VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELGAVI – 590014

A MINI PROJECT REPORT ON

MEETING SCHEDULE

IN

COMPUTER SCIENCE & ENGINEERING

By

AMULYA GL (1DB20CS012)

ANANYA S (1DB20CS015)

DEV H GOWDA (1DB20CS035)

K AKASH KIRAN (1DB20CS054)

UNDER GUIDANCE OF

Prof. KOMALA D

Assistant Professor

Department of CSE,

DBIT

DON BOSCO INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BENGALURU – 560074, KARNATAKA

2022 – 2023

DON BOSCO INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BENGALURU – 560074, KARNATAKA

CERTIFICATE

This is to certify that the Mini Project Report entitled "**Meeting Schedule**" is a bonafide Mini Project work carried out by Amulya G L (1DB20CS012), Ananya S (1DB20CS016), Dev H Gowda (1DB20CS035), K Akash Kiran (1DB20CS054) in partial fulfilment of VI semester for the Degree of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi, during the academic year 2022-23. It is certified that all corrections/suggestions indicated for Internal Assessments have been incorporated with the degree mentioned.

Prof. Komala D	**Dr K B Shivakumar**
Assistant Professor	Head of The Department
Dept. of CSE DBIT, Bangalore	Dept. of CSE DBIT Bangalore

External Viva

Name of the Examiners	Signature with Date
1. _____	_____
2. _____	_____

DON BOSCO INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BENGALURU – 560074, KARNATAKA

DECLARATION

We **Amulya G L, Ananya S, Dev H Gowda, K Akash Kiran** hereby declare that the dissertation entitled, *" **Meeting Schedule**"* is completed and written by us under the supervision of my guide **Komala D,** Assistant Professor, Department of Computer Science and Engineering, Don Bosco Institute of Technology, Bengaluru, of the Visvesvaraya Technological University, Belagavi - 590014, during the academic year 2022-2023.The dissertation report is original and it has not been submitted for any other degree in any university.

Amulya G L	**1DB20CS012**
Ananya S	**1DB20CS015**
Dev H Gowda	**1DB20CS035**
K Akash Kiran	**1DB20CS054**

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of the 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 beacons of light and crowned the effort with success.

We are indebted to the Management of Don Bosco Institute of Technology, Bengaluru for providing an environment that helped us complete our mini project.

I would like to thank Principal Dr Nagabhushana B S, Don Bosco Institute of Technology for his support though out this mini project.

I express my whole hearted gratitude to Dr K B ShivaKumar, who is our respectable Associate Professor and Head of Dept, Department of Computer Science and Engineering. I wish to acknowledge for his valuable help and Encouragement.

In this regard, a heartfelt gratitude to guide Prof. Komala D, Assistant Professor of Department of Computer Science and Engineering, for her timely advice on the mini project and regular assistance throughout the mini project work.

I would also like to thank the staff members of Department of Computer Science and Engineering for their co- operation.

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

Amulya G L	**1DB20CS012**
Ananya S	**1DB20CS015**
Dev H Gowda	**1DB20CS035**
K Akash Kiran	**1DB20CS054**

ABSTRACT

The proposed meeting scheduler Android application aims to provide users with a convenient and user-friendly way to schedule and manage their meetings. By offering various features such as storing meeting details in a database, displaying appropriate toast messages for different actions, setting reminders for scheduled meetings, and sending notifications to users, the application enhances the overall meeting management experience.

When a user wants to schedule a meeting, they are prompted to provide the necessary details such as meeting title, date, time, location, and participants. Once the user submits the details, the application stores the information in a database, ensuring that it is securely saved for future reference.

To enhance the user experience and provide real-time feedback, the application utilizes toast messages. These messages are displayed to the user based on their actions. For example, when a meeting is successfully scheduled, a toast message may appear saying "Meeting scheduled successfully!" Alternatively, if there are any errors or missing information, appropriate error messages can be displayed to guide the user in providing accurate details.

One of the key features of the application is the reminder functionality. When a meeting is scheduled, the application sets a reminder for the user. This reminder is triggered a specific time before the meeting, such as 15 minutes prior. When the reminder is activated, a notification is sent to the user, displaying a message like "You have a meeting scheduled in 15 minutes." This notification ensures that users are aware of their upcoming meetings and helps them stay organized and punctual.

Additionally, the application provides a feature to check if any meetings are scheduled on a given date. This functionality allows users to retrieve information about their scheduled meetings for a particular day. The application arranges and presents the meeting details, enabling users to quickly review their agenda. Toast messages may also be displayed to provide relevant information, such as "No meetings scheduled for today" or "You have 2 meetings scheduled for today."

The proposed meeting scheduler Android application offers a user-friendly interface for scheduling and managing meetings, with features like database storage, toast messages for feedback, reminders, and notifications. It enhances the meeting management experience, providing convenience, organization, and real-time updates. The improved UI ensures a better user experience and overall application performance.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

In recent years, smartphones have transformed the concept of mobile phones, becoming an integral part of people's communication and daily lives. The Android system has emerged as a dominant platform in the smartphone market. Its open-source nature and availability of free development tools have contributed to its widespread popularity. This has resulted in a vast array of Android applications that have enriched people's lives and communication experiences.

Developers have been drawn to the Android system due to its convenient hardware platform and the ease with which they can bring their ideas to fruition. Using the Java programming language, along with development tools like Eclipse, Android ADT, and the Android SDK, developers have been able to create mobile applications with visually appealing interfaces and seamless functionality. The abundance of resources and documentation available for Android development has further accelerated the growth of the platform.

The success of Android applications can be attributed to their emphasis on delivering a smooth and intuitive user experience. Developers strive to create applications with user-friendly interfaces that prioritize ease of use and functionality. By incorporating best design practices and leveraging the capabilities of the Android platform, these applications provide a seamless and enjoyable experience for users.

Privacy and data security are paramount in the Android ecosystem. Developers are encouraged to adhere to strict privacy guidelines and implement robust security measures to protect user information. This commitment to privacy instills confidence in users and fosters trust in the platform. Users can interact with Android applications, knowing that their personal data is safeguarded.

Smartphones powered by the Android system have transformed mobile communication, providing a diverse range of user-friendly applications. With continuous advancements and improvements, the Android ecosystem promises an even better user experience in the future. It has become an integral part of our lives, offering convenience, functionality, and endless possibilities

1.1 Android:

Android is a mobile operating system developed by Google, based on the Linux kernel and designed for touchscreen devices like smartphones and tablets. It features a user interface that emphasizes direct manipulation through touch gestures such as swiping and tapping. Google has expanded Android's reach to other devices, including televisions (Android TV), cars (Android Auto), and wristwatches (Android Wear). Variants of Android are also utilized in notebooks, game consoles, and digital cameras. Initially developed by Android, Inc., it was acquired by Google in 2005 and unveiled in 2007. The Google Play store hosts over a million Android apps, with billions of downloads, making it a popular platform among developers.

Figure 1.1 Android Image

1.2 Software Development Kit:

A software development kit (SDK or "devkit") is typically a set of software development tools. that allows the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform. To create applications you have to download this software development kit. For example, if you want to create an Android app you require an SDK with java programming, for iOS apps you require an iOS SDK with swift language, and to develop MS Windows apps you require the.net language. There are also SDKs that are installed in apps to provide analytics and data about activity. Prominent examples include Google and Facebook.

1.3. Android Studio:

Android Studio is an integrated development environment (IDE) for developing for the Android platform. It was announced on May 16, 2013 at the Google I/O conference. Android Studio is freely available under the Apache License 2.0. Android Studio was in early access preview stage starting from version. Figure 1.3 shows the android studio image.

0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0 to currently Arctic Fox Based on JetBrains' IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, Mac OS X and Linux, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

Figure 1.3 Android Studio

1.4. Android Manifest:

The AndroidManifest.xml file contains information of your package, including components of the application such as activities, services, broadcast receivers, content providers etc.

It performs some other tasks also:

- It is responsible to protect the application to access any protected parts by providing the permissions.
- It also declares the android API that the application is going to use.
- It lists the instrumentation classes. The instrumentation classes provide profiling and other information’s. These information’s are removed just before the application is published etc.

1.5. Main Activity:

The Main Activity File

The main activity code is a Java file MainActivity.java. This is the actual application file which ultimately gets converted to a Dalvik executable and runs your application.

Following is the default code generated by the application wizard for Hello World! application − package com.example.helloworld;

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
public class MainActivity extends AppCompatActivity {
        @Override   protected void
        onCreate(BundlesavedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
                                        }}
```

Here, R.layout.activity_main refers to the activity_main.xml file located in the res/layout folder. The onCreate() method is one of many methods that are figured.

CHAPTER 2

PROBLEM STATEMENT AND OBJECTIVES

1.Problem Statement:

Develop a content provider application with an activity called “Meeting Schedule” whichtakes Date, Time and Meeting Agenda as input from the user and store this information into the SQLite database. Create another application with an activity called “MeetingInfo” having DatePicker control, which on the selection of a date should display the Meeting Agenda information for that particular date, else it should display a toast message saying “No Meeting on this Date”. Add a number of participants to the meeting.

2.Objectives:

The main objective of this Project is to create an application which meets the user needsin the most helpful manner to ease the complexity levels. Objective IT specialise in developing native and cross platform mobile app that allow you to extend your internal systems and databases and use them on the move.

We design an application with two tabs named “Meeting Schedule” and the other one name “Meeting Info”. In the first tab, meeting details like, Date, Time and Meeting Agenda is been taken from the user and the data is been stored in SQLite database and for the second tab we have a calendar provided to the user, so that they can select a date from that to see if any meeting is been scheduled for the day. The corresponding toast messages are also been added for the action along the activity.

CHAPTER 3

LITERATURE SURVEY

The paper "Design and Implementation of a Mobile Meeting Scheduler" by Wei-Hua Lin, Zhihui Du, and Hsuan-Sheng Chen introduces a mobile meeting scheduler that leverages personal information management techniques to offer intelligent meeting scheduling assistance on mobile devices. The project aims to address the challenges of scheduling meetings efficiently and conveniently while considering users' preferences and availability.

The mobile meeting scheduler incorporates various features to facilitate the scheduling process. It utilizes personal information management techniques to collect and organize relevant user data, such as calendars, contact lists, and availability. By analyzing this information, the scheduler can provide intelligent suggestions and recommendations for meeting times and locations that are suitable for all participants. The system also takes into account the participants' preferences and priorities, ensuring that the scheduling process aligns with individual requirements.

Furthermore, the mobile meeting scheduler implements a user-friendly interface optimized for mobile devices, enabling users to easily access and manage their meeting schedules on the go. Notifications and reminders are incorporated to keep participants informed about upcoming meetings, avoiding scheduling conflicts and ensuring efficient coordination.

Overall, the project focuses on enhancing the meeting scheduling experience on mobile devices by leveraging personal information management techniques and providing intelligent assistance. By streamlining the process and considering individual preferences, the mobile meeting scheduler offers a convenient and efficient solution for users to schedule and manage their meetings effectively.

In addition to the features mentioned above, the mobile meeting scheduler also incorporates a cancel option, allowing users to easily cancel scheduled meetings if necessary. Moreover, to enhance efficiency, the scheduler automatically sends meeting information to the users' email addresses, ensuring they have access to all relevant details and reducing the need for manual information retrieval.

CHAPTER 4

SYSTEM REQUIREMENT

3.1. HARDWARE REQUIREMENT:

Minimum RAM	:-	8 GB or more.
Processor	:-	Intel core i5 9th Gen.
Disk Space	:-	20 GB of available disk space.
Display	:-	1280 x 800 minimum screen resolution.

3.2. SOFTWARE REQUIREMENT:

Operating System	:-	64-bit Microsoft Windows 8/10.
Emulator	:-	Pixel 4 api 30.
Target SDK version	:-	30.
Language Used	:-	xml , java.
JDK version	:-	java development kit 16.

EXECUTION

RUN ON EMULATOR:

1. Run the Android Studio on laptop or desktop.

2. Create a project and enter the source codes of xml and java.

3. In Android Studio, Create AVD, That the emulator can use to run and install in your App

4. In the toolbar, select your app from the run/debug configurations drop-down menu.

5. From the target device drop-down menu, select the AVD that you want to run your app on.

6. Click run . Android Studio installs the app on the AVD and starts the emulator.

CHAPTER 5

IMPLIMENTATION

XML CODE:

5.1.activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
    <androidx.coordinatorlayout.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

<com.google.android.material.appbar.AppBarLayout
android:layout_width="match_parent"
android:layout_height="wrap_content" android:background="#075378"
android:theme="@style/Theme.MeetingInfo.AppBarOverlay">

<TextView android:id="@+id/title"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:background="#00EE8F"
android:gravity="center"
android:minHeight="?actionBarSize"
android:padding="@dimen/appbar_padding"
android:text="Meeting Details"
android:textAlignment="center"
android:textAppearance="@style/TextAppearance.Widget.AppCompat.Toolbar.Title" />

<com.google.android.material.tabs.TabLayout
```

```
android:id="@+id/tabs" android:layout_width="match_parent"
android:layout_height="37dp" android:background="#8DFF3B"
app:tabIndicatorColor="#175138" />
</com.google.android.material.appbar.AppBarLayout>

<androidx.viewpager.widget.ViewPager android:id="@+id/view_pager"
android:layout_width="match_parent" android:layout_height="499dp"
app:layout_behavior="@string/appbar_scrolling_view_behavior">

<TextView android:id="@+id/textView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"android:text="TextView" />
</androidx.viewpager.widget.ViewPager>

</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

5.2.fragment_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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:id="@+id/constraintLayout"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".ui.main.PlaceholderFragment">

    <TextView
    android:id="@+id/section_label"
    android:text="Schedule Meeting"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="@dimen/activity_horizontal_margin"
        android:layout_marginTop="@dimen/activity_vertical_margin"
        android:layout_marginEnd="@dimen/activity_horizontal_margin"
        android:layout_marginBottom="@dimen/activity_vertical_margin"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="@+id/constraintLayout"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintTop_creator="1" />

 </androidx.constraintlayout.widget.ConstraintLayout>
```

5.1 fragment2_layout.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="Select Date to get Meeting Details"
        android:textColor="#09655C"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
```

```
    android:id="@+id/editTextDate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="text"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.512"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView2"
    app:layout_constraintVertical_bias="0.062" />

<CalendarView
    android:id="@+id/calendarView"
    android:layout_width="296dp"
    android:layout_height="313dp"
    android:layout_marginTop="32dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.582"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editTextDate" />

<Button
    android:id="@+id/btn2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Search to get Meeting Details"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.582"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/calendarView"
    app:layout_constraintVertical_bias="0.35" />

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

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Cancle meeting"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/textView2"
        app:layout_constraintVertical_bias="0.937" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

5.3 fragment_layout1.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    android:backgroundTint="#1A6670">

    <EditText
        android:id="@+id/txtDate"
        android:layout_width="212dp"
        android:layout_height="50dp"
        android:layout_marginTop="16dp"
        android:ems="10"
        android:hint="DD/MM/YYYY"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.946"
        app:layout_constraintStart_toEndOf="@+id/mDate"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/txtTime"
```

```
    android:layout_width="211dp"
    android:layout_height="63dp"
    android:layout_marginTop="64dp"
    android:layout_marginEnd="8dp"
    android:ems="10"
    android:hint="hh:mm"
    android:inputType="time"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="@+id/txtDate" />

<TextView
    android:id="@+id/mDate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="24dp"
    android:text="Date:"
    android:textAppearance="@style/TextAppearance.AppCompat.Large"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/txt2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:text="Time:"
    android:textAppearance="@style/TextAppearance.AppCompat.Large"
    app:layout_constraintEnd_toStartOf="@+id/txtTime"
    app:layout_constraintHorizontal_bias="0.087"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/mDate" />

<TextView
    android:id="@+id/txt3"
    android:layout_width="152dp"
```

```
    android:layout_height="61dp"
    android:layout_marginStart="4dp"
    android:text="Meeting Agenda:"
    android:textAppearance="@style/TextAppearance.AppCompat.Large"
    app:layout_constraintBottom_toTopOf="@+id/btn1"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/txt2"
    app:layout_constraintVertical_bias="0.083" />

<Button
    android:id="@+id/btn1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#117380"
    android:backgroundTint="#246770"
    android:text="Add Meeting Schedule"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.576"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/txtAgenda"
    app:layout_constraintVertical_bias="0.985" />

<EditText
    android:id="@+id/txtAgenda"
    android:layout_width="233dp"
    android:layout_height="58dp"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:inputType="textPersonName"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.279"
    app:layout_constraintStart_toEndOf="@+id/txt3"
    app:layout_constraintTop_toBottomOf="@+id/txtTime" />

<CalendarView
```

```
            android:id="@+id/mCal"
            android:layout_width="385dp"
            android:layout_height="305dp"
            android:background="#FFFFFF"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.615"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.774" />

    </androidx.constraintlayout.widget.ConstraintLayout>
```

JAVA CODE:

MainActivity.java:

```
package com.example.meetinginfo;

import android.os.Bundle;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;
import com.google.android.material.tabs.TabLayout;

import androidx.viewpager.widget.ViewPager;
import androidx.appcompat.app.AppCompatActivity;

import android.view.Menu;
import android.view.MenuItem;
import android.view.View;

import com.example.meetinginfo.ui.main.SectionsPagerAdapter;

public class MainActivity extends AppCompatActivity {

   @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        SectionsPagerAdapter sectionsPagerAdapter = new SectionsPagerAdapter(this,
getSupportFragmentManager());
        ViewPager viewPager = findViewById(R.id.view_pager);
        viewPager.setAdapter(sectionsPagerAdapter);
        TabLayout tabs = findViewById(R.id.tabs);
        tabs.setupWithViewPager(viewPager);
    }
}
```

Fragment1.java:

```
package com.example.meetinginfo;

import android.content.Context;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.CalendarView;
import android.widget.EditText;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

import static androidx.core.content.ContextCompat.getSystemService;

public class Fragment1 extends Fragment {
    EditText date,time,agenda;
    DataBaseConn dbc;
    CalendarView calendarView;
```

```
    Button btn;
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup
container, @Nullable Bundle savedInstanceState) {
        View view=inflater.inflate(R.layout.fragment_layout1,container,false);
        date=view.findViewById(R.id.txtDate);
        time=view.findViewById(R.id.txtTime);
        agenda=view.findViewById(R.id.txtAgenda);
        btn=view.findViewById(R.id.btn1);
        calendarView=view.findViewById(R.id.mCal);
        dbc=new DataBaseConn(getActivity());    //need to initialize here only
        calendarView.setVisibility(View.INVISIBLE);
        date.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                closeKeyBoard();
                calendarView.setVisibility(View.VISIBLE);
                calendarView.setOnDateChangeListener(new CalendarView.OnDateChangeListener()
{
                    @Override
                    public void onSelectedDayChange(@NonNull CalendarView view, int year, int
month, int dayOfMonth) {
                        String d=dayOfMonth+"/"+(month+1)+"/"+year;
                        date.setText(d);
                        calendarView.setVisibility(View.INVISIBLE);
                    }
                });
            }
        });
        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String mdate,mTime,mAgenda;
                mdate=date.getText().toString();
                mTime=time.getText().toString();
```

```
            mAgenda=agenda.getText().toString();

            Boolean insert=dbc.insertvalue(mdate,mTime,mAgenda);
            if(insert==true){
                Toast.makeText(getActivity(),"Meeting
Scheduled",Toast.LENGTH_SHORT).show();

            }
            else

Toast.makeText(getActivity(),"DataNOTInserted",Toast.LENGTH_SHORT).show();
            //txt.setText("NOT INSERTED");
          }
       });
       return view;
   }
   private void closeKeyBoard(){
       View  view = getActivity().getCurrentFocus();
       if (view != null) {
           InputMethodManager imm = (InputMethodManager)
                   getActivity().getSystemService(Context.INPUT_METHOD_SERVICE);
           imm.hideSoftInputFromWindow(view.getWindowToken(), 0);
       }

   }

}
```

Fragment2.java:

```
package com.example.meetinginfo;

import static com.example.meetinginfo.R.id.button;

import android.database.Cursor;
import android.os.Bundle;
import android.view.LayoutInflater;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.CalendarView;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.time.Duration;

import java.time.Duration;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;



import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatDialogFragment;
import androidx.fragment.app.Fragment;

import com.example.meetinginfo.R;

import java.time.temporal.Temporal;
import java.util.Calendar;

import papaya.in.sendmail.SendMail;

public class Fragment2 extends Fragment {
    EditText date;
    CalendarView cal;
    Button  btn1;
    Button cancle;
    DataBaseConn dbc;
    TextView t;
   // String med="";
   // String med1="";
```

```
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup
container, @Nullable Bundle savedInstanceState) {
        View view=inflater.inflate(R.layout.fragment2_layout,container,false);
        date=view.findViewById(R.id.editTextDate);
        cal=view.findViewById(R.id.calendarView);
        btn1=view.findViewById(R.id.btn2);
        cancle=view.findViewById(button);
        dbc=new DataBaseConn(getActivity());
       // t=()

        cal.setOnDateChangeListener(new CalendarView.OnDateChangeListener() {
            @Override
            public void onSelectedDayChange(@NonNull CalendarView view, int year, int month,
int dayOfMonth) {
                String d=dayOfMonth+"/"+(month+1)+"/"+year;
                date.setText(d);
            }
        });

        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String d1=date.getText().toString();
                StringBuffer res=new StringBuffer();
                Cursor c=dbc.fetch(d1);
                int count=c.getCount();
                c.moveToFirst();
                if(count>0) {
                    do {

res.append(c.getString(c.getColumnIndex("agenda"))+"\t"+"at"+"\t"+c.getString(c.getColumnIn
dex("time")));
                        res.append("\n");
```

```
                //med = (String.valueOf(c.getString(c.getColumnIndex("agenda"))));
                //med1 = (String.valueOf(c.getString(c.getColumnIndex("time"))));
            }while (c.moveToNext());
                Toast.makeText(getActivity(), res, Toast.LENGTH_LONG).show();
        }
        else
        {
            Toast.makeText(getActivity(), "No Meeting on This Day... ",
Toast.LENGTH_LONG).show();

        }

    }
});

    cancle.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String d1=date.getText().toString();
            StringBuffer res=new StringBuffer();
            Cursor c=dbc.fetch(d1);
            int count=c.getCount();
            c.moveToFirst();
            if(count>0) {
                do {

                }while (c.moveToNext());
                SendMail mail = new SendMail("devhgowda26@gmail.com", "ekumtkemdufjbpfh",
                        "amulyaglm@gmail.com",
                        "Reg. Cancled Meeting",
                        "The Meeting Scheduled is Cancled by the user\nEnjoy Your Day!.");
                mail.execute();

                Toast.makeText(getActivity(),  "The Meeting on This Day is Cancled... ",
Toast.LENGTH_LONG).show();
            }
```

```
            else
            {
                Toast.makeText(getActivity(), "No Meeting on This Day... ",
Toast.LENGTH_LONG).show();
            }
          }
        });
        return view;
    }
    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }
}
```

DataBaseConn.java:

```
package com.example.meetinginfo;
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;
import androidx.fragment.app.FragmentActivity;

public class DataBaseConn extends SQLiteOpenHelper {
    public DataBaseConn(Context context) {
        super(context,"MeetingDB.db",null,1);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create Table meetingTbl(date TEXT,time TEXT, agenda TEXT)");
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("drop Table if exists meetingTbl");
```

```
    }
    public boolean insertvalue(String d, String t, String agd){
        SQLiteDatabase DB=this.getWritableDatabase();
        ContentValues cv = new ContentValues();
        cv.put("date",d);
        cv.put("time",t);
        cv.put("agenda",agd);
        long res=DB.insert("meetingTbl",null,cv);  //query to insert
        if(res==-1){
            return  false;
        }
        else
            return true;
    }

    public Cursor fetch(String d){
        SQLiteDatabase DB=this.getReadableDatabase();
        // String sqlquery="select name from MDTbl where date='19/3/21' AND time='Afternoon'";
        // Cursor c = DB.rawQuery(sqlquery,null);
        Cursor c = DB.rawQuery("Select time,agenda from meetingTbl where date='"+d+"' ",null);
        return c;
    }
    public String fetchi(String d){
        SQLiteDatabase DB=this.getReadableDatabase();
        // String sqlquery="select name from MDTbl where date='19/3/21' AND time='Afternoon'";
        // Cursor c = DB.rawQuery(sqlquery,null);
        String t = String.valueOf(DB.rawQuery("Select time from meetingTbl where date='"+d+"'
",null));
        return t;
    }
}
```

CHAPTER 6

RESULTS AND SNAPSHOTS

6.1Main Page:

The Figure 6.1 consists of the main page of the android application of meeting schedule. The project first page also includes designs, creatives, templates and other graphics and drawings.

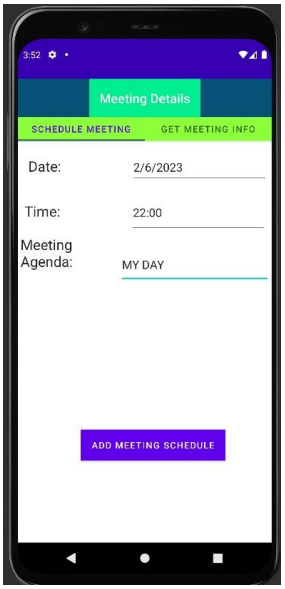


Fig 6.1. Main Page

6.2. Scheduling Meeting with Calander view:

The Figure 6.2 allows us to schedule the meeting with calendar view the user can easily get the access to set the desired meeting with respect to time.

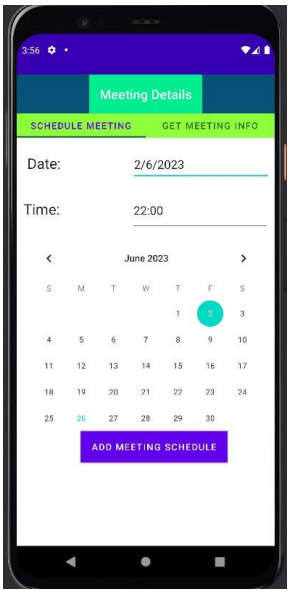


Fig 6.2. Scheduled Meeting with Calander view

6.3. Meeting Details.

Figure 6.3 allows us to fill the information about the meeting date, time, meeting agenda etc. This page gives the details of meeting that is scheduled.

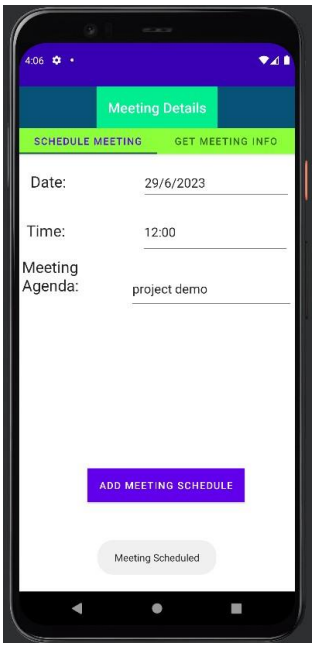


Fig.6.3. Meeting Details

6.4. Scheduled Meeting Information.

The Figure 6.4 shows the information about the scheduled meeting. The user can easily get the access to set the desired meeting with respect to time.

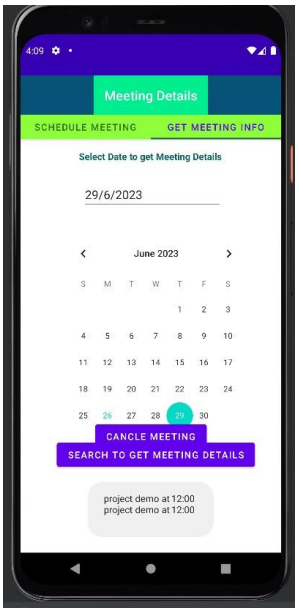
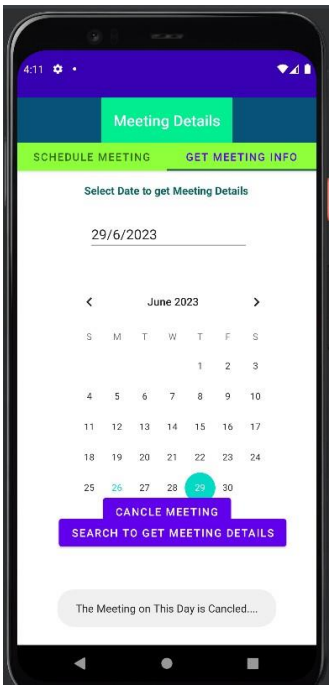


Fig.6.4. Scheduled Meeting Info

6.5. Cancel Meeting.

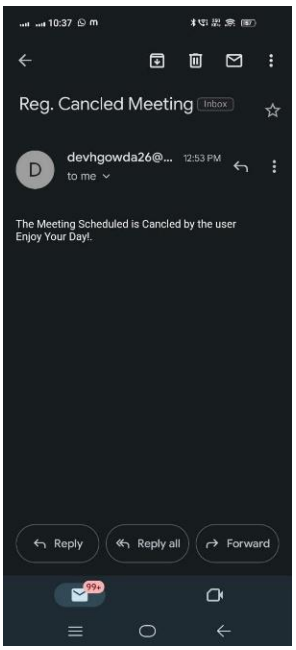
The Figure 6.5 shows the information about the cancellation of meeting. The user can easily get the access to cancel the desired meeting with respect to time.



6.5. Cancel Meeting

6.6. Sending email to mention email-id:

The Figure 6.6 shows the information about sending the details to mentioned mail id. The user can easily get the access to know that scheduled meeting has been cancelled.



6.6. Sending email to mention email id.

CONCLUSION

Developing an Android app presents a variety of challenges that can both motivate and satisfy developers. The "Meeting Scheduler" project exemplifies these experiences by utilizing SQLite to store meeting details such as dates, times, and agendas. One key challenge in Android app development lies in understanding and adapting to the platform's diverse ecosystem, which encompasses different devices and screen sizes. This necessitates careful optimization of layout, user interface, and performance to ensure a consistent user experience across various devices.

The integration of SQLite into the "Meeting Scheduler" project introduces its own set of challenges. Developers must handle tasks such as creating databases and tables, managing connections, and implementing data validation to maintain data integrity. Efficient management of SQLite queries and transactions is crucial for effective data manipulation within the app.

Another challenge is handling user input and interactions. In the context of the "Meeting Scheduler" project, incorporating a date picker enables users to select a specific date and retrieve associated meeting details. This requires capturing and validating user input, as well as seamlessly integrating it with the app's functionality. Additionally, developers must handle scenarios where no meeting is scheduled on the selected date, providing appropriate feedback to the user through toast messages.

Despite the challenges, the process of developing an Android app can be highly rewarding. Overcoming technical obstacles enhances a developer's skills and contributes to the dynamic Android app ecosystem. The opportunity to create powerful, user-friendly applications for the most popular mobile operating system can be incredibly motivating and satisfying, making Android app development an exciting endeavor.

FUTURE SCOPE

The inclusion of a cancel option in the mobile meeting scheduler opens up future scopes for further improving the user experience. One potential enhancement could be the integration of automated notifications, allowing participants to receive timely reminders and prompts to cancel meetings they are unable to attend. This proactive approach would streamline the cancellation process and ensure that all attendees are promptly notified of any changes.

Sending meeting information to the user's email address is a valuable addition to the mobile meeting scheduler. To expand on this feature, future development could focus on providing users with personalized meeting summaries that include key details such as agenda, attendees, and location. This would enable participants to have a comprehensive overview of each meeting at their fingertips, even without accessing the mobile app, enhancing efficiency and ensuring that everyone is well-informed.

In terms of efficiency, there are several avenues for improvement. One potential enhancement could involve integrating intelligent algorithms to optimize meeting scheduling based on factors such as participants' preferences, availability patterns, and historical data. By leveraging machine learning and data analytics techniques, the scheduler could suggest the most suitable meeting times and locations, reducing the need for manual adjustments and saving valuable time for all involved.

Looking ahead, future iterations of the mobile meeting scheduler could explore integration with other productivity tools and platforms. For instance, integration with popular team collaboration platforms or project management software could enable seamless synchronization of meeting schedules, task assignments, and related documents. This integration would promote a more holistic and streamlined workflow, providing users with a unified platform for managing all aspects of their work and meetings.

By continuously exploring and implementing these future scopes, the mobile meeting scheduler can evolve into a powerful tool that not only simplifies meeting scheduling but also enhances overall productivity and collaboration among users.

REFERENCES

Website:

[1]. YouTube : https://www.youtube.com/

[2]. https://developer.android.com/studio

[3]. https://github.com/

[4].https://www.gitbook.com/book/google-developer-training/android-developer[5].fundamentals-course- concepts/details

Reference Book:

Google Developer Training, "Android Developer Fundamentals Course – Concept Reference", Google Developer Training Team, 2017.