.StartActivityForResult(),  
 result -> {  
 if (result.getResultCode() == Activity.RESULT\_OK) {  
 Intent data = result.getData();  
 if (data == null) {  
 return;  
 }  
 try {  
 Uri uri = data.getData();  
 CommonFunctions.customLog(data.getData().toString());  
 CommonFunctions.customLog(uri.getPath());  
 final InputStream imageStream = getContentResolver().openInputStream(uri);  
 final Bitmap selectedImage = BitmapFactory.decodeStream(imageStream);  
 imageView.setImageBitmap(selectedImage);  
 imagePath = uri.getPath();  
 imageUri = uri;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 });  
 btnSubmit.setOnClickListener(view -> onSubmitData());  
  
 }  
  
 //initializing view  
 void initView() {  
 imageView = findViewById(R.id.userImageView\_id);  
 edtRestaurantName = findViewById(R.id.edtRestaurantName\_id);  
 edtRestaurantDescp = findViewById(R.id.edtRestaurantDesp\_id);  
 btnSubmit = findViewById(R.id.btnSubmit\_id);  
 }  
  
 //First check data is filled and then upload data to firebase  
 void onSubmitData() {  
 String name = edtRestaurantName.getText().toString().trim();  
 String description = edtRestaurantDescp.getText().toString().trim();  
 if (name.isEmpty()) {  
 CommonFunctions.showToast("Please add restaurant name", getApplicationContext());  
 return;  
 }  
 if (description.isEmpty()) {  
 CommonFunctions.showToast("Please add restaurant description", getApplicationContext());  
 return;  
 }  
 if (imagePath.isEmpty()) {  
 CommonFunctions.showToast("Please select any image", getApplicationContext());  
 return;  
 }  
 CommonFunctions.uploadImage(imagePath, this, imageUri, new ImageUploadResult() {  
 @Override  
 public void onUploadSuccess(String imageUrl) {  
 FireStore.addRestaurant(imageUrl, description, name);  
 finish();  
 }  
  
 @Override  
 public void onUploadFailure() {  
 CommonFunctions.showToast("Something went wrong", getApplicationContext());  
 }  
 });  
 }  
}

## AddStreetFoodActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.app.Activity;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.RadioGroup;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.interfaces.ImageUploadResult;  
import com.devexpert.forfoodiesbyfoodies.interfaces.OnResult;  
import com.devexpert.forfoodiesbyfoodies.models.StreetFood;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
  
import java.io.InputStream;  
  
public class AddStreetFoodActivity extends AppCompatActivity {  
 private ImageView imageView;  
 private EditText edtName, edtDescription, edtLocation;  
 private RadioGroup radioGroup;  
 private String foodType = Constants.vegetarian;  
 private String imagePath = "";  
 private static final int PICK\_IMAGE = 1;  
 private Button btnSubmit;  
 private String userId = "";  
 private Uri imageUri;  
 private ActivityResultLauncher<Intent> someActivityResultLauncher;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_add\_street\_food);  
 initView();  
 //get uuid of user from firebase  
 userId = FireStore.getCurrentUserUUid();  
 CommonFunctions.customLog("UserId: "+userId);  
 btnSubmit.setOnClickListener(view -> submitData());  
  
 //click on image view to select image for street food from gallery  
 imageView.setOnClickListener(view -> {  
 Intent intent = new Intent();  
 intent.setType("image/\*");  
 intent.setAction(Intent.ACTION\_GET\_CONTENT);  
 someActivityResultLauncher.launch(intent);  
 });  
 // You can do the assignment inside onAttach or onCreate, i.e, before the activity is displayed  
 someActivityResultLauncher = registerForActivityResult(  
 new ActivityResultContracts.StartActivityForResult(),  
 result -> {  
 if (result.getResultCode() == Activity.RESULT\_OK) {  
 // There are no request codes  
 Intent data = result.getData();  
 if (data == null) {  
 //error  
 return;  
 }  
 try {  
 Uri uri = data.getData();  
 CommonFunctions.customLog(data.getData().toString());  
 CommonFunctions.customLog(uri.getPath());  
 final InputStream imageStream = getContentResolver().openInputStream(uri);  
 final Bitmap selectedImage = BitmapFactory.decodeStream(imageStream);  
 imageView.setImageBitmap(selectedImage);  
 imagePath = uri.getPath();  
 imageUri = uri;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 });  
  
 //radio group for vege and non vege food type  
 radioGroup.setOnCheckedChangeListener((group, checkedId) -> {  
 switch (checkedId) {  
 case R.id.radioVege:  
 foodType = Constants.vegetarian;  
 break;  
 case R.id.radioNonVege:  
 foodType = Constants.no\_vegetarian;  
 break;  
 }  
 CommonFunctions.customLog("Food type: " + foodType);  
 });  
  
 }  
  
 //initializing view  
 void initView() {  
 imageView = findViewById(R.id.image\_id);  
 edtName = findViewById(R.id.edtName\_id);  
 edtDescription = findViewById(R.id.edtDescription\_id);  
 edtLocation = findViewById(R.id.edtLocation\_id);  
 radioGroup = findViewById(R.id.radioGroup);  
 btnSubmit = findViewById(R.id.submitStreetFooBtn\_id);  
 }  
  
 //when click submit button it check the data validity and already exist in db or not  
 //if not then add it to firebase  
 void submitData() {  
 String name = edtName.getText().toString();  
 String description = edtDescription.getText().toString();  
 String location = edtLocation.getText().toString();  
  
 if (name.isEmpty()) {  
 CommonFunctions.showToast("Please fill the name", getApplicationContext());  
 return;  
 }  
  
 if (description.isEmpty()) {  
 CommonFunctions.showToast("Please fill the description", getApplicationContext());  
 return;  
 }  
 if (location.isEmpty()) {  
 CommonFunctions.showToast("Please fill the location", getApplicationContext());  
 return;  
 }  
 if (imagePath.isEmpty()) {  
 CommonFunctions.showToast("Please select some picture", getApplicationContext());  
 return;  
 }  
  
 FireStore.db.collection(Constants.rootCollectionStreetFood).whereEqualTo(Constants.name, name).addSnapshotListener((value, error) -> {  
 if (value.getDocuments().size() > 0) {  
 CommonFunctions.showToast("Already exist", getApplicationContext());  
 } else {  
 uploadImage(name, description, location);  
  
 }  
 });  
 }  
  
 //return you the selected image data  
 @Override  
 public void onActivityResult(int requestCode, int resultCode, Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
 if (requestCode == PICK\_IMAGE && resultCode == Activity.RESULT\_OK) {  
 if (data == null) {  
 //error  
 return;  
 }  
 try {  
 Uri uri = data.getData();  
 CommonFunctions.customLog(data.getData().toString());  
 CommonFunctions.customLog(uri.getPath());  
 final InputStream imageStream = getContentResolver().openInputStream(uri);  
 final Bitmap selectedImage = BitmapFactory.decodeStream(imageStream);  
 imageView.setImageBitmap(selectedImage);  
 imagePath = uri.getPath();  
 imageUri = uri;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 //Function for uploading image to firebase storage  
 private void uploadImage(String name, String description, String location) {  
  
 CommonFunctions.uploadImage(imagePath, getApplicationContext(), imageUri, new ImageUploadResult() {  
 @Override  
 public void onUploadSuccess(String imageUrl) {  
 StreetFood streetFood = new StreetFood(name, description, location, imageUrl, foodType, userId);  
 FireStore.addStreetFoodStall(streetFood, new OnResult() {  
 @Override  
 public void onComplete() {  
 finish();  
 }  
  
 @Override  
 public void onFailure() {  
 CommonFunctions.showToast("Error while uploading data", getApplicationContext());  
 }  
 });  
 }  
  
 @Override  
 public void onUploadFailure() {  
 CommonFunctions.showToast("Error while uploading image", getApplicationContext());  
 }  
 });  
 }  
}

## ChatActivity Class

package com.devexpert.forfoodiesbyfoodies.activities;  
  
import androidx.annotation.RequiresApi;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.content.Intent;  
import android.os.Build;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.adapters.ChatAdapter;  
import com.devexpert.forfoodiesbyfoodies.models.Chat;  
import com.devexpert.forfoodiesbyfoodies.models.User;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.firebase.firestore.DocumentChange;  
import com.google.firebase.firestore.EventListener;  
import com.google.firebase.firestore.Query;  
import com.google.firebase.firestore.QuerySnapshot;  
  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
  
  
public class ChatActivity extends AppCompatActivity {  
 private String documentId;  
 private User userData;  
 private EditText editText;  
 private RecyclerView recyclerView;  
 private ChatAdapter adapter;  
 private final List<Chat> chatMessages = new ArrayList<>();  
  
 @RequiresApi(api = Build.VERSION\_CODES.*N*)  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_chat*);  
 CustomSharedPreference yourPreference = CustomSharedPreference.*getInstance*(getApplicationContext());  
 String userId = yourPreference.getData(Constants.*userId*);  
  
 //initializing view  
 editText = findViewById(R.id.*edt\_message\_id*);  
 Button btnSendMessage = findViewById(R.id.*btnSend\_id*);  
 recyclerView = findViewById(R.id.*chat\_recyclerview*);  
 recyclerView.setLayoutManager(new LinearLayoutManager(getApplicationContext()));  
 adapter = new ChatAdapter(chatMessages, userId);  
 recyclerView.setAdapter(adapter);  
  
 //get user data from firestore  
 FireStore.*getData*(userId, user -> userData = user);  
  
 //Gets data that sends fro previous activity  
 Intent intent = getIntent();  
 documentId = intent.getStringExtra(Constants.*docId*);  
  
 listenMessage(); //it will listen all the messages including new one too.  
  
  
 btnSendMessage.setOnClickListener(view -> sendMessage()); //click to send message  
 }  
  
 void sendMessage() {  
 String msg = editText.getText().toString().trim();  
 if (msg.isEmpty()) { //check is message is empty or not  
 CommonFunctions.*showToast*("Type something!", getApplicationContext());  
 return;  
 }  
 //if message is not empty then send it to firestore  
 Chat chat = new Chat(msg, CommonFunctions.*CurrentDateTime*(), userData.getUserId(), userData.getFirstName());  
 FireStore.*sendMessage*(documentId, chat);  
 editText.setText(""); //after sending message clear edit text  
 }  
  
 //method for listening all messages also new one too  
 private void listenMessage() {  
 FireStore.*db*.collection(Constants.*rootCollectionChannels*).document(documentId).collection(Constants.*messages*).orderBy("timestamp", Query.Direction.*ASCENDING*).addSnapshotListener(eventListener);  
 }  
  
 //return all messages  
 private final EventListener<QuerySnapshot> eventListener = (value, error) -> {  
 if (error != null) {  
 return;  
 }  
 if (value != null) {  
 int count = chatMessages.size();  
 for (DocumentChange documentChange : value.getDocumentChanges()) {  
 if (documentChange.getType() == DocumentChange.Type.*ADDED*) {  
 Chat chat = new Chat();  
 chat.setText(documentChange.getDocument().get("text").toString());  
 Date creationDate = documentChange.getDocument().getDate("timestamp");  
 chat.setTimestamp(creationDate);  
 chat.setUserId(documentChange.getDocument().get(Constants.*userId*).toString());  
 chat.setUserName(documentChange.getDocument().get("userName").toString());  
 chat.setMessageId(documentChange.getDocument().getId());  
 chatMessages.add(chat);  
 }  
 }  
 if (count == 0) {  
 if (chatMessages.size() > 0) {  
 recyclerView.smoothScrollToPosition(chatMessages.size() - 1);  
 }  
 adapter.notifyDataSetChanged();  
 } else {  
 adapter.notifyItemRangeInserted(chatMessages.size(), chatMessages.size());  
 recyclerView.smoothScrollToPosition(chatMessages.size() - 1);  
 }  
 recyclerView.setVisibility(View.*VISIBLE*);  
 }  
 };  
  
}

## RestaurantsFragment Class

package com.devexpert.forfoodiesbyfoodies.fragments;  
  
import android.content.Intent;  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ProgressBar;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.activities.AddRestaurantActivity;  
import com.devexpert.forfoodiesbyfoodies.adapters.RestaurantRecyclerViewAdapter;  
import com.devexpert.forfoodiesbyfoodies.models.Restaurant;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
import com.google.firebase.firestore.DocumentChange;  
import com.google.firebase.firestore.EventListener;  
import com.google.firebase.firestore.Query;  
import com.google.firebase.firestore.QuerySnapshot;  
  
import java.util.ArrayList;  
import java.util.List;  
  
  
public class RestaurantsFragment extends Fragment {  
 private RestaurantRecyclerViewAdapter adapter;  
 private final List<Restaurant> restaurantList = new ArrayList<>();  
 private ProgressBar progressBar;  
 private RecyclerView recyclerView;  
 private FloatingActionButton fab;  
  
 public RestaurantsFragment() {  
 }  
  
 @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 View view = inflater.inflate(R.layout.*fragment\_resturants*, container, false);  
 listenNewRestaurant(); //listen all restaurant in db and also new added restaurant too  
  
 //get user id from local storage  
 CustomSharedPreference sharedPreference = CustomSharedPreference.*getInstance*(getContext());  
 String userId = sharedPreference.getData(Constants.*userId*);  
  
 //get user data from firebase  
 FireStore.*getData*(userId, users -> {  
 //if user ia admin then show option for adding new restaurant other wise hide the option  
 if (users.isAdmin()) {  
 fab.setVisibility(View.*VISIBLE*);  
 fab.setOnClickListener(view1 -> {  
 //navigate to AddRestaurantActivity  
 Intent intent = new Intent(getContext(), AddRestaurantActivity.class);  
 startActivity(intent);  
 });  
 }  
 });  
 progressBar = view.findViewById(R.id.*progressbar\_id*);  
 fab = view.findViewById(R.id.*fab*);  
 recyclerView = view.findViewById(R.id.*restaurantRecyclerview\_id*);  
 recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));  
 adapter = new RestaurantRecyclerViewAdapter(getContext(), restaurantList);  
 recyclerView.setAdapter(adapter);  
  
 return view;  
 }  
  
  
 private void listenNewRestaurant() {  
 FireStore.*db*.collection(Constants.*rootCollectionRestaurant*).orderBy(Constants.*restaurantName*, Query.Direction.*ASCENDING*).addSnapshotListener(eventListener);  
 }  
  
 private final EventListener<QuerySnapshot> eventListener = (value, error) -> {  
 if (error != null) {  
 return;  
 }  
 if (value != null) {  
 int count = restaurantList.size();  
 for (DocumentChange documentChange : value.getDocumentChanges()) {  
 if (documentChange.getType() == DocumentChange.Type.*ADDED*) {  
 String imageUrl = documentChange.getDocument().getData().get("restaurantImageUrl").toString();  
 String description = documentChange.getDocument().getData().get("restaurantDescription").toString();  
 String name = documentChange.getDocument().getData().get(Constants.*restaurantName*).toString();  
 String id = documentChange.getDocument().getData().get(Constants.*id*).toString();  
  
 Restaurant restaurant = new Restaurant(imageUrl, description, name, id);  
 restaurantList.add(restaurant);  
 }  
 }  
 if (count == 0) {  
 adapter.notifyDataSetChanged();  
 } else {  
 adapter.notifyItemRangeInserted(restaurantList.size(), restaurantList.size());  
 }  
 recyclerView.setVisibility(View.*VISIBLE*);  
 }  
 progressBar.setVisibility(View.*GONE*);  
  
 };  
  
}

## ProfileFragment Class

package com.devexpert.forfoodiesbyfoodies.fragments;  
  
import android.app.Activity;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.os.Bundle;  
  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.fragment.app.Fragment;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.interfaces.ImageUploadResult;  
import com.devexpert.forfoodiesbyfoodies.interfaces.OnResult;  
import com.devexpert.forfoodiesbyfoodies.models.User;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.services.CustomSharedPreference;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import com.squareup.picasso.Picasso;  
  
import java.io.InputStream;  
  
public class ProfileFragment extends Fragment {  
 private ImageView imageView;  
 private EditText edtName, edtLastName, edtEmail, edtPassword;  
 private User userData;  
 private String imagePath = "";  
 private Uri imageUri;  
 FirebaseStorage storage;  
 StorageReference storageReference;  
 private ActivityResultLauncher<Intent> someActivityResultLauncher;  
  
 public ProfileFragment() {  
 // Required empty public constructor  
 }  
  
  
 @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 View view = inflater.inflate(R.layout.*fragment\_profile*, container, false);  
 imageView = view.findViewById(R.id.*userImageView\_id*);  
 edtName = view.findViewById(R.id.*nameEditText\_id*);  
 edtLastName = view.findViewById(R.id.*lastNameEditText\_id*);  
 edtEmail = view.findViewById(R.id.*emailEditText\_id*);  
 edtPassword = view.findViewById(R.id.*passwordEditText\_id*);  
  
 Button btnSubmit = view.findViewById(R.id.*btnSubmit\_id*);  
 storage = FirebaseStorage.*getInstance*();  
 storageReference = storage.getReference();  
  
 //fetching user id from local storage  
 CustomSharedPreference sharedPreference = CustomSharedPreference.*getInstance*(getContext());  
 String userId = sharedPreference.getData(Constants.*userId*);  
  
 //gets data of user from firebase and sets into the fields  
 FireStore.*getData*(userId, user -> {  
 userData = user;  
 CommonFunctions.*customLog*(userData.getDocumentId());  
 edtEmail.setText(user.getEmail());  
 edtName.setText(user.getFirstName());  
 edtLastName.setText(user.getLastName());  
 edtPassword.setText(user.getPassword());  
 Picasso.*get*().load(user.getImageUrl()).fit().centerCrop().  
 placeholder(R.drawable.*placeholder\_image*).error(R.drawable.*error\_image*).into(imageView);  
 });  
  
 //when click on image, open the gallery to pick one image  
 imageView.setOnClickListener(view12 -> {  
 Intent intent = new Intent();  
 intent.setType("image/\*");  
 intent.setAction(Intent.*ACTION\_GET\_CONTENT*);  
 someActivityResultLauncher.launch(intent);  
 });  
  
 // You can do the assignment inside onAttach or onCreate, i.e, before the activity is displayed  
 someActivityResultLauncher = registerForActivityResult(  
 new ActivityResultContracts.StartActivityForResult(),  
 result -> {  
 if (result.getResultCode() == Activity.*RESULT\_OK*) {  
 // There are no request codes  
 Intent data = result.getData();  
 if (data == null) {  
 //error  
 return;  
 }  
 try {  
 Uri uri = data.getData();  
 System.*out*.println(data.getData() + "::::::::::::" + uri.getPath());  
 final InputStream imageStream = getActivity().getContentResolver().openInputStream(uri);  
 final Bitmap selectedImage = BitmapFactory.*decodeStream*(imageStream);  
 imageView.setImageBitmap(selectedImage);  
 imagePath = uri.getPath();  
 imageUri = uri;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 });  
  
 btnSubmit.setOnClickListener(view1 -> {  
 String firstName = edtName.getText().toString().trim();  
 String lastName = edtLastName.getText().toString().trim();  
 String email = edtEmail.getText().toString().trim();  
 String password = edtPassword.getText().toString().trim();  
 if (firstName.isEmpty()) {  
 CommonFunctions.*showToast*("Please fill first name!", getContext());  
 return;  
 }  
 if (lastName.isEmpty()) {  
 CommonFunctions.*showToast*("Please fill last name!", getContext());  
 return;  
 }  
 if (email.isEmpty() || !CommonFunctions.*isEmailValid*(email)) {  
 CommonFunctions.*showToast*("Please fill email correctly!", getContext());  
 return;  
 }  
 if (password.isEmpty() || password.length() < 8) {  
 CommonFunctions.*showToast*("Please fill password correctly!", getContext());  
 return;  
 }  
 if (imagePath != null && imageUri != null) {  
 //upload image and then update user data  
 CommonFunctions.*uploadImage*(imagePath, getContext(), imageUri, new ImageUploadResult() {  
 @Override  
 public void onUploadSuccess(String imageUrl) {  
 User user = new User(firstName, lastName, email, userData.getUserId(), password, imageUrl, userData.isUser(), userData.isCritic(), userData.isAdmin());  
  
 //update user data  
 FireStore.updateUserData(userData.getDocumentId(), user, new OnResult() {  
 @Override  
 public void onComplete() {  
 CommonFunctions.showToast("Data Update successfully", getContext());  
 }  
  
 @Override  
 public void onFailure() {  
 CommonFunctions.showToast("Error while updating user data", getContext());  
 }  
 });  
 }  
  
 @Override  
 public void onUploadFailure() {  
 CommonFunctions.showToast("Error while uploading image", getContext());  
 }  
 });  
  
 } else {  
 //without image upload update data in firebase  
 User user = new User(firstName, lastName, email, userData.getUserId(), password, userData.getImageUrl(), userData.isUser(), userData.isCritic(), userData.isAdmin());  
  
 FireStore.updateUserData(userData.getDocumentId(), user, new OnResult() {  
 @Override  
 public void onComplete() {  
 CommonFunctions.showToast("Data save successfully", getContext());  
 }  
  
 @Override  
 public void onFailure() {  
 CommonFunctions.showToast("Something went wrong", getContext());  
 }  
 });  
 }  
 });  
 return view;  
 }  
}

## AddUserByAdmin Class

package com.devexpert.forfoodiesbyfoodies.fragments;  
  
import android.app.ProgressDialog;  
import android.os.Bundle;  
  
import androidx.fragment.app.Fragment;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.RadioGroup;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.models.User;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.utils.CommonFunctions;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.firebase.auth.FirebaseAuth;  
  
  
public class AddUserByAdminFragment extends Fragment {  
  
 private EditText edtFirstName, edtLastName, edtEmail, edtPassword;  
 private boolean isCritic = true;  
 private boolean isAdmin = false;  
 private FirebaseAuth auth;  
  
 public AddUserByAdminFragment() {  
 // empty public constructor  
 }  
  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 // Inflate the layout for this fragment  
 View view = inflater.inflate(R.layout.fragment\_add\_user\_by\_admin, container, false);  
 //Get Firebase auth instance  
 auth = FirebaseAuth.getInstance();  
 edtFirstName = view.findViewById(R.id.firstNameField\_id);  
 edtLastName = view.findViewById(R.id.lastNameField\_id);  
 edtEmail = view.findViewById(R.id.emailField\_id);  
 edtPassword = view.findViewById(R.id.passwordField\_id);  
 RadioGroup radioGroup = view.findViewById(R.id.radioGroupAddUser\_id);  
 Button btnSubmit = view.findViewById(R.id.signUpBtn\_id);  
 radioGroup.setOnCheckedChangeListener((group, checkedId) -> {  
 switch (checkedId) {  
 case R.id.radioCritic:  
 isCritic = true;  
 isAdmin = false;  
 break;  
 case R.id.radioAdmin:  
 isAdmin = true;  
 isCritic = false;  
 break;  
 }  
 CommonFunctions.customLog("isCritics: " + isCritic + " isAdmin: " + isAdmin);  
 });  
 btnSubmit.setOnClickListener(view1 -> createAnAccount()); //create an account  
  
 return view;  
 }  
  
 //Function for creating an account  
 void createAnAccount() {  
 String firstName = edtFirstName.getText().toString().trim();  
 String lastName = edtLastName.getText().toString().trim();  
 String email = edtEmail.getText().toString().trim();  
 String password = edtPassword.getText().toString().trim();  
 if (firstName.isEmpty() || lastName.isEmpty() || email.isEmpty() || password.isEmpty()) {  
 CommonFunctions.showToast("Please fill the data properly.", getContext());  
 } else {  
 if (CommonFunctions.isEmailValid(email)) {  
  
 if (password.length() >= 8) {  
 //do signUp  
 final ProgressDialog progressDialog = new ProgressDialog(getContext());  
 progressDialog.setTitle("Creating an account please wait!");  
 progressDialog.show();  
  
 auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(task -> {  
 progressDialog.cancel();  
 if (!task.isSuccessful()) {  
 CommonFunctions.showToast("Error while creating account!", getContext());  
 } else {  
 User user = new User(firstName, lastName, email, task.getResult().getUser().getUid(),  
 password, Constants.defaultImageUrl, false, isCritic, isAdmin);  
 FireStore.addUserToFireStore(user);  
 edtFirstName.setText("");  
 edtLastName.setText("");  
 edtEmail.setText("");  
 edtPassword.setText("");  
 CommonFunctions.showToast("Create account Successful.", getContext());  
  
 }  
 }).addOnFailureListener(e -> {  
 progressDialog.cancel();  
 CommonFunctions.showToast("Error while creating account!", getContext());  
  
 });  
 } else {  
 CommonFunctions.showToast("Password should contain at-least 8 characters.", getContext());  
 }  
 } else {  
 CommonFunctions.showToast("Email is not valid", getContext());  
 }  
 }  
 }  
}

## StreeFoodFragment Class

package com.devexpert.forfoodiesbyfoodies.fragments;  
  
import android.content.Intent;  
import android.os.Build;  
import android.os.Bundle;  
  
import androidx.annotation.RequiresApi;  
import androidx.fragment.app.Fragment;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ProgressBar;  
  
import com.devexpert.forfoodiesbyfoodies.R;  
import com.devexpert.forfoodiesbyfoodies.activities.AddStreetFoodActivity;  
import com.devexpert.forfoodiesbyfoodies.adapters.StreetFoodRecyclerviewAdapter;  
import com.devexpert.forfoodiesbyfoodies.models.StreetFood;  
import com.devexpert.forfoodiesbyfoodies.models.User;  
import com.devexpert.forfoodiesbyfoodies.services.FireStore;  
import com.devexpert.forfoodiesbyfoodies.utils.Constants;  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
import com.google.firebase.firestore.DocumentChange;  
import com.google.firebase.firestore.EventListener;  
import com.google.firebase.firestore.Query;  
import com.google.firebase.firestore.QuerySnapshot;  
  
import java.util.ArrayList;  
import java.util.List;  
  
  
public class StreetFoodFragment extends Fragment {  
 private StreetFoodRecyclerviewAdapter adapter;  
 private ProgressBar progressBar;  
 private RecyclerView recyclerView;  
 private final List<StreetFood> streetFoodList = new ArrayList<>();  
 private final User userData;  
  
 public StreetFoodFragment(User user) {  
 this.userData = user;  
 }  
  
 @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 }  
  
 @RequiresApi(api = Build.VERSION\_CODES.*N*)  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 View view = inflater.inflate(R.layout.*fragment\_street\_food*, container, false);  
  
 progressBar = view.findViewById(R.id.*progressbar\_id*);  
 recyclerView = view.findViewById(R.id.*streetFoodRecyclerview\_id*);  
 recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));  
 adapter = new StreetFoodRecyclerviewAdapter(getContext(), streetFoodList, userData);  
 recyclerView.setAdapter(adapter);  
 listenNewStreetFoodRestaurant(); //listen all street food entries  
  
 FloatingActionButton fab = view.findViewById(R.id.*fab*);  
 fab.setOnClickListener(view1 -> {  
 //navigate to AddStreetFoodActivity  
 Intent intent = new Intent(getContext(), AddStreetFoodActivity.class);  
 startActivity(intent);  
 });  
  
 return view;  
 }  
  
 //Function for listening all street food restaurant including new one too  
 private void listenNewStreetFoodRestaurant() {  
 FireStore.*db*.collection(Constants.*rootCollectionStreetFood*).orderBy(Constants.*name*, Query.Direction.*ASCENDING*)  
 .addSnapshotListener(eventListener);  
 progressBar.setVisibility(View.*GONE*);  
  
 }  
  
 private final EventListener<QuerySnapshot> eventListener = (value, error) -> {  
 if (error != null) {  
 return;  
 }  
 if (value != null) {  
 int count = streetFoodList.size();  
 for (DocumentChange documentChange : value.getDocumentChanges()) {  
 if (documentChange.getType() == DocumentChange.Type.*ADDED*) {  
 StreetFood streetFood = new StreetFood();  
 streetFood.setDescription(documentChange.getDocument().get("description").toString());  
 streetFood.setLocation(documentChange.getDocument().get("location").toString());  
 streetFood.setName(documentChange.getDocument().get(Constants.*name*).toString());  
 streetFood.setPicture(documentChange.getDocument().get("picture").toString());  
 streetFood.setType(documentChange.getDocument().get("type").toString());  
 streetFood.setUserId(documentChange.getDocument().get(Constants.*userId*).toString());  
 streetFood.setId(documentChange.getDocument().getId());  
 streetFoodList.add(streetFood);  
 }  
 }  
 if (count == 0) {  
 adapter.notifyDataSetChanged();  
 } else {  
 adapter.notifyItemRangeInserted(streetFoodList.size(), streetFoodList.size());  
 }  
 recyclerView.setVisibility(View.*VISIBLE*);  
 }  
 progressBar.setVisibility(View.*GONE*);  
  
 };  
}

## Resources

## activity\_login.xml

<?xml version="1.0" encoding="utf-8"?>  
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".activities.LoginActivity">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:gravity="center"  
 android:orientation="vertical"  
 android:padding="10dp">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:fontFamily="monospace"  
 android:text="@string/login"  
 android:textColor="@color/black"  
 android:textSize="50sp" />  
  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
  
 <EditText  
 android:id="@+id/emailField\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/email"  
 android:hint="@string/email"  
 android:inputType="textEmailAddress"  
 android:minHeight="48dp" />  
  
  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
  
 <EditText  
 android:id="@+id/passwordField\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/password"  
 android:hint="@string/password"  
 android:inputType="textPassword"  
 android:minHeight="48dp" />  
 </com.google.android.material.textfield.TextInputLayout>  
  
  
 <TextView  
 android:id="@+id/forgetPasswordTextView\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:gravity="end"  
 android:text="@string/forget\_password"  
 android:textColor="@color/teal\_200" />  
  
 <Button  
 android:id="@+id/loginBtn\_id"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:text="@string/login"  
 tools:ignore="DuplicateSpeakableTextCheck" />  
  
 <ProgressBar  
 android:id="@+id/progressBar\_id"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:layout\_gravity="center|bottom"  
 android:layout\_marginBottom="20dp"  
 android:visibility="gone" />  
  
 <TextView  
 android:id="@+id/signUpTextView\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:gravity="center"  
 android:text="@string/create\_a\_new\_account"  
 android:textColor="@color/teal\_200" />  
 </LinearLayout>  
  
</ScrollView>

## activity\_signup.xml

<?xml version="1.0" encoding="utf-8"?>  
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".activities.SignUpActivity">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:gravity="center"  
 android:orientation="vertical"  
 android:padding="10dp">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:fontFamily="monospace"  
 android:text="@string/signUp"  
 android:textColor="@color/black"  
 android:textSize="50sp" />  
  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
 <EditText  
 android:id="@+id/firstNameField\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/firstName"  
 android:hint="@string/firstName"  
 android:inputType="text"  
 android:minHeight="48dp" />  
  
 </com.google.android.material.textfield.TextInputLayout>  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
 <EditText  
 android:id="@+id/lastNameField\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/lastName"  
 android:hint="@string/lastName"  
 android:inputType="text"  
 android:minHeight="48dp" />  
  
 </com.google.android.material.textfield.TextInputLayout>  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
  
 <EditText  
 android:id="@+id/emailField\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/email"  
 android:hint="@string/email"  
 android:inputType="textEmailAddress"  
 android:minHeight="48dp" />  
  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
  
 <EditText  
 android:id="@+id/passwordField\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/password"  
 android:hint="@string/password"  
 android:inputType="textPassword"  
 android:minHeight="48dp" />  
  
 </com.google.android.material.textfield.TextInputLayout>  
  
  
 <ProgressBar  
 android:id="@+id/progressBar\_id"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:layout\_gravity="center|bottom"  
 android:layout\_marginBottom="20dp"  
 android:visibility="gone" />  
 <Button  
 android:id="@+id/signUpBtn\_id"  
 android:layout\_width="200dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:text="@string/signUp"  
 tools:ignore="DuplicateSpeakableTextCheck" />  
  
 <TextView  
 android:id="@+id/loginTextView\_id"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:gravity="center"  
 android:textColor="@color/teal\_200"  
 android:text="@string/already\_have\_an\_account" />  
  
 </LinearLayout>  
  
</ScrollView>

## activity\_forget\_password.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.coordinatorlayout.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_gravity="center"  
 android:fitsSystemWindows="true"  
 tools:context=".activities.ForgetPasswordActivity">  
  
 <LinearLayout  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="30dp"  
 android:gravity="center"  
 android:orientation="vertical"  
 android:padding="@dimen/activity\_horizontal\_margin">  
  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal"  
 android:padding="10dp"  
 android:text="@string/lbl\_forgot\_password"  
 android:textColor="@android:color/black"  
 android:textSize="20dp" />  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dp"  
 android:gravity="center\_horizontal"  
 android:padding="@dimen/activity\_horizontal\_margin"  
 android:text="@string/forgot\_password\_msg"  
 android:textColor="@android:color/darker\_gray"  
 android:textSize="14sp" />  
  
 <com.google.android.material.textfield.TextInputLayout  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal">  
  
 <EditText  
 android:id="@+id/email"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:autofillHints="@string/email"  
 android:hint="@string/email"  
 android:inputType="textEmailAddress"  
 android:minHeight="48dp" />  
  
  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <!-- Login Button -->  
  
 <Button  
 android:id="@+id/btn\_reset\_password"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="20dip"  
 android:text="@string/btn\_reset\_password" />  
  
 <Button  
 android:id="@+id/btn\_back"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:backgroundTint="@color/white"  
 android:text="@string/login"  
 android:textColor="@color/purple\_200" />  
  
 </LinearLayout>  
  
 <ProgressBar  
 android:id="@+id/progressBar"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:layout\_gravity="center|bottom"  
 android:layout\_marginBottom="20dp"  
 android:visibility="gone" />  
</androidx.coordinatorlayout.widget.CoordinatorLayout>

## Values

## colors.xml

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="purple\_200">#FFBB86FC</color>  
 <color name="purple\_500">#FF6200EE</color>  
 <color name="purple\_700">#FF3700B3</color>  
 <color name="teal\_200">#FF03DAC5</color>  
 <color name="teal\_700">#FF018786</color>  
 <color name="black">#FF000000</color>  
 <color name="white">#FFFFFFFF</color>  
 <color name="golden">#FFD700</color>  
 <color name="grey">#CAC9C9</color>  
 <color name="yellowColor">#fece2f</color>  
 <color name="green">#50C878</color>  
  
</resources>

## dimens.xml

<resources>  
 <!-- Default screen margins, per the Android Design guidelines. -->  
 <dimen name="activity\_horizontal\_margin">16dp</dimen>  
 <dimen name="activity\_vertical\_margin">16dp</dimen>  
 <dimen name="nav\_header\_vertical\_spacing">8dp</dimen>  
 <dimen name="nav\_header\_height">176dp</dimen>  
 <dimen name="fab\_margin">16dp</dimen>  
</resources>

## strings.xml

<resources>  
 <string name="app\_name">For foodies by foodies</string>  
 <string name="title\_activity\_main">MainActivity</string>  
 <string name="login">Login</string>  
 <string name="signUp">SignUp</string>  
 <string name="email">Email</string>  
 <string name="password">Password</string>  
 <string name="firstName">First Name</string>  
 <string name="lastName">Last Name</string>  
 <string name="forget\_password">Forget Password</string>  
 <string name="create\_a\_new\_account">Create a new Account!</string>  
 <string name="already\_have\_an\_account">Already have an account!</string>  
 <!-- *TODO: Remove or change this placeholder text* -->  
 <string name="hello\_blank\_fragment">Hello blank fragment</string>  
 <string name="main\_drawer\_open">open drawer</string>  
 <string name="main\_drawer\_close">close drawer</string>  
 <string name="restaurants">restaurants</string>  
 <string name="view">View</string>  
 <string name="view\_review">View Review</string>  
 <string name="add\_review">Add Review</string>  
 <string name="reservation">Reservation</string>  
 <string name="title\_activity\_reservation">Reservation</string>  
 <!-- Strings used for fragments for navigation -->  
 <string name="first\_fragment\_label">First Fragment</string>  
 <string name="second\_fragment\_label">Second Fragment</string>  
 <string name="next">Next</string>  
 <string name="previous">Previous</string>  
  
 <string name="hello\_first\_fragment">Hello first fragment</string>  
 <string name="hello\_second\_fragment">Hello second fragment. Arg: %1$s</string>  
 <string name="user\_image">user image</string>  
 <string name="title\_activity\_reviews">ReviewsActivity</string>  
 <string name="cancel">Cancel</string>  
 <string name="done">Done</string>  
 <string name="add\_your\_rating">Add your rating!</string>  
 <string name="name">Name</string>  
 <string name="description">Description</string>  
 <string name="location">Location</string>  
 <string name="vegetarian">Vegetarian</string>  
 <string name="no\_vegetarian">No-Vegetarian</string>  
 <string name="user\_name">User Name</string>  
 <string name="submit">Submit</string>  
 <string name="submit\_changes">Submit changes</string>  
 <string name="food\_consumer">Food Connoisseur</string>  
 <string name="\_3\_5">3/5</string>  
 <string name="create\_an\_account">Create an account!</string>  
 <string name="admin">Admin</string>  
 <string name="critic">Critic</string>  
 <string name="logo">TODO</string>  
 <string name="restaurant\_name">Restaurant Name</string>  
 <string name="channels">Channels</string>  
 <string name="send">Send</string>  
 <string name="type\_message">Type message</string>  
 <string name="for\_adding\_chat\_topics">for adding chat topics</string>  
 <string name="restaurant">Restaurants</string>  
 <string name="profile">Profile</string>  
 <string name="street\_food">Street Food</string>  
 <string name="chat\_forum">Chat Forum</string>  
 <string name="add\_critic">Add Critic</string>  
 <string name="for\_adding\_new\_restaurant">For adding new restaurant</string>  
 <string name="for\_adding\_street\_food\_stalls">For adding street food stalls</string>  
 <string name="lbl\_forgot\_password">Forget Password</string>  
 <string name="forgot\_password\_msg">To reset password, enter your emial, press the button anc check mail to follow instructions.</string>  
 <string name="btn\_reset\_password">Reset Password</string>  
  
</resources>

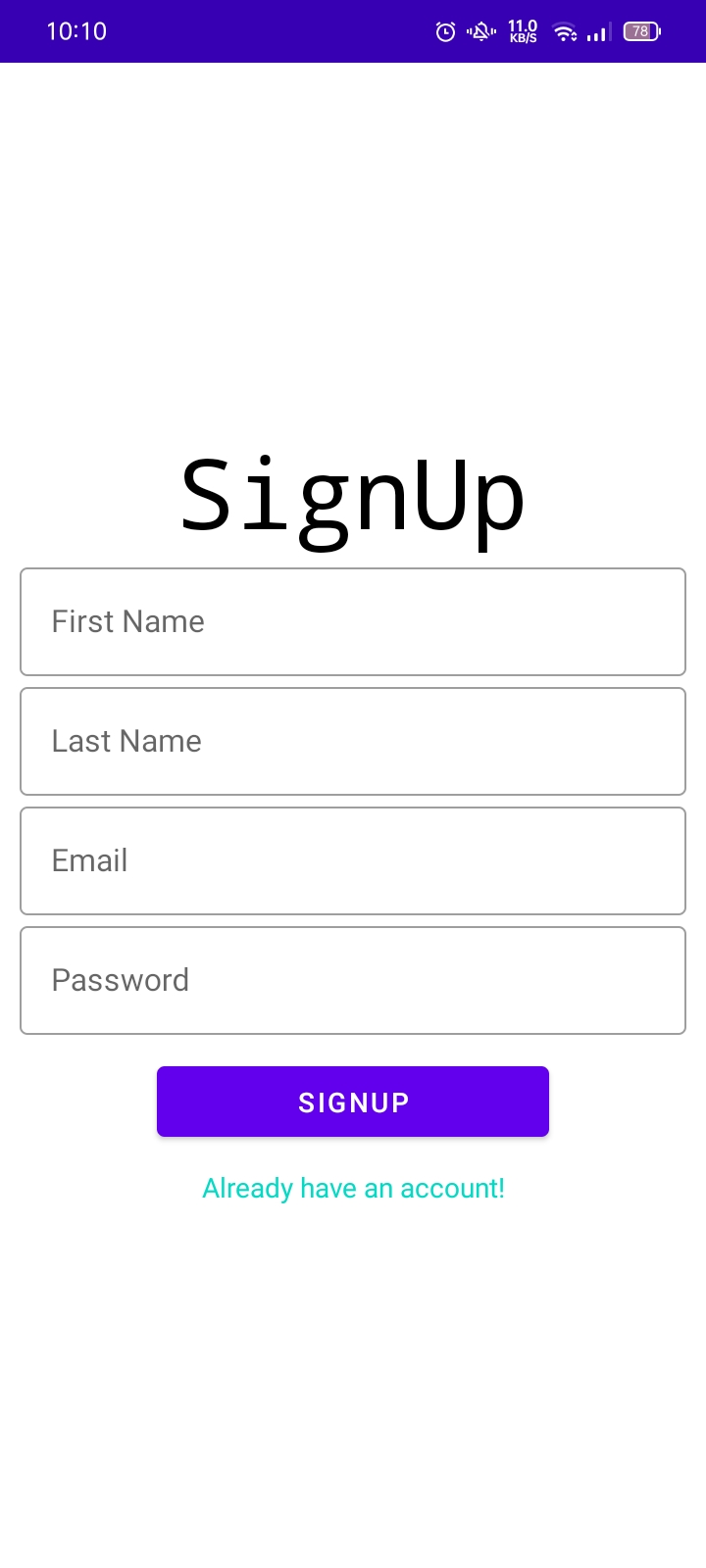
## Manifest.xml Permissions

<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

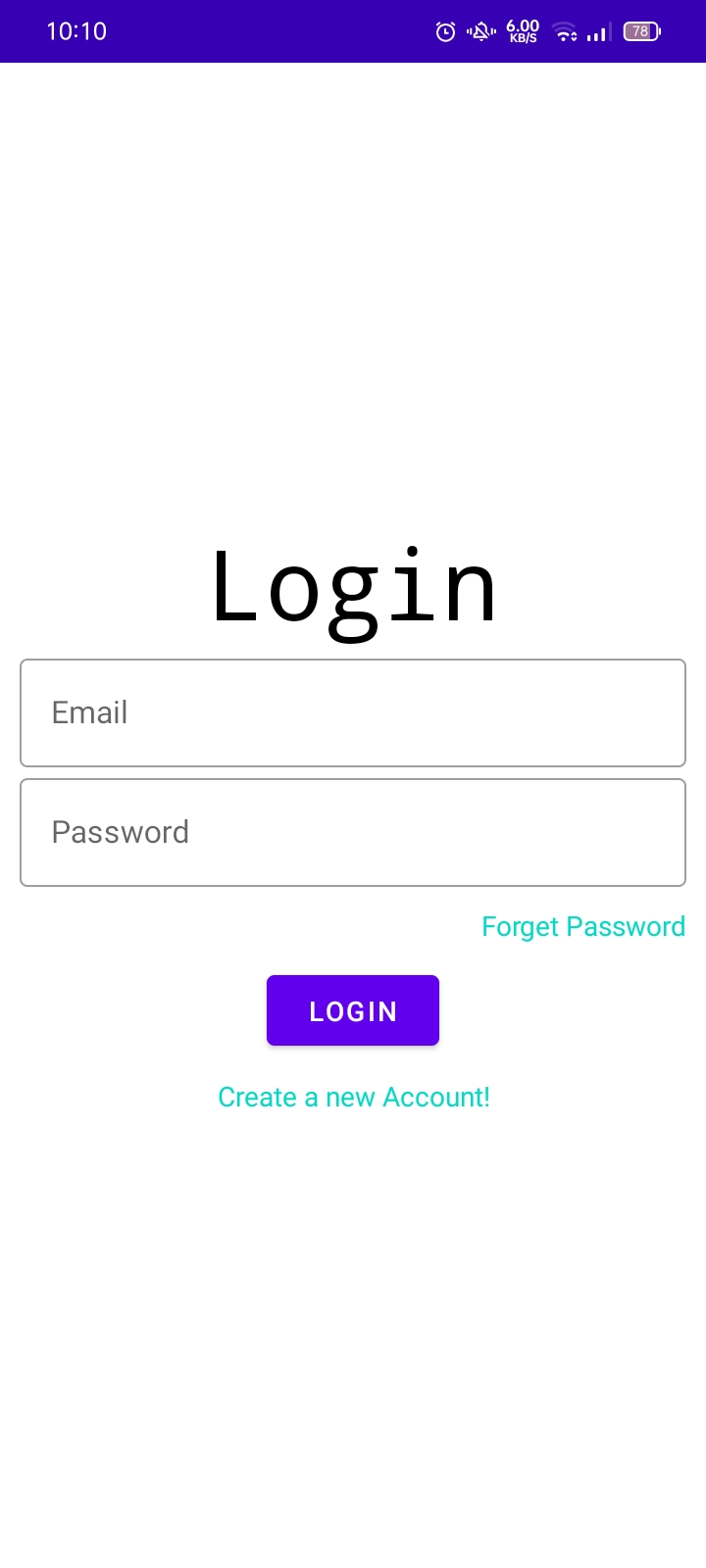
## Splash View



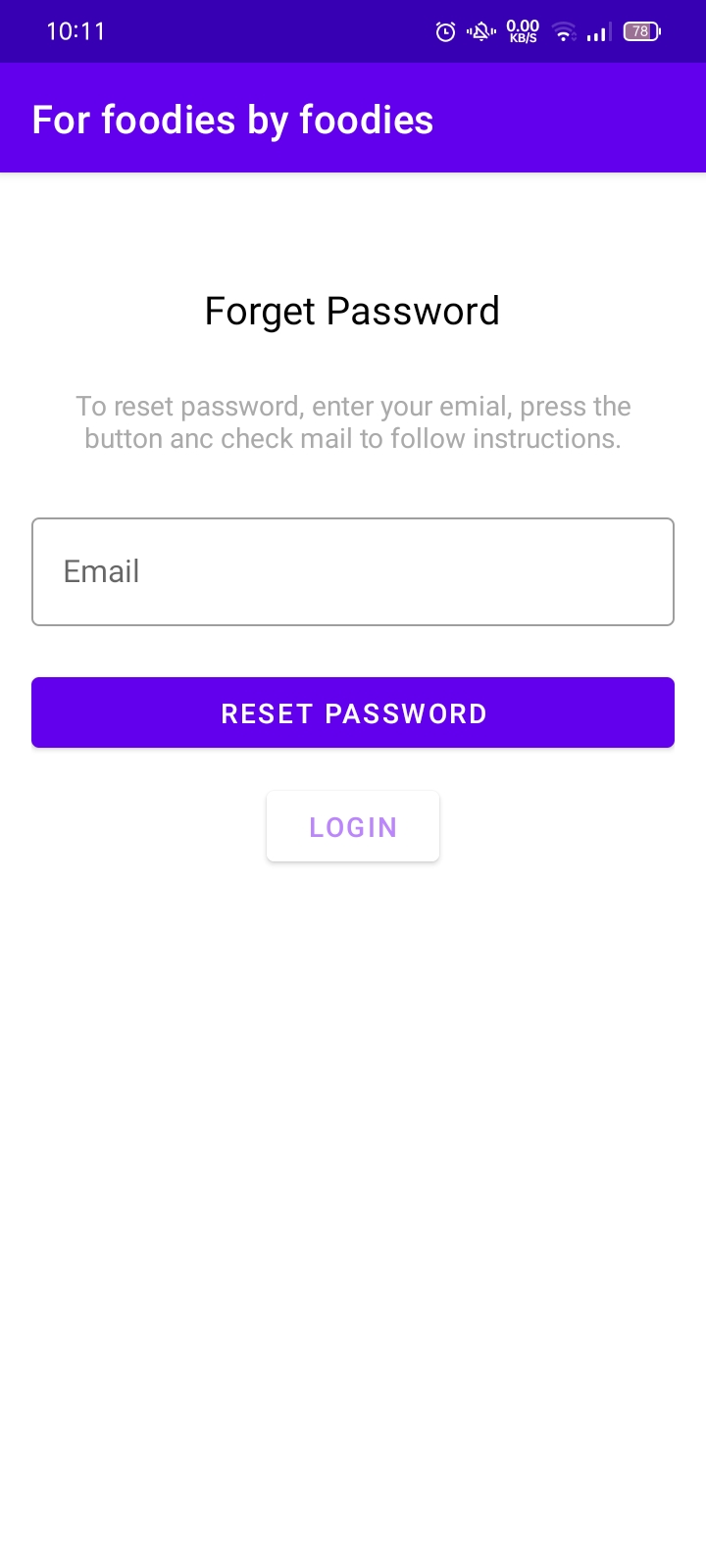
## Signup View



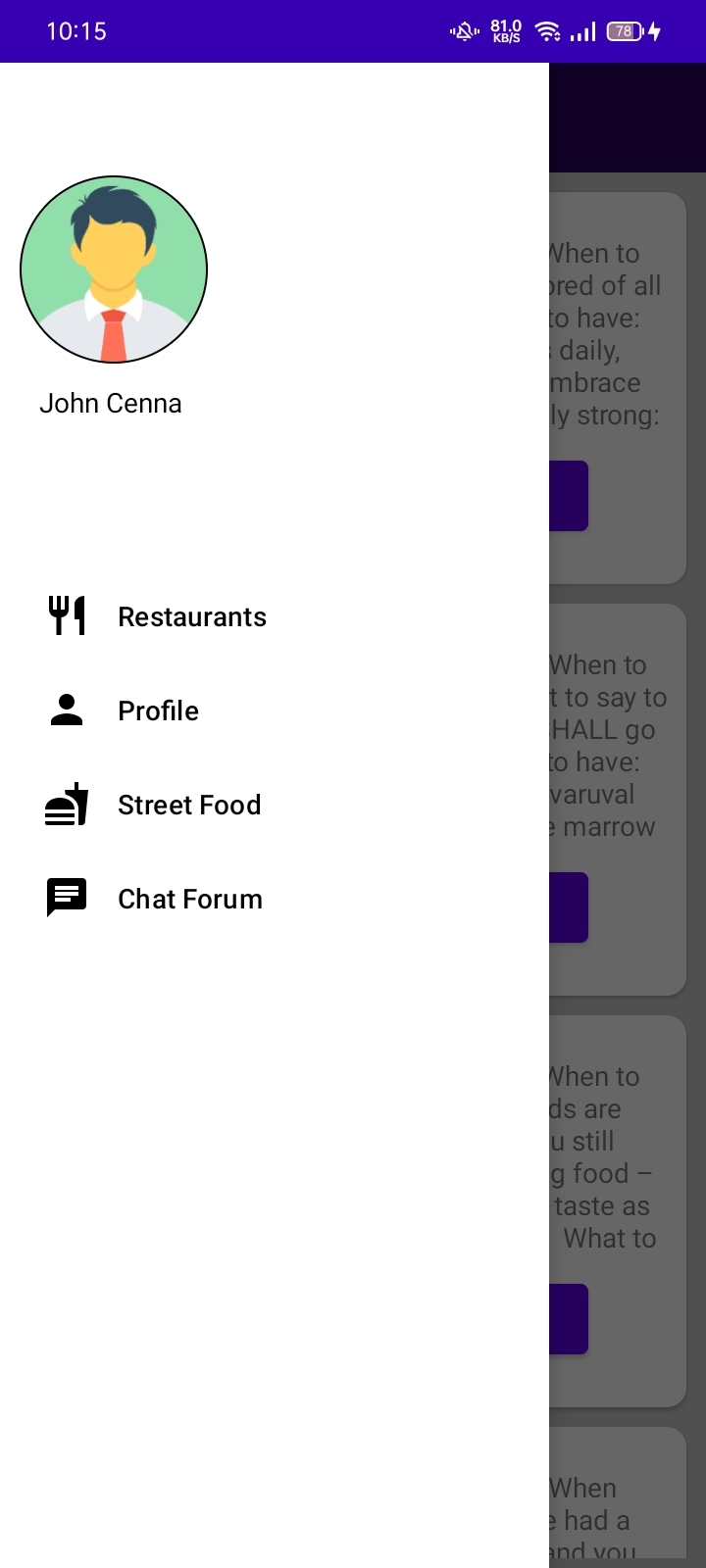
## Login View



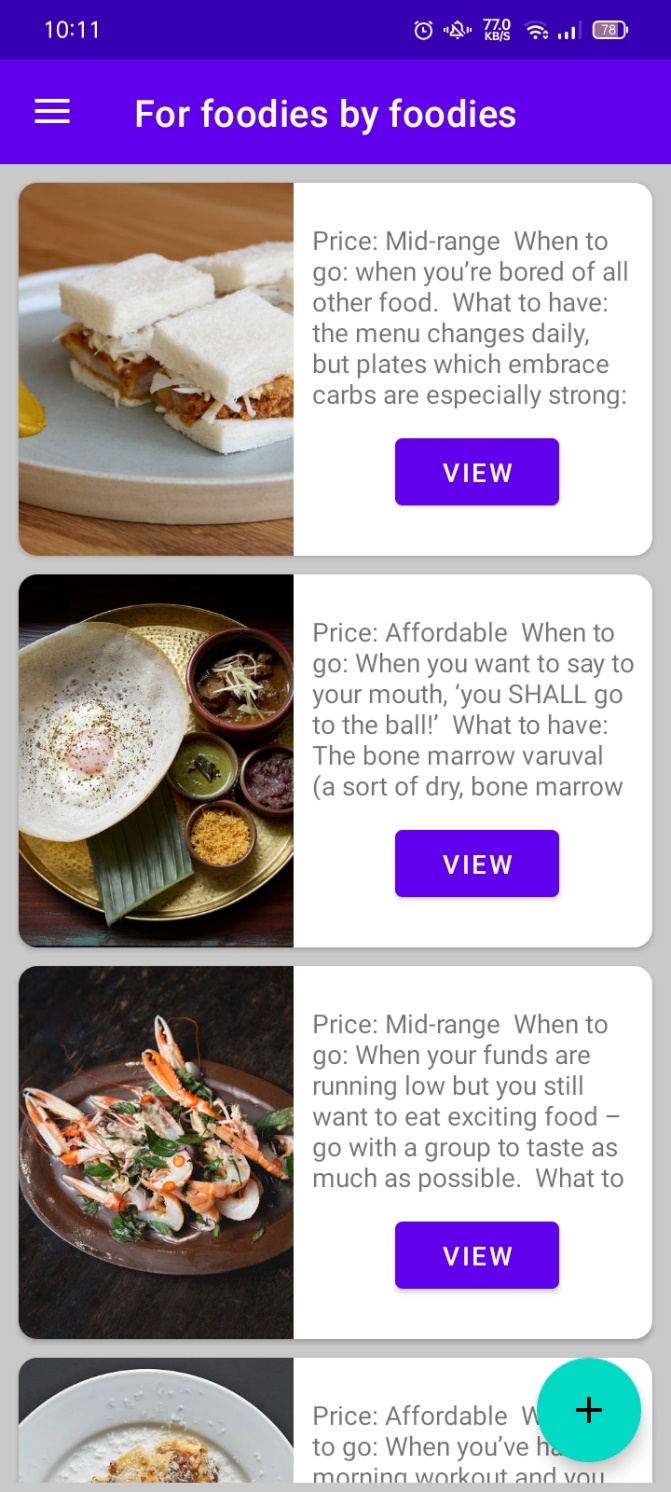
## ForgetPassword View



## Menu View



## DashBoard View

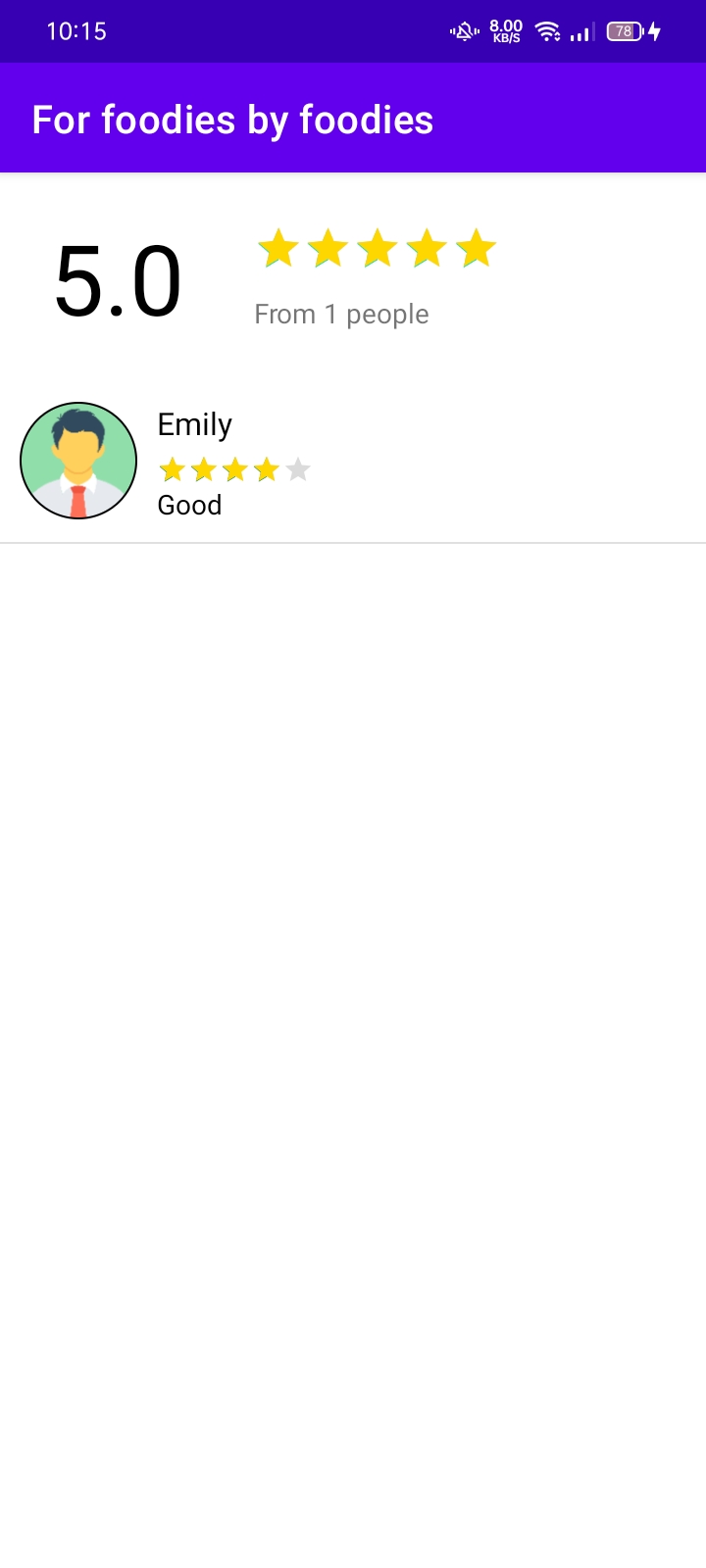


Disclosure: diagrams, pictures or any other images used in this assignment are for educational purposes only.

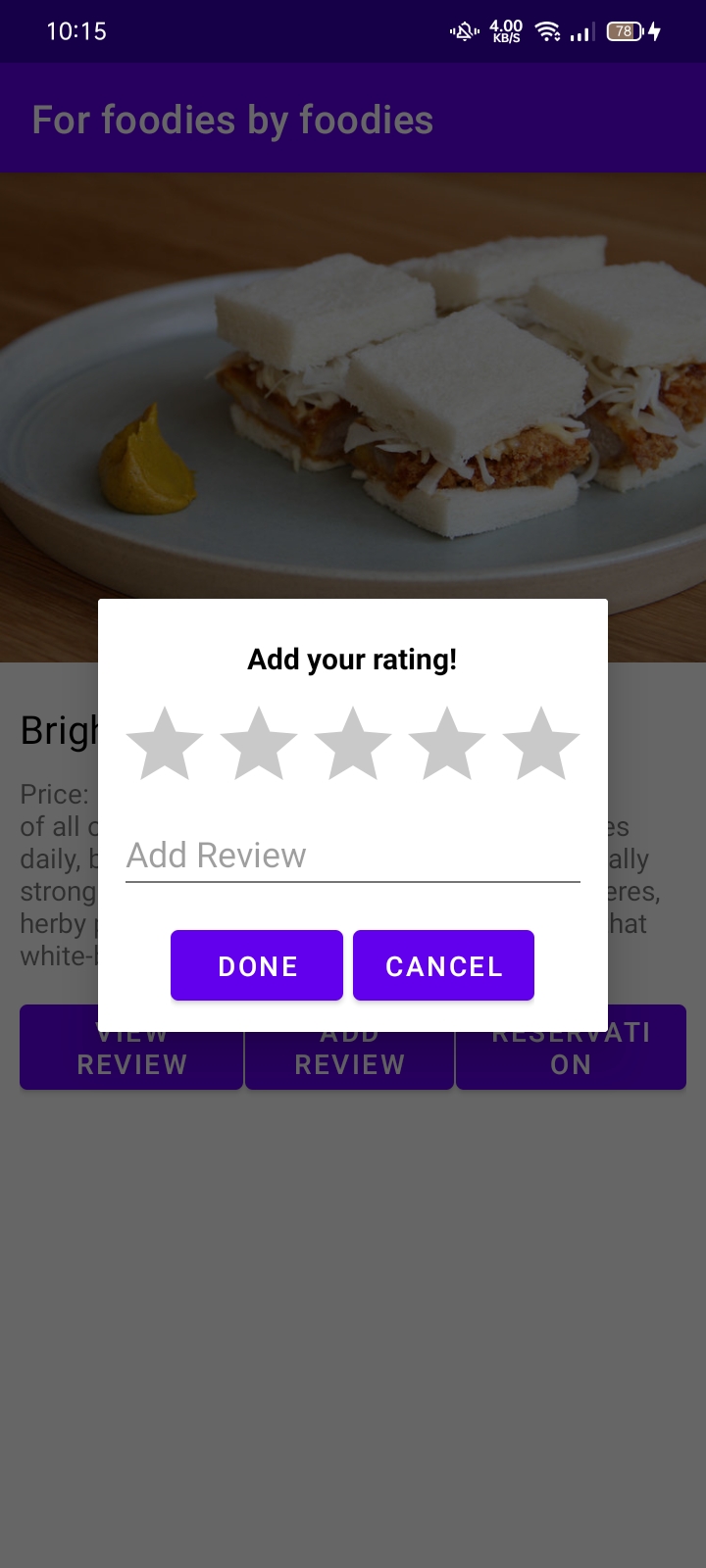
## Restaurant Detail View



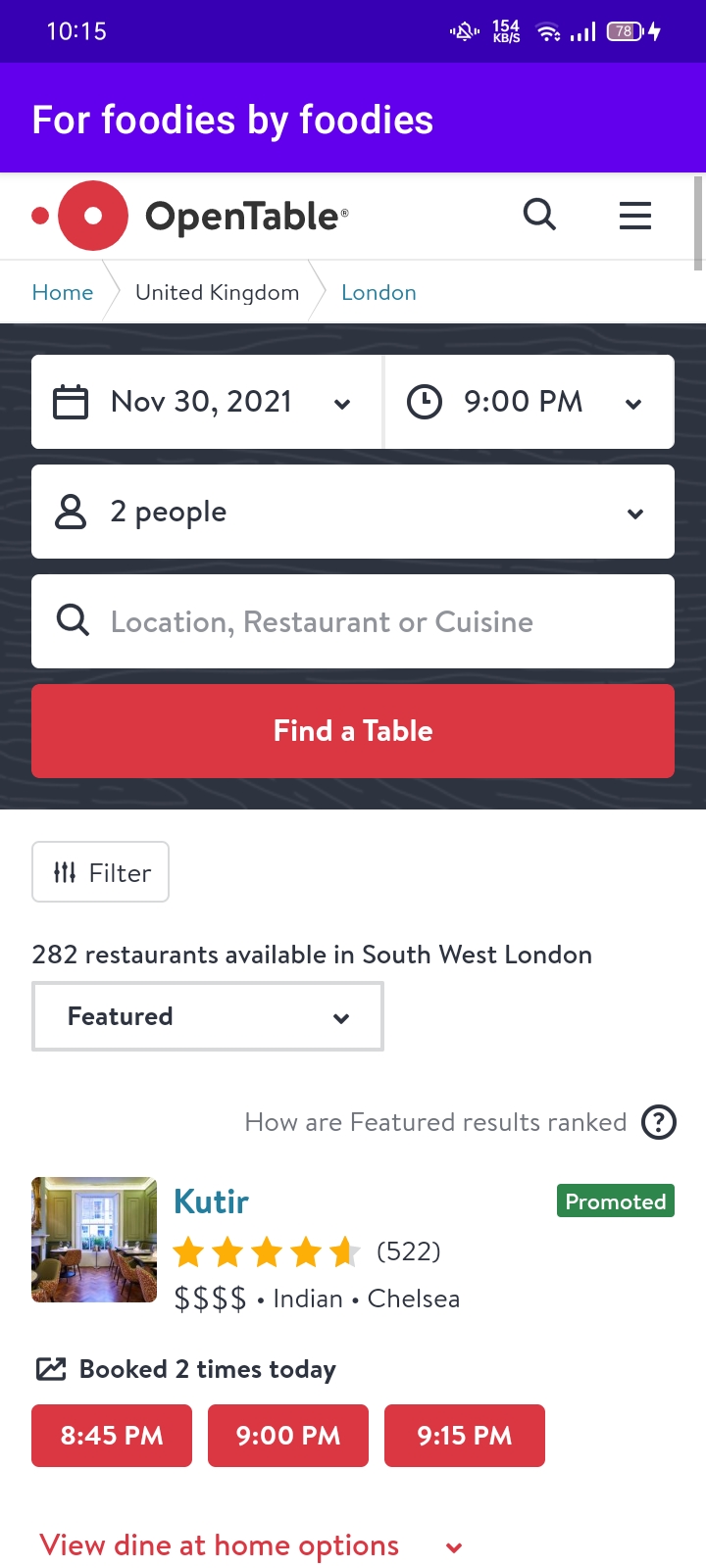
## Reviews View



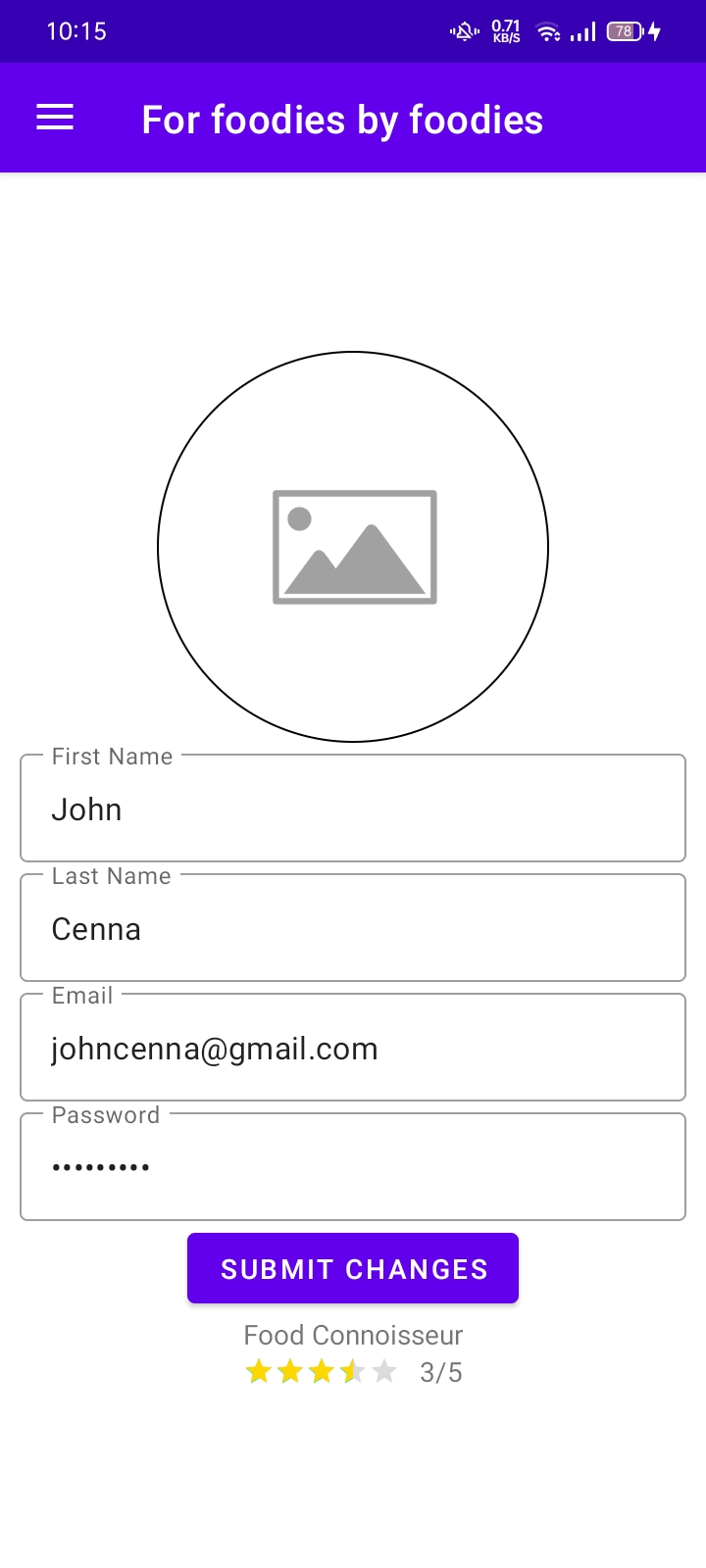
## Add rating View



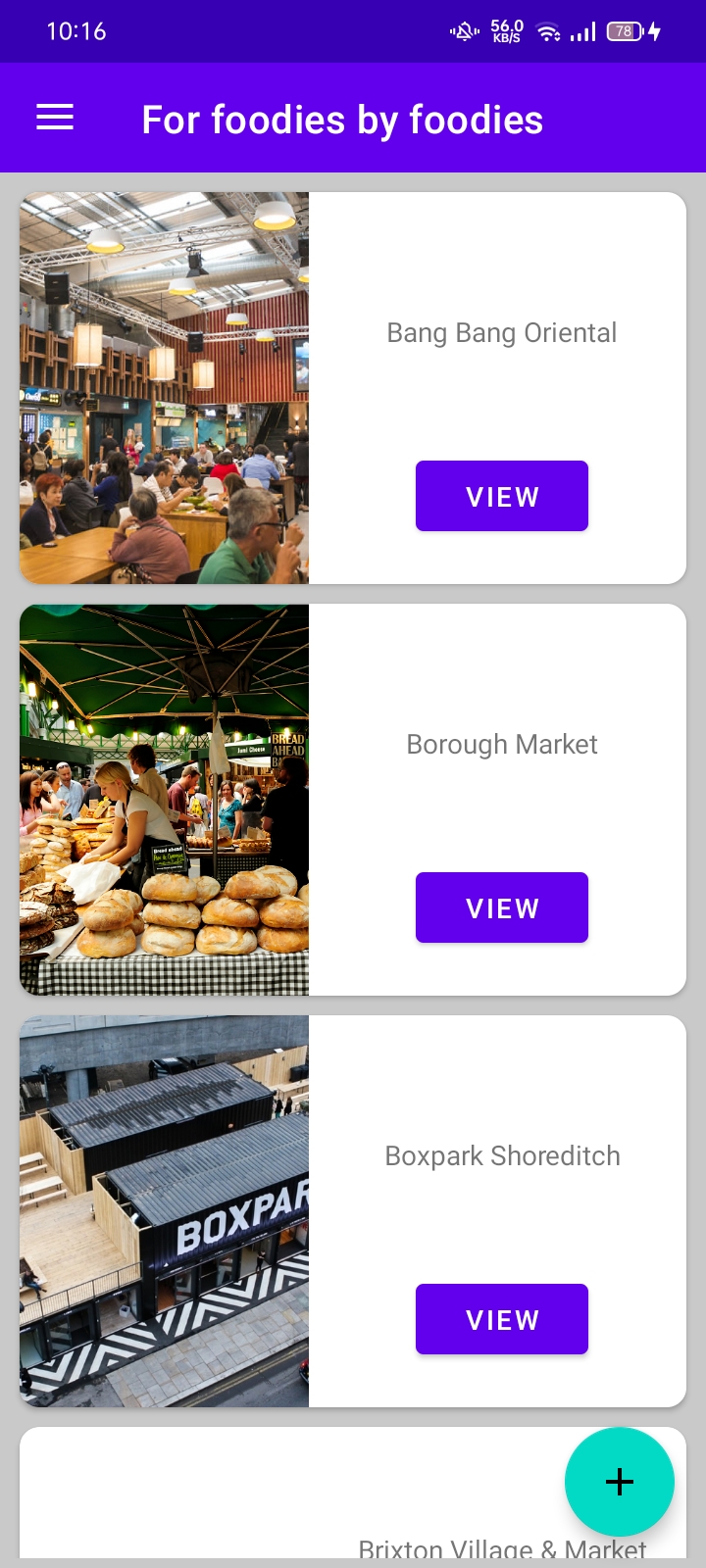
## Reservation View



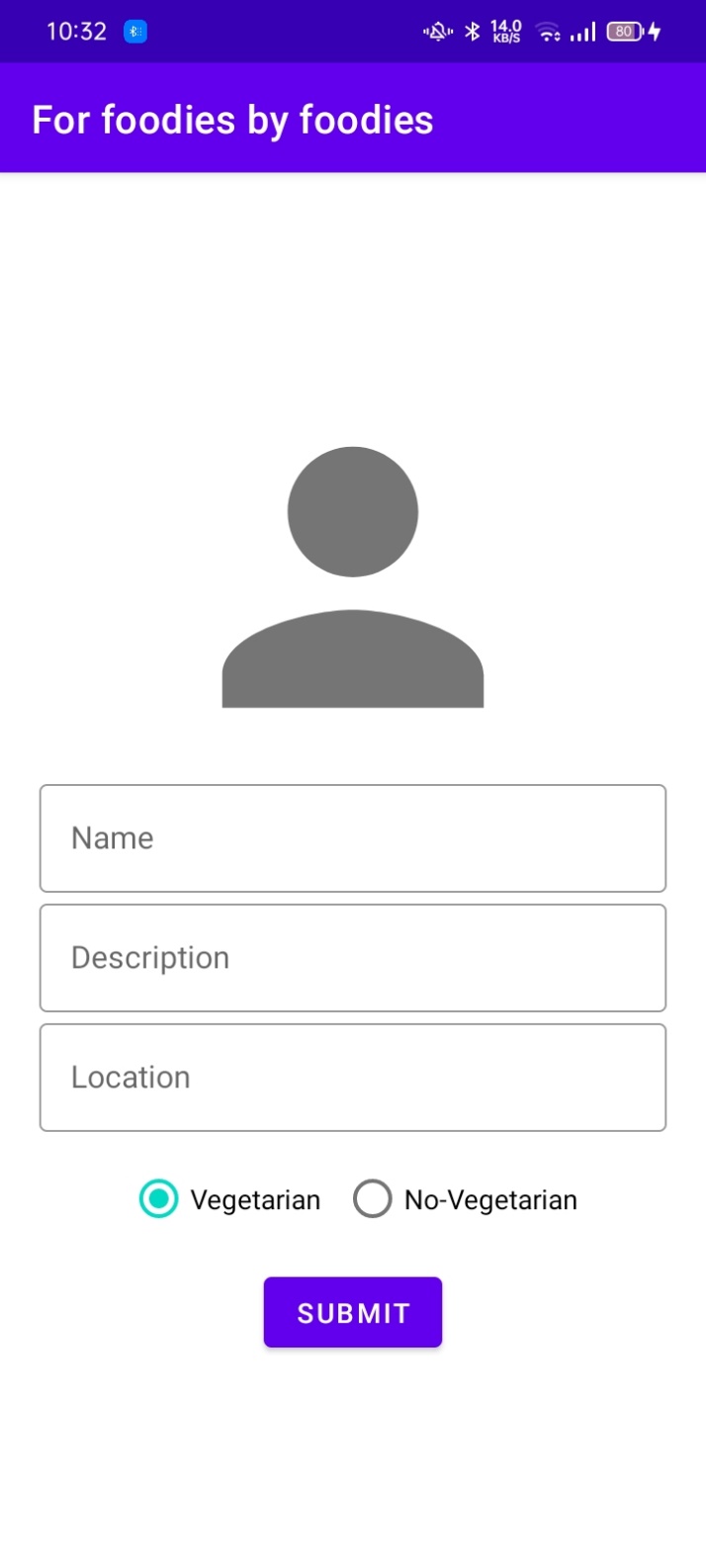
## Profile View



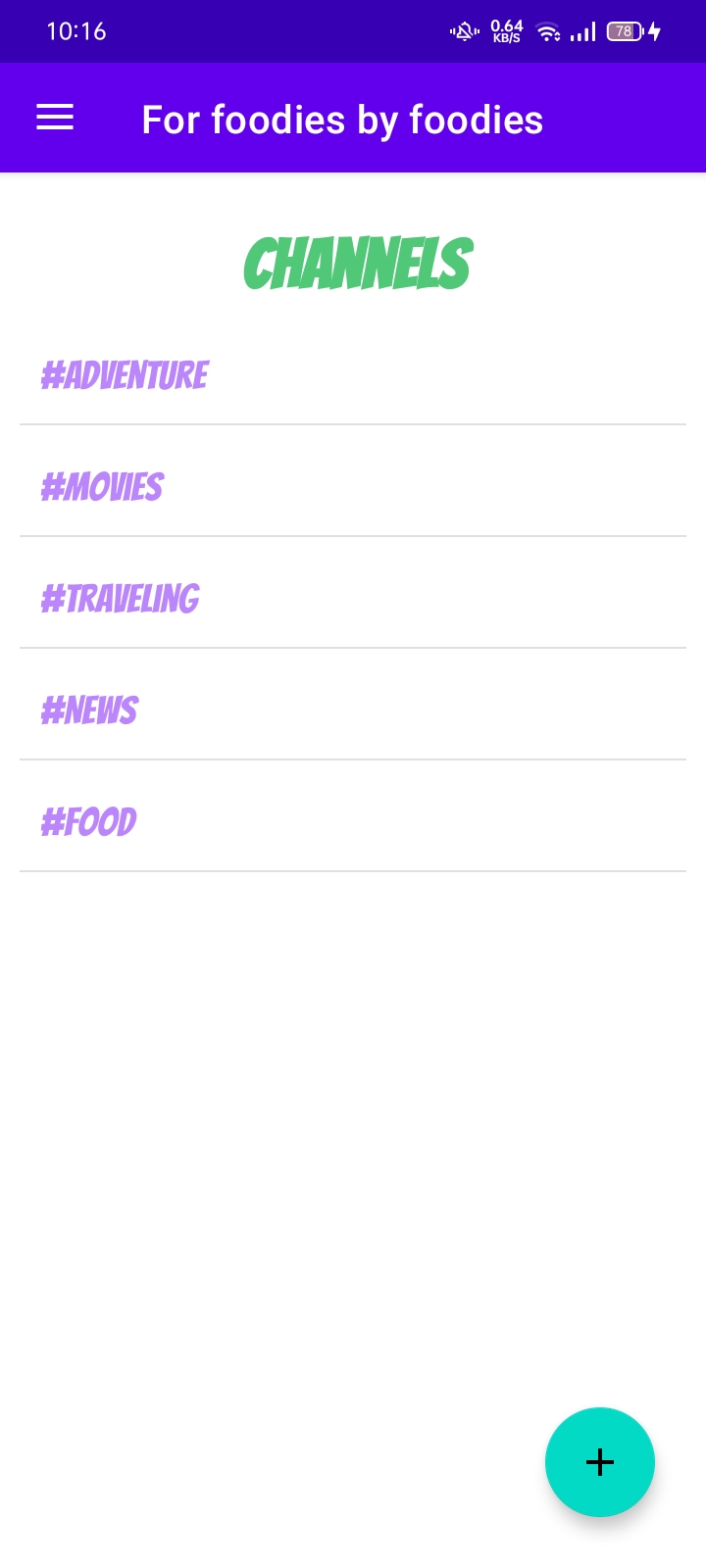
## Street Food View



## Add Street Food Stall View



## Chat Channel View



## Chat View



# **Test Documentation**

## Testing- user

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Input | Output | Result |
| Start Application | Tap the app’s icon | The application starts in the main view showing the splash screen | Successful |
| Enter email and password | Tap login | The user will navigate to dashboard. | Successful |
| Enter wrong email | Tab login | Email invalid toast will be display | Successful |
| Set some fields empty | Tap signup button | Toast will be display | Successful |
| Select restaurant | Tap view button | Navigate to detail screen | Successful |
| Add review | Tap add review | Nothing will appear | Successful |
| View reviews | Tap view reviews | Show all the restaurant related reviews | Successful |
| Select reservation | Tap reservation | Reservation screen will be open | Successful |
| Sign in without email | Tap sign in-no email and password entered | Message “Empty email” will be displayed to the user | Successful |
| Sign in with invalid email format | Tap sign in using a non-valid email format | Message “Invalid email” will be displayed to the user | Successful |
| Entering an incorrect email with correct password format | Tap sign in using incorrect email | Message “Login failed” will be displayed to the user | Successful |
| Entering correct email with short password | Tap sign in using correct email and password less than 8 characters | Message “The password must have at least 8 characters” will be displayed to the user | Successful |
| Entering a correct email with the wrong password | Tap sign in using correct email and password more than 8 characters, but wrong password | Message “Login failed” will be displayed to the user | Successful |
| Entering a correct email with correct password | Tap sign in using correct email and password | Message “Logged in successfully” will be displayed to the user and automatically returns to the main menu | Successful |
| Select” Sign up” | Tap” Sign up” | Register page will be available to the user | Successful |
| Select” Sign out” | Tap” Sign out” | Message “Sign out” will be displayed to the user, and the account is now anonymous | Successful |
| Select Profile | Tap Profile | User profile screen will be shown | Successful |
| Send Message to channel | Add message in field and tap send button | Message is added in list | Successful |
| Send message with empty field | Tap send with no message | Show message to enter some data | Successful |

## Testing- Critic user

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Input | Output | Result |
| Start Application | Tap the app’s icon | The application starts in the main view showing the splash screen | Successful |
| Enter email and password | Tap login | The user will navigate to dashboard. | Successful |
| Enter wrong email | Tab login | Email invalid toast will be display | Successful |
| Set some fields empty | Tap signup button | Toast will be display | Successful |
| Select restaurant | Tap view button | Navigate to detail screen | Successful |
| Add review | Tap add review | Nothing will appear | Successful |
| View reviews | Tap view reviews | Show all the restaurant related reviews | Successful |
| Select reservation | Tap reservation | Reservation screen will be open | Successful |
| Sign in without email | Tap sign in-no email and password entered | Message “Empty email” will be displayed to the user | Successful |
| Sign in with invalid email format | Tap sign in using a non-valid email format | Message “Invalid email” will be displayed to the user | Successful |
| Entering an incorrect email with correct password format | Tap sign in using incorrect email | Message “Login failed” will be displayed to the user | Successful |
| Entering correct email with short password | Tap sign in using correct email and password less than 8 characters | Message “The password must have at least 8 characters” will be displayed to the user | Successful |
| Entering a correct email with the wrong password | Tap sign in using correct email and password more than 8 characters, but wrong password | Message “Login failed” will be displayed to the user | Successful |
| Entering a correct email with correct password | Tap sign in using correct email and password | Message “Logged in successfully” will be displayed to the user and automatically returns to the main menu | Successful |
| Select” Sign up” | Tap” Sign up” | Register page will be available to the user | Successful |
| Select” Sign out” | Tap” Sign out” | Message “Sign out” will be displayed to the user, and the account is now anonymous | Successful |
| Select Profile | Tap Profile | User profile screen will be shown | Successful |
| Send Message to channel | Add message in field and tap send button | Message is added in list | Successful |
| Send message with empty field | Tap send with no message | Show message to enter some data | Successful |
| Add Restaurant Review | Press add button to add review | Show a dialogue with rating and input field | Successful |
| Add Restaurant with no data | Tap submit with no dtaa | Show message to fill the field | Successful |

## Testing- Admin user

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Input | Output | Result |
| Start Application | Tap the app’s icon | The application starts in the main view showing the splash screen | Successful |
| Enter email and password | Tap login | The user will navigate to dashboard. | Successful |
| Enter wrong email | Tab login | Email invalid toast will be display | Successful |
| Set some fields empty | Tap signup button | Toast will be display | Successful |
| Select restaurant | Tap view button | Navigate to detail screen | Successful |
| Add review | Tap add review | Nothing will appear | Successful |
| View reviews | Tap view reviews | Show all the restaurant related reviews | Successful |
| Select reservation | Tap reservation | Reservation screen will be open | Successful |
| Sign in without email | Tap sign in-no email and password entered | Message “Empty email” will be displayed to the user | Successful |
| Sign in with invalid email format | Tap sign in using a non-valid email format | Message “Invalid email” will be displayed to the user | Successful |
| Entering an incorrect email with correct password format | Tap sign in using incorrect email | Message “Login failed” will be displayed to the user | Successful |
| Entering correct email with short password | Tap sign in using correct email and password less than 8 characters | Message “The password must have at least 8 characters” will be displayed to the user | Successful |
| Entering a correct email with the wrong password | Tap sign in using correct email and password more than 8 characters, but wrong password | Message “Login failed” will be displayed to the user | Successful |
| Entering a correct email with correct password | Tap sign in using correct email and password | Message “Logged in successfully” will be displayed to the user and automatically returns to the main menu | Successful |
| Select” Sign up” | Tap” Sign up” | Register page will be available to the user | Successful |
| Select” Sign out” | Tap” Sign out” | Message “Sign out” will be displayed to the user, and the account is now anonymous | Successful |
| Select Profile | Tap Profile | User profile screen will be shown | Successful |
| Send Message to channel | Add message in field and tap send button | Message is added in list | Successful |
| Send message with empty field | Tap send with no message | Show message to enter some data | Successful |
| Create critic account | Tap submit to create account | Created successfully message shown | Successful |
| Create account with no data | Tap submit to create account | Shown message to fille data | Successful |
| Create account with invalid email and password | Tap submit button to create an account | Show a message error while creating account | Successful |

# **Appendix:**

## GitHub Page of the project:

## [radugaraz/Mobile-App-Design-and-Development (github.com)](https://github.com/radugaraz/Mobile-App-Design-and-Development)

## Archived project:



# References

Bassot, B. (2016). *The Reflective Practice Guide: An Interdisciplinary Approach to Critical Reflection.* Abingdon: Routledge.

(Accessed: 27/11/2021).

Google LLC, 2019. *Build a Responsive UI with ConstraintLayout | Android Developers.* [Online]   
Available at: https://developer.android.com/training/constraint-layout  
(Accessed: 28/11/2021).

Google LLC, 2019. *Cloud Firestore | Firebase.* [Online]   
Available at: https://firebase.google.com/docs/firestore  
(Accessed: 28/11/2021).

Google LLC, 2019. *Cloud Storage | Firebase.* [Online]   
Available at: https://firebase.google.com/docs/storage  
(Accessed: 02/12/2021).

Google LLC, 2019. *Guide to app architecture | Android Developers.* [Online]   
Available at: https://developer.android.com/jetpack/docs/guide  
(Accessed: 03/12/2021).

Google LLC, 2019. *LiveData Overview | Android Developers.* [Online]   
Available at: https://developer.android.com/topic/libraries/architecture/livedata  
(Accessed: 03/12/2021).

Google LLC, 2019. *Understand the Activity Lifecycle | Android Developers.* [Online]   
Available at: https://developer.android.com/guide/components/activities/activity-lifecycle  
(Accessed: 04/12/2021).

Google LLC, 2019. *ViewModel Overview | Android Developers.* [Online]   
Available at: https://developer.android.com/topic/libraries/architecture/viewmodel  
(Accessed: 01/12/2021).

Microsoft Corp., 2019. *Architectural principles | Microsoft Docs.* [Online]   
Available at: https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/architectural-principles  
(Accessed: 01/12/2021).

# **Bibliography:**

* [Griffiths, D. and Griffiths, D. (2017) Head first Android development. (2nd ed.) Sebastopol: O'Reilly Media.](https://books.qabusinessschool.com/HeritageScripts/Hapi.dll/search2?searchTerm0=Dawn%20Griffiths%20)
* <https://developer.android.com/>
* <https://firebase.google.com/docs/>