Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Fall, Year: 2025), B.Sc. in CSE (Day)

MealMate

Course Title: Mobile Application Lab
Course Code: CSE 426
Section: 221 D4

<u>Students Details</u>

Name	ID
MD Abu Sufian	221002136
Md Arif Billah	221002520

Submission Date: 27/08/2025
Course Teacher's Name: Abida Sultana

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	Signature:
Comments:	Date:

Contents

Chapter 1

Introduction

1.1 Overview

MealMate is an Android-based donation application designed to minimize food wastage and support underprivileged communities. The application connects donors—such as restaurants, event organizers, and households—with organizations or individuals in need. Apart from food, the app also facilitates the donation of other essential items such as clothes, books, and utensils. By integrating modern technology with social responsibility, MealMate aims to foster sustainable practices and reduce the adverse effects of excessive waste on the environment.

1.2 Motivation

The motivation behind MealMate arises from the growing problem of food wastage in Bangladesh, particularly in urban areas where large events and restaurants generate significant amounts of unused food. Despite this surplus, millions of people still go hungry every day. Additionally, unmanaged waste contributes to environmental pollution and carbon emissions. The absence of a centralized, convenient platform for donating excess food and other essentials further intensifies the problem. MealMate is designed to address these issues by providing a structured system that encourages donations, reduces waste, and helps underprivileged communities.

1.3 Problem Definition

1.3.1 Problem Statement

Despite the availability of surplus food and essential items, there is no effective system in place to bridge the gap between donors and recipients. As a result, valuable resources are wasted, leading to environmental degradation, while underprivileged communities continue to suffer from hunger and lack of basic necessities.

1.3.2 Complex Engineering Problem

Elaborated in below Table:

Table 1.1: Summary of the attributes touched by the mentioned projects

Name of the P Attributess	Explain how to address
P1: Depth of knowledge required	Moderate understanding of mobile app development, real-time database management, and cloud integration.
P2: Range of conflicting requirements	Balancing user-friendliness, performance, and data security for smooth operation.
P3: Depth of analysis required	Understanding donation patterns, user behavior, and optimizing location-based allocation.
P4: Familiarity of issues	Awareness of food safety, logistical challenges, and data privacy concerns.
P5: Extent of applicable codes	Implementing secure coding practices, authentication mechanisms, and map-based services.
P6: Extent of stakeholder involvement and conflicting requirements	Addressing needs of donors, recipients, and volunteers while maintaining operational efficiency.
P7: Interdependence	Integrating real-time database, map API, and notification systems for seamless functionality.

1.4 Design Goals/Objectives

The primary objectives of MealMate are:

Ob1: Minimize food wastage by enabling efficient collection and distribution of excess food.

Ob2: Offer a seamless platform for donating other essentials such as clothes, books, and utensils.

Ob3: Promote sustainable consumption practices and encourage a culture of sharing and social responsibility.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

This chapter focuses on the design, architecture, algorithms, and implementation details of the MealMate Android application. The development process follows a structured approach, ensuring usability, performance, and scalability while addressing the problem of food wastage and resource donation.

2.2 Architectural design

MealMate follows a three-tier architecture:

1. Presentation Layer (Front-End)

Developed using Android Studio.

Provides user interface for donors, recipients, and volunteers.

Integrates map API for location selection and real-time navigation.

2. Application Layer (Business Logic)

Handles user authentication, donation requests, and status updates.

Processes donation information and matches donors with recipients based on proximity and availability.

3. Data Layer (Back-End)

Firebase/MongoDB for storing user profiles, donation records, and real-time updates.

Ensures secure data storage with authentication and encryption.

2.3 Generalized Algorithm

Step 1: User Registration/Login using Firebase Authentication.
Step 2: Donor lists food or essential items for donation.
Step 3: App checks recipient requests and matches availability.
Step 4: Location and pickup/delivery details shared through Map API.
Step 5: Donation marked as completed once collected by recipient/volunteer.
Step 6: Database updated and user notified.

2.4 Project Details

MealMate is an Android-based donation application designed to minimize food wastage and support underprivileged communities. It provides a platform where donors can list surplus food, clothes, books, and utensils, while recipients can browse and request available items. The app integrates Firebase for real-time data storage, authentication, and notifications, along with Google Maps API for location tracking and pickup coordination. A Subtractive Filtering Algorithm ensures only valid donations are displayed, while the Reverse Affine Method reallocates unclaimed donations to nearby recipients. MealMate combines technology and social responsibility to promote sustainability and community welfare.

.

.

2.5 Procedure

2.5.1 Functional Requirements

- User Authentication: Secure login for donors and recipients.
- Donation Listing: Option to list food, clothes, books, and utensils.
- Real-Time Updates: Notifications for donation acceptance, pickup, and completion.
- Location Services: Map API integration for precise drop-off and collection points.
- Data Storage: Cloud-based database for records and analytics.

2.5.2 Non-Functional Requirements

- Scalability: Ability to handle large user base and data growth.
- Security: Encrypted communication between app and database.
- Performance: Quick response time for database queries and UI interactions.
- Usability: Intuitive design for users with minimal technical knowledge.
- Availability: High uptime with Firebase hosting and database services.

2.5.3 SDLCModel Selection

- The project adopts the Agile Model of Software Development Life Cycle.
- Allows iterative development and quick incorporation of user feedback.
- Facilitates regular updates, testing, and enhancement.
- Ensures faster delivery of core features while improving functionality over time.

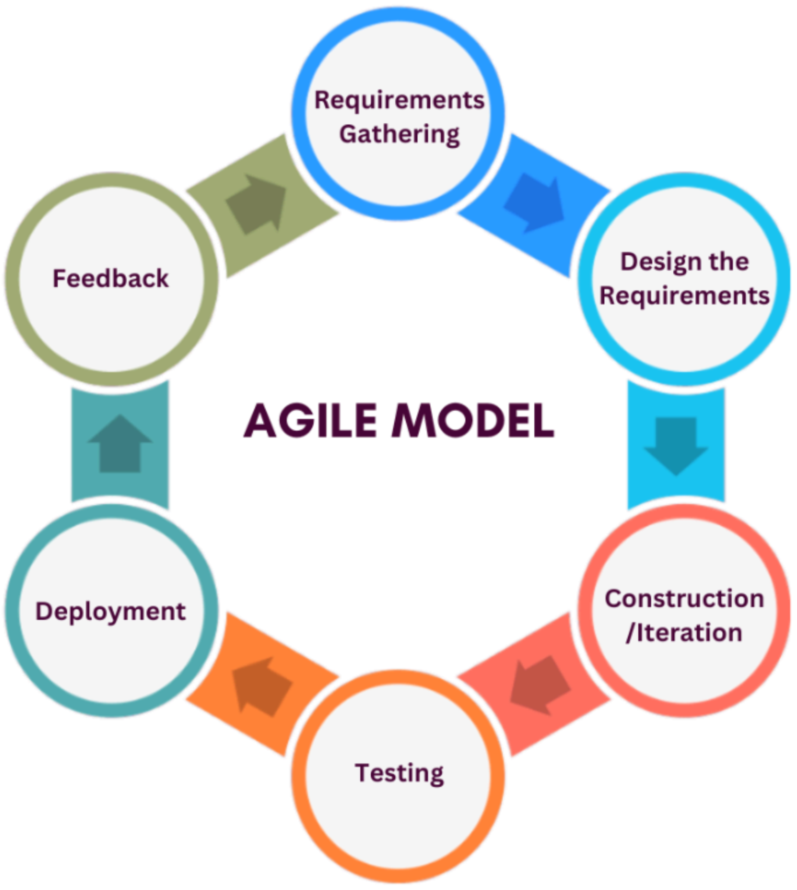


Figure 2.4: Agile Model

.

2.6 Implementation

This Mealmate details the implementation

2.6.4 Code

About

```
package com.example.mealmate_v2;

import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;

public class About extends AppCompatActivity {

   CardView instagram, facebook, twitter;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_about);

       instagram = findViewById(R.id.instagram);
       facebook = findViewById(R.id.facebook);
       twitter = findViewById(R.id.twitter);

       instagram.setOnClickListener(v -> {
           Intent myWebLink = new Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.instagram.com"));
           startActivity(myWebLink);
       });

       facebook.setOnClickListener(v -> {
           Intent myWebLink = new Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.facebook.com"));
           startActivity(myWebLink);
       });

       twitter.setOnClickListener(v -> {
           Intent myWebLink = new Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.twitter.com"));
           startActivity(myWebLink);
       });
   }
}
```

Contact

```java
package com.example.mealmate_v2;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class Contact extends AppCompatActivity {

    EditText name, email, message;
    Button submit;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contact);

        name = findViewById(R.id.name);
        email = findViewById(R.id.email);
        message = findViewById(R.id.message);
        submit = findViewById(R.id.submit);

        submit.setOnClickListener(v -> {
            String n = name.getText().toString().trim();
            String e = email.getText().toString().trim();
            String m = message.getText().toString().trim();

            if(n.isEmpty() || e.isEmpty() || m.isEmpty()){
                Toast.makeText(Contact.this, "Please fill all fields",
Toast.LENGTH_SHORT).show();
            } else {
                sendEmail(n, e, m);
            }
        });
    }

    private void sendEmail(String name, String email, String message) {
        String subject = "Contact Form Message from " + name;
        String body = "Name: " + name + "\nEmail: " + email +
"\n\nMessage:\n" + message;

        Intent intent = new Intent(Intent.ACTION_SENDTO);

intent.setData(Uri.parse("mailto:221002136@student.green.edu.bd")); //
recipient
        intent.putExtra(Intent.EXTRA_SUBJECT, subject);
        intent.putExtra(Intent.EXTRA_TEXT, body);

        try {
            startActivity(Intent.createChooser(intent, "Send email
via"));
        } catch (android.content.ActivityNotFoundException ex) {
            Toast.makeText(this, "No email clients installed.",
Toast.LENGTH_SHORT).show();
```

```
        }

        // Clear fields after sending
        this.name.setText("");
        this.email.setText("");
        this.message.setText("");
    }
}
```

Donate

```
package com.example.mealmate_v2;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Bundle;
import android.text.TextUtils;
import android.widget.Button;
import android.widget.EditText;
import android.widget.FrameLayout;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

public class Donate extends AppCompatActivity {

   EditText donorName, foodItem, phone, description;
   Button submit;
   SQLiteHelper dbHelper;

   private GoogleMap mMap;
   private double selectedLat = 0.0, selectedLng = 0.0;
   private static final int LOCATION_PERMISSION_REQUEST = 100;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_donate);

       donorName = findViewById(R.id.donorname);
       foodItem = findViewById(R.id.fooditem);
       phone = findViewById(R.id.phone);
       description = findViewById(R.id.description);
```

```
        submit = findViewById(R.id.submit);
        dbHelper = new SQLiteHelper(this);

        // Map initialization
        FrameLayout mapContainer = findViewById(R.id.mapContainer);
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.mapContainer);

        if (mapFragment == null) {
            mapFragment = SupportMapFragment.newInstance();
            getSupportFragmentManager().beginTransaction()
                    .replace(R.id.mapContainer, mapFragment)
                    .commit();
        }

        mapFragment.getMapAsync(googleMap -> {
            mMap = googleMap;
            mMap.getUiSettings().setZoomControlsEnabled(true);

            // Enable user location if permission granted
            if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)
                    == PackageManager.PERMISSION_GRANTED) {
                mMap.setMyLocationEnabled(true);

                // Move camera to user's current location
                FusedLocationProviderClient fusedLocationClient =
LocationServices.getFusedLocationProviderClient(this);

fusedLocationClient.getLastLocation().addOnSuccessListener(location ->
{
                    if (location != null) {
                        LatLng userLoc = new
LatLng(location.getLatitude(), location.getLongitude());

mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(userLoc, 15));
                    }
                });
            } else {
                ActivityCompat.requestPermissions(this,
                        new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
                        LOCATION_PERMISSION_REQUEST);
            }

            // Map click listener for selecting donation location
            mMap.setOnMapClickListener(latLng -> {
                mMap.clear(); // remove previous marker
                mMap.addMarker(new MarkerOptions()
                        .position(latLng)
                        .title("Donation Location"));
                selectedLat = latLng.latitude;
                selectedLng = latLng.longitude;
            });
        });
        // Submit donation
        submit.setOnClickListener(v -> {
            String name = donorName.getText().toString().trim();
            String item = foodItem.getText().toString().trim();
            String ph = phone.getText().toString().trim();
            String desc = description.getText().toString().trim();

            if (TextUtils.isEmpty(name)) {
                donorName.setError("Donor Name is required");
                return;
            }
            if (TextUtils.isEmpty(item)) {
                foodItem.setError("Food Item is required");
                return;
            }
            if (TextUtils.isEmpty(ph)) {
                phone.setError("Phone is required");
                return;
            }
            if (TextUtils.isEmpty(desc)) {
                description.setError("Description is required");
                return;
            }
            if (selectedLat == 0.0 && selectedLng == 0.0) {
                Toast.makeText(Donate.this, "Please select a location
on the map", Toast.LENGTH_SHORT).show();
                return;
            }

            // Save donation in SQLite with location
            boolean inserted = dbHelper.insertDonation(name, item, ph,
desc, selectedLat, selectedLng);
            if (inserted) {
                Toast.makeText(Donate.this, "Donation submitted!",
Toast.LENGTH_SHORT).show();
                donorName.setText("");
                foodItem.setText("");
                phone.setText("");
                description.setText("");
                mMap.clear();
                selectedLat = 0.0;
                selectedLng = 0.0;
            } else {
                Toast.makeText(Donate.this, "Error submitting
donation", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Food Map

```java
package com.example.mealmate_v2;

import android.database.Cursor;
import android.os.Bundle;
import androidx.fragment.app.FragmentActivity;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

public class FoodMap extends FragmentActivity implements
OnMapReadyCallback {

   private GoogleMap mMap;
   private SQLiteHelper dbHelper;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_food_map);

       dbHelper = new SQLiteHelper(this);

       SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
               .findFragmentById(R.id.google_map);
       if (mapFragment != null) {
           mapFragment.getMapAsync(this);
       }
   }

   @Override
   public void onMapReady(GoogleMap googleMap) {
       mMap = googleMap;
       mMap.getUiSettings().setZoomControlsEnabled(true);

       // Fetch all donations from database
       Cursor cursor = dbHelper.getAllDonations();
       if (cursor != null && cursor.getCount() > 0) {
           while (cursor.moveToNext()) {
               String donor =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_DONOR_N
AME));
               String food =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_FOOD_IT
EM));
               String desc =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_DESCRIP
TION));
               double lat =
cursor.getDouble(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_LAT));
               double lng =
cursor.getDouble(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_LNG));
```

```
               LatLng loc = new LatLng(lat, lng);

               // Check if this is a receiver record (food_item =
"Received Food")
               boolean isReceiver = food.equalsIgnoreCase("Received
Food");

               mMap.addMarker(new MarkerOptions()
                       .position(loc)
                       .title(donor + " - " + food)
                       .snippet(desc)
                       .icon(isReceiver ?
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_GREE
N)
                               :
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_RED)
));
           }
           cursor.close();
       }

       // Move camera to Dhaka as default
       LatLng dhaka = new LatLng(23.8103, 90.4125);
       mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(dhaka, 12));
   }
}
```

History

```
package com.example.mealmate_v2;

import android.database.Cursor;
import android.os.Bundle;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;
import android.content.Context;
import android.view.ViewGroup;

public class History extends AppCompatActivity {

   LinearLayout showData;
   SQLiteHelper dbHelper;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_history);

       showData = findViewById(R.id.showdata);
       dbHelper = new SQLiteHelper(this);
       Context context = this;
```

```
        // Fetch all donations from database
        Cursor cursor = dbHelper.getAllDonations();

        if (cursor.getCount() == 0) {
            Toast.makeText(this, "No history found",
Toast.LENGTH_SHORT).show();
        } else {
            while (cursor.moveToNext()) {
                // Columns from your donations table
                String donorName =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_DONOR_N
AME));
                String foodItem =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_FOOD_IT
EM));
                String description =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_DESCRIP
TION));

                // Combine info for display
                String dataTextString = donorName + " - " + foodItem +
"\n" + description;

                // Create CardView dynamically
                CardView cardView = new CardView(context);
                LinearLayout.LayoutParams cardParams = new
LinearLayout.LayoutParams(
                        LinearLayout.LayoutParams.MATCH_PARENT,
                        LinearLayout.LayoutParams.WRAP_CONTENT
                );
                cardParams.setMargins(0, 0, 0, 10);
                cardView.setLayoutParams(cardParams);
                cardView.setRadius(12f);
                cardView.setCardElevation(5f);

cardView.setCardBackgroundColor(getResources().getColor(R.color.gray));
                cardView.setClickable(true);

                TextView dataText = new TextView(context);
                dataText.setText(dataTextString);

dataText.setTextColor(getResources().getColor(R.color.white));
                dataText.setTextSize(16);
                dataText.setPadding(20, 20, 20, 20);

                cardView.addView(dataText);
                showData.addView(cardView);
            }
        }
        cursor.close();

        // Delete button
        CardView delete = findViewById(R.id.delete);
        delete.setOnClickListener(v -> {
```

```
dbHelper.getWritableDatabase().delete(SQLiteHelper.TABLE_DONATIONS,
null, null);
            showData.removeAllViews(); // remove all cards from UI
            Toast.makeText(History.this, "History deleted",
Toast.LENGTH_SHORT).show();
        });
    }
}
```

History Adapter

```
package com.example.mealmate_v2;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import java.util.ArrayList;

public class HistoryAdapter extends
RecyclerView.Adapter<HistoryAdapter.HistoryViewHolder> {

    Context context;
    ArrayList<HistoryItem> list;

    public HistoryAdapter(Context context, ArrayList<HistoryItem> list)
{
        this.context = context;
        this.list = list;
    }

    @NonNull
    @Override
    public HistoryViewHolder onCreateViewHolder(@NonNull ViewGroup
parent, int viewType) {
        View v =
LayoutInflater.from(context).inflate(R.layout.item_history, parent,
false);
        return new HistoryViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull HistoryViewHolder holder, int
position) {
        HistoryItem item = list.get(position);
        holder.name.setText(item.getName());
        holder.type.setText(item.getType());
        holder.description.setText(item.getDescription());

        holder.delete.setOnClickListener(v -> {
            list.remove(position);
            notifyItemRemoved(position);
        });
    }

    @Override
    public int getItemCount() {
        return list.size();
    }

    public static class HistoryViewHolder extends
RecyclerView.ViewHolder {

        TextView name, type, description;
        View delete;

        public HistoryViewHolder(@NonNull View itemView) {
            super(itemView);
            name = itemView.findViewById(R.id.name);
            type = itemView.findViewById(R.id.type);
            description = itemView.findViewById(R.id.description);
            delete = itemView.findViewById(R.id.delete);
        }
    }
}
```

History Item

```
package com.example.mealmate_v2;

public class HistoryItem {

    private String name, type, description;

    public HistoryItem(String name, String type, String description) {
        this.name = name;
        this.type = type;
        this.description = description;
    }

    public String getName() { return name; }
    public String getType() { return type; }
    public String getDescription() { return description; }
}
```

Landing Page

```
package com.example.mealmate_v2;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;

public class LandingPage extends AppCompatActivity {

   CardView cardLogin, cardRegister, cardAboutus;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_landingpage);

       cardLogin = findViewById(R.id.cardLogin);
       cardRegister = findViewById(R.id.cardRegister);
       cardAboutus = findViewById(R.id.cardAboutus);

       cardLogin.setOnClickListener(v -> startActivity(new
Intent(LandingPage.this, Logup.class)));
       cardRegister.setOnClickListener(v -> startActivity(new
Intent(LandingPage.this, Signup.class)));

       cardAboutus.setOnClickListener(v -> startActivity(new
Intent(LandingPage.this, About.class)));
   }
}
```

LogUp

```
package com.example.mealmate_v2;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.text.TextUtils;
import android.widget.*;

public class Logup extends AppCompatActivity {

   EditText mEmail, mPassword;
   Button mLoginBtn;
   TextView mRegister; // for signup page link
   SQLiteHelper dbHelper;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_logup);

       mEmail = findViewById(R.id.email);
       mPassword = findViewById(R.id.password);
       mLoginBtn = findViewById(R.id.login);
```

```
        mRegister = findViewById(R.id.register);

        dbHelper = new SQLiteHelper(this);

        // LOGIN
        mLoginBtn.setOnClickListener(v -> {
            String email = mEmail.getText().toString().trim();
            String password = mPassword.getText().toString().trim();

            if (TextUtils.isEmpty(email)) {
                mEmail.setError("Email is Required.");
                return;
            }
            if (TextUtils.isEmpty(password)) {
                mPassword.setError("Password is Required.");
                return;
            }

            if (dbHelper.checkUser(email, password)) {
                Toast.makeText(Logup.this, "Login Successful",
Toast.LENGTH_SHORT).show();

                // -------- SESSION SAVE --------
                SharedPreferences prefs =
getSharedPreferences("LoginPrefs", MODE_PRIVATE);
                SharedPreferences.Editor editor = prefs.edit();
                editor.putBoolean("isLoggedIn", true);
                editor.apply();

                // Go to MainActivity
                startActivity(new Intent(Logup.this,
MainActivity.class));
                finish();
            } else {
                Toast.makeText(Logup.this, "Invalid Email or Password",
Toast.LENGTH_SHORT).show();
            }
        });

        // REGISTER (signup page)
        mRegister.setOnClickListener(v -> {
            Intent intent = new Intent(Logup.this, Signup.class); // <-
your signup activity
            startActivity(intent);
        });
    }
}
```

MainActivity

```
package com.example.mealmate_v2;

import android.content.Intent;
```

```java
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;

import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;

public class MainActivity extends AppCompatActivity {

   CardView donate, receive, logout, foodmap, about, contact, mypin,
history;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);

       donate = findViewById(R.id.cardDonate);
       receive = findViewById(R.id.cardReceive);
       logout = findViewById(R.id.cardLogout);
       foodmap = findViewById(R.id.cardFoodmap);
       mypin = findViewById(R.id.cardMyPin);
       history = findViewById(R.id.cardHistory);
       about = findViewById(R.id.cardAboutus);
       contact = findViewById(R.id.cardContact);

       // -------- SESSION CHECK ----------
       SharedPreferences prefs = getSharedPreferences("LoginPrefs",
MODE_PRIVATE);
       boolean isLoggedIn = prefs.getBoolean("isLoggedIn", false);

       if (!isLoggedIn) {
           Intent intent = new Intent(MainActivity.this, Logup.class);
           intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP |
Intent.FLAG_ACTIVITY_CLEAR_TASK | Intent.FLAG_ACTIVITY_NEW_TASK);
           startActivity(intent);
       }

       // --------- CARD CLICK EVENTS ---------
       donate.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), Donate.class)));

       receive.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), Receive.class)));

       foodmap.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), FoodMap.class)));

       about.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), About.class)));

       mypin.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), MyPin.class)));

       history.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), UserdataActivity.class)));
```

```
        contact.setOnClickListener(v -> startActivity(new
Intent(getApplicationContext(), Contact.class)));

        // --------- LOGOUT ---------
        logout.setOnClickListener(v -> {
            SharedPreferences.Editor editor = prefs.edit();
            editor.putBoolean("isLoggedIn", false);
            editor.apply();

            Intent intent = new Intent(MainActivity.this, Logup.class);
            intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP |
Intent.FLAG_ACTIVITY_CLEAR_TASK | Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(intent);
        });
    }
}
```

MyPin

```
package com.example.mealmate_v2;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Bundle;
import androidx.annotation.NonNull;
import androidx.core.app.ActivityCompat;
import androidx.fragment.app.FragmentActivity;

import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.tasks.OnSuccessListener;

public class MyPin extends FragmentActivity implements
OnMapReadyCallback {

    private GoogleMap mMap;
    private FusedLocationProviderClient fusedLocationClient;
    private static final int LOCATION_PERMISSION_REQUEST = 1000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my_pin);

        fusedLocationClient =
LocationServices.getFusedLocationProviderClient(this);
```

```
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.google_map);
        if(mapFragment != null) {
            mapFragment.getMapAsync(this);
        }
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        // Check permission
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED
                && ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
LOCATION_PERMISSION_REQUEST);
            return;
        }

        mMap.setMyLocationEnabled(true); // show blue dot

        // Get last known location
        fusedLocationClient.getLastLocation().addOnSuccessListener(this,
location -> {
            if (location != null) {
                LatLng userLocation = new LatLng(location.getLatitude(),
location.getLongitude());
                mMap.addMarker(new
MarkerOptions().position(userLocation).title("You are here"));

mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(userLocation, 15f));
            }
        });
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull
String[] permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
        if (requestCode == LOCATION_PERMISSION_REQUEST &&
grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
            onMapReady(mMap); // permission granted, reload map
        }
    }
}
```

Receive

```
package com.example.mealmate_v2;

import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;

public class Receive extends AppCompatActivity implements
OnMapReadyCallback {

   private EditText receiverName, description;
   private Button submit;
   private SQLiteHelper dbHelper;

   private GoogleMap mMap;
   private LatLng selectedLocation = null;
   private Marker selectedMarker = null;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_receive);

       // Initialize views
       receiverName = findViewById(R.id.receivername);
       description = findViewById(R.id.description);
       submit = findViewById(R.id.submit);

       // Initialize database helper
       dbHelper = new SQLiteHelper(this);

       // Initialize map
       SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.mapContainer);
       if (mapFragment != null) {
           mapFragment.getMapAsync(this);
       }

       // Submit button listener
       submit.setOnClickListener(v -> {
           String name = receiverName.getText().toString().trim();
           String desc = description.getText().toString().trim();

           if (name.isEmpty() || desc.isEmpty()) {
               Toast.makeText(this, "Enter all fields",
Toast.LENGTH_SHORT).show();
               return;
           }

           if (selectedLocation == null) {
               Toast.makeText(this, "Please select a location on the
map", Toast.LENGTH_SHORT).show();
               return;
           }

           // Insert donation into database
           boolean inserted = dbHelper.insertDonation(
                   name,
                   "Received Food",
                   null, // No phone for receiver
                   desc,
                   selectedLocation.latitude,
                   selectedLocation.longitude
           );

           if (inserted) {
               Toast.makeText(this, "Submitted successfully",
Toast.LENGTH_SHORT).show();
               receiverName.setText("");
               description.setText("");
               selectedLocation = null;

               // Remove marker after submission
               if (selectedMarker != null) {
                   selectedMarker.remove();
                   selectedMarker = null;
               }
           } else {
               Toast.makeText(this, "Failed to submit",
Toast.LENGTH_SHORT).show();
           }
       });
   }

   @Override
   public void onMapReady(GoogleMap googleMap) {
       mMap = googleMap;
       mMap.getUiSettings().setZoomControlsEnabled(true);

       // Default location: Dhaka
       LatLng dhaka = new LatLng(23.8103, 90.4125);
       mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(dhaka, 12));

       // Handle map clicks
       mMap.setOnMapClickListener(latLng -> {
           selectedLocation = latLng;

           // Remove old marker
           if (selectedMarker != null) {
               selectedMarker.remove();
           }

           // Add new marker
           selectedMarker = mMap.addMarker(new 
com.google.android.gms.maps.model.MarkerOptions()
                   .position(latLng)
                   .title("Selected Location")
                   .snippet("Tap Submit to save")
                    
.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HU
E_GREEN)));

           selectedMarker.showInfoWindow();
       });
   }
}
```

SignUp

```
package com.example.mealmate_v2;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.*;

public class Signup extends AppCompatActivity {

    EditText mFullName, mEmail, mPassword, mPhone;
    Button mRegisterBtn;
    TextView mLoginBtn;
    SQLiteHelper dbHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_signup);

        mFullName = findViewById(R.id.name);
        mEmail = findViewById(R.id.email);
        mPassword = findViewById(R.id.password);
        mPhone = findViewById(R.id.phone);
        mRegisterBtn = findViewById(R.id.register);
        mLoginBtn = findViewById(R.id.login);

        dbHelper = new SQLiteHelper(this);

        // REGISTER
        mRegisterBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name = mFullName.getText().toString().trim();
                String email = mEmail.getText().toString().trim();
                String password = mPassword.getText().toString().trim();
                String phone = mPhone.getText().toString().trim();

                if (TextUtils.isEmpty(name)) {
                    mFullName.setError("Name is Required.");
                    return;
                }
                if (TextUtils.isEmpty(email)) {
                    mEmail.setError("Email is Required.");
                    return;
                }
                if (TextUtils.isEmpty(password)) {
                    mPassword.setError("Password is Required.");
                    return;
                }
                if (password.length() < 6) {
                    mPassword.setError("Password Must be >= 6
Characters");
                    return;
                }

                boolean inserted = dbHelper.insertUser(name, email,
phone, password);
                if (inserted) {
                    Toast.makeText(Signup.this, "User Registered
Successfully", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(Signup.this,
MainActivity.class));
                    finish();
                } else {
                    Toast.makeText(Signup.this, "User Already Exists!",
Toast.LENGTH_SHORT).show();
                }
            }
        });

        // Go to Login
        mLoginBtn.setOnClickListener(v -> {
            startActivity(new Intent(Signup.this, Logup.class));
        });
    }
}
```

SQLLite Helper

```
package com.example.mealmate_v2;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
```

```java
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class SQLiteHelper extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "MealMate.db";
    private static final int DATABASE_VERSION = 3; // bumped version
for schema change

    // Users table
    public static final String TABLE_USERS = "users";
    public static final String COL_USER_ID = "id";
    public static final String COL_USER_NAME = "name";
    public static final String COL_USER_EMAIL = "email";
    public static final String COL_USER_PHONE = "phone";
    public static final String COL_USER_PASSWORD = "password";

    // Donations table
    public static final String TABLE_DONATIONS = "donations";
    public static final String COL_DONATION_ID = "id";
    public static final String COL_DONOR_NAME = "donor_name";
    public static final String COL_FOOD_ITEM = "food_item";
    public static final String COL_DONOR_PHONE = "donor_phone";
    public static final String COL_DESCRIPTION = "description";
    public static final String COL_LAT = "latitude";    // New column
    public static final String COL_LNG = "longitude";   // New column

    public SQLiteHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        // Users table
        db.execSQL("CREATE TABLE " + TABLE_USERS + " (" +
                COL_USER_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                COL_USER_NAME + " TEXT, " +
                COL_USER_EMAIL + " TEXT UNIQUE, " +
                COL_USER_PHONE + " TEXT, " +
                COL_USER_PASSWORD + " TEXT)");

        // Donations table with lat/lng
        db.execSQL("CREATE TABLE " + TABLE_DONATIONS + " (" +
                COL_DONATION_ID + " INTEGER PRIMARY KEY AUTOINCREMENT,
" +
                COL_DONOR_NAME + " TEXT, " +
                COL_FOOD_ITEM + " TEXT, " +
                COL_DONOR_PHONE + " TEXT, " +
                COL_DESCRIPTION + " TEXT, " +
                COL_LAT + " REAL, " +
                COL_LNG + " REAL)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_USERS);
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_DONATIONS);
        onCreate(db);
    }

    // -------- Users methods --------
    public boolean insertUser(String name, String email, String phone,
String password) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(COL_USER_NAME, name);
        values.put(COL_USER_EMAIL, email);
        values.put(COL_USER_PHONE, phone);
        values.put(COL_USER_PASSWORD, password);

        long result = -1;
        try {
            result = db.insertOrThrow(TABLE_USERS, null, values);
        } catch (Exception e) {
            return false; // duplicate email or error
        }
        return result != -1;
    }

    public boolean checkUser(String email, String password) {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_USERS +
                        " WHERE " + COL_USER_EMAIL + "=? AND " +
COL_USER_PASSWORD + "=?",
                new String[]{email, password});
        boolean exists = cursor.getCount() > 0;
        cursor.close();
        return exists;
    }

    // -------- Donations methods --------
    public boolean insertDonation(String donorName, String foodItem,
String phone, String description, double lat, double lng) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(COL_DONOR_NAME, donorName);
        values.put(COL_FOOD_ITEM, foodItem);
        values.put(COL_DONOR_PHONE, phone);
        values.put(COL_DESCRIPTION, description);
        values.put(COL_LAT, lat);   // save latitude
        values.put(COL_LNG, lng);   // save longitude

        long result = db.insert(TABLE_DONATIONS, null, values);
        return result != -1;
    }

    public Cursor getAllDonations() {
        SQLiteDatabase db = this.getReadableDatabase();
        return db.rawQuery("SELECT * FROM " + TABLE_DONATIONS, null);
    }
}
```

UserDataActivity

```
package com.example.mealmate_v2;

import android.database.Cursor;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import java.util.ArrayList;

public class UserdataActivity extends AppCompatActivity {

   RecyclerView recView;
   HistoryAdapter adapter;
   ArrayList<HistoryItem> itemList;
   SQLiteHelper dbHelper;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.userdata);

       recView = findViewById(R.id.rec_view);
       recView.setLayoutManager(new LinearLayoutManager(this));

       dbHelper = new SQLiteHelper(this);
       itemList = new ArrayList<>();


       Cursor cursor = dbHelper.getAllDonations();
       if (cursor != null && cursor.getCount() > 0) {
           while (cursor.moveToNext()) {
               String donorName =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_DONOR_N
AME));
               String foodItem =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_FOOD_IT
EM));
               String description =
cursor.getString(cursor.getColumnIndexOrThrow(SQLiteHelper.COL_DESCRIP
TION));

               // Add to list
               itemList.add(new HistoryItem(donorName, foodItem,
description));
           }
           cursor.close();
       }

       adapter = new HistoryAdapter(this, itemList);
       recView.setAdapter(adapter);
   }
}
```

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

In this section, we delve into the intricacies of the experimental setup and procedures employed to simulate outcomes effectively

3.1.1 Environment Configuration

The MealMate application was developed and tested on Android Studio with an emulator running Android 12. The backend services were hosted on Firebase, providing authentication, database storage, and real-time synchronization. Google Maps API was integrated to enable location-based services. Performance testing was conducted using both emulator and real Android devices to ensure compatibility, responsiveness, and scalability across various screen sizes and network conditions.

3.1.2 User Scenarios

Testinginvolved multiple real-world user scenarios, such as:

1. A donor listing food items after an event.
2. A recipient browsing nearby donations and requesting pickup.
3. The system reallocating unclaimed donations to other recipients using the Reverse Affine Method.
4. Removal of expired or already-claimed donations through the Subtractive Filtering Algorithm.
5. These scenarios ensured that core features functioned seamlessly under normal and edge-case conditions.

3.2 Results Analysis/Testing

2.6.4 Output

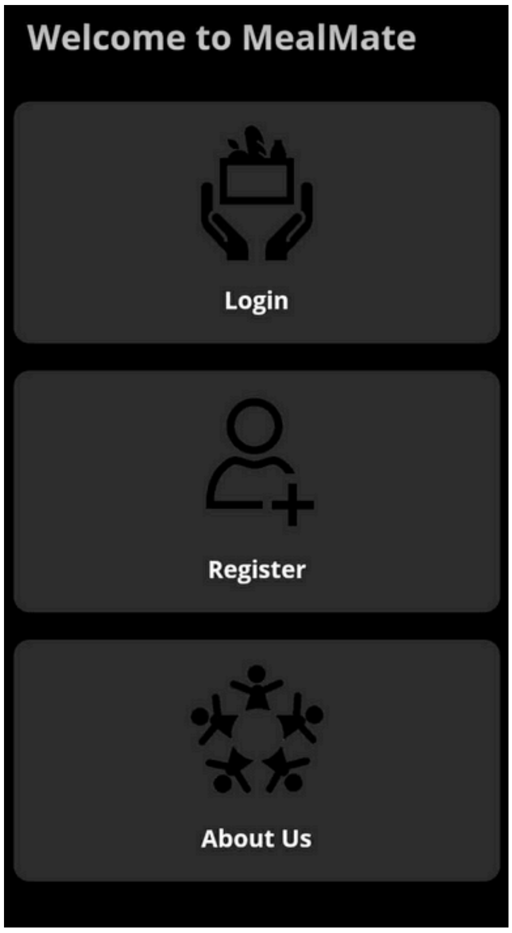


Figure 1.0: Welcome Page

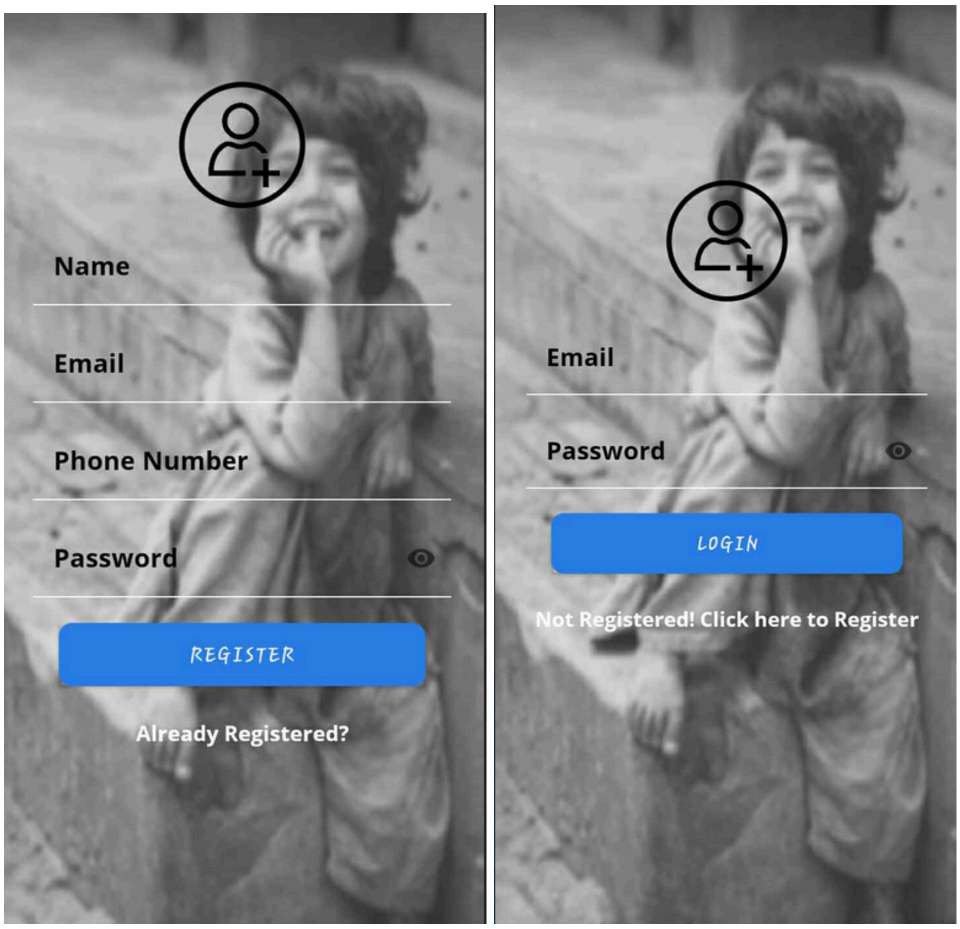


Figure 1.2: Login & Registration

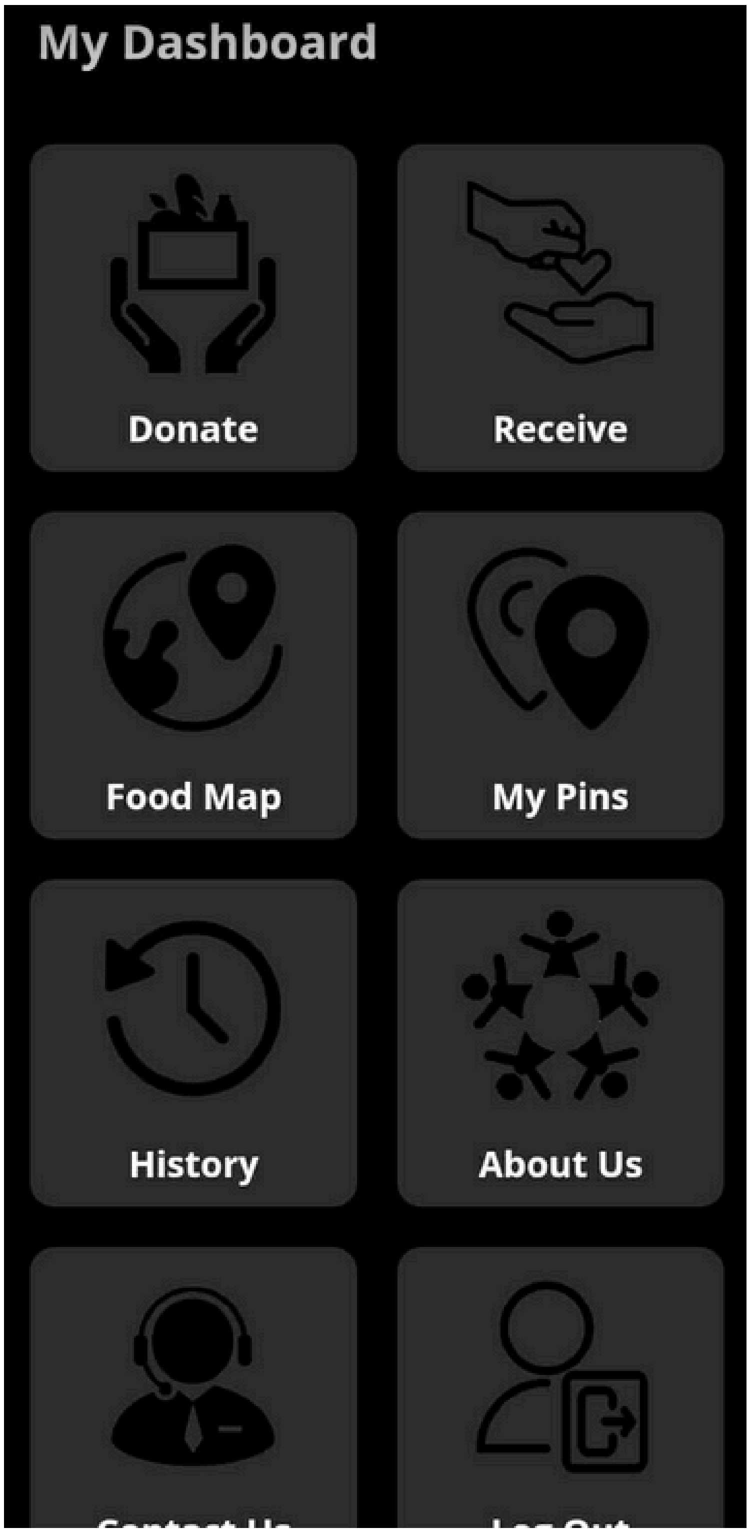


Figure 1.3: Dashboard

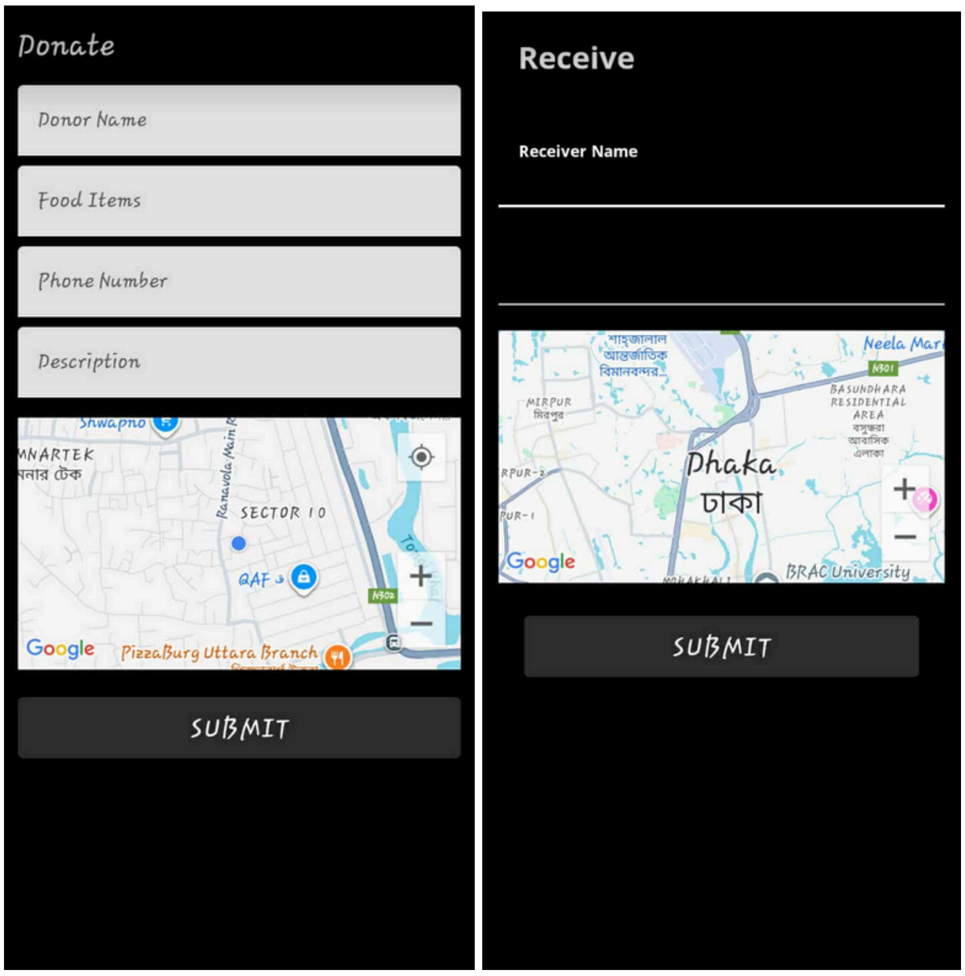


Figure1.4: Donation and Receive.

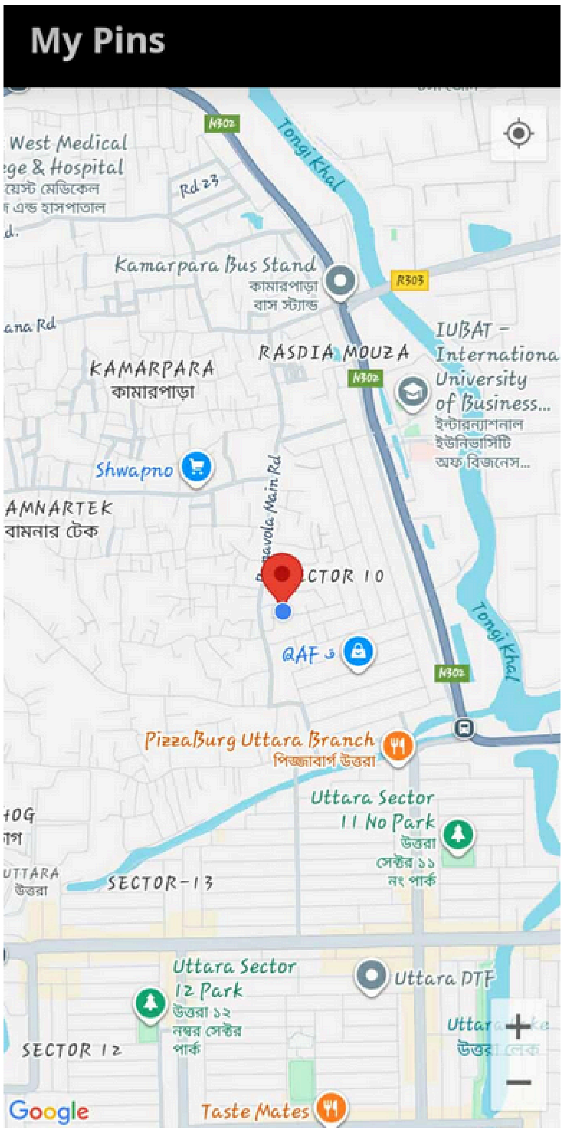
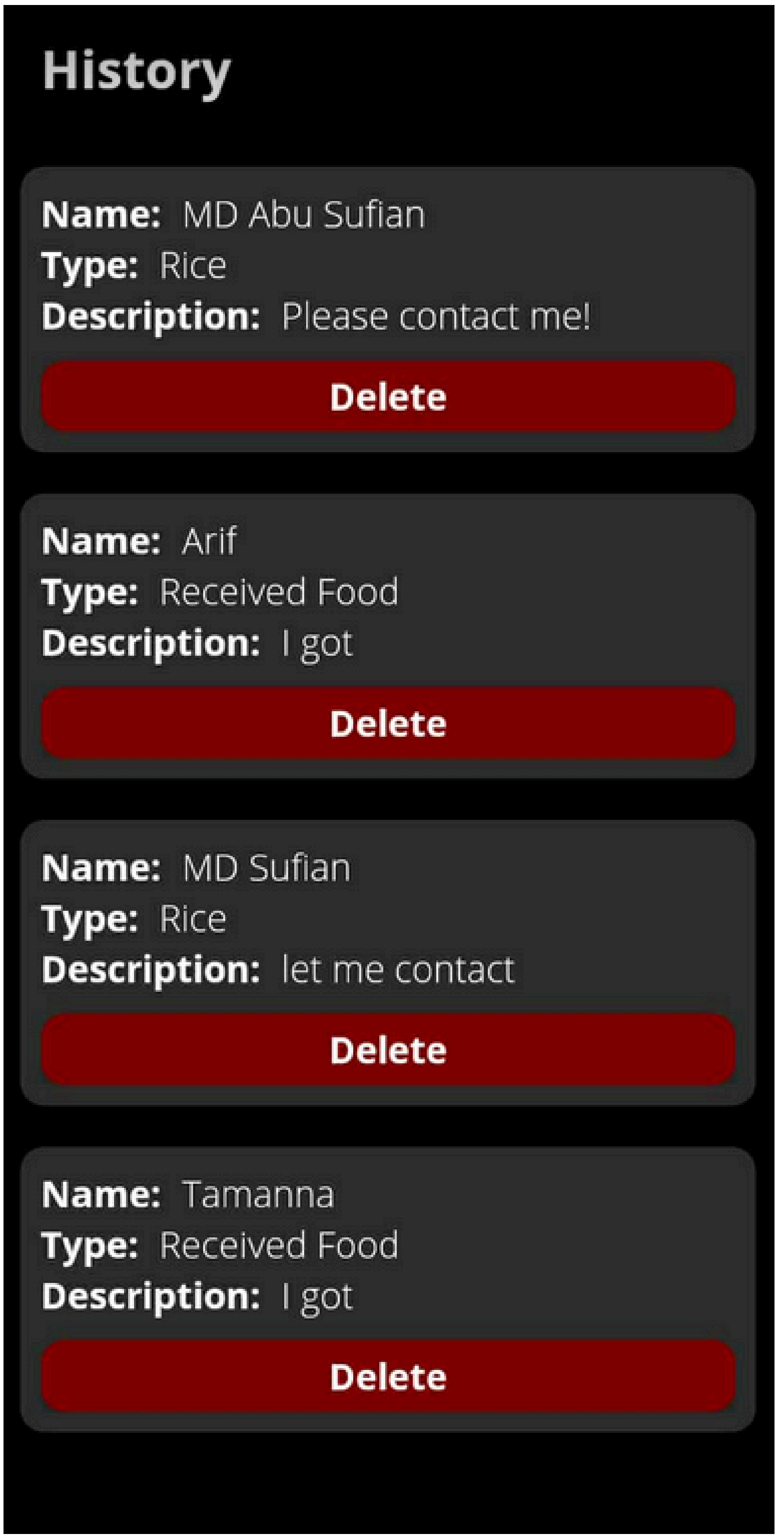


Figure 1.5: My and Food Location


History

Name: MD Abu Sufian
Type: Rice
Description: Please contact me!

Name: Arif
Type: Received Food
Description: I got

Name: MD Sufian
Type: Rice
Description: let me contact

Name: Tamanna
Type: Received Food
Description: I got

Figure 1.6: Meal History

.

3.3 Results Overall Discussion

Testing confirmed that MealMate performs efficiently across different use cases. The Subtractive Filtering Algorithm effectively maintained a clean and accurate donation list, while the Reverse Affine Method ensured optimal reallocation of resources. The system handled concurrent users without noticeable lag, maintained real-time synchronization, and provided a user-friendly experience. These results demonstrate the feasibility of using MealMate as a scalable and reliable platform for donation management.

Chapter 4

Conclusion

4.1 Discussion

MealMate addresses a pressing issue in Bangladesh—food wastage amidst widespread poverty—by creating a technological bridge between donors and recipients. The app successfully integrates real-time database management, location-based services, and efficient filtering/reallocation algorithms to minimize waste and maximize social impact.

4.2 Limitations

Despite its promising performance, the project has certain limitations:

- Limited to Android users; iOS support is yet to be developed.
- Relies heavily on internet connectivity for real-time updates.
- Volunteer-based delivery system not yet automated.
- Does not include AI-based prediction for demand and supply optimization.

4.3 Scope of Future Work

Future development of MealMate will focus on enhancing its usability, scalability, and overall impact. Planned improvements include creating an iOS version to extend accessibility beyond Android users and integrating AI-based analytics to predict donation demand and optimize resource distribution. Additional features, such as a reputation and rating system for donors, recipients, and volunteers, will improve reliability and accountability. Expanding partnerships with NGOs, charities, and government organizations will further strengthen the donation network. Moreover, introducing offline functionality and automated logistics support can make the app more versatile in regions with limited internet connectivity.