INDEX

Ex:no: 01 CREATE LOGIN PAGE AND DISPLAY TOAST MESSAGE

Date:18.12.2024

Aim:

The aim of this task is to demonstrate how to use intents to navigate between activities in an Android application.

Procedure:

1. Create Android project with proper name, package, and configurations in Android Studio.
2. Optionally, design layouts for activities using XML in res/layout directory.
3. Create Java classes for activities, extending the Activity class for each.
4. Define intents to navigate between activities.
5. Optionally, pass data between activities using extras in intents.
6. Retrieve data passed through intent using getIntent().getExtras() in target activity.
7. Handle intent in target activity by performing actions based on received data.
8. Test application functionality on emulator or physical Android device.

Source Code:

Acitvity_main.xml:

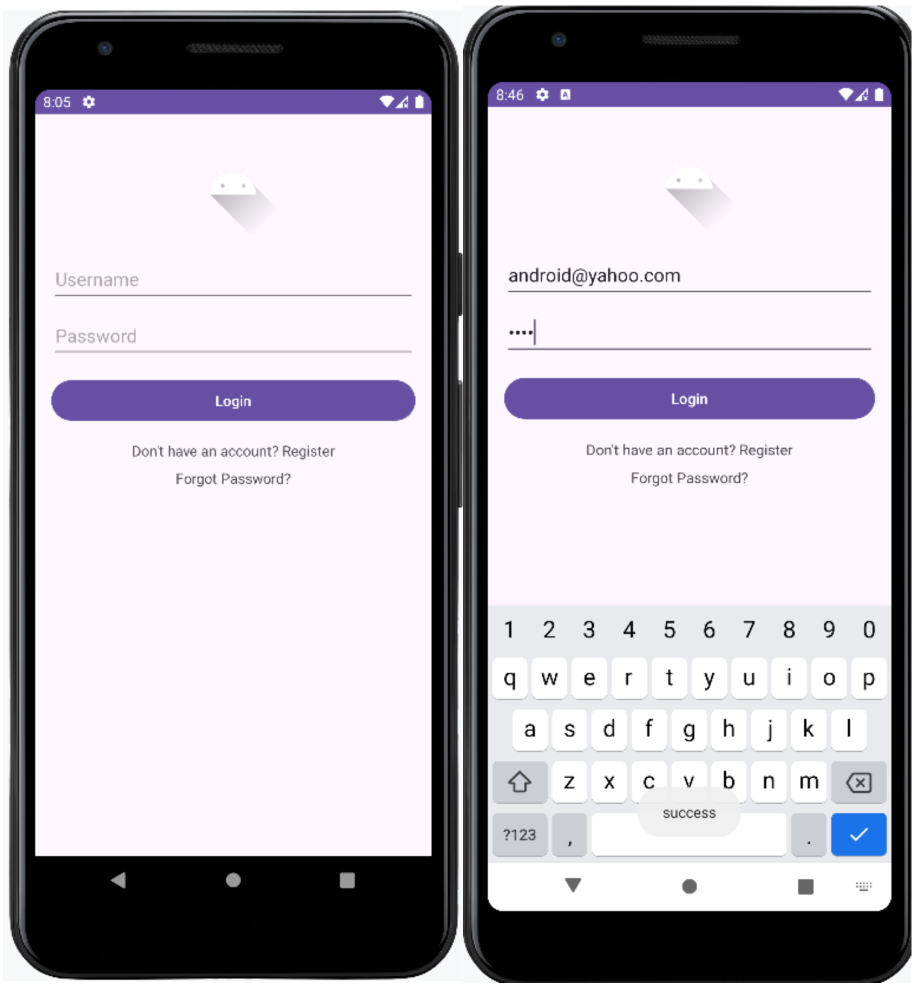
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"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

<LinearLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
android:padding="16dp">
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginBottom="16dp"
    android:src="@drawable/ic_launcher_foreground"
    android:importantForAccessibility="no" />
    <EditText
        android:id="@+id/etUsername"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:hint="Username"
        android:importantForAutofill="no"
        android:inputType="text"
        android:minHeight="48dp"
        tools:ignore="HardcodedText" />
    <EditText
        android:id="@+id/etPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp"
        android:hint="Password"
        android:importantForAutofill="no"
        android:inputType="textPassword"
        android:minHeight="48dp"
        tools:ignore="HardcodedText" />
<Button
    android:id="@+id/button1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Login"
    tools:ignore="HardcodedText" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Don't have an account? Register"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="16dp"
    tools:ignore="HardcodedText" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Forgot Password?"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="8dp"
    tools:ignore="HardcodedText" />
</LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java:

```
package com.example.ex_1;
import androidx.appcompat.app.AppCompatActivity;
import android.app.Activity;
import android.view.View;
import android.view.Menu;
import android.widget.*;
import android.os.Bundle;
```

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button b1 = (Button) findViewById(R.id.button1);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                Toast.makeText(getBaseContext(), "success",
                        Toast.LENGTH_LONG).show();
            }
        });
    }
}
```

Output:



RESULT:

Thus, the Android Application to display toast message is developed and executed successfully

Ex:no:02

DEMONSTRATE INTENT EXPLICIT

Date:21.12.2024

Aim:

The aim of implementing implicit intents in an Android application is to enable communication between different components of an app

Procedure:

1. Define action like Intent.ACTION_VIEW ,Intent.ACTION_SEND for specific functionality.

2. Create Intent object, setting action and optionally other parameters like data or type.

3. Optionally, verify available activity for intent using PackageManager for robustness.

4. Start activity with startActivity() ,startActivityForResult() for passing intent object.

5. Optionally, declare intent filters in manifest for handling implicit intents from other apps.

6. Test app on emulator or physical device to ensure intent and navigation functionality.

Source Code:

Activity_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:background="@android:color/darker_gray"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="137dp"
        android:layout_height="33dp"
        android:fontFamily="sans-serif"
        android:gravity="center_horizontal"
        android:text="INTENT"
        android:textAlignment="center"
        android:textSize="20dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.491"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.227" />

    <Button
        android:id="@+id/btnActivity1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="164dp"
        android:layout_marginLeft="164dp"
        android:layout_marginEnd="148dp"
        android:layout_marginRight="148dp"
        android:text="Next"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.09"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.459"
        tools:text="Next" />

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

Activity_main2.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:background="@color/white"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity2">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="298dp"
        android:layout_marginBottom="414dp"
        android:text="This is Next Page"
        android:textSize="34dp"
        android:fontFamily="sans-serif"
        android:textAppearance="@style/TextAppearance.AppCompat.Display2"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.572"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>
    <TextView
```

```
        android:id="@+id/textView"
        android:layout_width="137dp"
        android:layout_height="33dp"
        android:fontFamily="sans-serif"
        android:gravity="center_horizontal"
        android:text="INTENT-2"
        android:textAlignment="center"
        android:textSize="20dp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.491"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.227" />

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

MainActivity.java:

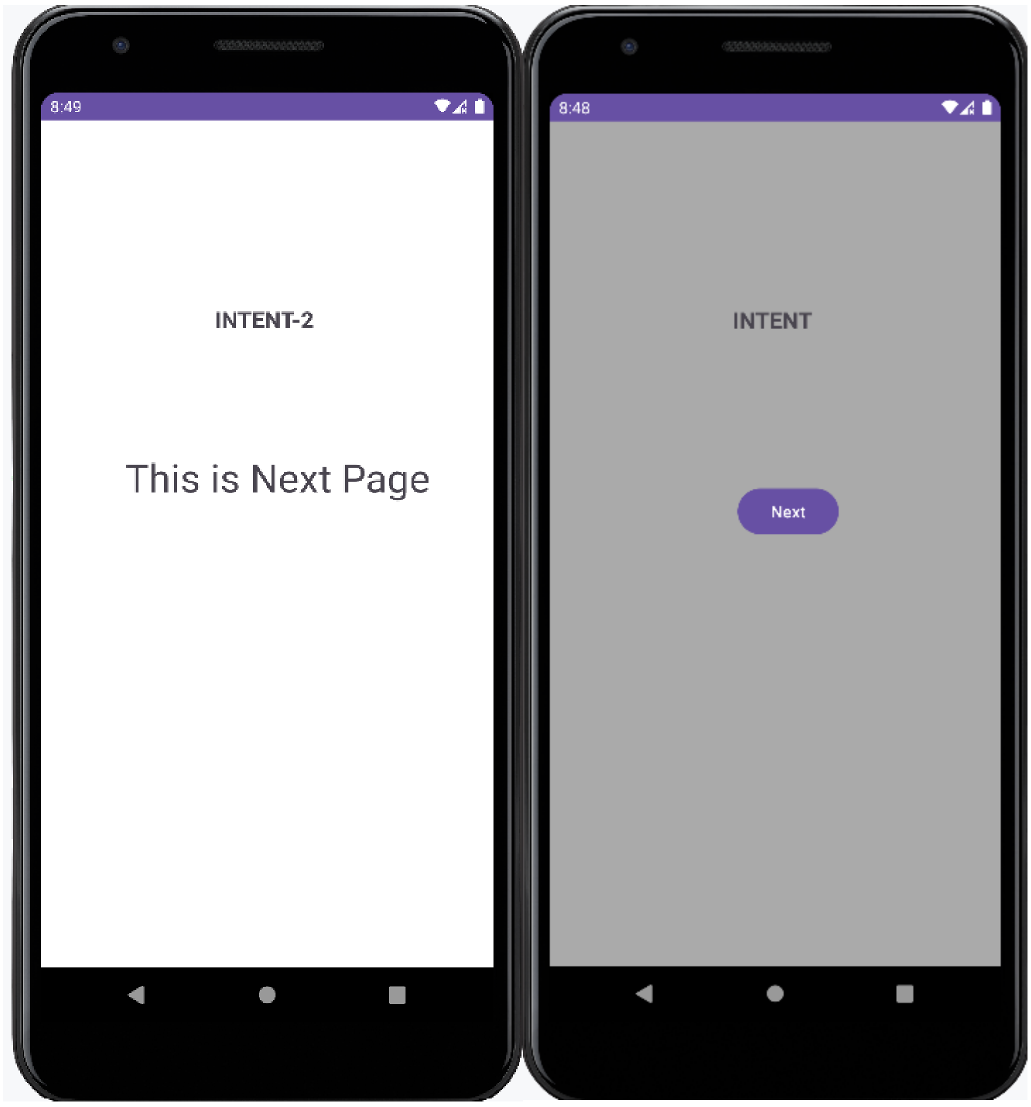
```
package com.example.ex_2;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.view.View;
import android.widget.Button;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button butActivity2 = findViewById(R.id.btnActivity1);
        butActivity2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent ActivIntent = new
                        Intent(MainActivity.this,MainActivity2.class);
                startActivity(ActivIntent);
            }
        });
    }
}
```

MainActivity2.java:

```
package com.example.ex2;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
```

```
public class MainActivity2 extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
    }
}
```

Output:



RESULT:

Thus, the Android Application to implement explicit intent is developed and executed successfully.

Ex No:03 Develop an application for student Registration confirmation using intent Explicit

Date:02.01.2024

Aim:

To develop an application for student Registration confirmation using intent Explicit

Procedure:

1.Open Android Studio and create a new project.

2. Design the layout XML file (**activity_main.xml**) with appropriate fields for student information and a registration button.

3.Add another layout XML file (**activity_confirmation.xml**) for displaying the confirmation message.

4.In the **MainActivity.java** file, retrieve user input from EditText fields when the registration button is clicked.

5. Create a new activity (**ConfirmationActivity.java**) to display the confirmation message.

Retrieve data from the intent passed from MainActivity and display it in the ConfirmationActivity layout.

6.Run the application on an Android emulator or a physical device.

Activity_main.xml

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

    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Name" />

    <EditText
        android:id="@+id/editTextEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextName"
        android:layout_marginTop="84dp"
        android:hint="Email" />

    <Button
        android:id="@+id/buttonRegister"
        android:layout_width="wrap_content"
        android:layout_height="126dp"
        android:layout_below="@id/editTextEmail"
        android:layout_alignParentStart="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="156dp"
        android:layout_marginTop="277dp"
        android:layout_marginEnd="142dp"
        android:layout_marginBottom="224dp"
        android:text="Register" />

</RelativeLayout>
```

Activity_confirmation.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ConfirmationActivity">

    <TextView
        android:id="@+id/textViewConfirmation"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:textSize="18sp"
        android:textStyle="bold"/>

</RelativeLayout>
```

MainActivity.java

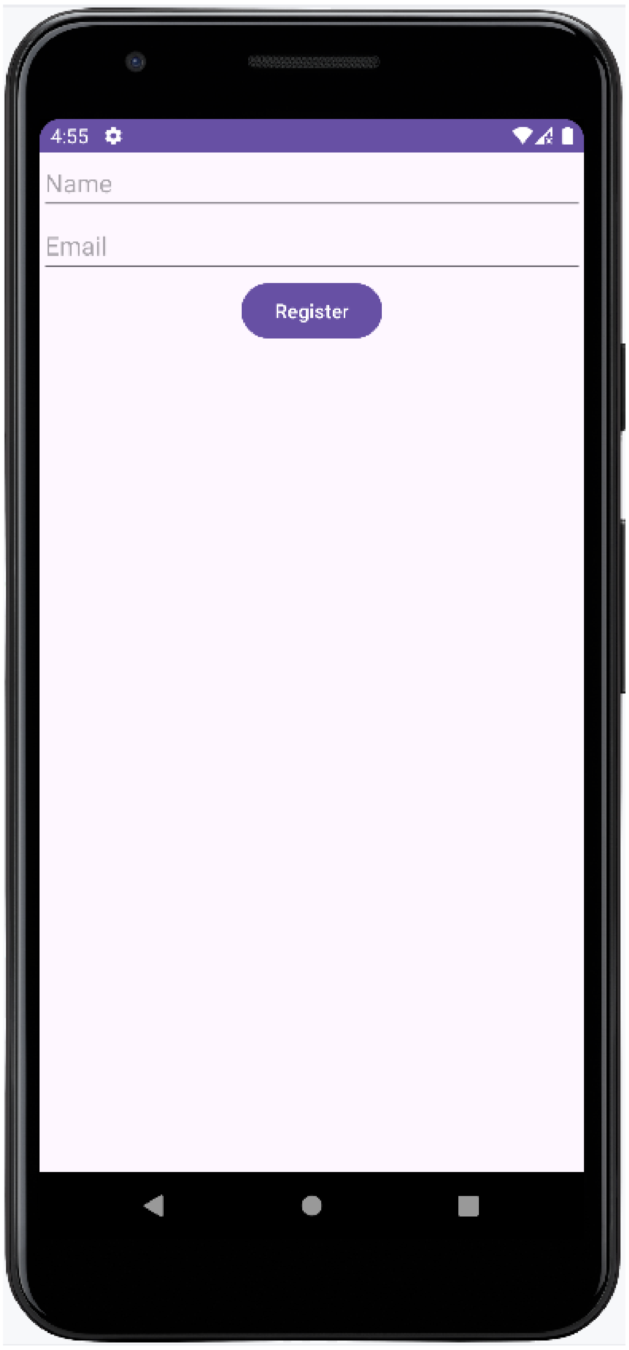
```
package com.example.reg_explicit;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import androidx.appcompat.app.AppCompatActivity;
```

```
public class MainActivity extends AppCompatActivity {

    EditText editTextName, editTextEmail;
    Button buttonRegister;

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

        editTextName = findViewById(R.id.editTextName);
        editTextEmail = findViewById(R.id.editTextEmail);
        buttonRegister = findViewById(R.id.buttonRegister);

        buttonRegister.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Get the input from EditText fields
                String name = editTextName.getText().toString();
                String email = editTextEmail.getText().toString();

                Intent intent = new Intent(MainActivity.this, ConfirmationActivity.class);
                // Pass data to ConfirmationActivity
                intent.putExtra("NAME", name);
                intent.putExtra("EMAIL", email);
                startActivity(intent);
            }
        });
    }
}
```

ConfirmationActivity.java

```
package com.example.reg_explicit;
import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class ConfirmationActivity extends AppCompatActivity {

    TextView textViewConfirmation = findViewById(R.id.textViewConfirmation);

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_confirmation);
        Intent intent = getIntent();
        String name = intent.getStringExtra("NAME");
        String email = intent.getStringExtra("EMAIL");

        String confirmationMessage = "Registration confirmed!\nName: " + name + "\nEmail: "
+ email;
        textViewConfirmation.setText(confirmationMessage);
    }
}
```

Output:



Result

Thus, the Android Application for student Registration confirmation using intent Explicit is developed and executed successfully.

Ex No:04 NAVIGATE TO DIFFERENT WEBSITES USING INTENT IMPLICIT

Date:05.01.2024

Aim:

The aim of implementing Implicit intent to navigate different websites in android studio.

Procedure:

1. Create Android project with proper name, package, and configurations in Android Studio.

2. Optionally, design layouts for activities using XML in res/layout directory.

3. Create Java classes for activities, extending the Activity class for each.

4.Add Permissions: Make sure to add the internet permission to your AndroidManifest.xml file. This is necessary for accessing the internet.

5.In your activity layout file (e.g., activity_main.xml), add a Button that users will click to navigate to a website.

6. Test app on emulator or physical device to ensure implicit intetnt and navigation functionality.

Source Code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:padding="8dp"
   tools:context=".MainActivity">

   <Button
      android:layout_width="170dp"
      android:layout_height="54dp"
      android:layout_alignParentStart="true"
      android:layout_alignParentTop="true"
      android:layout_alignParentEnd="true"
      android:layout_alignParentBottom="true"
      android:layout_marginStart="125dp"
      android:layout_marginTop="383dp"
      android:layout_marginEnd="108dp"
      android:layout_marginBottom="286dp"
      android:onClick="GetUrlFromIntent"
      android:text="Go!!"
      tools:ignore="TouchTargetSizeCheck" />

   <EditText
      android:id="@+id/editTextText2"
      android:layout_width="267dp"
```

```
        android:layout_height="53dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="81dp"
        android:layout_marginTop="222dp"
        android:layout_marginEnd="55dp"
        android:layout_marginBottom="448dp"
        android:ems="10"
        android:inputType="text"
        android:minHeight="48dp"
        android:hint="Username"
        tools:ignore="TouchTargetSizeCheck" />

    <EditText
        android:id="@+id/editTextText3"
        android:layout_width="262dp"
        android:layout_height="58dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="88dp"
        android:layout_marginTop="283dp"
        android:layout_marginEnd="53dp"
        android:layout_marginBottom="382dp"
        android:ems="10"
        android:inputType="text"
        android:hint="Password" />
</RelativeLayout>
```

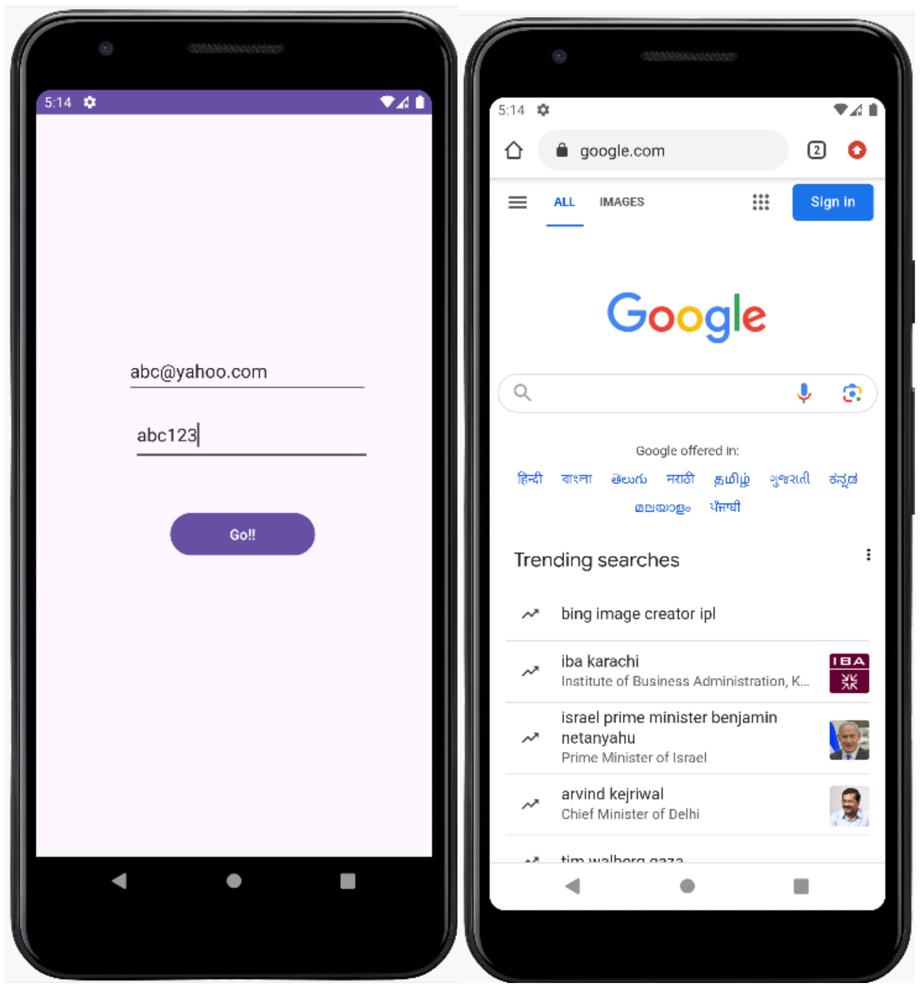
MainActivity.java

```
package com.example.login;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void GetUrlFromIntent(View view) {
        String url = "http://www.google.com";
        Intent i = new Intent(Intent.ACTION_VIEW);
        i.setData(Uri.parse(url));
        startActivity(i);
    }
}
```

AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Login"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Output:


abc@yahoo.com

abc123

Result:

Thus, the Android Application to implement implicit intent is developed and executed successfully

Ex No:05

PASSING MESSAGES FROM ONE ACTIVITY TO ANOTHER USING INTENT.

Date:09.01.2024

Aim:

The aim of implementing message passing using Intent in android studio.

Procedure:

1.Create Project: Start a new Android Studio project.

2.Design layouts for sender and receiver activities with EditText and Button.

3.Sender Activity:Create SenderActivity.java.Get references to EditText and Button.

Set OnClickListener for the send Button.Inside the listener, create an Intent, put the message as an extra, and start ReceiverActivity.

4. Receiver Activity: Create ReceiverActivity.java.Get reference to TextView. Retrieve the message from the Intent using getIntent().getStringExtra().Set the received message to the TextView.

5.Manifest Declaration: Declare both activities in the AndroidManifest.xml file.

6.Testing: Run the application to test sending and receiving messages between activities.

Source Code:

Activity_main.xml

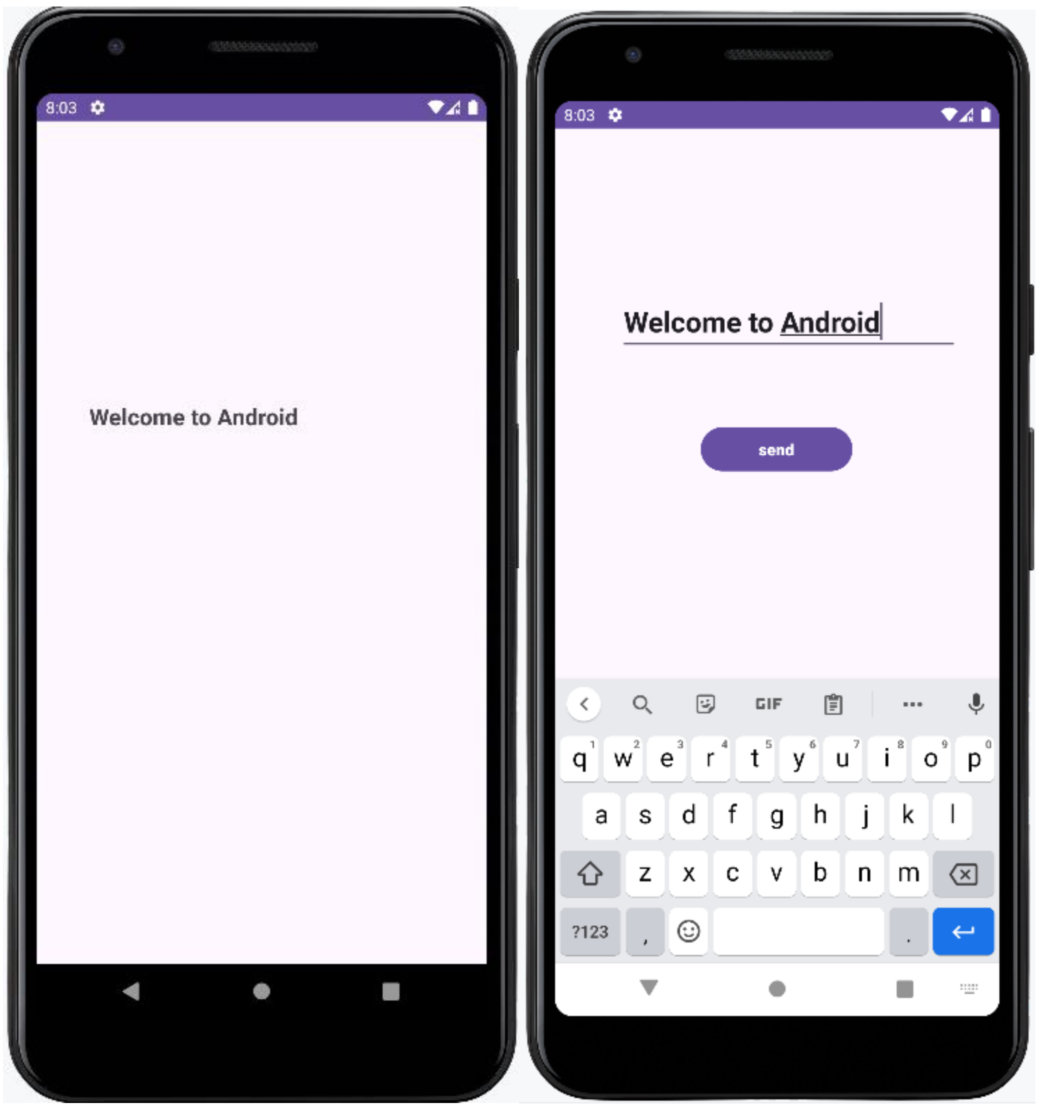
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">

   <EditText
      android:id="@+id/editTextPhone"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Phone Number" />
   <EditText
      android:id="@+id/editTextMessage"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextPhone"
        android:hint="Message" />
    <Button
        android:id="@+id/buttonSend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextMessage"
        android:text="Send"
        android:onClick="sendMessage" />
</RelativeLayout>
```

MainActivity.java

```
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    private EditText editTextPhone, editTextMessage;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        editTextPhone = findViewById(R.id.editTextPhone);
        editTextMessage = findViewById(R.id.editTextMessage);
    }
    public void sendMessage(View view) {
        String phoneNo = editTextPhone.getText().toString();
        String message = editTextMessage.getText().toString();
        Uri uri = Uri.parse("smsto:" + phoneNo);
        Intent intent = new Intent(Intent.ACTION_SENDTO, uri);
        intent.putExtra("sms_body", message);
        startActivity(intent);
    }
}
```

Output:



RESULT:

Thus, the Android Application to implement message passing is developed and executed successfully

Ex:no:06 **DEMONSTRATE DATE PICKER**

Date:12.01.2024

Aim:

The aim of implementing a DatePicker view in an Android application using Android Studio.

Procedure:

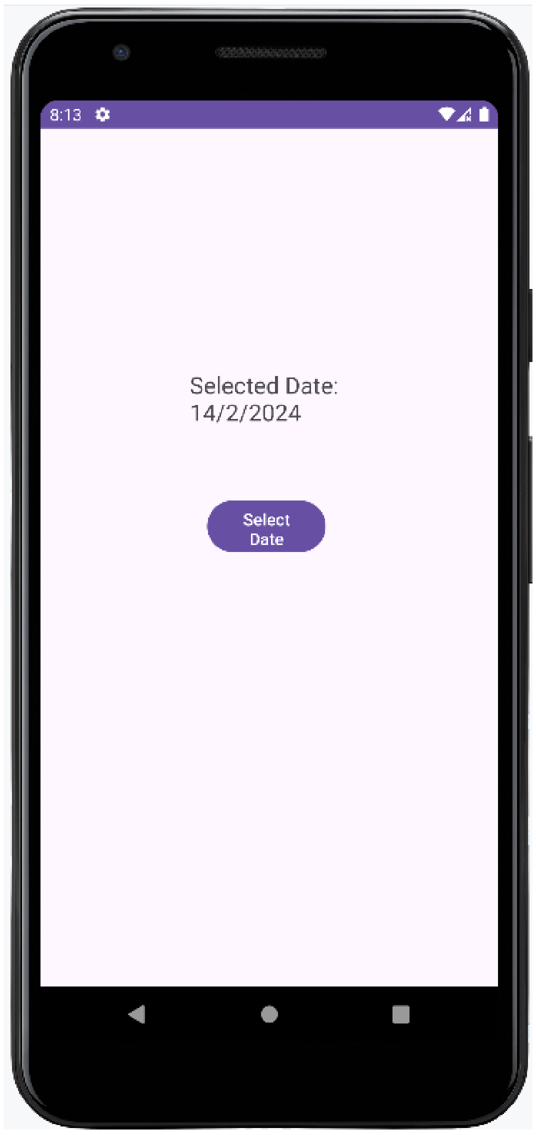
1. Create new Android project or open existing project in Android Studio.
2. Design layout: Add DatePicker widget to layout XML, adjust attributes.
3. Handle DatePicker: Declare and initialize DatePicker object in MainActivity.
4. Retrieve selected date: Use getYear(), getMonth(), getDayOfMonth() methods.
5. Handle selected date in app logic, such as displaying or setting reminders.
6. Optionally, customize DatePicker with attributes like min/max dates, initial date, etc.

Source Code:

MainActivity.java

```
package com.example.datepickerexample;
import androidx.appcompat.app.AppCompatActivity;
import android.app.DatePickerDialog;
import android.os.Bundle;
import android.view.View;
 import android.widget.Button;
 import android.widget.DatePicker;
import android.widget.TextView;
import java.util.Calendar;
public class MainActivity extends AppCompatActivity {
private TextView tvSelectedDate;
private Button btnDatePicker;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
tvSelectedDate = findViewById(R.id.tvSelectedDate);
btnDatePicker = findViewById(R.id.btnDatePicker);
// Set OnClickListener to show DatePickerDialog when the button is clicked
btnDatePicker.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) { showDatePickerDialog();
} }); }
 private void showDatePickerDialog() {
// Get current date final
Calendar calendar = Calendar.getInstance();
int year = calendar.get(Calendar.YEAR);
int month = calendar.get(Calendar.MONTH);
int dayOfMonth = calendar.get(Calendar.DAY_OF_MONTH);
// Create DatePickerDialog and show it
DatePickerDialog datePickerDialog = new DatePickerDialog( this, new
DatePickerDialog.OnDateSetListener() {
 @Override
 public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
// Update TextView with the selected date
String selectedDate = dayOfMonth + "/" + (month + 1) + "/" + year;
tvSelectedDate.setText("Selected Date: " + selectedDate);
 } },
year, month, dayOfMonth );
datePickerDialog.show();
 }
}
```

Output:



Selected Date: 14/2/2024

RESULT:

Thus, the Android Application to implement Date Picker view is developed and executed successfully

Ex No:07 **DEMONSTRATE TIME PICKER**

Date:19.01.2024

Aim:

The implement a DatePicker view in an Android application using Android Studio.

Procedure:

1. Create new Android project or open existing project in Android Studio.
2. Design layout: Add TimPicker widget to layout XML, adjust attributes.
3. Handle TimePicker: Declare and initialize TimePicker object in MainActivity.
4. Retrieve selected date: Use getHourOftheDay(),getMinute() methods.
5. Handle selected time in app logic, such as displaying or setting reminders.
6. Optionally, customize TimePicker with attributes like hours,minutes.

Source Code:

Activity_main.xml:

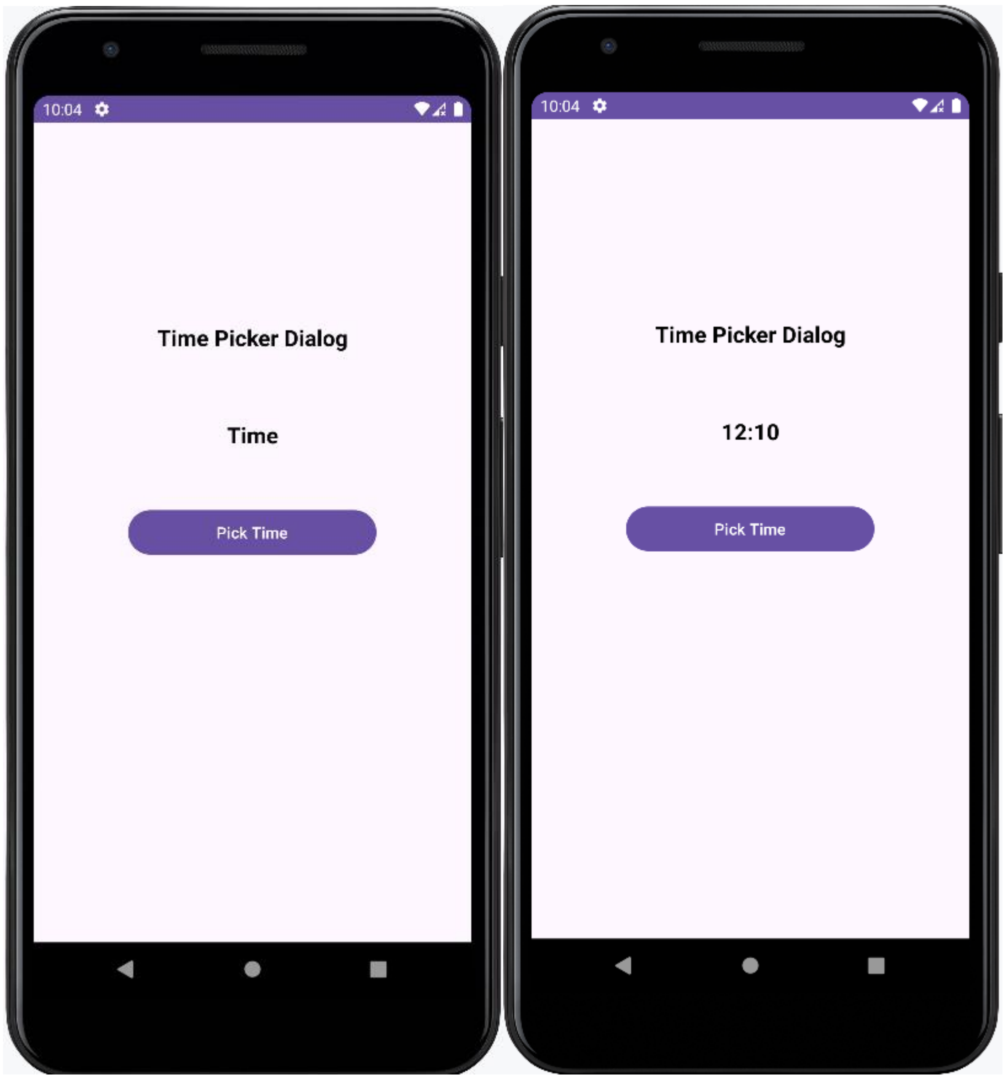
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/idRLContainer"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   tools:context=".MainActivity">

   <TextView
      android:id="@+id/idTVHeading"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_above="@id/idTVSelectedTime"
      android:layout_centerInParent="true"
      android:layout_margin="20dp"
      android:gravity="center"
      android:padding="10dp"
      android:text="Time Picker Dialog"
      android:textAlignment="center"
      android:textColor="@color/black"
      android:textSize="20sp"
      android:textStyle="bold" />

   <TextView
      android:id="@+id/idTVSelectedTime"
      android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_above="@id/idBtnPickTime"
        android:layout_centerInParent="true"
        android:layout_margin="20dp"
        android:gravity="center"
        android:padding="10dp"
        android:text="Time"
        android:textAlignment="center"
        android:textColor="@color/black"
        android:textSize="20sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/idBtnPickTime"
        android:layout_width="223dp"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:layout_marginStart="20dp"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="20dp"
        android:layout_marginEnd="20dp"
        android:layout_marginRight="20dp"
        android:layout_marginBottom="20dp"
        android:text="Pick Time"
        android:textAllCaps="false" />

</RelativeLayout>
```

MainActivity.java

```
package com.example.timepick;

import android.app.TimePickerDialog;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.TimePicker;
import androidx.appcompat.app.AppCompatActivity;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {

    private Button pickTimeBtn;
    private TextView selectedTimeTV;

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

        pickTimeBtn = findViewById(R.id.idBtnPickTime);
        selectedTimeTV = findViewById(R.id.idTVSelectedTime);

        pickTimeBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final Calendar c = Calendar.getInstance();

                int hour = c.get(Calendar.HOUR_OF_DAY);
                int minute = c.get(Calendar.MINUTE);

                TimePickerDialog timePickerDialog = new TimePickerDialog(MainActivity.this,
                        new TimePickerDialog.OnTimeSetListener() {
                            @Override
                            public void onTimeSet(TimePicker view, int hourOfDay,
                                                  int minute) {
                                selectedTimeTV.setText(hourOfDay + ":" + minute);
                            }
                        }, hour, minute, false);
                timePickerDialog.show();
            }
        });
    }
}
```

Output:



RESULT:

Thus, the Android Application to implement Time Picker view is developed and executed successfully

Ex No:08 Develop an application for event management using Picker view

Date:24.01.2024

Aim:

The aim of implementing a event management using Picker view in an Android application using Android Studio.

Procedure:

1. Create new Android project or open existing project in Android Studio

2.Design the UI by adding various basic views like TextView, EditText, Button, ImageView, and CheckBox.

3.Add event handling logic, such as setting an OnClickListener for the Button to handle its click event and showing a toast message.

4.Similarly, set an OnClickListener for the CheckBox to handle its click event and show a toast message accordingly.

5. Testing: Run your application and test the functionality to ensure that data is being stored, retrieved, updated, and deleted correctly.

Source Code:

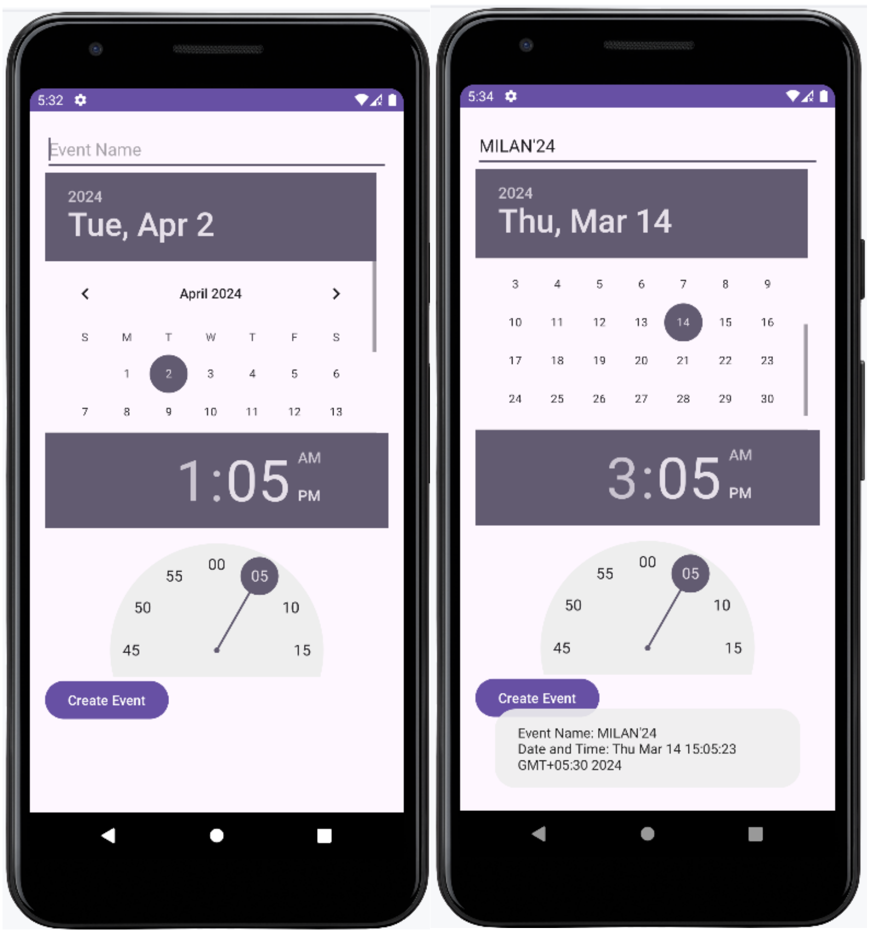
Activity_main.xml:

```
<!-- activity_main.xml -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/eventNameEditText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Event Name"
        android:minHeight="48dp" />

    <DatePicker
        android:id="@+id/datePicker"
        android:layout_width="wrap_content"
        android:layout_height="273dp" />

    <TimePicker
        android:id="@+id/timePicker"
        android:layout_width="wrap_content"
        android:layout_height="256dp" />

    <Button
        android:id="@+id/createEventButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Create Event" />

</LinearLayout>
```

MainActivity.java:

```
package com.example.events;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TimePicker;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.Calendar;

public class MainActivity extends AppCompatActivity {

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

        final EditText eventNameEditText = findViewById(R.id.eventNameEditText);
        final DatePicker datePicker = findViewById(R.id.datePicker);
        final TimePicker timePicker = findViewById(R.id.timePicker);
        Button createEventButton = findViewById(R.id.createEventButton);

        createEventButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Get event name
                String eventName = eventNameEditText.getText().toString();

                // Get selected date from DatePicker
                int year = datePicker.getYear();
                int month = datePicker.getMonth();
                int day = datePicker.getDayOfMonth();

                // Get selected time from TimePicker
                int hour = timePicker.getHour();
                int minute = timePicker.getMinute();

                // Create a Calendar object and set it to the selected date and time
                Calendar calendar = Calendar.getInstance();
                calendar.set(year, month, day, hour, minute);

                // You can now save this event data to a database, file, or any storage mechanism
                // For now, let's just display a toast message with the event details
                String eventDateTime = calendar.getTime().toString();
                Toast.makeText(MainActivity.this, "Event Name: " + eventName + "\nDate and
Time: " + eventDateTime, Toast.LENGTH_LONG).show();
            }
        });
    }
}
```

Output:



```
Event Name: MILAN'24
Date and Time: Thu Mar 14 15:05:23
GMT+05:30 2024
```

Result:

Thus, the Android Application to implement event handling is developed and executed successfully

Ex:no:09

DEMONSTRATE DATA PERSISTENT USING SHARED PREFERENCES

Date:08.02.2024

Aim:

Demonstrate data persistence using shared preferences in an Android application using Android Studio.

Procedure:

1. Initialize a New Android Project: Start by creating a new Android Studio project.
2. Design User Interface: Design the user interface (UI) if your application requires it. For shared preferences, UI design is often minimal as it's primarily used for storing user preferences.
3. Implement Shared Preferences: "my_preferences" is the name of the SharedPreferences file. MODE_PRIVATE ensures that only your application can access this file.
4. Write Data to SharedPreferences: You can store various types of data (string, int, boolean, etc.) using appropriate methods like putString(), putInt(), putBoolean(), etc.
5. Read Data from SharedPreferences: Provide a default value in case the key does not exist in SharedPreferences
6. Update Data in SharedPreferences: To update an existing value, simply call the corresponding put method with the new value and apply the changes.
7. Testing: Run your application and test the functionality to ensure that data is being stored, retrieved, updated, and deleted correctly.

Source Code:

Mainactivity1.java

```
package com.example.shar;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class MainActivity extends AppCompatActivity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      Button butActivity2 = findViewById(R.id.btnActivity1);
      butActivity2.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View view) {
            Intent ActivIntent = new
                  Intent(MainActivity.this, MainActivity2.class);
            startActivity(ActivIntent);
            }
        });
    }
}
```

MainActivity2.java

```
package com.example.shar;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity2 extends AppCompatActivity {

    private SharedPreferences sharedPreferences;
    private TextView textView;
    private EditText editText;

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

        textView = findViewById(R.id.textView);

        // Initialize SharedPreferences
        sharedPreferences = getSharedPreferences("MyPrefs", MODE_PRIVATE);

// Check if a value with key "message" exists
        if (sharedPreferences.contains("message")) {
            // If it exists, retrieve the value and display it
            String message = sharedPreferences.getString("message", "");
            textView.setText(message);
        } else {
            // If it doesn't exist, set a default value
            textView.setText("No message stored");
        }

        // Save a value to SharedPreferences
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString("message", "ANDROID-APP-II");
        editor.apply();
    }
}
```

Activitymain.xml:

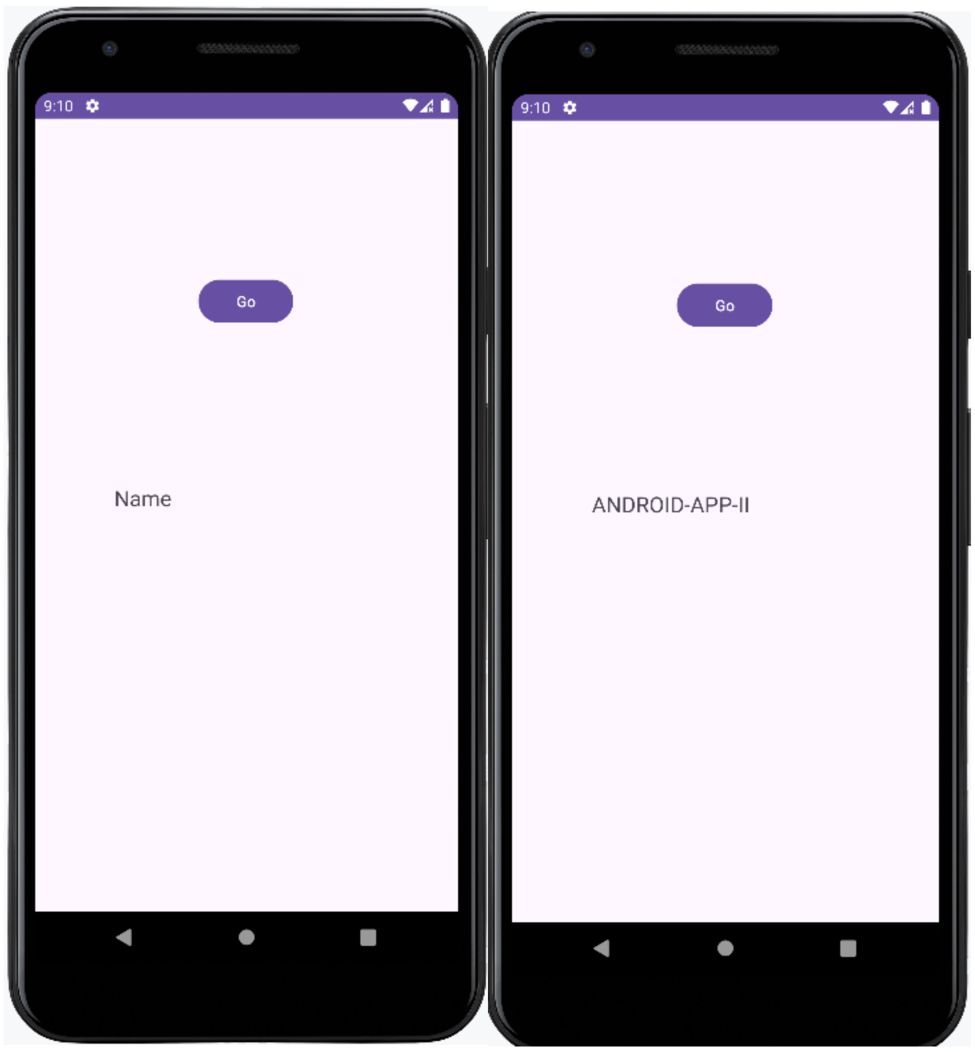
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
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

    <Button
        android:id="@+id/btnActivity1"
        android:layout_width="wrap_content"
        android:layout_height="48dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="32dp"
        android:layout_marginEnd="24dp"
        android:text="Go"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.512"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.173"
        tools:ignore="HardcodedText" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="245dp"
        android:layout_height="59dp"
        android:layout_marginStart="100dp"
        android:layout_marginEnd="100dp"
        android:text="Name"
        android:textSize="20sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.509"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="HardcodedText,TextSizeCheck" />

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

Activitymain2.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"
    tools:context=".MainActivity2">

    <TextView
        android:id="@+id/textView"
        android:layout_width="291dp"
        android:layout_height="56dp"
        android:layout_marginStart="100dp"
        android:layout_marginEnd="100dp"
        android:text="Name"
        android:textSize="20dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.35"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.407" />

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

Output:



RESULT:

Thus the Android Application to implement Shared preferences is developed and executed successfully

Ex:no:10

DEVELOP APPLICATION TO STORE AND RETRIEVE DATA USING FILES

Date:15.02.2024

Aim:

The aim of implementing a File storing in an Android application using Android Studio.

Procedure:

1.Initialize a New Android Project: Start by creating a new Android Studio project.

2.Design User Interface: Design the user interface (UI) if your application requires it. For shared preferences, UI design is often minimal as it's primarily used for storing user preferences.

3. Implement File Storage:Internal Storage:Files are stored in the app's private directory. Use `getFilesDir()` to get the path. External Storage Files are stored on the device's external storage. Use `getExternalFilesDir()` to get the path.

4. Shared Preferences: For creating files and directories, you can use classes like `FileOutputStream`, `FileInputStream`, `File`, `BufferedReader`, `BufferedWriter`, etc.

5.Testing: Run your application and test the functionality to ensure that data is being stored, retrieved, updated, and deleted correctly.

Source Code:

MainActivity.java

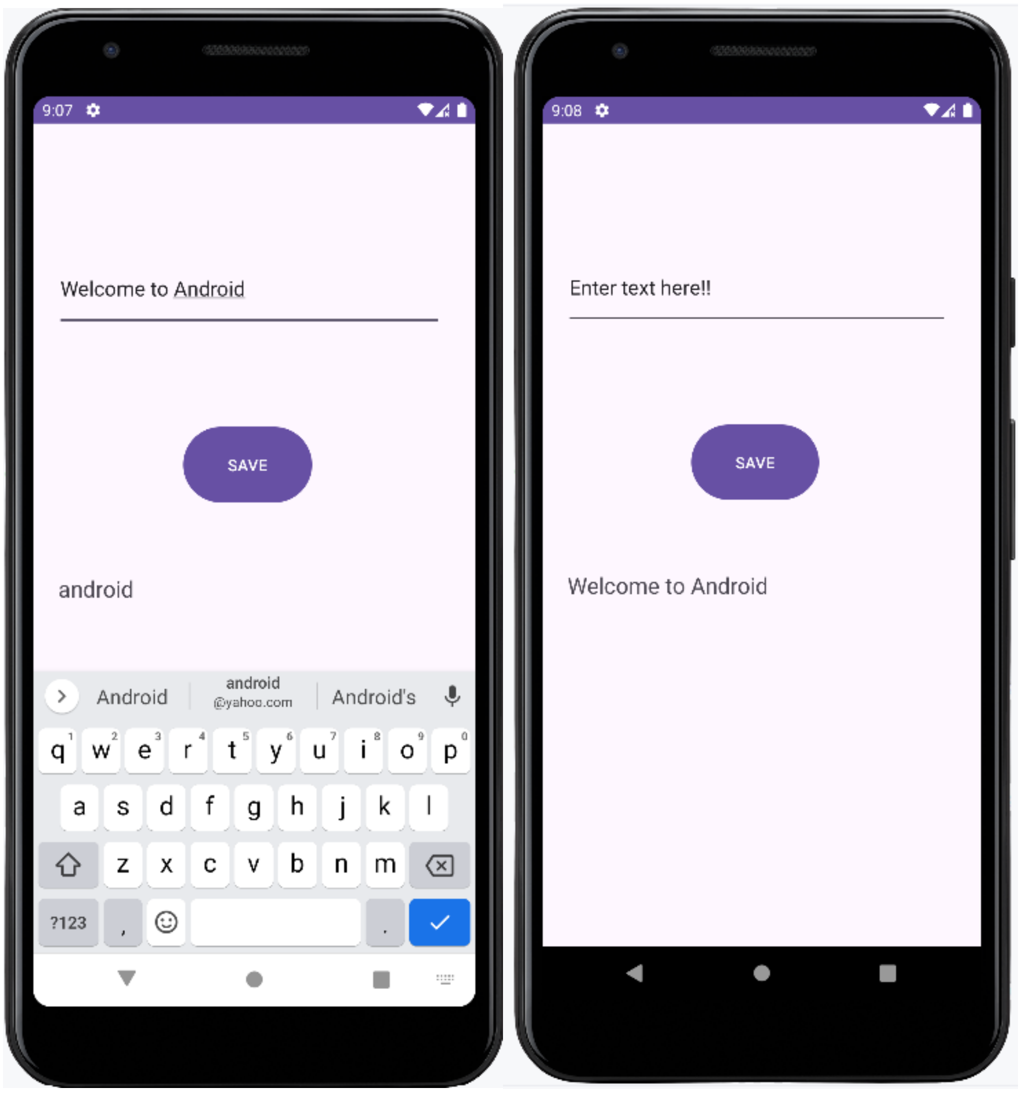
```
import android.content.Context;

import android.os.Bundle;

import android.support.v7.app.AppCompatActivity;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import java.io.BufferedReader;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.InputStreamReader;


public class MainActivity extends AppCompatActivity {
```

```
EditText editText;
TextView textView;
Button saveButton;
private static final String FILE_NAME = "example.txt";

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

    editText = findViewById(R.id.edit_text);
    textView = findViewById(R.id.text_view);
    saveButton = findViewById(R.id.save_button);

    saveButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            saveData();
        }
    });
    // Load data when the app starts
    loadData();
}
private void saveData() {
    String text = editText.getText().toString();
    FileOutputStream fos = null;
    try {
        fos = openFileOutput(FILE_NAME, Context.MODE_PRIVATE);
        fos.write(text.getBytes());
        editText.getText().clear();
        textView.setText("Data saved to file.");
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            if (fos != null)
                fos.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }    }
private void loadData() {
    FileInputStream fis = null;

    try {
        fis = openFileInput(FILE_NAME);
        InputStreamReader isr = new InputStreamReader(fis);
        BufferedReader br = new BufferedReader(isr);
        StringBuilder sb = new StringBuilder();
        String text;
        while ((text = br.readLine()) != null) {
            sb.append(text).append("\n");
        }
        textView.setText(sb.toString());
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            if (fis != null)
                fis.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".MainActivity">
    <EditText
        android:id="@+id/edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter text here"/>
    <Button
        android:id="@+id/save_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/edit_text"
        android:layout_marginTop="16dp"
        android:text="Save"/>
    <TextView
        android:id="@+id/text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/save_button"
        android:layout_marginTop="16dp"/>
</RelativeLayout>
```

Output:



RESULT:

Thus, the Android Application to implement Files is developed and executed successfully

Ex:no:11 IMPLEMENTATION OF SQLITE IN ANDROID APPLICATION

Date:19.02.2024

Aim:

To implement Sqlite in an Android application using Android Studio.

Procedure:

1. Initialize a New Android Project: Start by creating a new Android Studio project.
2. Design User Interface: Design the user interface (UI) if your application requires it. For shared preferences, UI design is often minimal as it's primarily used for storing user preferences.
3. Create a Database Helper Class: Create a class that extends `SQLiteOpenHelper`.Override `onCreate()` and `onUpgrade()` methods to create and upgrade your database schema
4. Define Database Schema: table names, column names, data types, constraints, etc.
5. Perform Database Operations: `SQLiteDatabase` object obtained from your `SQLiteOpenHelper` to perform CRUD (Create, Read, Update, Delete) operations.
6. Close Database Connections:Ensure that you close database connections (`SQLiteDatabase` objects) properly to release resources.

Source Code:

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".MainActivity">
    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Name" />

    <EditText
        android:id="@+id/editTextAge"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextName"
        android:layout_marginTop="16dp"
        android:hint="Enter Age"
        android:inputType="number" />
    <Button
        android:id="@+id/buttonAdd"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextAge"
        android:layout_marginTop="16dp"
        android:text="Add Student" />
    <ListView
        android:id="@+id/listViewStudents"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/buttonAdd"
        android:layout_marginTop="16dp" />
</RelativeLayout>
```

MainActivity.java

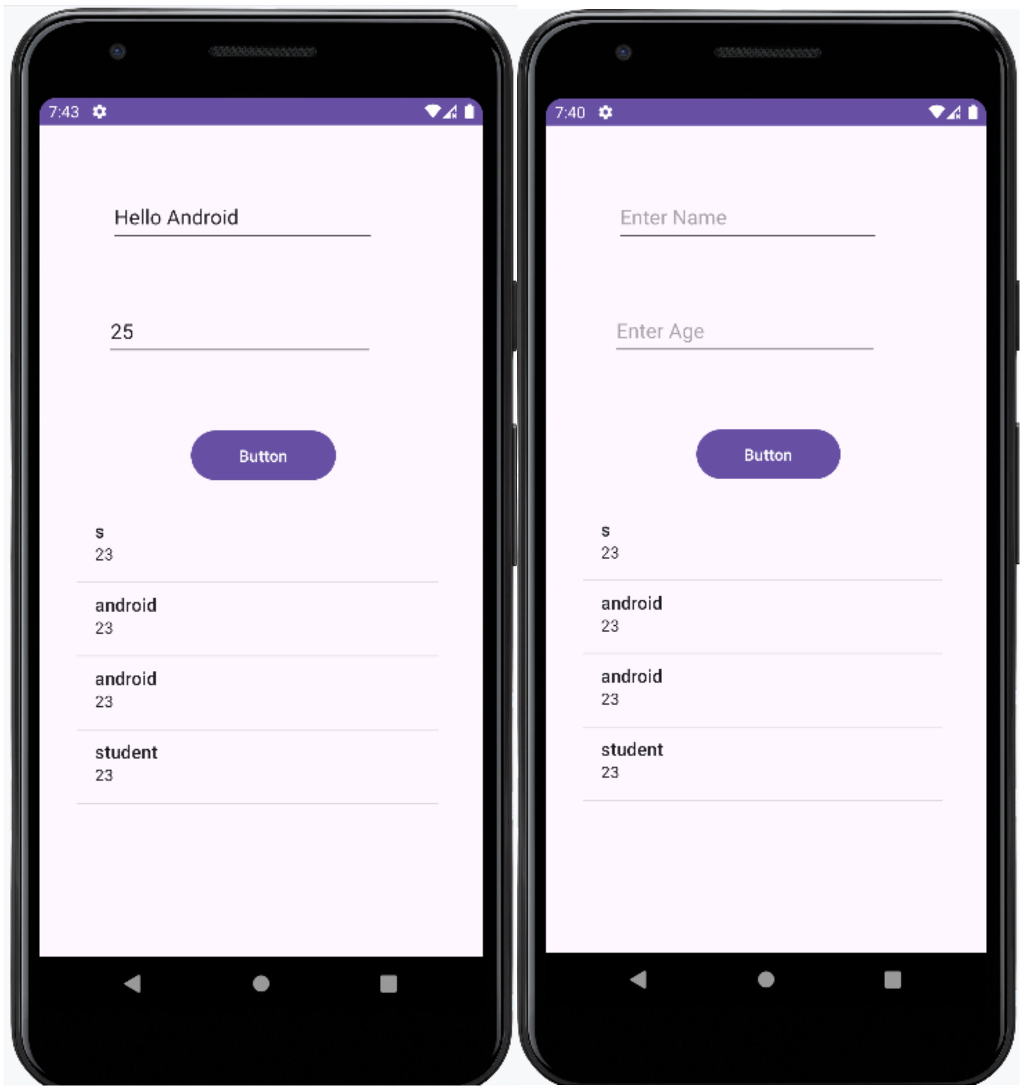
```
import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private EditText editTextName, editTextAge;
    private Button buttonAdd;
    private ListView listViewStudents;
    private DatabaseHelper dbHelper;

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

        editTextName = findViewById(R.id.editTextName);
        editTextAge = findViewById(R.id.editTextAge);
        buttonAdd = findViewById(R.id.buttonAdd);
        listViewStudents = findViewById(R.id.listViewStudents);
        dbHelper = new DatabaseHelper(this);

        buttonAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name = editTextName.getText().toString();
                String age = editTextAge.getText().toString();

                if (!name.isEmpty() && !age.isEmpty()) {
                    long result = dbHelper.insertData(name, Integer.parseInt(age));
                    if (result != -1) {
                        Toast.makeText(MainActivity.this, "Student added successfully",
Toast.LENGTH_SHORT).show();
                        displayStudents();
                        editTextName.setText("");
                        editTextAge.setText("");
                    } else {
                        Toast.makeText(MainActivity.this, "Failed to add student",
Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText(MainActivity.this, "Please enter name and age",
Toast.LENGTH_SHORT).show();
                }
            }
        })
        displayStudents();
    }

    private void displayStudents() {
        Cursor cursor = dbHelper.getAllData();
        String[] fromColumns = {DatabaseHelper.COL_NAME, DatabaseHelper.COL_AGE};
        int[] toViews = {android.R.id.text1, android.R.id.text2};
        SimpleCursorAdapter adapter = new SimpleCursorAdapter(this,
                android.R.layout.simple_list_item_2, cursor, fromColumns, toViews, 0);
        listViewStudents.setAdapter(adapter);
    }
}
```

DatabaseHelper.java

```
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
```

```
public class DatabaseHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "students.db";
    private static final int DATABASE_VERSION = 1;
    public static final String TABLE_NAME = "students";
    public static final String COL_ID = "_id";
    public static final String COL_NAME = "name";
    public static final String COL_AGE = "age";
    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        String createTableQuery = "CREATE TABLE " + TABLE_NAME + " (" +
                COL_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                COL_NAME + " TEXT, " +
                COL_AGE + " INTEGER)";
        db.execSQL(createTableQuery);
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }
    public long insertData(String name, int age) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_NAME, name);
        contentValues.put(COL_AGE, age);
        return db.insert(TABLE_NAME, null, contentValues);
    }
    public Cursor getAllData() {
        SQLiteDatabase db = this.getReadableDatabase();
        return db.query(TABLE_NAME, null, null, null, null, null, null);
    }
}
```

Output:



7:43

Hello Android

25

Button

s
23

android
23

android
23

student
23

7:40

Enter Name

Enter Age

Button

s
23

android
23

android
23

student
23

RESULT:

Thus, the Android Application to implement Sqlite is developed and executed successfully

Ex No:12 DEVELOP AN ANDROID APPLICATION FOR SMS MESSAGING

Date:22.02.2024

Aim:

To implement sms messaging in android application using Android studio.

Procedure:

1. Set Up Your Android Project: Create a new Android project in Android Studio.
2. Add Permission in Manifest File: Add the `<uses-permission android:name="android.permission.SEND_SMS"/>` permission to your `AndroidManifest.xml` file.
3. Check and Request SMS Permission at Runtime: In your activity or fragment, check if the SMS permission is granted. If not granted, request the permission from the user.
4. Handle Permission Request Result: Override `onRequestPermissionsResult()` to handle the permission request result.Proceed with sending SMS if permission is granted; otherwise, handle the denial gracefully.
5. Implement Sending SMS Functionality Write a method to send SMS using `SmsManager`.
6. Test Your Implementation: Test sending SMS functionality on devices and emulators.Verify permission requests and SMS sending behavior.

Source Code:

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/editTextPhone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Phone Number"/>

    <EditText
        android:id="@+id/editTextMessage"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextPhone"
        android:hint="Message"/>

    <Button
        android:id="@+id/buttonSend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/editTextMessage"
    android:text="Send"/>

</RelativeLayout>
```

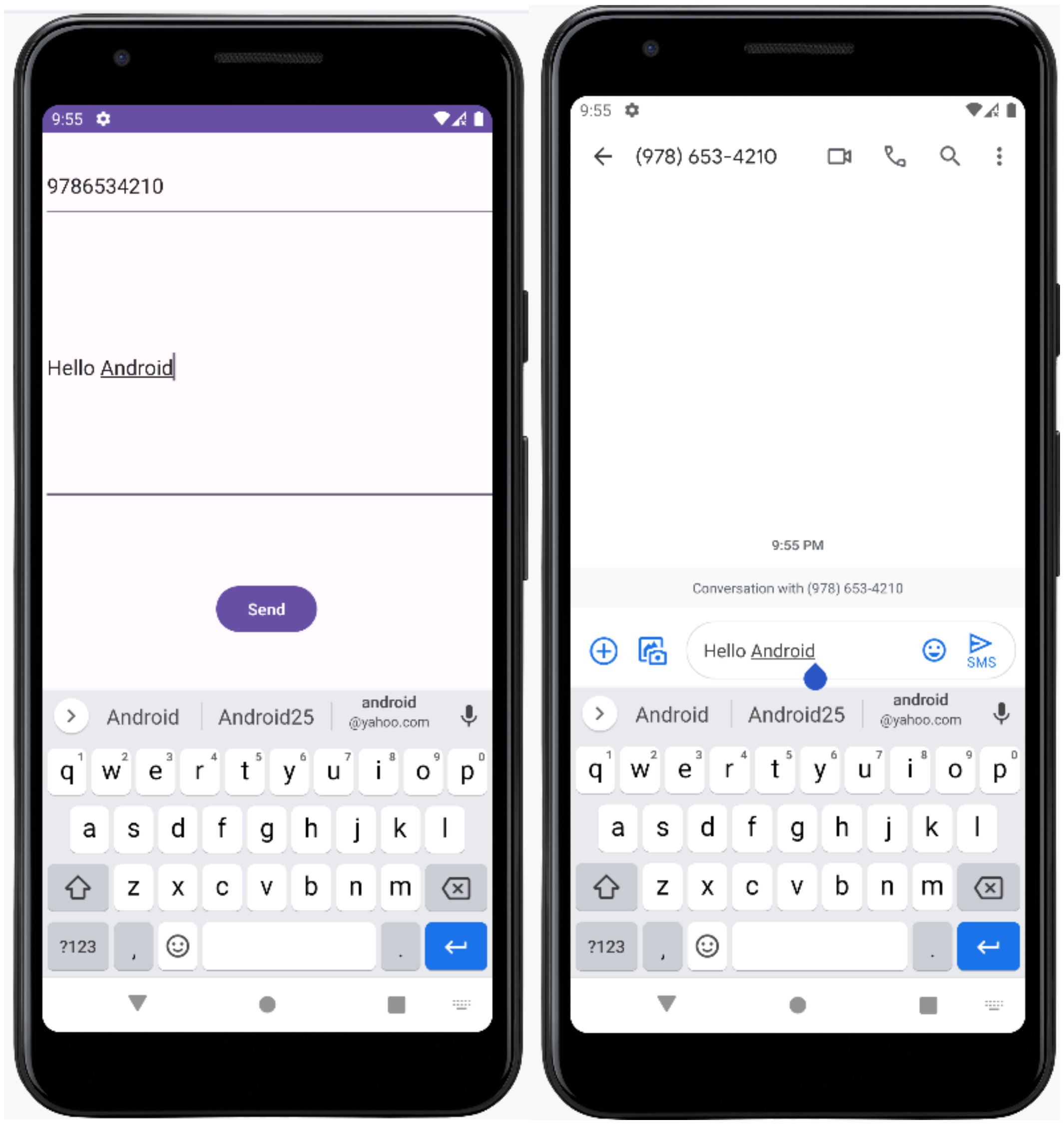
MainActivity.java

```
package com.example.myapplication;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

public class MainActivity extends AppCompatActivity {
    private EditText editTextPhone, editTextMessage;
    private Button buttonSend;
    private static final int PERMISSION_REQUEST_CODE = 1;

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

        editTextPhone = findViewById(R.id.editTextPhone);
        editTextMessage = findViewById(R.id.editTextMessage);
        buttonSend = findViewById(R.id.buttonSend);

        buttonSend.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (checkPermission()) {
                    sendSMS();
                } else {
                    requestPermission();
                }
            }
        });
    }

    private void requestPermission() {
        ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.SEND_SMS}, PERMISSION_REQUEST_CODE);
    }
    private boolean checkPermission() {
        int result = ContextCompat.checkSelfPermission(getApplicationContext(),
Manifest.permission.SEND_SMS);
        return result == PackageManager.PERMISSION_GRANTED;
    }

    private void sendSMS() {
        String phoneNo = editTextPhone.getText().toString();
        String message = editTextMessage.getText().toString();

        try {
            SmsManager smsManager = SmsManager.getDefault();
            smsManager.sendTextMessage(phoneNo, null, message, null, null);
            Toast.makeText(getApplicationContext(), "SMS sent successfully",
Toast.LENGTH_LONG).show();
        } catch (Exception ex) {
            Toast.makeText(getApplicationContext(), "SMS sending failed",
Toast.LENGTH_LONG).show();
            ex.printStackTrace();
        }
    }
    @Override
    public void onRequestPermissionsResult(int requestCode, String permissions[], int[]
grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        switch (requestCode) {
            case PERMISSION_REQUEST_CODE: {
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    sendSMS();
                } else {
                    Toast.makeText(getApplicationContext(), "Permission denied",
Toast.LENGTH_LONG).show();
                }
            }
        }
    }
}
```

AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-feature
        android:name="android.hardware.telephony"
        android:required="false" />
    <uses-permission android:name="android.permission.SEND_SMS">
    </uses-permission>
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Output:



9:55

9786534210

Hello Android

9:55

(978) 653-4210

9:55 PM

Conversation with (978) 653-4210

Hello Android

RESULT:

Thus, the Android Application to implement Sms messaging is developed and executed successfully

Ex No:13 DEMONSTRATE SMS MESSAGING FOR SENDER AND RECEIVER

Date:26.02.2024

Aim:

To Implement sms messaging in android application using Android studio.

Procedure:

1. Add Permissions to Your Manifest: Declare permissions in your `AndroidManifest.xml` file to send and receive SMS messages:
2. Create a BroadcastReceiver for Receiving SMS: Create a class that extends `BroadcastReceiver`.Override the `onReceive()` method to handle incoming SMS messages.
3. Handle SMS Sending:Use the `SmsManager` class to send SMS messages. Obtain an instance of `SmsManager` using `SmsManager.getDefault()`.
4. Handle SMS Delivery Reports: If you want to receive delivery reports for sent SMS messages, implement a `PendingIntent` to receive delivery reports and pass it to `sendTextMessage()`.
5. Display SMS in Your Application: If you want to display received SMS messages within your application, create an activity or fragment to handle this.
6. Use content providers like `Telephony.Sms.Inbox.CONTENT_URI` to query SMS messages stored in the device's SMS database.
7. Test Your SMS Messaging Functionality:Test sending and receiving SMS messages on various devices and Android versions.

Source Code:

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    android:padding="15sp"
    tools:context=".MainActivity">

    <EditText
        android:background="@android:drawable/editbox_background"
        android:inputType="phone"
        android:maxLength="10"
        android:padding="15sp"
        android:id="@+id/editTextPhone"
        android:hint="@string/enter_phone_number"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:autofillHints="" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editTextMessage"
        android:hint="@string/enter_message"
        android:padding="15sp"
        android:inputType="textMultiLine"
        android:lines="10"
        android:background="@android:drawable/editbox_background"
        android:autofillHints="" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/btnSent"
        android:text="@string/send_sms"
        android:textAllCaps="false"
        android:layout_marginTop="30dp"
        />
</LinearLayout>
```

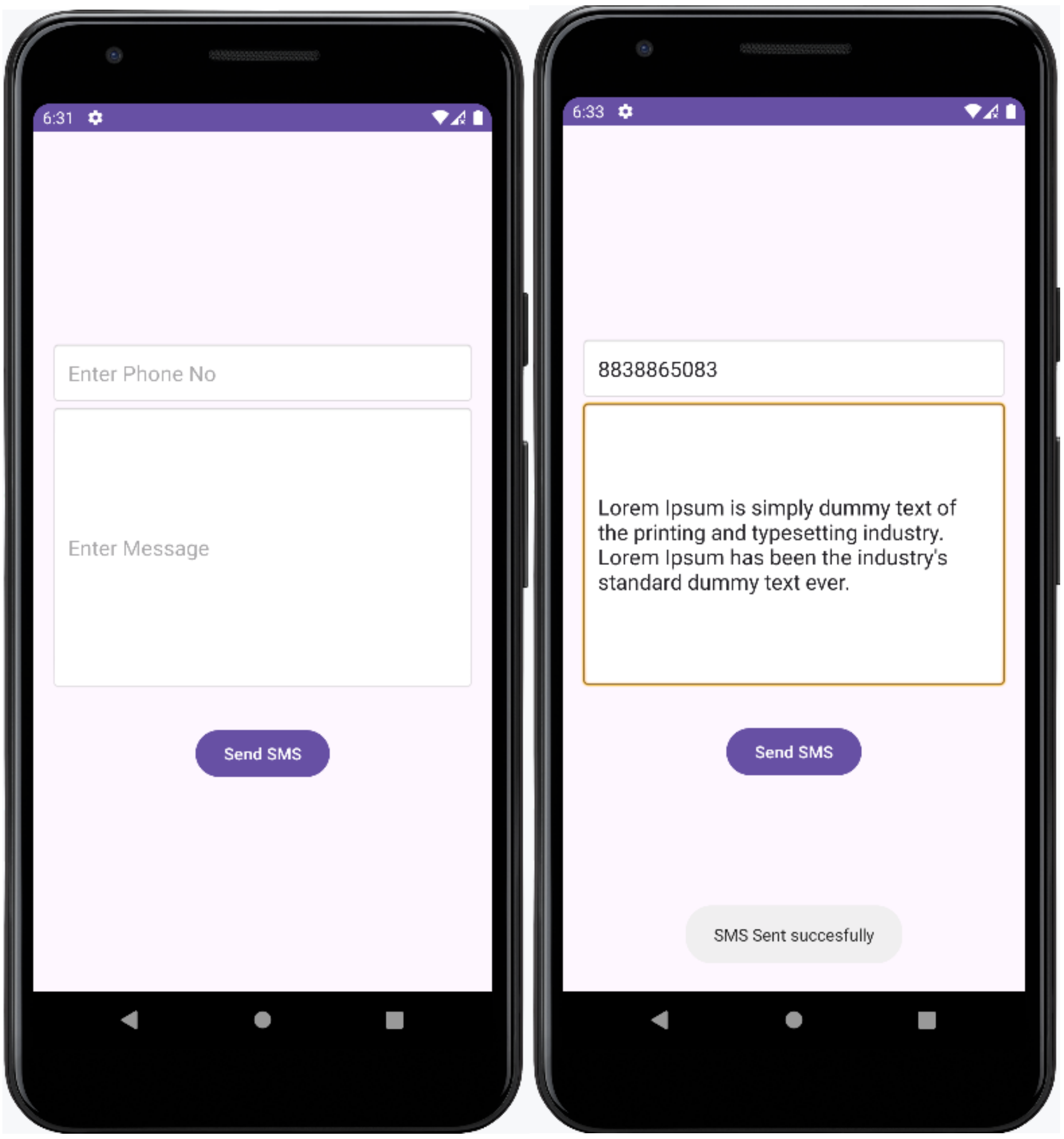
AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="co.ls.Messenger">
    <uses-permission android:name="android.permission.SEND_SMS"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Messenger"
        tools:targetApi="33">
        <activity
            android:name="com.example.messenger.MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

MainActivity.java

```
package com.example.messenger;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

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

public class MainActivity extends AppCompatActivity {
    //initialize variabe
    EditText editTextPhone,editTextMessage;
    Button btnSent;

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

        //assign variable
        editTextPhone = findViewById(R.id.editTextPhone);
        editTextMessage = findViewById(R.id.editTextMessage);
        btnSent = findViewById(R.id.btnSent);

        btnSent.setOnClickListener(v -> {
            //check condition for permission
            if
(ContextCompat.checkSelfPermission(MainActivity.this,Manifest.permission.SEND_SMS)
                    == PackageManager.PERMISSION_GRANTED) {
                //when permission is granted
                //create a method
                sendSMS();
            }else {
                //when permission is not granted
                //request for permission
                ActivityCompat.requestPermissions(MainActivity.this, new
String[]{Manifest.permission.SEND_SMS},
                        100);
            }
        });
    }
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        //check condition
        if (requestCode == 100 && grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
            //permission is granted
            //call method
            sendSMS();
        } else {
            //when permission is denied
            //display toast msg
            Toast.makeText(this, "Permission Denied", Toast.LENGTH_SHORT).show();
        }

    }

    private void sendSMS() {
            //get value from editText
            String phone=editTextPhone.getText().toString();
            String message=editTextMessage.getText().toString();

            //check condition if string is empty or not
            if (!phone.isEmpty() && !message.isEmpty()) {
                //initalize SMS Manager
                SmsManager smsManager = SmsManager.getDefault();
                //send message
                smsManager.sendTextMessage(phone,null,message,null,null);
                //display Toast msg
                Toast.makeText(this,"SMS Sent succesfully",Toast.LENGTH_SHORT).show();
            }else {
                //when string is empty then dispaly toast msg
                Toast.makeText(this,"Please Enter Phone
Number",Toast.LENGTH_SHORT).show();
            }

        }
    }
```

Output:



8838865083

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever.

SMS Sent succesfully

Result:

Thus, the Android Application to implement SMS messaging is developed and executed successfully

Ex No:14 DEVELOP AN APPLICATION USING VARIOUS BASIC VIEW

Date:04.03.2024

Aim:

The implement basic views in android application using Android studio.

Procedure:

1.Create a New Project: Open Android Studio and create a new project with an appropriate name and package

2.Create a New Project: Open Android Studio and create a new project with an appropriate name and package.

3.Add Views: Drag and drop views (like TextView, EditText, Button, etc.) from the palette onto the layout editor or directly add them in the XML code.

4.Connect Views with Java Code: Open the corresponding Java or Kotlin file for your activity (e.g., MainActivity.java or MainActivity.kt). Inside this file, you'll find a method named onCreate().

5.Run the App: After implementing the views and their functionality, run the app on an emulator or a physical device to see the output.

Source Code:

Activity_main.xml:

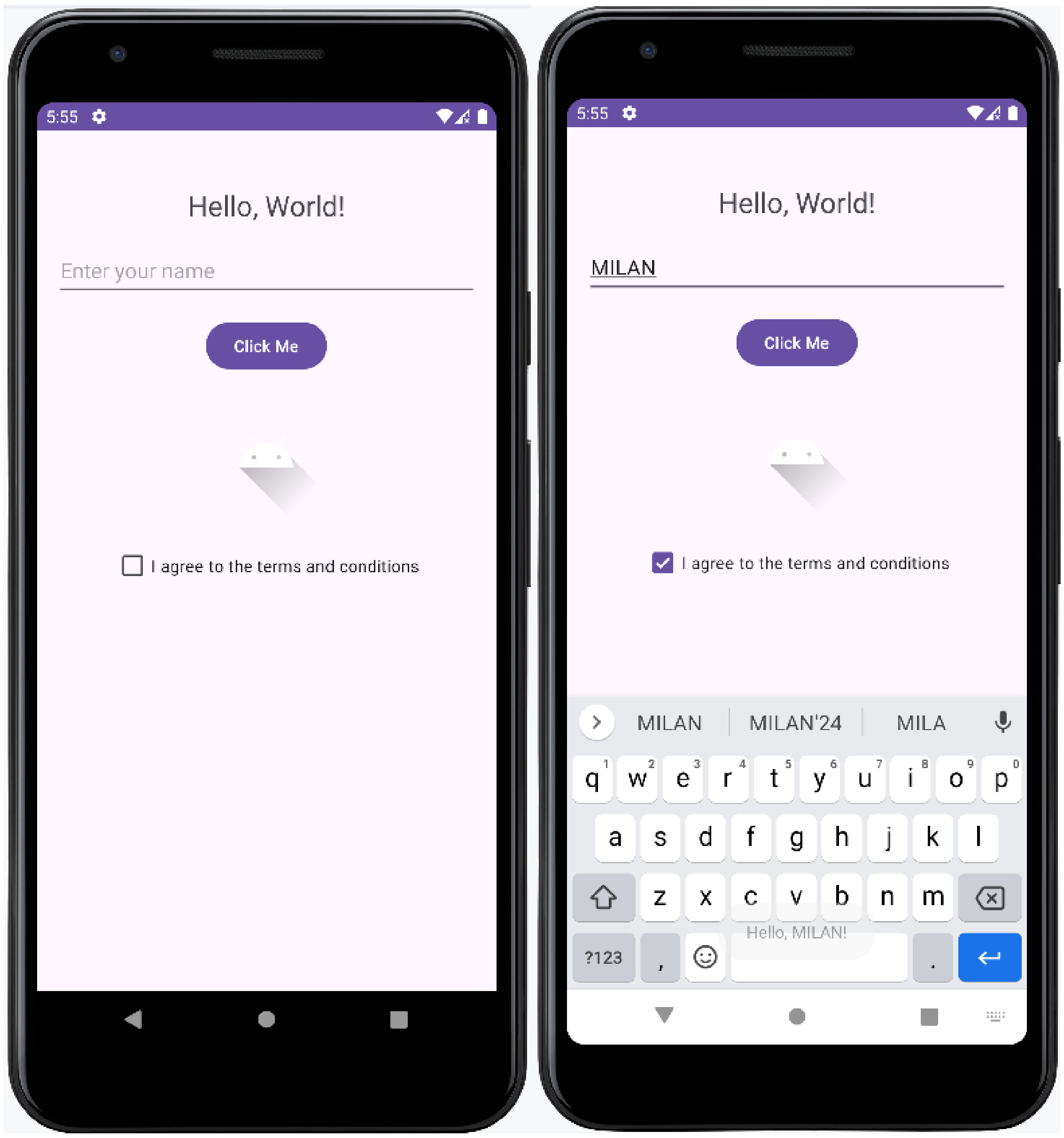
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:padding="16dp"
   tools:context=".MainActivity">

   <TextView
      android:id="@+id/textView"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Hello, World!"
      android:textSize="24sp"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="32dp"/>

   <EditText
      android:id="@+id/editText"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_below="@id/textView"
      android:layout_marginTop="16dp"
```

```
        android:hint="Enter your name"
        android:minHeight="48dp" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Click Me"
        android:layout_below="@id/editText"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"/>

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/ic_launcher_foreground"
        android:layout_below="@id/button"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"/>

    <CheckBox
        android:id="@+id/checkBox"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I agree to the terms and conditions"
        android:layout_below="@id/imageView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"/>

</RelativeLayout>
```

MainActivity.java:

```
package com.example.basicviewsapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final EditText editText = findViewById(R.id.editText);
        Button button = findViewById(R.id.button);
        final CheckBox checkBox = findViewById(R.id.checkBox);

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name = editText.getText().toString();
                if (!name.isEmpty()) {
                    Toast.makeText(MainActivity.this, "Hello, " + name + "!",
Toast.LENGTH_SHORT).show();
                } else {
                    Toast.makeText(MainActivity.this, "Please enter your name",
Toast.LENGTH_SHORT).show();
                }
            }
        });

        checkBox.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (checkBox.isChecked()) {
                    Toast.makeText(MainActivity.this, "Checkbox checked",
Toast.LENGTH_SHORT).show();
                } else {
                    Toast.makeText(MainActivity.this, "Checkbox unchecked",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Output:



Result:

Thus, the Android Application to implement basic views developed and executed successfully

Ex No:15 IMPLEMENTATION OF LIST VIEW AND SCROLL VIEW

Date:07.03.2024

Aim:

To implement the list view and scroll view in android application using Android Studio.

Procedure:

1.First, create a new Android project in Android Studio.

2.Then create activity_main.xml: Layout file for the main activity.

3.Also create item.xml: Layout file for the list item.

4.Then create MainActivity.java: Implementation of the main activity.you'll have a ScrollView containing a ListView with 20 items. Customize it as per your needs.

5. Test app on emulator or physical device to ensure scroll view and list view functionality.

Source Code:

Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="-5dp"
        android:layout_marginTop="140dp"
        android:layout_marginEnd="5dp"
        android:layout_marginBottom="-140dp">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">


            <ListView
                android:id="@+id/list_view"
                android:layout_width="match_parent"
                android:layout_height="285dp" />
        </LinearLayout>
    </ScrollView>


</RelativeLayout>
```

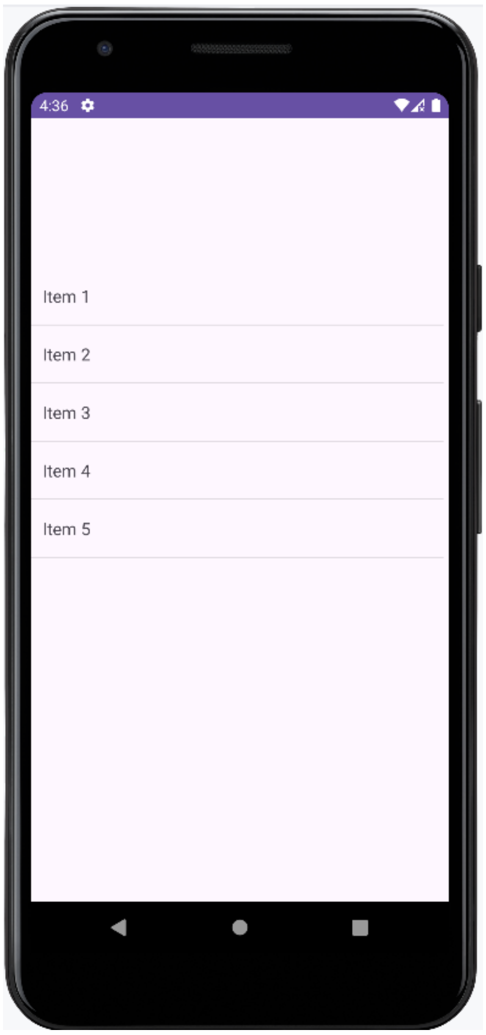
List_item.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/text_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="16dp"
    android:textSize="16sp" />
```

MainActivity.java:

```
package com.example.scrollv;

import android.app.Activity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class MainActivity extends Activity {

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

        ListView listView = findViewById(R.id.list_view);
        String[] data = new String[]{"Item 1", "Item 2", "Item 3", "Item 4", "Item 5", "Item 6",
"Item 7", "Item 8", "Item 9", "Item 10"};

        ArrayAdapter<String> adapter = new ArrayAdapter<>(this, R.layout.list_item,
R.id.text_view, data);
        listView.setAdapter(adapter);
    }
}
```

Output:



4:36

Item 1

Item 2

Item 3

Item 4

Item 5

Result:

Thus, the Android Application to implement Scroll view and list view is developed and executed successfully

Ex No:16 **IMPLEMENTATION OF EMAIL MESSAGING**

Date:18.03.2024

Aim:

To implement the email messaging in android application using Android Studio.

Procedure:

1. Create a new project in android studio.
2. Open activity_main.xml.
3. Add Three EditText to get the input from the user.
4. Add Textview to show the result and button to perform the action.
5. Open MainActivity.java.
6. Create a method as sendMail().
7. Declare the variable To, Subject, Compose Email and send button.
8. To Send the Email to many recipient the comma used as an seprator.
9. select the AVD and run the program

Source Code:

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/to"
        android:textAppearance="@style/TextAppearance.AppCompat.Large" />

    <EditText
        android:id="@+id/edit_text_to"
        android:layout_width="380dp"
        android:layout_height="50dp"
        android:inputType="textEmailAddress"
        android:autofillHints="emailAddress"
        tools:ignore="LabelFor,SpeakableTextPresentCheck"/>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/subject"
        android:textAppearance="@style/TextAppearance.AppCompat.Large" />

    <EditText
        android:id="@+id/edit_text_subject"
        android:layout_width="380dp"
        android:layout_height="50dp"
        android:inputType="textEmailSubject"
        android:autofillHints="emailAddress"
        tools:ignore="LabelFor,SpeakableTextPresentCheck"/>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/composeemail"
        android:textAppearance="@style/TextAppearance.AppCompat.Large" />

    <EditText
        android:id="@+id/edit_text_message"
        android:layout_width="380dp"
        android:layout_height="200dp"
        android:gravity="start|top"
        android:inputType="textLongMessage"
        android:autofillHints="emailAddress"
        android:lines="10"
        tools:ignore="LabelFor,SpeakableTextPresentCheck"/>

    <Button
        android:id="@+id/button_send"
        android:layout_width="374dp"
        android:layout_height="wrap_content"
        android:text="@string/SEND" />

</LinearLayout>
```

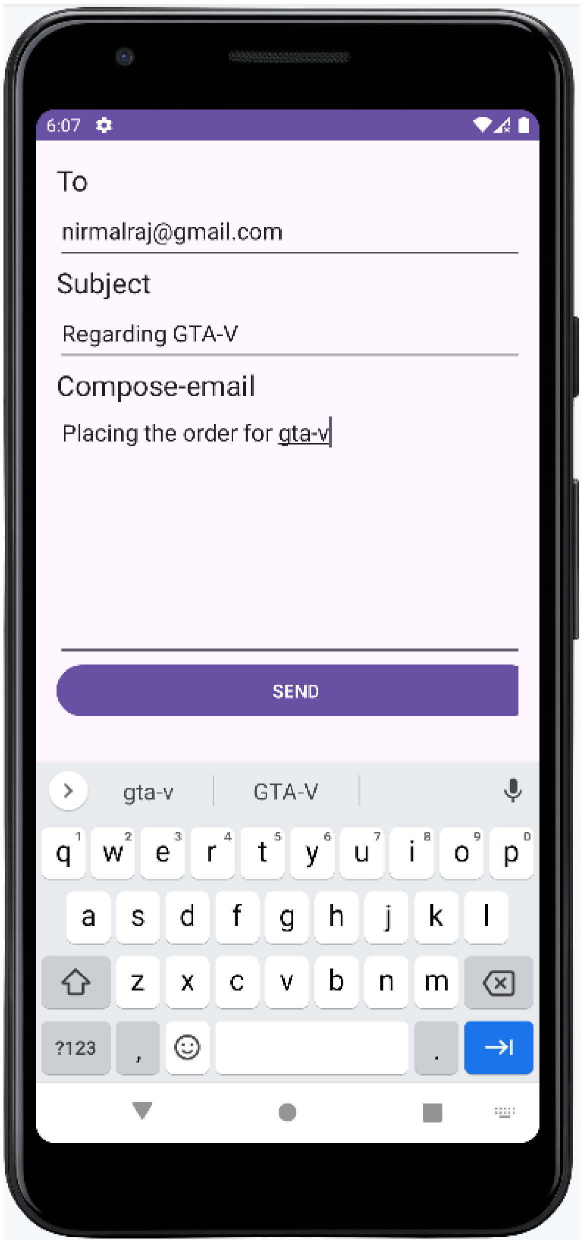
AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.EMail"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
    </application>

</manifest>
```

MainActivity.java

```
package com.example.email;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;


        import android.content.Intent;
        import android.widget.Button;
        import android.widget.EditText;


public class MainActivity extends AppCompatActivity {
    private EditText mEditTextTo;
    private EditText mEditTextSubject;
    private EditText mEditTextMessage;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mEditTextTo = findViewById(R.id.edit_text_to);
        mEditTextSubject = findViewById(R.id.edit_text_subject);
        mEditTextMessage = findViewById(R.id.edit_text_message);

        Button buttonSend = findViewById(R.id.button_send);
        buttonSend.setOnClickListener( v -> sendMail() );
    }

    private void sendMail() {
        String recipientList = mEditTextTo.getText().toString();
        String[] recipients = recipientList.split(",");

        String subject = mEditTextSubject.getText().toString();
        String message = mEditTextMessage.getText().toString();

        Intent intent = new Intent(Intent.ACTION_SEND);
        intent.putExtra(Intent.EXTRA_EMAIL, recipients);
        intent.putExtra(Intent.EXTRA_SUBJECT, subject);
        intent.putExtra(Intent.EXTRA_TEXT, message);

        intent.setType("message/rfc822");
        startActivity(Intent.createChooser(intent, "Choose an email client"));
    }
}
```

Output:


To

nirmalraj@gmail.com

Subject

Regarding GTA-V

Compose-email

Placing the order for gta-v

Result:

Thus, the Android Application to implement Email messaging is developed and executed successfully