MEHSANA DISTRICT EDUCATION FOUNDATION SANCHALIT
U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA - 382 711 DIST. MEHSANA. (N.G.)

Name :- Meha Nileshkumar Bhatt

Enrollment no. :- 21012011049

Class :- CEIT - B

Batch :- 5B-1

Subject :- Mobile Application Development

Assignment :- 01

Assignment - 1

1.) Based on your understanding, identify a recent business trend that has influenced the Android platform. Explain how this trend impacts Android app developers and businesses in the mobile app industry.

Ans:- Some of the recent trends that has influenced the Android platform are :-

(i) The rise of foldable devices :- Foldable devices have been gaining popularity in recent years, and this trend is having a significant impact on the Android platform. The rise of foldable devices presents new opportunities for Android app developers. By optimizing their apps for foldable devices, developers can reach a new audience and create more immersive experience for their users.

Businesses in the mobile app industry can also benefit from the rise of foldable devices. For example, a business could develop a foldable-friendly e-commerce app that allows customers to browse products and make purchases more easily.

(ii.) The rise of mobile gaming :- Mobile gaming is now the largest segment of the video

game industry, and Android is the most pop mobile gaming platform. This trend is having significant impact on the development and distribution of Android apps.

2) What is the purpose of an Inflater of layou in Android development, and how does it fit into the architecture of Android layouts

Ans:- The purpose of an Inflater of layout in Android development is to convert an XML layout file into a View hierarchy. This mean that the Inflater takes the XML layout file and creates a corresponding View object for each element in the file. The View hierar is then displayed on the screen.

→ The Inflater is typically used by the Andro system to create the initial View hierarchy an Activity or Fragment. It can also be used by developers to create View hierarchi dynamically.

Here is a simple example of how to use Inflater to create a View hierarchy from an X layout file:

Code:-

```
// Get the LayoutInflater source
LayoutInflater inflater = getLayoutInflater();
// Inflate the XML layout file
View view = inflater.inflate(R.layout.activity_
                                              main,
```

02.

// set the view as the content of the Activity.
setContentView (view);

→ This code will inflate the layout file activity_main.xml and set the resulting View hierarchy as the content of the Activity.

3) Explain the concept of a CustomDialogBox in Android applications. Provide examples to illustrate its use.

Ans:- A CustomDialogBox in Android is a dialog box that is created by the developer which is used to display any type of content, such as text, images, buttons and other View objects. CustomDialogBoxes are often used to display information to the user, to collect input from the user, or to confirm a user action.

Examples of CustomDialogBoxes :-

→ A dialog box that asks the user to confirm that they want to delete a file.

→ A dialog box that displays a list of items and allows the user to select one.

→ A dialog box that allows the user to customize the app's settings.

Example :-

xml file code :-

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

    <TextView
        android:id="@+id/dialog_text"
        android:layout="wrap_content"
        android:layout="wrap_content"
        android:text="This is custom dialog."
        android:textSize="18sp" />

    <Button
        android:id="@+id/dialog_button"
        android:layout_width="wrap_conte
        android:layout_height="wrap_content
        android:text="close" />

</LinearLayout>
```

04.

MainActivity.kt :-

```
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import androidx.fragment.app.DialogFragment

class CustomDialogFragment : DialogFragment()
{
  override fun onCreateView (
    inflater: LayoutInflater, container: ViewGroup
                                          ?,
    savedInstanceState: Bundle?): View? {
      return inflater.inflate(R.layout.custom-
                              dialog, container, false)
  }

  override fun onViewCreated (view: View,
                              savedInstanceState: Bundle?){
    super.onViewCreated (view, savedInstanceState)
    dialog_text.text = "This is a custom dialog"
    dialog_button.setOnClickListener {
    dismiss()
    }
  }
}
```

4) How do activities, services and the Android Manifest file work together to make an Android app? Can you describe

their main roles and provide a basic exam
of how they cooperate to design a mobile
Ans:- Activities, services, and the Android Manife
file are the three main building blocks
an Android app.

Activities are the basic building blocks
the user interface (UI). They are responsible
for displaying a single screen and handlin
user interactions. Activities are typically
launched by users, but they can also
be launched by other activities or services

Services are background components that per
long-running operations or provide function
to other apps. Services do not have a UI,
they can continue to run even when the
user is not interacting with the app.

Android Manifest file is an XML file that
describes the components of an Android a
to the Android system. It also declares
the permissions that the app needs.

→ Here is an example of how activities,
services and the Android Manifest file
cooperate to design a mobile app:

(1.) The user taps on the app icon to
launch it.

(2.) The Android system looks up the app's

manifest file to determine which activity to launch first.

(3.) The Android system launches the specified activity.

(4.) The activity displays its UI and handles user interactions.

(5.) The activity may need to perform some long-running operation, such as downloading a file or uploading data to a server.

(6.) The activity can start a service to perform the long-running operation in the background.

(7.) The service performs the long-running operation and updates the activity with the results.

(8.) The activity updates its UI to reflect the results of the long-running operation.

5.) How does the Android Manifest file work together to make an Android app? Provide an example to demonstrate its significance.

Ans:- The Android Manifest file is a crucial component in the development of an Android application. It serves several important purposes and its content significantly impacts how the android system interacts and manage the app.
Significance of Android Manifest file are:

- App configuration
- Component declaration
- Permissions
- App lifecycle
- Intent filters

- Code :

```
<manifest xmlns: android = "http://schemas.android.com/apk/res/android"
    package = "com.example.myapplication" >

<application
    android : allowBackup = "true"
    android : icon = "@mipmap/ic_launcher"
    android : label = "@string/app_name"
    android : supportRtl = "true"
    android : theme = "@style/AppTheme" >

<activity android : name = ".MainActivity" >
<intent-filter>
<action android : name = "android.intent.action.
    MAIN" />
```

08

<category android:name="android.intent.category.launcher"/>

</intent-filter>
</activity>

<activity android:name=".secondActivity">
// declare additional activities here
</activity>

<user-permission android:name="android.permission.INTENT" />

</application>
</manifest>

6.) What is the role of resources in Android development? Discuss the various types of resources and their significance in creating well-structured applications. Provide examples to clarify your points.

Ans:- Resources play a fundamental role in Android development by providing a structured way to manage assets, values, layouts and other elements. The various types of resources and their significance are :-

(i.) Layout Resources :- xml files in 'res/layout'

directory. It defines the structure and appearance of user interface.

Example :-

xml file :-

```
<Button
    android : id = "@+id/button"
    android : layout_width = "wrap_content"
    android : layout_height = "wrap_content"
    android : text = "click me" />
```

(ii.) **Drawable resources** :- Images and drawable are in the drawable folder. It stores images, icons, graphics used in your app.

Example :- 'ic_launcher.png' is the app's launcher icon.

(iii.) **String resources** :- Strings defined in xml files present in 'res/values' directory. It stores text strings making it easier to provide translations and maintain consistency.

Example :-

```
<string name = "app_name"> My App </string>
<string name = "welcome"> Welcome </string>
```

(iv.) **Color Resources** :- Color defined in xml files under 'res/values'. It stores color values, ensuring consistency in the app's design.

Example :-

```
<color name = "primary_color"> #007ACC </color>
```

(v.) Style resources :- Styles defined in xml files under 'res/values'. It defines reusable styles for UI components.
Example :-
<Style name = "MyButtonStyle">
<item name = "android : background">
@drawable/my_button</item>
</Style>

(vi.) Dimension Resources :- Dimensions defined in xml files under 'res/values' directory. It stores dimension values ensuring a consistent layout.
Example :-
<dimen name = "margin_layout">16dp
</dimen>

(vii.) Raw resources :- Files such as JSON data, audio, video etc.
Example :- Store a JSON file for app configuration

7.) How does an Android service contribute to the functionality of a mobile application ? Describe the purpose process of developing an Android service.

Ans :- Android Service contribute to the functionality of mobile applications in a number of ways. They can be used to perform long-running operations in the

background, such as playing music, fetching data from the network or performing file. They can also be used to interact with other applications, such as providing data to a widget or handling notifications.

Examples :-

Music player apps :- A music player app might use a service to play music in the background, even when the user is not actively using the app.

Weather apps :- A weather app might use a service to fetch the lastest weather forecast in the background, so that the user can see it as soon as it changes.

You can use following methods :-

onStartCommand () :- When the service is started

OnBind () :- called when a client binds to the service

onUnbind () :- called when a client unbinds from the service

Once you have implemented these methods, you can register your service in the Android manifest file. You can then start the service by calling the startService () method from another component of the app, such as an activity or another service.

Code :-

```
class MyService : Service () {
    override fun onStartCommand (intent : Int
                                 flags: Int, startId : In
                                 Int {
```

```
    return Service.START_NOT_STICKY
}

override fun onBind(intent: Intent?): IBinder? {
    return null
}
}
```

→ To start this service, you would call the startService() method from another component of your app.

```
val intent = Intent(this, MyService::class.java)
startService(intent)
```

5/10/23