Contacts List App

Activity Lifecycle and UI basics.

Create a **mini Android app** that displays a list of contacts (name + phone number), handle clicks, and includes an **app icon**.

Project: Simple Contacts List App

- ✓ **Project creation**
- ✓ **Layout (activity_main.xml)**
- ✓ **Code (MainActivity.kt)**
- ✓ **Custom icon setup**

Step 1: Create the Project

1. **Open Android Studio**
2. **File → New → New Project → Empty Activity**
3. **Name:** ContactsListApp
4. **Package name:** com.example.contactslistapp
5. **Save location:** (choose a folder)
6. **Language:** Kotlin
7. **Minimum SDK:** API 24 (Android 7.0) or above
8. Click **Finish**

Step 2: Design the UI
Open **res/layout/activity_main.xml** and replace everything with this:

```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:id="@+id/tvTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Contacts List"
        android:textSize="24sp"
        android:textStyle="bold"
        android:layout_gravity="center_horizontal"
        android:layout_marginBottom="10dp"/>
```

```
   <ListView
      android:id="@+id/listViewContacts"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"/>
</LinearLayout>
```

This layout includes:

- A title at the top
- A **ListView** to show the list of contacts

Step 3: Write the Kotlin Code
Now open **MainActivity.kt** and replace the content with:

```
package com.example.contactslistapp

import android.os.Bundle
import android.widget.ArrayAdapter
import android.widget.ListView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

   override fun onCreate(savedInstanceState: Bundle?) {
      super.onCreate(savedInstanceState)
      setContentView(R.layout.activity_main)

      val listViewContacts = findViewById<ListView>(R.id.listViewContacts)

      // Step 1: Create an array of contact names
      val contacts = arrayOf(
         "Rahul - 9876543210",
         "Priya - 9876501234",
         "Amit - 9988776655",
         "Sneha - 9876123456",
         "Kiran - 9090909090"
      )

      // Step 2: Create an adapter to connect data and ListView
      val adapter = ArrayAdapter(
         this,
         android.R.layout.simple_list_item_1,
         contacts
      )
```

```
        // Step 3: Assign adapter to the ListView
        listViewContacts.adapter = adapter

        // Step 4: Handle click on each list item
        listViewContacts.setOnItemClickListener { _, _, position, _ ->
            val selectedContact = contacts[position]
            Toast.makeText(this, "Selected: $selectedContact", Toast.LENGTH_SHORT).show()
        }
    }
}
```

Step 4: Note

Step	Description
ArrayAdapter	Converts the array into a format that can be displayed by the ListView
listViewContacts.adapter = adapter	Links the ListView with your contact data
setOnItemClickListener	Handles what happens when the user taps a contact
Toast	Displays a short popup message with the selected contact name

Step 5: Add an App Icon

1. Prepare an image (PNG, 512x512 recommended) — e.g., contact_icon.png.
   You can create one using free sites like https://icons8.com or https://flaticon.com.

2. In Android Studio:
   - **Right-click** res → **New → Image Asset**
   - **Icon Type:** Launcher Icons (Adaptive and Legacy)
   - **Path:** Select your contact_icon.png
   - Click **Next → Finish**

This automatically generates icons in all required sizes under mipmap folders.

3. To verify: open app/src/main/AndroidManifest.xml
   You'll see something like this inside the <application> tag:
4. android:icon="@mipmap/ic_launcher"
5. android:roundIcon="@mipmap/ic_launcher_round"
6. android:label="@string/app_name"   **(If not, add it manually.)**

Step 6: Run and Test

1. Click **Run** ▶
2. App displays a **list of contacts**
3. Tap any contact → a **Toast** message appears showing the selected name

Output: interactive list app with a custom icon.