

NAME: SAKSHI VINAYAK KITTURKAR  
CSU ID: 2860273  
HOMEWORK 2

Develop a contact management app version 1:

- The main activity would start with no contact, but it has a button where the user could use to add a new contact.
- On clicking an "Add Contact" button in the main activity, it goes to the second activity, where the user could add the contact information such as name, phone number, email, etc.
- After entering a new contact, the user could click the submit button in the second activity, which the second activity will be closed, and the app goes back to the main activity.
- The main activity now should display the new contact information.

= I have built an app where first screen shows you the contact information such as name, phone number, email but it is just blank to add data to it you must visit the second page where you can enter the data and it saves and take you back to first screen where you see the information has been updates.

In layout: Activity\_One.java this code demonstrates the use of the AndroidX library for handling results between activities. It allows Activity\_One to start Activity\_Two and receive data back from it to update its UI elements.

startForResult Launcher Activity\_One uses the AndroidX ActivityResultLauncher to launch Activity\_Two for a result. The launcher is registered using registerForActivityResult.

```
private final ActivityResultLauncher<Intent> startForResult = registerForActivityResult (new
ActivityResultContracts.StartActivityForResult(),
    new ActivityResultCallback<ActivityResult> () {
        @Override
        public void onActivityResult (ActivityResult result) {
            if (result.getResultCode () == Activity.RESULT_OK) {
                Bundle results = result.getData (). getExtras ();

                // Update TextView elements with data from Activity_Two
                ((TextView) findViewById(R.id.textViewName)). setText(results.getString("name"));
                ((TextView) findViewById(R.id.textViewPhone)).setText(results.getString("phone"));
                ((TextView) findViewById(R.id.textViewEmail)).setText(results.getString("email"));
            }
        }
    });
```

In layout: Activity\_Two.java is designed to capture user input and return the entered data to the calling activity. This structure allows for seamless communication and data exchange between different parts of the Android application.

Submit Button Click Listener the buttonSubmit is associated with a click listener that captures user input from EditText elements and creates an Intent to send the data back to the calling activity, Activity\_One.

```
buttonSubmit.setOnClickListener(view -> {  
    CharSequence userName = editTextName.getText();  
    CharSequence userPhone = editTextPhone.getText();  
    CharSequence userEmail = editTextEmail.getText();  
  
    Intent resultIntent = new Intent();  
    resultIntent.putExtra("name", userName.toString());  
    resultIntent.putExtra("phone", userPhone.toString());  
    resultIntent.putExtra("email", userEmail.toString());  
  
    setResult(Activity.RESULT_OK, resultIntent);  
    finish();  
});
```

In layout: backup\_rules.xml this file serves as a template for specifying rules for full backups in an Android app. Developers can customize it by uncommenting and modifying the include and exclude elements based on their app's backup requirements. The comments provide guidance on how to use the file and where to find additional information in the Android documentation. Additionally, it mentions that this file is ignored for devices older than API 31, and it provides a link for further information about backup and restore in Android.

```
``xml  
<?xml version="1.0" encoding="utf-8"?>  
<!--  
    Sample backup rules file; uncomment and customize as necessary.  
    See https://developer.android.com/guide/topics/data/autobackup  
    for details.  
    Note: This file is ignored for devices older than API 31  
    See https://developer.android.com/about/versions/12/backup-restore  
-->  
<full-backup-content>  
    <!--  
    <include domain="sharedpref" path="."/>
```

```
<exclude domain="sharedpref" path="device.xml"/>
-->
</full-backup-content>
```

In layout: data\_extraction\_rules.xml this XML file offers a starting point for developers to define data extraction rules, ensuring precise control over the backup and restore operations in their Android applications. The comments and documentation link provide guidance on how to tailor the rules to meet specific application requirements.

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  Sample data extraction rules file; uncomment and customize as necessary.
  See https://developer.android.com/about/versions/12/backup-restore#xml-changes
  for details.
-->
<data-extraction-rules>
  <cloud-backup>
    <!-- TODO: Use <include> and <exclude> to control what is backed up.
    <include .../>
    <exclude .../>
    -->
  </cloud-backup>
  <!--
  <device-transfer>
    <include .../>
    <exclude .../>
  </device-transfer>
  -->
</data-extraction-rules>
```

First screen with a button to add contact information.



1:59

Name:

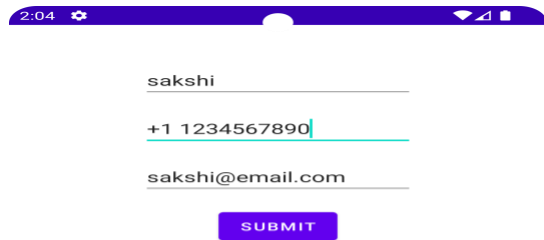
Phone:

Email:

+

A mobile app interface with a purple status bar at the top showing the time 1:59, a settings gear icon, and signal/battery icons. The main area is white and contains three labels: 'Name:', 'Phone:', and 'Email:'. Below these labels is a large, light blue circular button with a white plus sign inside, indicating an 'add' or 'submit' action.

Second screen to enter data.



2:04

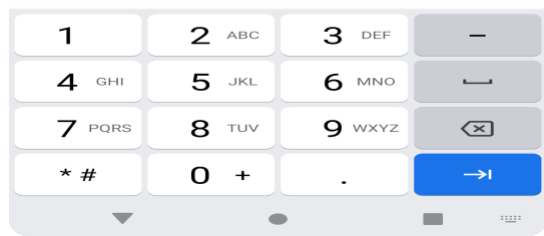
sakshi

+1 1234567890

sakshi@email.com

SUBMIT

A mobile app interface with a purple status bar at the top showing the time 2:04, a settings gear icon, and signal/battery icons. The main area is white and contains three text input fields. The first field contains 'sakshi', the second contains '+1 1234567890', and the third contains 'sakshi@email.com'. Below the input fields is a blue rectangular button with the word 'SUBMIT' in white capital letters.



A standard mobile numeric keypad with a light gray background. It consists of a 4x3 grid of buttons. The first three columns contain numbers 1 through 9, each with its corresponding letters (e.g., 2 has ABC, 3 has DEF). The fourth column contains a minus sign, a left arrow, a delete icon (X), and a right arrow. The bottom row contains buttons for asterisk/hash, 0 with a plus sign, a period, and a right arrow. The keypad is shown on a white background with a gray shadow.

Back to first screen with the updated contact information.



**Name:** sakshi

**Phone:** +1 1234567890

**Email:** sakshi@email.com

