

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY,
BELAGAVI - 590 018**



**Mini Project Report On
“HOMIFY APP”**

**A report submitted in partial fulfillment of the requirements for
MOBILE APPLICATION DEVELOPMENT LABORATORY (18CSMP68)**

**in
COMPUTER SCIENCE AND ENGINEERING**

Submitted by

ADITH P KOTIAN

4AL20CS006

ISHWAR PAVAN

4AL20CS050

Under the Guidance of

Mrs. Reena Lobo

Assistant Professor



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY
MOODBIDRI-574225, KARNATAKA**

2022 – 2023

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY

MIJAR, MOODBIDRI D.K. -574225

KARNATAKA



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Mini Project entitled “**HOMIFY APP**” has been successfully completed by

ADITH P KOTIAN

4AL20CS006

ISHWAR PAVAN

4AL20CS050

in the partial fulfillment for the award of Degree of Bachelor of Engineering in Computer and Engineering of the Visvesvaraya Technological University, Belagavi during the year 2022-2023. It is certified that all corrections/suggestions indicated have been incorporated in the report. The Mini project report has been approved as it satisfies the academic requirements in respect of Mini Project Work prescribed for the award of Bachelor of Engineering Degree.

Mrs. Reena Lobo,
Mini Project Guide

Dr. Manjunath Kotari,
HOD CSE

External Viva

Name of the Examiners

Signature with Date

1.

2.

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY
MIJAR, MOODBIDRI D.K. -574225
KARNATAKA



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Declaration

We,

ADITH P KOTIAN
ISHWAR PAVAN

hereby declare that the dissertation entitled, “**HOMIFY APP**” is completed and written by us under the supervision of my guide Mrs. REENA LOBO, Assistant Professor, **Department of Computer Science and Engineering**. Alva's Institute of Engineering And Technology, Moodbidri, **DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING** of the **VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI** during the academic year 2022-2023. The dissertation report is original and it has not been submitted for any other degree in any university.

ADITH P KOTIAN	4AL20CS006
ISHWAR PAVAN	4AL20CS050

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.

So, with gratitude we acknowledge all those whose guidance and encouragement served as beacon of light and crowned the effort with success.

The selection of this mini project work as well as the timely completion is mainly due to the interest and persuasion of our mini project coordinator **Mrs. Reema Lobo, Assistant Professor**, Department of Computer Science & Engineering. We will remember her contribution for ever.

We sincerely thank, **Dr. Manjunath Kotari**, Professor and Head, Department of Computer Science & Engineering who has been the constant driving force behind the completion of the project.

We thank our beloved Principal **Dr. Peter Fernandes**, for his constant help and support throughout.

We are indebted to **Management of Alva's Institute of Engineering and Technology, Mijar, Moodbidri** for providing an environment which helped us in completing our mini project.

Also, we thank all the teaching and non-teaching staff of Department of Computer Science & Engineering for the help rendered.

ADITH P KOTIAN

4AL20CS006

ISHWAR PAVAN

4AL20CS050

HOMIFY APP

ABSTRACT

A Home Maintenance (HOMIFY) mobile app is a software application designed to help homeowners manage and maintain their homes efficiently. The app typically provides a range of features and tools to help users keep track of various aspects of their homes, providing homeowners with a one-stop-shop to manage. It provides a convenient way to accomplish to-do-list tasks, important contacts, track purchases, maintain home inventory, and improve budgeting. Overall this app helps to keep up with all necessary maintenance tasks. Our app will cover all such functionalities and make the entire process as relevant user friendly as possible.

The required data is stored and retrieved using SQLite database. The app is coded in Java using Android Studio.

ADITH P KOTIAN	4AL20CS006
ISHWAR PAVAN	4AL20CS050

TABLE OF CONTENTS

CHAPTER NO.	DESCRIPTION	PAGE NO.
	DECLARATION.....	i
	ACKNOWLEDGEMENT.....	ii
	ABSTRACT.....	iii
1	INTRODUCTION	
1.1	OVER VIEW OF PROJECT.....	3
1.2	IMPORTANCE.....	3
1.3	OBJECTIVE.....	4
2	SYSTEM REQUIREMENT AND SPECIFICATION	
2.1	INTRODUCTION TO ANDROID STUDIO.....	5
2.2	FUNCTIONAL REQUIREMENTS.....	6
2.3	USER REQUIREMENTS.....	6
2.4	SOFTWARE REQUIREMENTS.....	6
2.5	HARDWARE REQUIREMENTS.....	7
3	SYSTEM DESIGN	
3.1	DESIGN OF PROJECT.....	8
3.2	DATABASE DESIGN.....	9
4	IMPLEMENTATION	
4.1	SOLUTION APPROACH / METHODOLOGY.....	10
4.2	STORAGE.....	10
4.3	ANDROID APP.....	11
4.4	JAVA CODES OF APPLICATION.....	13
5	SNAPSHOTS.....	23
6	CONCLUSION AND FUTURE WORKS.....	28
	REFERENCES.....	29

CHAPTER 1 INTRODUCTION

1.1 Overview of the project

Home maintenance app is to provide a comprehensive set of tools and resources to help homeowners manage and maintain their homes. The app can help simplify the process and save time and money efficiently, without any hassle.

The Home Management App project aims to develop a comprehensive mobile application that assists users in efficiently managing various aspects of their home. With the increasing complexity of modern lifestyles, homeowners require tools to simplify and streamline their day-to-day activities. The Home Management App serves as a one-stop solution, providing features and functionalities to organize tasks, monitor expenses, manage schedules, and enhance overall home management.

1.2 Importance

A home maintenance app provides a centralized hub for all your home maintenance needs with improved accuracy. Also provides reminders and real time updates. It can be accessed from anywhere with an internet connection. Helpful for busy homeowners who may forget or overlook certain tasks. The application is implemented on android platform which is linked to the SQLite Database for accessing the data.

This project report presents an overview of the Home Management App, outlining its objectives, scope, and key features. It also discusses the development process, technologies utilized, challenges faced, and future enhancements. The report provides a comprehensive understanding of the project, its significance, and the potential benefits it offers to homeowners.

The Home Management App is a mobile application designed to simplify and streamline various aspects of home management for users. In today's fast-paced and complex lifestyles, homeowners require efficient tools to effectively handle their household tasks, track expenses, manage schedules, and enhance overall home management. This project

report provides an overview of the Home Management App, its objectives, scope, and key features.

1.3 Objective

- Providing reminders: The app helps homeowners stay on top of routine maintenance tasks.
- Facilitating communication: The app should allow homeowners to communicate with service providers, such as plumbers or electricians etc
- Saving money: The app should help homeowners save money by preventing costly repairs and identifying potential issues before they become major problems.
- Budgeting: The app can help users track their home maintenance expenses as well as the day-to-day expenses and create a budget for future repairs and improvements.
- Simplifying home maintenance: The app should make it easy for homeowners to manage and prioritize home maintenance tasks by providing a user-friendly interface.

CHAPTER 2

SYSTEM REQUIREMENT AND SPECIFICATION

2.1 Introduction to Android Studio

Android Studio is a powerful integrated development environment (IDE) specifically designed for Android application development. Developed by Google, Android Studio provides a comprehensive set of tools, libraries, and features that streamline the process of building Android apps. The Electric Eel version, also known as Android Studio 4.3, represents the latest iteration of the IDE, introducing new enhancements and features to support developers in creating cutting-edge Android applications.

In this project report, we will provide an overview of Android Studio Electric Eel Version, highlighting its key features, improvements, and benefits for Android developers. We will explore the capabilities of the IDE, discuss the development workflow, and delve into the various tools and functionalities it offers to accelerate the app development process.

Requirements are during early stages of a system development as a specification of what should be implemented or as a constraint of some kind of on the system. They may be a user level facility description, a detailed specification of expected system behaviour, a general system property, a specific constraint on the system, and information on how to carry out some computation or a constraint on the development of the system. The end product of the requirement analysis phase is a requirement specification. The requirement specification is a reconstruction of the result of this analysis phase. Its purpose is to communicate this result to others. System requirements are more detailed descriptions of the user requirements. They may serve as the basis for a contract to the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point of system design. In principle, the system requirements should state what the system should do and not how it should be implemented. However, at the level of detail required to specify the system completely, it is virtually impossible to exclude all design information.

Natural language is often used to write system requirements specifications. Further problems with natural language can arise when it is used for more detailed specification:

1. Natural language understanding relies on the specification of the readers and writers using the same words for the same concept. This leads to misunderstandings because of the ambiguity of the natural language.
2. A natural language requirements specification is over-flexible. You can say the same thing in completely different ways. It is up to the reader to find out when requirements are same and when they are distinct.

2.2 Functional Requirements

The functional requirements are the statement of services the system should provide, how system reacts to particular inputs and how system should behave in particular situation. It describes the functionality that the system provides.

Our app requires:

- I) Active internet connection.

2.3 User Requirements

User requires active internet connection to use the app.

2.4 Software Requirements

- Operating System: Windows 10
- IDE: Android studio Electric Eel
- Front End: XML
- Back End: Java
- Database: SQLite
- Version Control: Git (GitHub)

2.5 HARDWARE REQUIREMENTS

The hardware requirements for the Home Maintenance App depend on various factors such as the target platform, scalability needs, and anticipated user base. Here are some general hardware requirements to consider for developing and deploying the app:

1. Minimum 4 GB RAM (8GB recommended).
2. 5GB free disk space
3. USB 2.0 or higher
4. Android Device

It's important to note that these hardware requirements are general guidelines, and the actual hardware needs may vary based on the specific project requirements, expected user base, and performance considerations. It is advisable to conduct performance testing and capacity planning to identify the optimal hardware configuration for the Home Maintenance App based on its unique needs.

CHAPTER 3

SYSTEM DESIGN

3.1 DESIGN OF PROJECT

The system design for the Home Maintenance App project involves defining the architecture, components, and interactions necessary to develop a robust and scalable application. The following sections outline the key aspects of the system design.

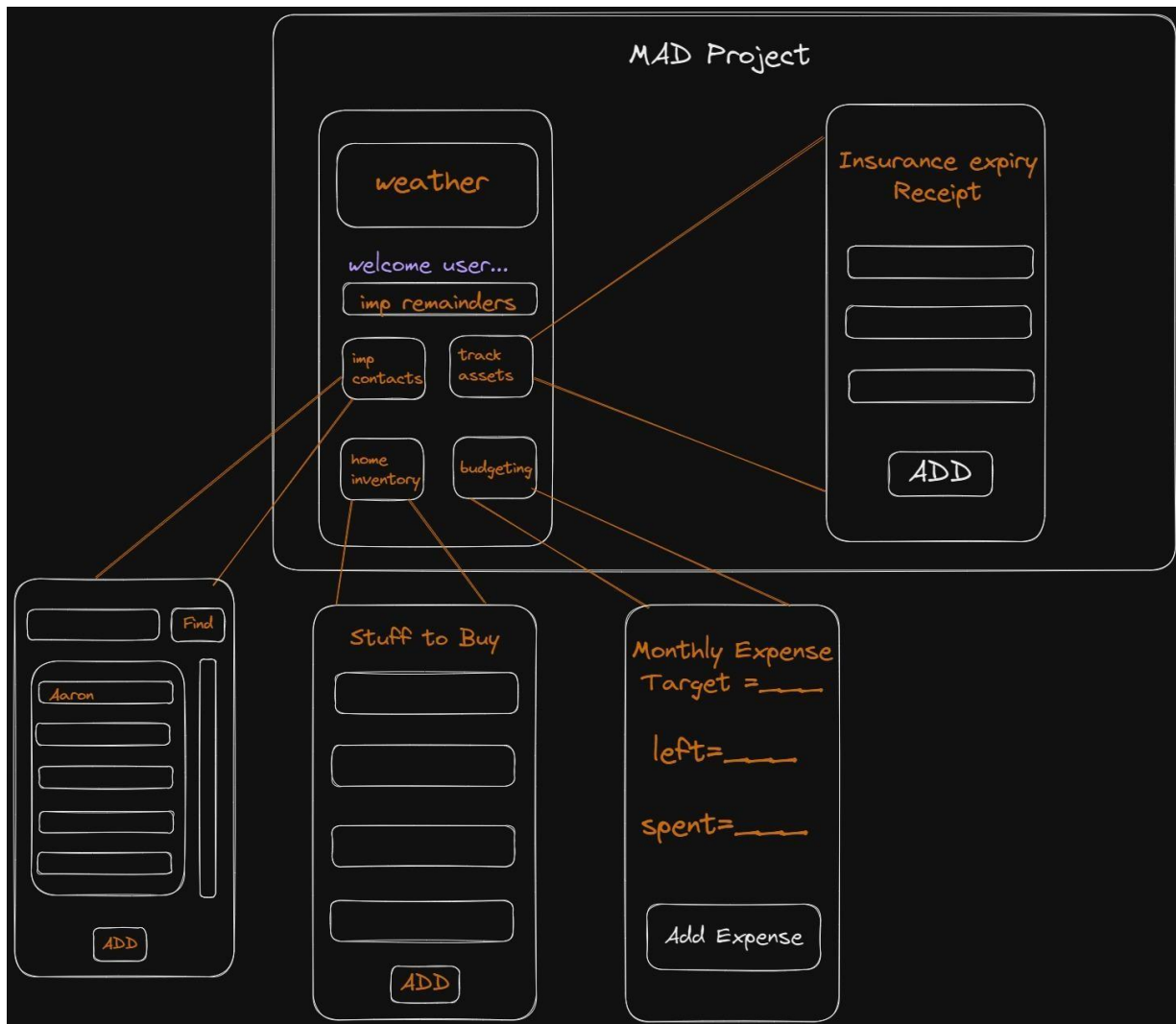


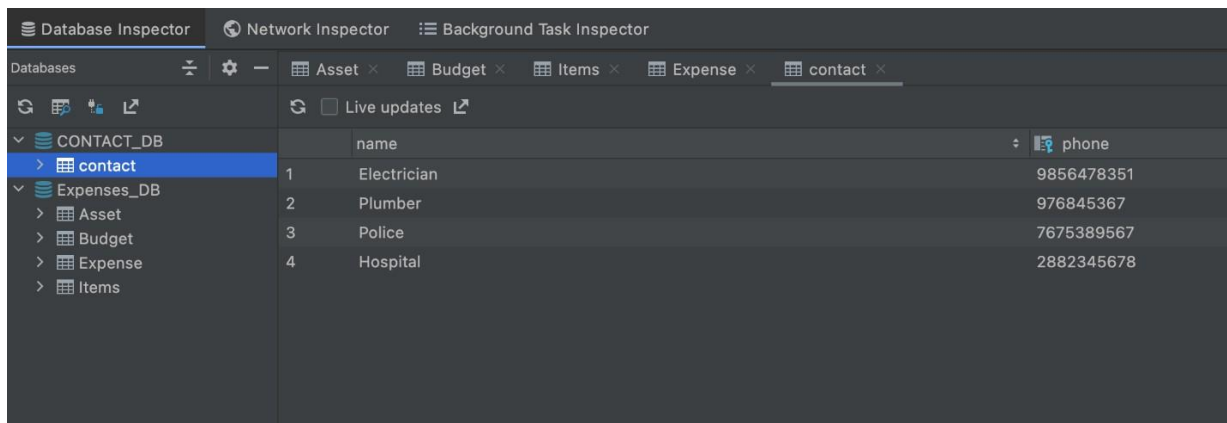
Fig 3.1: System Design

3.2 DATABASE DESIGN

Database Management System: A relational or NoSQL database is used to store and retrieve data related to users, tasks, expenses, schedules, and other relevant entities.

The database is designed using SQLite in which data is stored in form of table(rows and columns). Queries are used to access data from the table.

A snapshot of database:



	name	phone
1	Electrician	9856478351
2	Plumber	976845367
3	Police	7675389567
4	Hospital	2882345678

CHAPTER 4 IMPLEMENTATION

The implementation of the Home Maintenance App involves the development and integration of various components to create a functional and user-friendly application. This section provides an introduction to the implementation process and highlights the key steps involved.

4.1 SOLUTION APPROACH/METHODOLOGY

The methodology for developing the Home Maintenance App involves a systematic approach to ensure efficient project execution and successful delivery.

We are here using xml and java for the front end and firebase for the backend as a server.

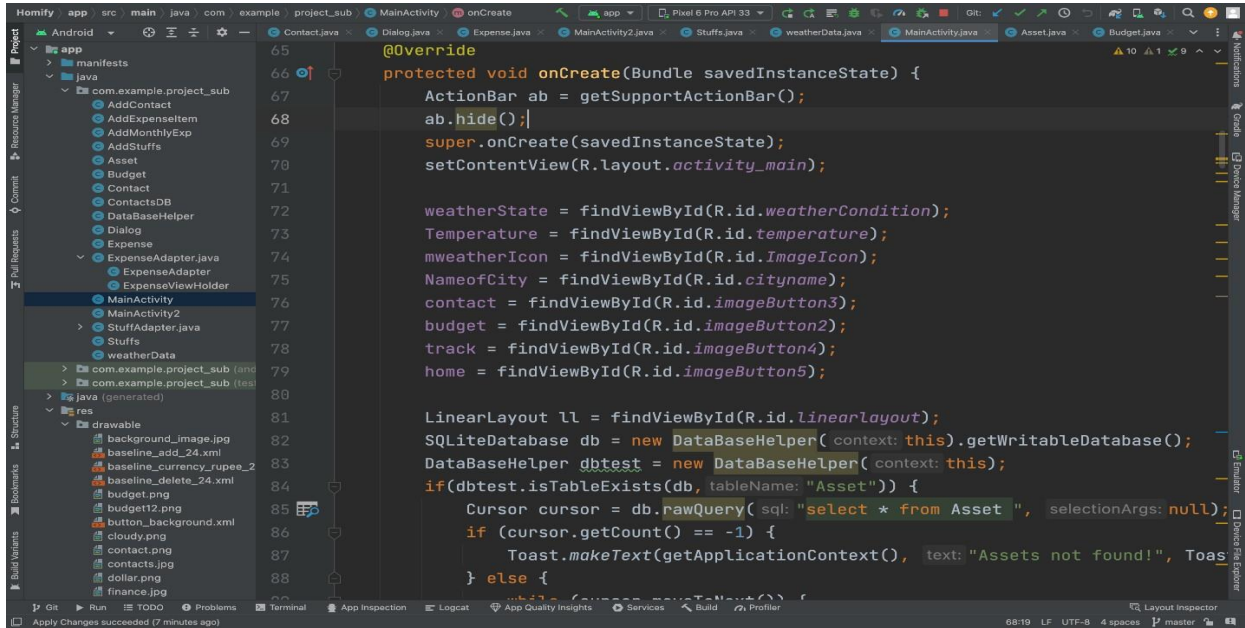
4.2 STORAGE

The Home Maintenance App requires appropriate storage solutions to store and manage various types of data, including user information, task details, expenses, schedules, and communication data. It is essential to follow security best practices, including data encryption, proper access controls, and regular security audits, to protect user data and maintain data privacy.

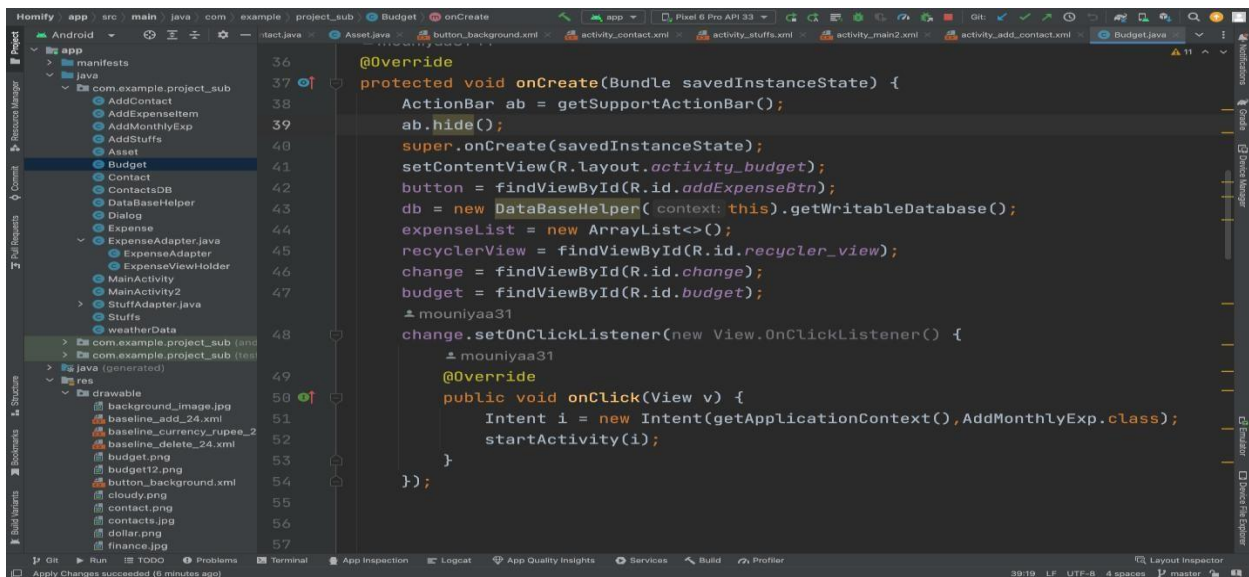
By considering the storage information and implementing appropriate storage solutions, the Home Maintenance App can efficiently handle and manage data related to user accounts, tasks, expenses, schedules, and communication, providing users with a reliable and secure experience. The files like images, audio, video etc can be stored in the app. The data stored is highly secured and is robust in nature means it resumes from the last point if any network error occurs.

4.3 ANDROID APP

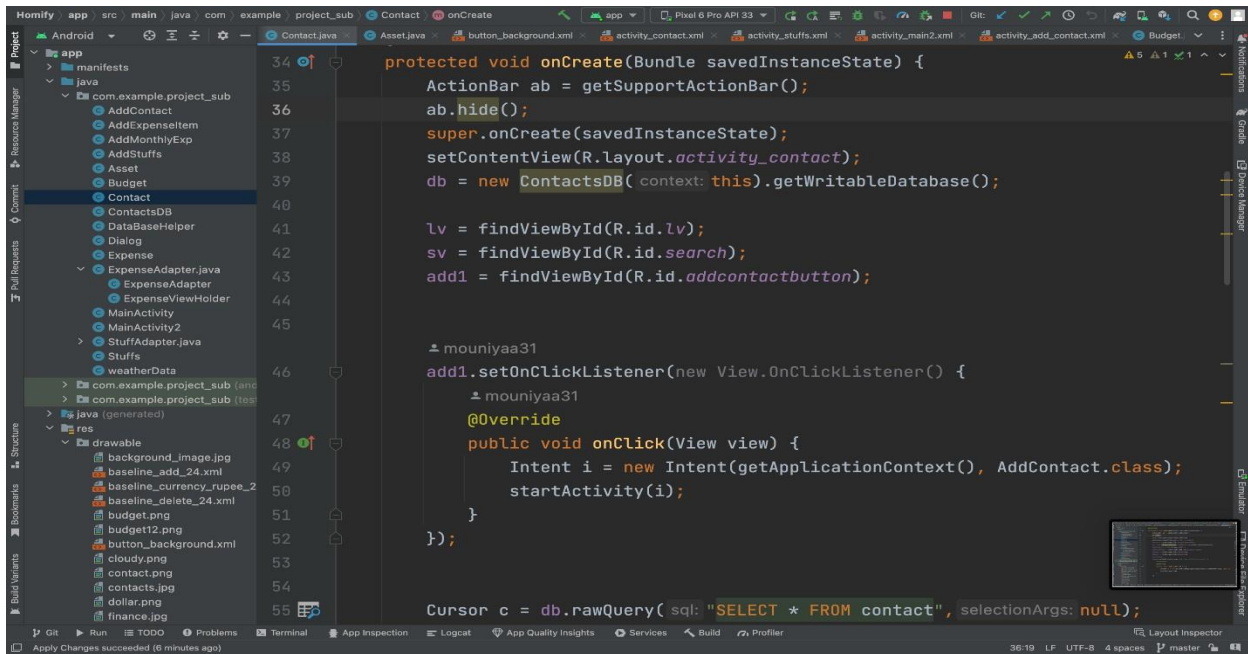
Step 1: A Home Screen is designed and developed, which displays the weather according to the location (using openweathermap.org). Then contains a reminder section. ImageButtons are used for navigating through different functionalities of the app.



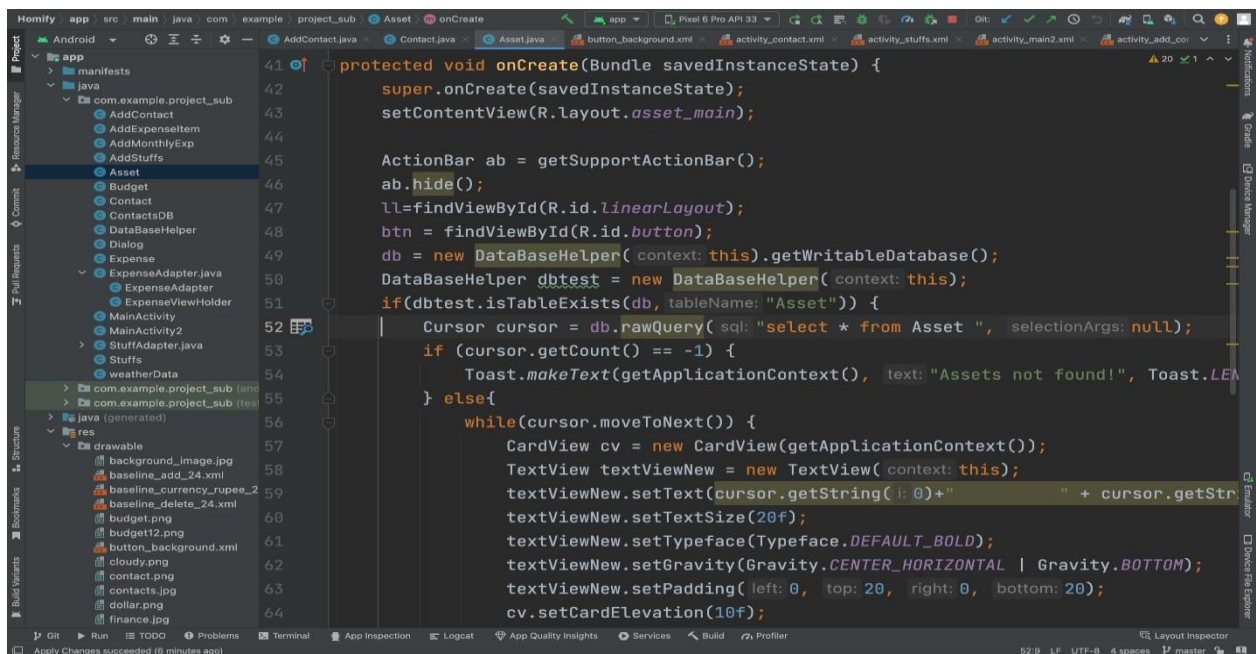
Step 2: Enable users to input their monthly budget and track their spending against that budget. This includes features such as automatic deduction of expenses from the budget, the ability to set a monthly savings goal.



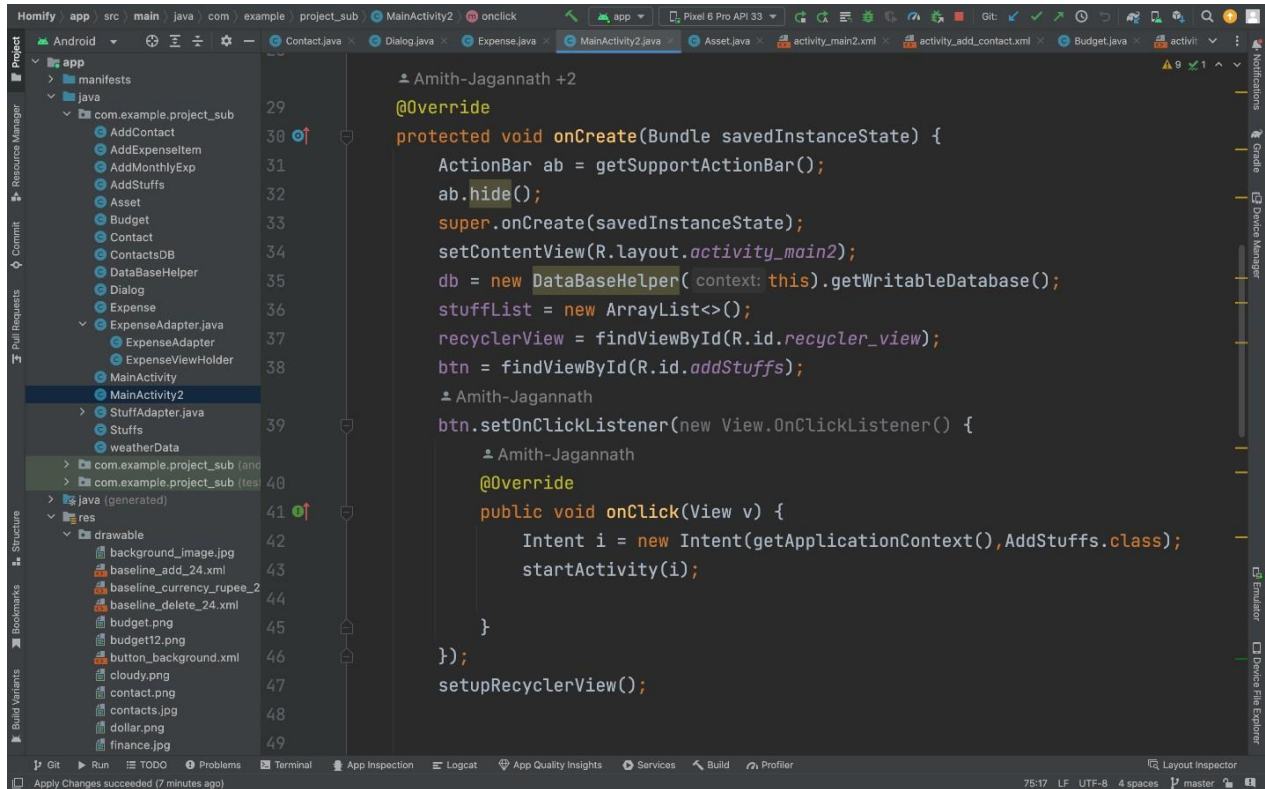
Step 3: Contact Activity contains ListView of contacts including a search bar. New contacts can be add ,by entering the required details which is stored in the database.



Step 4: Tracking Assets contains a scrollView of assets such as your insurance details along with expiration date. It further has the feature to add more assets along with date. It is also connected to home screen to show urgent assets having today's deadline.



Step 5: User can create and manage a shopping list. This feature allows users to input items they need to purchase, and then delete them from the list once they have been bought. By incorporating a shopping list into the application, users can better plan and manage their spending, ensuring they don't forget important items and avoid making unnecessary purchases.



4.4 JAVA CODES OF THE APPLICATION

```

package com.example.project_sub;

import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

```

```
import android.graphics.Color;
import android.graphics.Typeface;
import android.graphics.drawable.ColorDrawable;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;
import androidx.core.app.ActivityCompat;

import com.loopj.android.http.AsyncHttpClient;
import com.loopj.android.http.JsonHttpResponseHandler;
import com.loopj.android.http.RequestParams;

import org.json.JSONObject;

import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Random;
```

```
import cz.msebera.android.httpclient.Header;

public class MainActivity extends AppCompatActivity {

    final String APP_ID = "dab3af44de7d24ae7ff86549334e45bd";
    final String WEATHER_URL = "https://api.openweathermap.org/data/2.5/weather";

    final long MIN_TIME = 5000;
    final float MIN_DISTANCE = 1000;
    final int REQUEST_CODE = 101;
    String Location_Provider = LocationManager.GPS_PROVIDER;

    TextView NameofCity, weatherState, Temperature;
    ImageView mweatherIcon;

    ImageButton contact,track,budget,home;

    LocationManager mLocationManager;
    LocationListener mLocationListner;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        ActionBar ab = getSupportActionBar();
        ab.hide();
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        weatherState = findViewById(R.id.weatherCondition);
        Temperature = findViewById(R.id.temperature);
        mweatherIcon = findViewById(R.id.ImageIcon);
        NameofCity = findViewById(R.id.cityname);
        contact = findViewById(R.id.imageButton3);
```

```
budget = findViewById(R.id.imageButton2);
track = findViewById(R.id.imageButton4);
home = findViewById(R.id.imageButton5);

LinearLayout ll = findViewById(R.id.linearlayout);
SQLiteDatabase db = new DataBaseHelper(this).getWritableDatabase();
DataBaseHelper dbtest = new DataBaseHelper(this);
if(dbtest.isTableExists(db,"Asset")) {
    Cursor cursor = db.rawQuery("select * from Asset ", null);
    if (cursor.getCount() == -1) {
        Toast.makeText(getApplicationContext(), "Assets not found!",
Toast.LENGTH_SHORT).show();
    } else {
        while (cursor.moveToNext()) {
            Calendar calendar = Calendar.getInstance();
            SimpleDateFormat dateFormat = new SimpleDateFormat("d/M/yyyy");
            String today = dateFormat.format(calendar.getTime());
            if(today.equals(cursor.getString(1))) {
                CardView cv = new CardView(getApplicationContext());
                TextView textViewNew = new TextView(this);
                textViewNew.setText(cursor.getString(0) + "      " + cursor.getString(1));
                textViewNew.setTextSize(20f);
                textViewNew.setTypeface(Typeface.DEFAULT_BOLD);
                textViewNew.setGravity(Gravity.CENTER_HORIZONTAL |
Gravity.BOTTOM);
                textViewNew.setPadding(0, 20, 0, 20);
                cv.setCardElevation(10f);
                cv.setUseCompatPadding(false);
                cv.setCardBackgroundColor(Color.WHITE);
                cv.setMaxCardElevation(12f);
                cv.setPreventCornerOverlap(true);
```

```
        cv.setBackground(new ColorDrawable(Color.rgb(255, 255, 255)));
        LinearLayout.LayoutParams layoutParams = new LinearLayout.LayoutParams(
            LinearLayout.LayoutParams.MATCH_PARENT,
            LinearLayout.LayoutParams.WRAP_CONTENT
        );
        layoutParams.setMargins(0, 16, 0, 16);
        cv.setLayoutParams(layoutParams);
        cv.addView(textViewNew);
        ll.addView(cv);
    }
}
}
```

```
contact.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent i = new Intent(getApplicationContext(), Contact.class);
        startActivity(i);
    }
});
```

```
budget.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent i = new Intent(getApplicationContext(), Budget.class);
        startActivity(i);
    }
});
```

```
home.setOnClickListener(new View.OnClickListener() {
```

```
@Override
public void onClick(View view) {
    Intent i = new Intent(getApplicationContext(), Asset.class);
    startActivity(i);
}
});

track.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent i = new Intent(getApplicationContext(), MainActivity2.class);
        startActivity(i);
    }
});
}
```

```
/* @Override
protected void onResume() {
    super.onResume();
    getWeatherForCurrentLocation();
}*/
```

```
@Override
protected void onResume() {
    super.onResume();
    Intent mIntent=getIntent();
    String city= mIntent.getStringExtra("City");
    if(city!=null)
    {
        getWeatherForNewCity(city);
    }
}
```

```
else
{
    getWeatherForCurrentLocation();
}
}

private void getWeatherForNewCity(String city)
{
    RequestParams params=new RequestParams();
    params.put("q",city);
    params.put("appid",APP_ID);
    letsdoSomeNetworking(params);

}

private void getWeatherForCurrentLocation() {

    mLocationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
    mLocationListner = new LocationListener() {
        @Override
        public void onLocationChanged(Location location) {

            String Latitude = String.valueOf(location.getLatitude());
            String Longitude = String.valueOf(location.getLongitude());

            RequestParams params =new RequestParams();
            params.put("lat" ,Latitude);
            params.put("lon",Longitude);
            params.put("appid",APP_ID);
            letsdoSomeNetworking(params);
        }
    }
}
```

```
@Override
public void onStatusChanged(String provider, int status, Bundle extras) {
}

@Override
public void onProviderEnabled(String provider) {
}

@Override
public void onProviderDisabled(String provider) {
    //not able to get location
}

};

if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
    // TODO: Consider calling
    //    ActivityCompat#requestPermissions
    // here to request the missing permissions, and then overriding
    //    public void onRequestPermissionsResult(int requestCode, String[] permissions,
    //                                           int[] grantResults)
    // to handle the case where the user grants the permission. See the documentation
    // for ActivityCompat#requestPermissions for more details.
    ActivityCompat.requestPermissions(this,new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},REQUEST_CODE);
    return;
}

mLocationManager.requestLocationUpdates(Location_Provider, MIN_TIME,
MIN_DISTANCE, mLocationListner);
}
```



```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);

    if(requestCode==REQUEST_CODE)
    {
        if(grantResults.length>0 &&
grantResults[0]==PackageManager.PERMISSION_GRANTED)
        {
            Toast.makeText(MainActivity.this,"Locationget
Succesffully",Toast.LENGTH_SHORT).show();
            getWeatherForCurrentLocation();
        }
        else
        {
            //user denied the permission
        }
    }
}

private void letsdoSomeNetworking(RequestParams params)
{
    AsyncHttpClient client = new AsyncHttpClient();
    client.get(WEATHER_URL,params,new JsonHttpResponseHandler()
    {
        @Override
        public void onSuccess(int statusCode, Header[] headers, JSONObject response) {

            Toast.makeText(MainActivity.this,"Data Get
Success",Toast.LENGTH_SHORT).show();
```

```
        weatherData weatherD=weatherData.fromJson(response);
        updateUI(weatherD);
        // super.onSuccess(statusCode, headers, response);
    }
    @Override
    public void onFailure(int statusCode, Header[] headers, Throwable throwable,
JSONObject errorResponse) {
        //super.onFailure(statusCode, headers, throwable, errorResponse);
    }
});
}

private void updateUI(weatherData weather){
    Temperature.setText(weather.getMTemperature());
    NameofCity.setText(weather.getMcity());
    System.out.println(weather.getMcity());
    weatherState.setText(weather.getMWeatherType());
    int
resourceID=getResources().getIdentifier(weather.getMicon(),"drawable",getPackageName());
    mweatherIcon.setImageResource(resourceID);
}
@Override
protected void onPause() {
    super.onPause();
    if(mLocationManager!=null)
    {
        mLocationManager.removeUpdates(mLocationListner);
    }
}
```

CHAPTER 5

SNAPSHOTS

Home Screen:

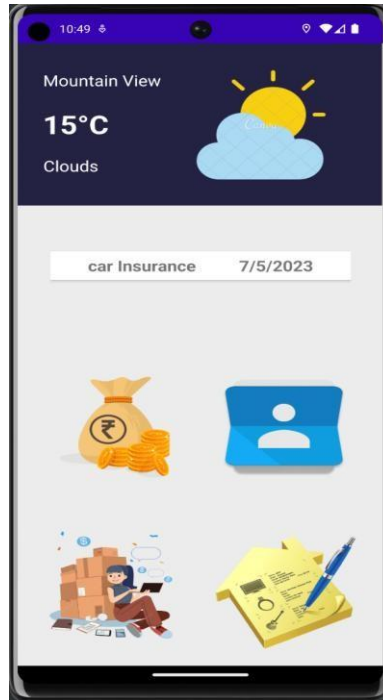


Fig: Home screen of HOMIFY app

Home Maintenance App is designed to make your life easier by providing you with a comprehensive solution for managing and maintaining your home. Whether you're a homeowner, a tenant, or a property manager, this app has all the tools you need to stay on top of your home maintenance tasks. Easily create and manage tasks for various home maintenance activities such as Budgeting, Contacts, Buy Assets, Inventory. Receive timely notifications and alerts about upcoming tasks, maintenance reminders, service provider appointments, and community updates.

Budget:

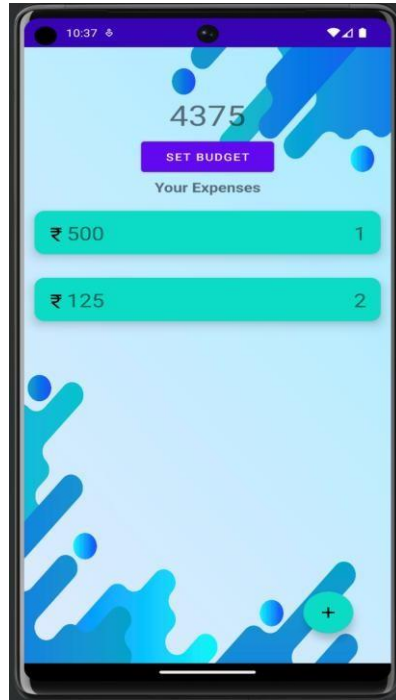


Fig: Home screen of HOMIFY app

Home Maintenance App offers robust budgeting and expense tracking features to help you manage your home maintenance expenses effectively. Set a budget for your home maintenance expenses and track your spending. Get insights into your expenditure patterns and plan your finances accordingly. Set a budget for your home maintenance expenses based on your financial goals and requirements. Define spending limits for different categories such as repairs, landscaping, cleaning, and more.

Effective budgeting and expense tracking are essential for maintaining financial stability and managing your home maintenance costs. Our app provides the tools and insights you need to stay in control of your expenses and ensure your home receives the care it deserves.

Contacts:

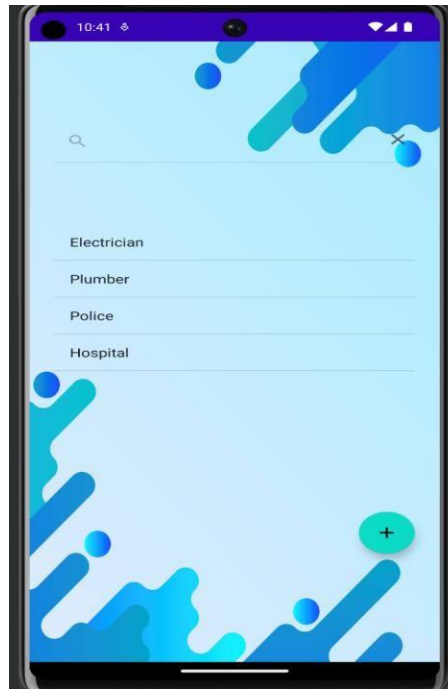


Fig: Contacts screen of HOMIFY app

In the Contacts page of our Home Maintenance App, you'll find all the necessary details and information to help you connect with service providers and professionals for your home maintenance needs. Find contact details for local authorities, homeowner associations, or any other relevant organizations related to home maintenance. This information can be valuable when dealing with specific issues or seeking guidance on regulations and guidelines.

Emergency Contacts Access a list of emergency contacts such as plumbers, electricians, and other professionals who offer 24/7 services. These contacts can be essential during urgent situations requiring immediate attention or repairs.

Assets:



Fig: Assets screen of HOMIFY app

In the Assets page of our Home Maintenance App, you can keep track of all your home possessions and assets in a convenient and organized manner. Capture important details for each asset, such as the make, model, serial number, purchase date, warranty information, and any relevant notes. This information helps you stay informed about your assets and plan for maintenance or repairs. Categorize and tag your assets based on rooms, types, or any custom categories you define. This helps in organizing and filtering your assets for easier access and management.

The Assets page provides a centralized and organized way to manage your home inventory, enabling you to keep track of your belongings, their maintenance history, warranties, and important documentation. By having all this information readily accessible, you can make informed decisions about repairs, replacements, and overall asset management within your home.

Stuffs to buy:

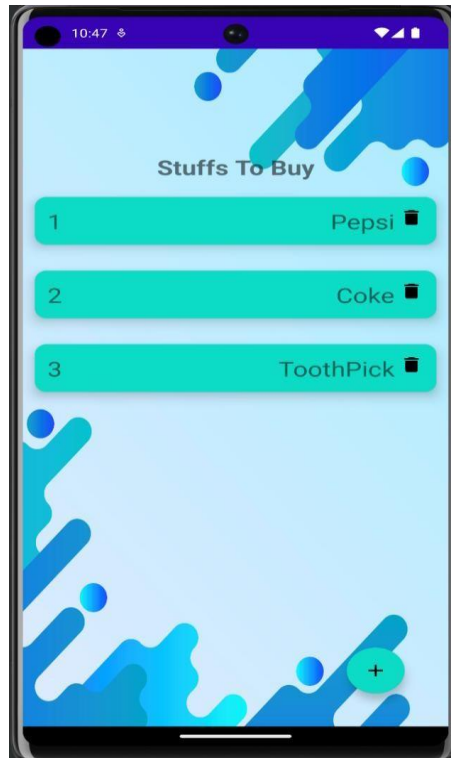


Fig: Contacts screen of HOMIFY app

In our Home Maintenance App, the "Stuffs to Buy" feature allows you to create and manage a shopping list for all the items you need to purchase for your home maintenance tasks. Easily create multiple shopping lists based on different home maintenance projects or areas of your home. For example, you can have separate lists for kitchen supplies, cleaning products, gardening tools, or repair materials. With the "Stuffs to Buy" feature, you can efficiently manage your shopping lists for home maintenance supplies and materials. This helps you remember specific details or requirements when purchasing the items. Arrange the items in your shopping lists based on priority or the order in which you plan to buy them. This allows you to streamline your shopping experience and ensure you don't forget any crucial items. It helps you stay organized, ensures you have all the necessary items for your tasks, and simplifies the overall shopping process, saving you time and effort. Specify the quantity needed for each item and add any additional notes or specifications.

CHAPTER 6 CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

In conclusion, the home maintenance app that we have designed and developed a powerful tool for homeowners who want to keep their homes in good condition. By including features such as maintenance reminders, budget tracking, emergency contacts and weather forecasting app provides users with everything they need to take care of their homes.

Our app is not only user-friendly but also provides a seamless experience that allows users to easily manage their homes from their smartphones. Overall, we believe that our home maintenance app has the potential to revolutionize the way that homeowners approach home maintenance. By providing users with a comprehensive set of tools and resources, our app empowers them to take control of their homes and ensure that they are well-maintained and safe for themselves and their families

6.2 FUTURE WORKS

In the future, we may extend this project by adding extra features to our android app like,

- **Voice assistant integration:** By integrating with popular voice assistants such as Amazon Alexa or Google Assistant, users could access the app's features hands-free.
- **Offering guidance and advice:** The app should provide homeowners with helpful tips and information on home maintenance best practices, DIY projects, and other related topics.
- **Integration with smart home devices:** As more and more households adopt smart home devices such as smart thermostats, security cameras, and lighting systems, integrating your home maintenance app with these devices could provide users with even greater control over their homes.

REFERENCES

- <https://firebase.google.com/docs/>
- Stack Overflow, <https://stackoverflow.com>
- YouTube, <https://www.youtube.com>
- [Geeksforgeeks ,www.geeksforgeeks.com](https://www.geeksforgeeks.com)