

ABSTRACT

The work deals with the development of android-based alarm triggered medicine database remainder, an integrated application for alarm based medicine remainder based upon user given input. This application will trigger the alarm when the input time matches the current time and it will also show the medicine name. Where the medicine database will be stored. User will give the input like medicine name, time medicine to be taken, and date.

CONTENTS

SI No.	Title	Pg. No.
	Acknowledgment	i
	Abstract	ii
1.	Introduction.....	01
1.1	Mobile Application Development	01
1.2	Overview of Project.....	01
2.	Software Requirement Specification.....	02
2.1	Functional Requirement	02
2.2	Hardware Requirement.....	02
2.3	Software Requirement	02
2.4	Android.....	03
3.	System Design.....	04
3.1	Android Architecture.....	04
3.2	Flow Chart of Medicine Remainder	05
3.3	Software Development Kit.....	08
4.	Implementation	10
4.1	Description of Implementation Modules.....	10
4.2	Code	10
5.	Results	21
6.	Conclusion and Future Enhancement.....	25
6.1	Conclusion	25
6.2	Future Enhancement	25
	References	26

LIST OF FIGURES

Figure No.	Title	Pg. No.
Figure 3.1	Android Architecture.....	04
Figure 3.2	Flow Chart of Medicine Remainder	05
Figure 5.1	Front Screen.....	21
Figure 5.2	Inserting Data	22
Figure 5.3	Fetching Data.....	23
Figure 5.4	Fetches Data.....	24

CHAPTER 1

INTRODUCTION

1.1 Mobile Application Development

Mobile application development is the process to making software for smartphones and digital assistants, most commonly for Android and iOS. The software can be preinstalled on the device, downloaded from a mobile app store or accessed through a mobile web browser. The programming and markup languages used for this kind of software development include Java, Swift, C# and HTML5.

Mobile app development is rapidly growing. From retail, telecommunications and e-commerce to insurance, healthcare and government, organizations across industries must meet user expectations for real-time, convenient ways to conduct transactions and access information. Today, mobile devices—and the mobile applications that unlock their value—are the most popular way for people and businesses to connect to the internet. To stay relevant, responsive and successful, organizations need to develop the mobile applications that their customers, partners and employees demand.

1.1 Overview of Project

This work deals with the development of android-based alarm triggered medicine database remainder, an integrated application for alarm based medicine remainder based upon user given input. This application will trigger the alarm when the input time matches the current time and it will also show the medicine name. Where the medicine database will be stored, user will give the input like medicine name, time medicine to be taken, and date.

CHAPTER 2

SOFTWARE REQUIREMENT SPECIFICATION

2.1 Functional Requirement

Functional Requirements describe how product must behave what its features and function. Generally, functional requirement describes system behaviour under specific condition.

Our project has the following modules:

- **Medicine Remainder Module:** Where the medicine database will store. user will give the input like medicine name, time medicine to be taken, and date.
- **Medicine Remainder User Interface Module :** A markup language created as a standard way to encode data in internet-based applications. Android applications use XML to create layout files.

2.2 Hardware Requirement

- PROCESSOR: Intel@core™2Duo
- SPEED: 2.10GHz
- RAM: 8.00GB Minimum
- SPACE ON DISC: 20GB Minimum

2.3 Software Requirement

- Operating System: windows 10
- Languages: XML and Java
- Screen: 1280 x 800 minimum screen resolution
- Software: IDE, Android SDK and Android Emulator

2.4 Android

Android is an open source and Linux-based **Operating System** for mobile devices such as smartphones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008.

CHAPTER 3

SYSTEM DESIGN

Software design is the process by which an agent creates specification of software artifact, intended to accomplish goals, using the set of primitive components and subject to constraints.

3.1 Android Architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

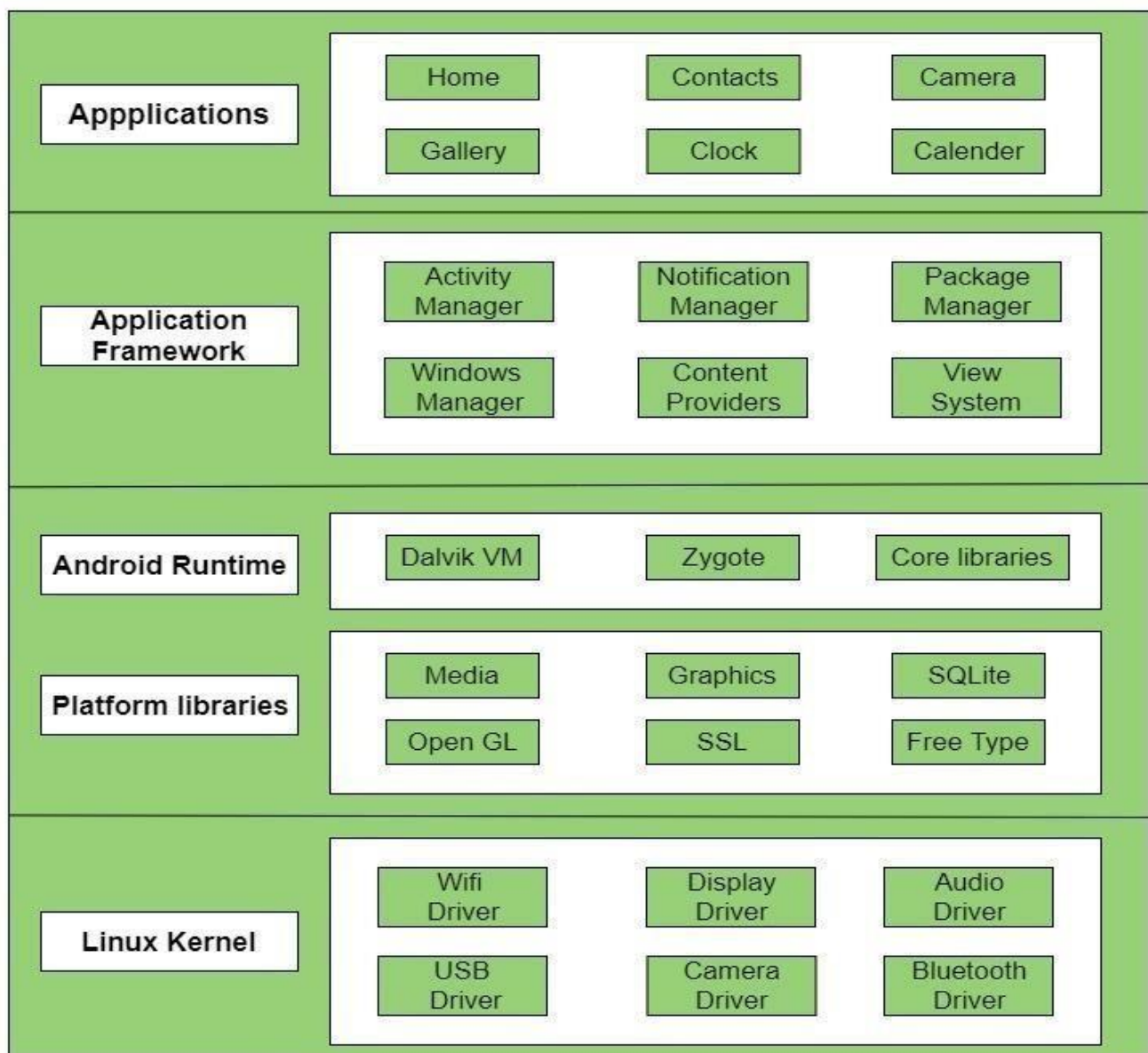


Fig 3.1: Android Architecture

3.2 Flow Chart

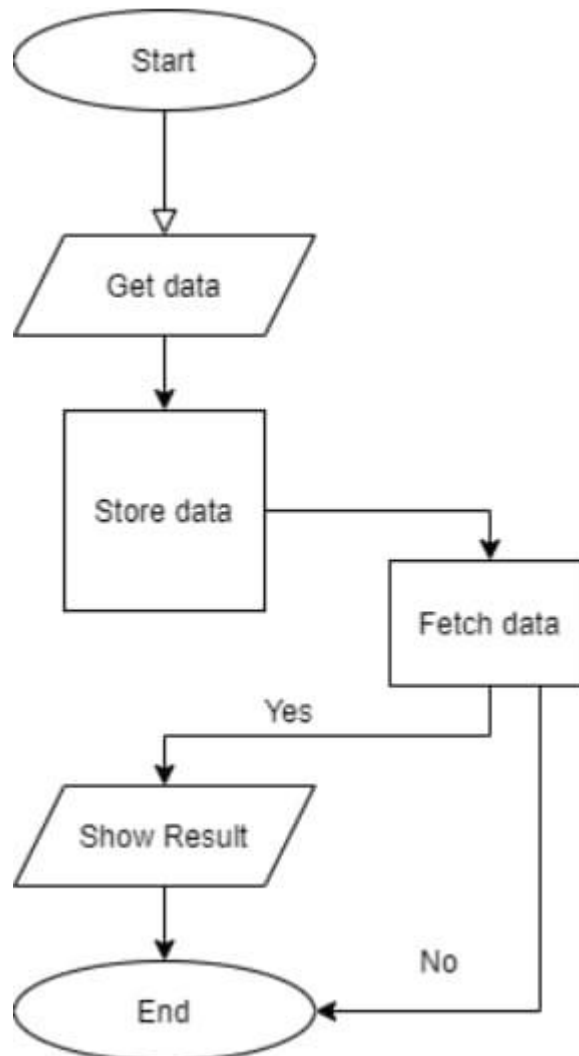


Fig 3.2: Flow Chart of Medicine Remainder

Linux kernel

At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches. This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is

really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.

Libraries

On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

Android Libraries

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows –

- **android.app** – Provides access to the application model and is the cornerstone of all Android applications.
- **android.content** – Facilitates content access, publishing and messaging between applications and application components.
- **android.database** – Used to access data published by content providers and includes SQLite database management classes.
- **android.opengl** – A Java interface to the OpenGL ES 3D graphics rendering API.
- **android.os** – Provides applications with access to standard operating system services including messages, system services and inter-process communication.
- **android.text** – Used to render and manipulate text on a device display.
- **android.view** – The fundamental building blocks of application user interface.

Android Runtime

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android.

The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine.

The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

The Android framework includes the following key services –

- **Activity Manager** – Controls all aspects of the application lifecycle and activity stack.
- **Content Providers** – Allows applications to publish and share data with other applications.
- **Resource Manager** – Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
- **Notifications Manager** – Allows applications to display alerts and notifications to the user.
- **View System** – An extensible set of views used to create application user interfaces.

Applications

You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games etc.

3.3 Software Development Kit

A software development kit (SDK) is a set of tools provided by the manufacturer of (usually) a hardware platform, operating system (OS), or programming language. SDKs help software developers create applications for that specific platform, system, or programming language. Think of it kind of like a toolkit, or the plastic bag of tools that comes packaged with the parts of a dresser you've bought to assemble yourself—only for app development. You have the building blocks—or development tools—you need to get the job done, and what's included in the kit varies from manufacturer to manufacturer.

Typically, a basic SDK will include a compiler, debugger, and application programming interfaces (APIs), but they can also include any of the following:

- Documentation
- Libraries
- Editors
- Runtime/development environments
- Testing/analysis tools
- Drivers
- Network protocols

Android Studio

Android Studio is the official Integrated Development Environment (IDE) for android application development. Android Studio provides more features that enhance our productivity while building Android apps.

Android Studio was announced on 16th May 2013 at the Google I/O conference as an official IDE for Android app development. It started its early access preview from version 0.1 in May 2013. The first stable built version was released in December 2014, starts from version 1.0.

Since 7th May 2019, Kotlin is Google's preferred language for Android application development. Besides this, other programming languages are supported by Android Studio.

Features of Android Studio

- It has a flexible Gradle based build system.
- It has a fast and feature rich emulator for app testing.
- Android Studio has a consolidated environment where we can develop for all Android devices.
- Apply changes to the resource code of our running app without restarting the app.
- Android Studio provides extensive testing tools and frameworks.
- It supports C++ and JDK.
- It provides built-in supports for Google Cloud Platform. It makes it easy to integrate Google Cloud Messaging and App Engine.

CHAPTER 4

IMPLEMENTATION

4.1 Description of Implementation Modules

Input: Medicine name, date, time Medicine to be taken.

Output: Alarm will trigger and it will show the name of the medicine.

According to the input given by the user, alarm will trigger along with the medicine name
Displays on screen.

4.2 Code

1.SQLiteOpenHelper:Public abstract class SQLiteOpenHelper extends Object implements AutoCloseable java.lang.Object is android.database.sqlite.SQLiteOpenHelper A helper class to manage database creation and version management. Creating a subclass implementing onCreate(SQLiteDatabase), onUpgrade(SQLiteDatabase, int, int) and optionally onOpen(SQLiteDatabase), and this class takes care of opening the database if it exists, creating it if it does not, and upgrading it as necessary. This class makes it easy for ContentProvider implementations to defer opening and upgrading the database until first use.

2.Alarm Manager: public class Alarm Manager extends Object java.lang.Object android.app.AlarmManager.

MainActivity.java

```
package com.example.pro10;
import androidx.appcompat.app.AppCompatActivity;

import android.app.AlarmManager;import android.app.AlertDialog;

import android.app.PendingIntent;

import android.content.Context;

import android.content.Intent;

import android.database.Cursor;

import android.os.Bundle;import android.view.View;
```

```
import android.widget.Button; import
android.widget.EditText; import android.widget.Toast;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {
    private Button insert,set_alarm;

    private EditText time,date,name,input1,input2;
    private Context context;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        DatabaseHandler db = new DatabaseHandler(this);
        setupUIViews();

        insert.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                boolean res = db.Insert_Data(name.getText().toString(), date.getText().toString(),
                time.getText().toString());

                if (res == true)

                Toast.makeText(MainActivity.this, "Inserted", Toast.LENGTH_SHORT).show(); else

                Toast.makeText(MainActivity.this, " Not Inserted",
                Toast.LENGTH_SHORT).show();

            }
        });
    }
}
```

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

    @override

    public void onClick(View v) {

        int time = 5;

        Intent i = new Intent(MainActivity.this, Alarm.class);

        PendingIntent pi = PendingIntent.getBroadcast(getApplicationContext(), 0, i, 0);

        AlarmManager am = (AlarmManager) getSystemService(ALARM_SERVICE);

        am.set(AlarmManager.RTC_WAKEUP, System.currentTimeMillis() + time * 1000, pi);

        Toast.makeText(MainActivity.this, "Alarm Set", Toast.LENGTH_SHORT).show();

    }

});

Intent intent2 = getIntent();

int TYPE = intent2.getIntExtra("type",4);

if(TYPE == 4)

{

    view_data();

}

}

public void view_data()

{

    DatabaseHandler db = new DatabaseHandler(this);

    Calendar calander = Calendar.getInstance(); Cursor

    res = db.getData("2021/03/4","12:00");
```

```
if(res.getCount()==0)
{
    ShowMessage("Error","Nothing Found");
}
else
{
    StringBuffer buffer=new StringBuffer();
    while(res.moveToNext()) { buffer.append("Name:" +
    res.getString(0) + "\n");
    }
    ShowMessage("Data",buffer.toString());
}
}

private void setupUIViews()
{
    insert = (Button)findViewById(R.id.b1);
    set_alarm = (Button)findViewById(R.id.b2);
    name = (EditText) findViewById(R.id.tv1);
    date= (EditText) findViewById(R.id.tv2); time
    = (EditText) findViewById(R.id.tv3);
}

public void ShowMessage(String title,String Message)
{
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setCancelable(true); builder.setTitle((title));

    builder.setTitle((title)); builder.setMessage(Message);

    builder.show();}
```



```
public void setAlarm() {  
    }  
}  
  
DatabaseHandler. java  
  
package com.example.pro10;  
  
import android.content.ContentValues;  
import android.content.Context; import  
android.database.Cursor;  
  
import android.database.sqlite.SQLiteDatabase; import  
android.database.sqlite.SQLiteOpenHelper; import  
androidx.annotation.Nullable;  
  
public class DatabaseHandler extends SQLiteOpenHelper {  
    private static final String DATABASE_NAME = "MEDICINE_DATABASE1.db";  
    private static final String TABLE_NAME = "MEDICINE_TABLE";  
  
    private static final String COL1_NAME = "MEDICINE_NAME";  
    private static final String COL2_NAME = "DATE";  
  
    private static final String COL3_NAME = "TIME_OF_DAY";  
    public DatabaseHandler(@Nullable Context context) {  
        super(context, DATABASE_NAME, null, 1);  
    }  
    @Override  
    public void onCreate(SQLiteDatabase db) {  
        db.execSQL("create table " + TABLE_NAME + "( " + COL1_NAME + " TEXT ," +  
COL2_NAME + " TEXT ," + COL3_NAME + " TEXT );");  
    }  
    @Override
```

```
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
    // Drop older table if existed  
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);  
    onCreate(db);  
}  
  
public boolean Insert_Data(String Medi_name, String Medi_date, String Medi_time) {  
    SQLiteDatabase db = this.getWritableDatabase();  
  
    ContentValues values = new ContentValues();  
    values.put(COL1_NAME, Medi_name); // Contact Name  
    values.put(COL2_NAME, Medi_date); // Contact Phone  
    values.put(COL3_NAME, Medi_time);  
  
    // Inserting Row  
    Long res = db.insert(TABLE_NAME, null, values); if  
    (res == -1)  
  
    return false;  
    else  
  
    return true;  
}  
  
public Cursor getData(String input1, String input2)  
{  
    SQLiteDatabase db = this.getWritableDatabase();  
  
    String selectQuery = "select "+COL1_NAME+" FROM "+TABLE_NAME+" where  
    "+COL2_NAME+" = ? and "+COL3_NAME+" = ?";  
  
    String args[] = {input1,input2};
```

```
Cursor res = db.rawQuery(selectQuery, args);  
return res;}}}
```

Alarm. java

```
import android.content.Context;  
package com.example.pro10
```

```
import android.content.BroadcastReceiver;  
import android.content.Intent;  
  
import android.widget.Toast;  
  
public class Alarm extends BroadcastReceiver {  
    @Override  
  
    public void onReceive(Context context, Intent intent) {  
        Toast.makeText(context, "alarm. .... ", Toast.LENGTH_SHORT).show();  
  
        Intent intent1 = new Intent(context, MainActivity.class);  
        intent1.putExtra("type", 4);  
        intent1.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);  
        context.startActivity(intent1);  
  
    }  
}
```

activity_Main.xml

```
<?xml version="1.0" encoding="utf-8"?>  
  
<LinearLayout 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="wrap_content" android:layout_height="wrap_content"
```

```
android:background="@color/cardview_light_background"
android:orientation="vertical"

tools:context=".MainActivity">
<TextView
    android:id="@+id/t"

    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="MEDICINE DATABASE"
    android:textSize="19dp"

    android:layout_gravity="center"
/>

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/t1"

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="MEDICINE_NAME:" />

    <EditText
        android:id="@+id/tv1"

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
```

```
android:ems="10"
android:inputType="text" />

</LinearLayout>

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/t2"

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"

        android:text="DATE:(yyyy-mm-dd)" />

    <EditText
        android:id="@+id/tv2"

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="date" />

</LinearLayout>

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/t3"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="TIME(h:m)" />

<EditText
android:id="@+id/tv3"

android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:ems="10"
android:inputType="time" />

</LinearLayout>

<view

android:layout_width="match_parent"

android:layout_height="4dp"
android:background="@android:color/black" />

<Button
android:id="@+id/b1"

android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="INSERT"
android:layout_gravity="center"

/>

<Button
android:id="@+id/b2"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="setalarm"
android:layout_gravity="center"

/>
<View
android:layout_width="match_parent"
android:layout_height="4dp"

android:background="@android:color/black" />
</LinearLayout>
```

CHAPTER 5

RESULTS

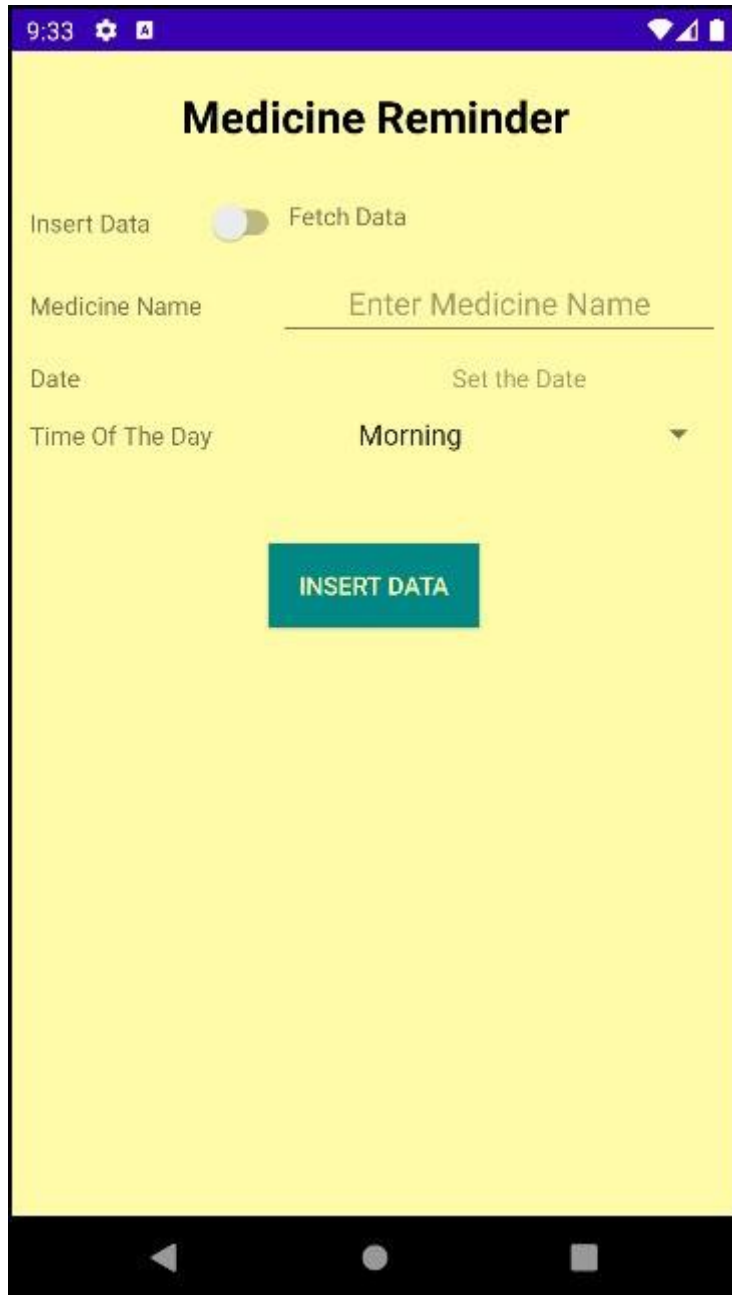


Fig 5.1: Front Screen

The screenshot shows a mobile application interface titled "Medicine Reminder". At the top, there is a status bar with the time 9:36 and icons for settings, a notification, and connectivity. Below the title, there are two toggle switches: "Insert Data" (which is currently turned on) and "Fetch Data". Below these toggles, there are three input fields: "Medicine Name" with the text "Paracetamol", "Date" with the value "6/18/2021", and "Time Of The Day" with a dropdown menu showing "Evening". A large green button labeled "INSERT DATA" is positioned below the input fields. The entire form is set against a yellow background, and the bottom of the screen shows the standard Android navigation bar.

Fig 5.2: Inserting Data

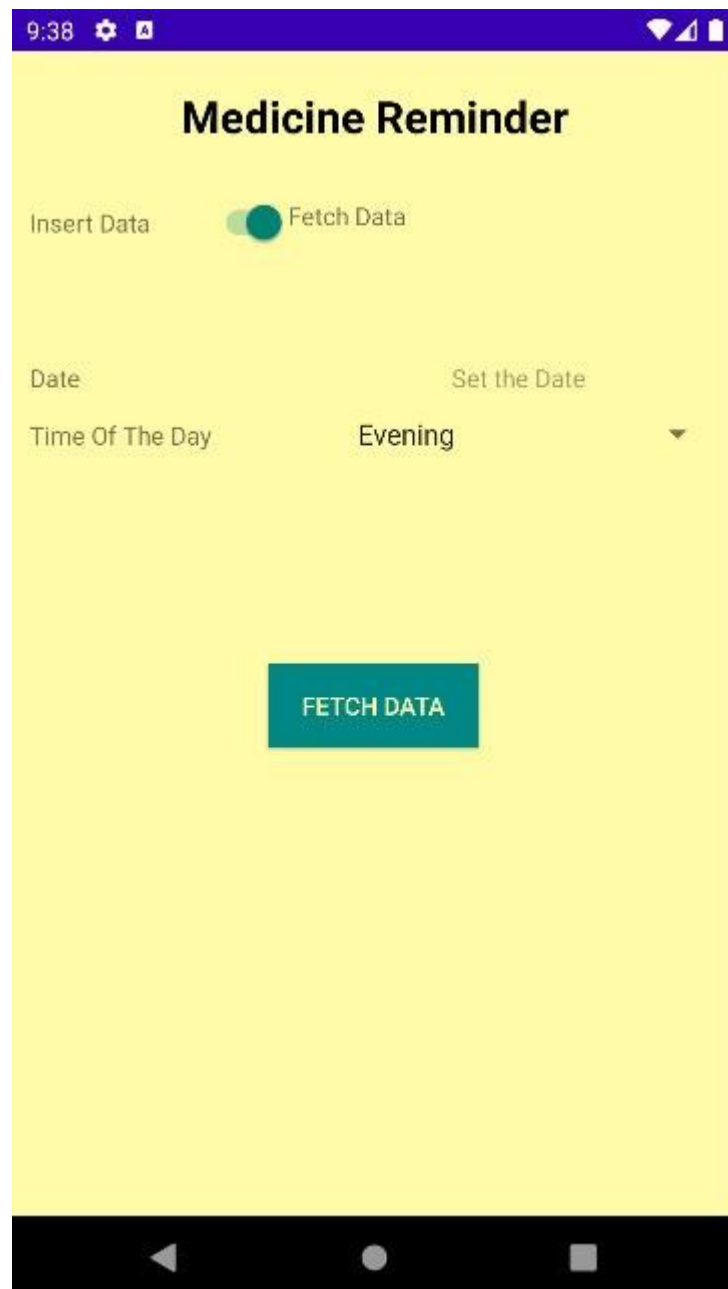


Fig 5.3: Fetching Data

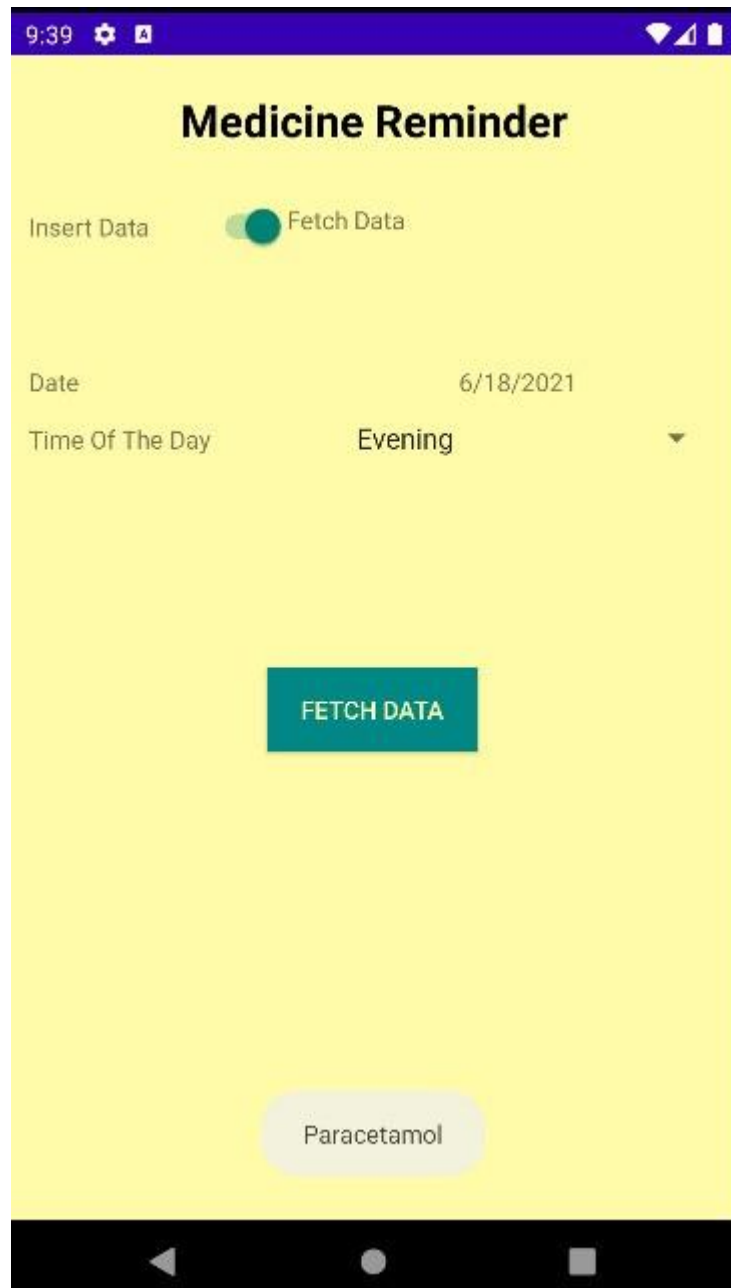


Fig 5.4: Fetched Data

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion

Alarm triggered medicine remainder is an native android application meant to aid the forgetful and busy with remembering to take their daily medicines.it is designed for users who need a little help keeping track of their medication schedule and who are dedicated to keeping the schedule. The application allows the user to store pill names and alarms for those pills. The user is able to view their pills name when alarm trigger.

6.2 Future Enhancement

- We can add more functionality and background color to the project for the realistic android application.
- We can also add a reset button to clear the result.
- We can improve our project by adding a storing option that gives previously inserted data.

REFERENCES

- [1] Google Developer Training, "Android Developer Fundamentals Course – Concept Reference", Google Developer Training Team, 2017. <https://www.gitbook.com/book/google-developer-training/android-developer-fundamentals-course-concepts/details>

- [2] Erik Hellman, “Android Programming – Pushing the Limits”, 1st Edition, Wiley India Pvt Ltd, 2014. ISBN-13: 978-8126547197

- [3] Bill Phillips, Chris Stewart and Kristin Marsicano, “Android Programming: The Big Nerd Ranch Guide”, 3rd Edition, Big Nerd Ranch Guides, 2017. ISBN-13: 978-0134706054