PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE – 411043

Department of Electronics & Telecommunication Engineering

BATCH AND ROLL NO: S-7
EXPERIMENT NO.2
TITLE: Design a mobile application to create home page using grid layout.
DATE OF PERFORMANCE:
DATE OF SUBMISSION:

Title: Designing of mobile application to create home page using grid layout.

Requirements:
1 Android studio
2. Java SDK

Theory:

Introduction

In the realm of mobile application development, the design and layout of user interfaces play a pivotal role in creating a seamless and visually appealing user experience. The choice of layout managers is crucial for efficiently organizing and presenting content on the screen. One popular layout manager for achieving a structured and responsive layout is the Grid Layout.

Grid Layout: Grid Layout is a versatile layout manager that arranges UI components in a grid structure. This layout is particularly useful for creating home pages and dashboards in mobile applications, allowing developers to organize content in rows and columns. It provides a flexible and dynamic structure that adapts well to various screen sizes and orientations.

Objective of the Lab: The primary objective of this lab is to guide you through the process of designing a home page for a mobile application using the Grid Layout. You will learn how to efficiently organize and display content, such as images, text, and interactive elements, in a grid format. By the end of this lab, you should be adept at using the Grid Layout to create visually appealing and responsive home pages for your mobile applications.

Lab Prerequisites:

- Basic understanding of mobile application development concepts.
- Familiarity with the chosen development environment (e.g., Android Studio).
- Prior knowledge of programming languages such as Java (for Android)

Steps:
Designing a Home Page Using Grid Layout:

Step 1: Set Up Your Development Environment

- Ensure you have a suitable development environment installed, such as Android Studio for Android development.
- Create a new project or open an existing one.

Step 2: Understand Grid Layout Basics

- Familiarize yourself with the basic concepts of the Grid Layout, including rows, columns, and grid items.
- Explore how the Grid Layout adapts to different screen sizes and orientations.

Step 3: Create Grid Layout in XML

- Open the XML layout file (for Android).
- Define a Grid Layout container with a specified number of rows and columns.

Step 4: Add UI Elements as Grid Items

- Identify the content you want to display on the home page.
- Add UI elements (e.g., ImageView, TextView, Button) as grid items within the rows and columns of the Grid Layout.

Step 5: Customize Grid Items

- Customize the appearance of each grid item by adjusting properties such as size, padding, and margins.
- Consider using features like span to merge multiple rows or columns for specific elements.

Step 6: Handle Interactions and Navigation

- If applicable, implement interaction elements such as buttons or clickable components.
- Set up navigation or actions for grid items, allowing users to navigate to other pages or perform specific tasks.

Step 7: Test Responsiveness

- Test your home page layout on various devices and screen sizes to ensure responsiveness.
- Adjust layout parameters as needed to optimize the appearance on different devices.

Step 8: Implement Dynamic Data

- If your home page involves displaying dynamic content (e.g., images from a server, user-specific information), implement the necessary logic to fetch and populate the data within the Grid Layout.

Step 9: Test and Debug

- Test your home page thoroughly, including user interactions and data retrieval.
- Use debugging tools to identify and address any issues that may arise during testing.

Step 10: Iterate and Enhance

- Gather user feedback and iterate on the design based on usability and user experience.
- Consider enhancing the home page with animations, transitions, or additional features to make it more engaging.

XML Code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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">

  <GridLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:rowCount="3"
      android:columnCount="2">

      <androidx.cardview.widget.CardView
          android:layout_height="wrap_content"
          android:layout_width="wrap_content"
          android:layout_column="0"
          android:layout_row="0"
          android:layout_columnWeight="1"
```

```
         android:layout_rowWeight="1">

  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_gravity="center_vertical|center_horizontal">

    <ImageView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:src="@drawable/baseline_home_24"/>

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Home"/>
  </LinearLayout>
</androidx.cardview.widget.CardView>

<androidx.cardview.widget.CardView
  android:layout_height="wrap_content"
  android:layout_width="wrap_content"
  android:layout_column="1"
  android:layout_row="0"
  android:layout_columnWeight="1"
  android:layout_rowWeight="1">

  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_gravity="center_vertical|center_horizontal">

    <ImageView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:src="@drawable/baseline_contacts_24"/>

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Contacts"/>
  </LinearLayout>
</androidx.cardview.widget.CardView>

<androidx.cardview.widget.CardView
  android:layout_width="133dp"
  android:layout_height="wrap_content"
  android:layout_row="1"
  android:layout_rowWeight="1"
  android:layout_column="0"
  android:layout_columnWeight="1"
  android:background="@drawable/card_border">

  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="63dp"
    android:layout_gravity="center_vertical|center_horizontal"
    android:gravity="center"
    android:orientation="vertical">

    <ImageView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:src="@drawable/baseline_account_balance_24"  />
```

```
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Account"  />
    </LinearLayout>
</androidx.cardview.widget.CardView>

<androidx.cardview.widget.CardView
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:layout_column="1"
    android:layout_row="1"
    android:layout_columnWeight="1"
    android:layout_rowWeight="1">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:gravity="center"
        android:layout_gravity="center_vertical|center_horizontal">

        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/baseline_access_time_24"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Transaction  History"/>
    </LinearLayout>
</androidx.cardview.widget.CardView>
```

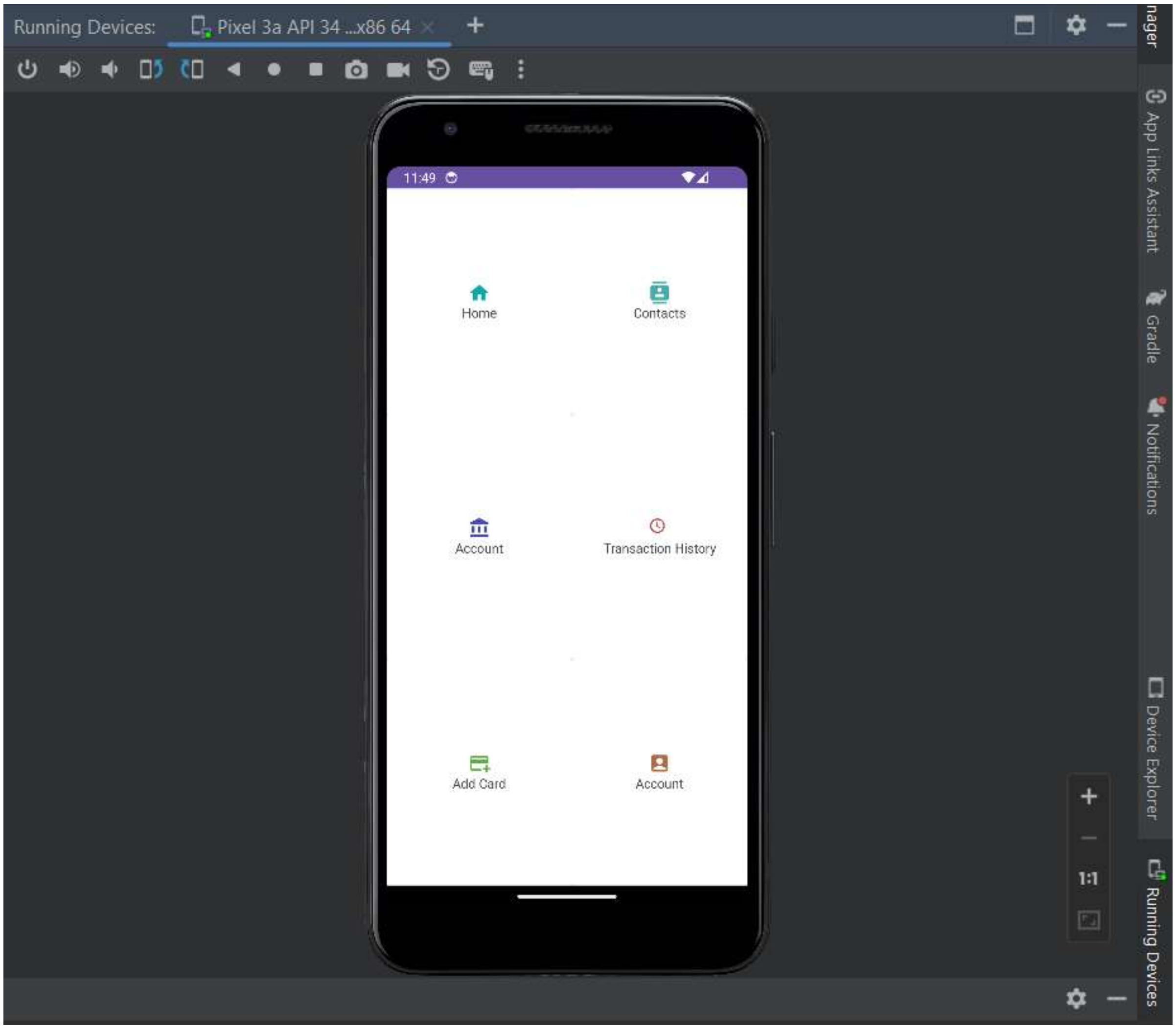
```
<androidx.cardview.widget.CardView
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:layout_column="0"
    android:layout_row="2"
    android:layout_columnWeight="1"
    android:layout_rowWeight="1">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:gravity="center"
        android:layout_gravity="center_vertical|center_horizontal">

        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/baseline_add_card_24"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Add Card"/>
    </LinearLayout>
</androidx.cardview.widget.CardView>

<androidx.cardview.widget.CardView
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:layout_column="1"
    android:layout_row="2"
    android:layout_columnWeight="1"
```

```
                      android:layout_rowWeight="1">

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:gravity="center"
            android:layout_gravity="center_vertical|center_horizontal">

            <ImageView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:src="@drawable/baseline_account_box_24"/>

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Account"/>
        </LinearLayout>
    </androidx.cardview.widget.CardView>

</GridLayout>


 </RelativeLayout>
```

Java Code:

```
package com.example.myapplication_gridlayout;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

Output:



Conclusion:

....

....

....