21012011093

GANPAT UNIVERSITY
॥ विद्यया समाजोत्कर्षः ॥
U V PATEL COLLEGE OF ENGINEERING

U. V. Patel College of Engineering

GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

Assignment : 1

Que-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.

→ Current Business trend: Subscription-Based Models on the Rise

- Impact on Android App Developers:
  - More steady Income: Apps with subscriptions can bring in a regular flow of money for developers, which is often more reliable than one time purchases.
  - Flexible App Design: Subscriptions allow developers to add more features and content to their apps without worrying about upfront costs.
  - Engaged users: Subscribed users tend to use the app more often since they're paying regularly.

- Example of successful Android subscription Apps:
  - Netflix
  - Amazon Prime Video
  - Hulu
  - spotify
  - YouTube Premium

→ The rise of subscription-based models has changed the game for Android. It offers developers a consistent income and keeps users engaged.

DEEP XEROX

Page No.

Que-2 What is the purpose of an Inflator or layout in Android development, and how does it fit into the architecture of Android layouts?

→ The purpose of an Inflater or layout in Android development is to convert XML layout files into view objects. This is done by parsing the XML file and creating the corresponding view objects, such as buttons, text views, and image views. The Inflater also handles setting the attributes of the view objects, such as their size, position, and text.

→ The Inflater is a key part of the Android layout architecture. It is responsible for creating the view hierarchy, which is the structure of all the view objects in an app. The view hierarchy is used by the Android system to render the app's UI.

→ how an Inflater fits into the architecture of Android layouts:

- Layout Files: These files are stored in the res/layout directory of an Android project.
- Activity: When an Activity needs to create or update its UI, it uses an Inflater to convert the layout file into view objects.
- Inflater: It parses the layout file and creates corresponding view objects, obtainable from the LayoutInflater class.
- View Hierarchy: The Inflater turns the design into a family tree of view objects, each representing UI elements.
- User Interface: This view hierarchy acts as the blueprint for the app's appearance and behavior, allowing developers to control and interact with the UI elements.

→ The Inflater is responsible for bridging the gap between layout files and the actual user interface of an Android app.

GANPAT UNIVERSITY
॥ विद्यया समाजोत्कर्षः ॥
U V PATEL COLLEGE OF ENGINEERING

U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

a) Intlater commonly used to:

1. Create the initial layout of an activity or fragment.
2. Generate dialog boxes and pop-up windows.
3. Dynamically inflate new layout views, such as when an adapter creates a new list item.

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

→ A customDialogBox in Android applications is a tailored dialog box that developers can create with their own design, layout, and functionality.

→ It is used for various purposes, such as displaying custom messages, gathering user input, offering choices, or showing complex layouts like forms and maps.

→ One common way to implement it is by using the AlertDialog class.

a) To create a CustomDialogBox, 'AlertDialog' classes.

→ AlertDialog is a simple way to create a custom dialog box. It allows you to specify a title, message, and buttons for the dialog box. You can also add a custom layout to the dialog box.

Ex My customAlerDialog.kt:

```
class MyCustomAlertDialog (context: context): AlertDialog (
context) {
```

```
private val titleView : TextView
private val messageView : TextView

init {
    val layout = layoutInflater.from(context).inflate(R.layout.
                    custom_dialog_layout, null)
    titleView = layout.findViewById(R.id.title_view)
    messageView = layout.findViewById(R.id.message_view)

    setContentView(layout)
}

fun setTitle(title:String){
    titleView.text = title
}

Fun setMessage(message:String){
    messageView.text = message
}}
```

•) activity_main.xml:

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

<TextView
    android:id="@+id/title_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="20sp"
    android:textStyle="bold" />

<TextView
    android:id="@+id/message_view"
```

U V PATEL COLLEGE OF ENGINEERING
U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

```
android:layout_width = "match_parent"
android:layout_height = "wrap_content"
android:textSize = "16sp" />
</LinearLayout>
```

⟹ To use the custom AlertDialog, you can create a new instance of the class and call the 'show()' method

o) Main Activity.kt

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

        val dialog = MyCustomAlertDialog (this)
        dialog.setTitle ("My Custom Alert Dialog")
        dialog.setMessage ("This is a custom alert dialog.")

        dialog.setPositiveButton ("OK") { dialog, -> dialog.dismiss()
        dialog.show()
    }
```

→ This code will show a custom AlertDialog with title "My custom Alert Dialog" and the message "This is a custom alert dialog". The dialog box will have a single "OK" button. When the user clicks the "OK" button, the dialog box will be dismissed.

DEEP XEROX

Que-4

How do activity, services, and the Android Manifest file work together to make an Android app? Can you describe their main roles and provide a basic example of how they cooperate to design a mobile app?

→ Activities, services, and the Android Manifest file are the three pillars of Android app development:

- Activities: These handle what you see on your screen and your interactions. They are like different pages of a book.
- Services: They work behind the scenes, doing tasks like downloading files or playing music without needing a visible interface.
- Android Manifest File: It's like an app's ID card, telling Android what the app contains and its permissions.

Ex :- Imagine a weather app. When you open it, the main screen (activity) shows today's weather. If you want more details, another screen (another activity) appears. Meanwhile, a service keeps the weather info up-to-date in the background.

2. How activities, services, and the Android Manifest file cooperate to design a mobile app:

1. The user launches the app. The Android system reads the app's manifest file and determines which activity to start.
2. The activity is created and displayed to the user. The activity can then interact with the user and display other activities as needed.
3. If the activity needs to perform a long-running background operation, it can start a service. The service will run in the background, even if the user leaves the app or turns off the screen.

GANPAT UNIVERSITY
U V PATEL COLLEGE OF ENGINEERING
U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

4) When the service is finished with its task, it can notify the activity or send it a broadcast message.
5) The activity can then take appropriate action, such as updating its user interface or displaying a notification to the user.

5) How does the Android Manifest file impact the development of an Android application? Provide an example to demonstrate its significance.

→ Certainly, here's a shorter yet unchanged version:

- How the Android Manifest file impacts the development of an Android application:
- The Android Manifest file is crucial in Android app development. It informs the Android system about your app's components, permissions, hardware/software needs, and metadata, without it, your app can't function.

Ex :- Manifest file:-

```
<manifest xmlns:android = "http://schemas.android.com/
                          apk/res/android">
  <application android:name = ".MyApp">
    <activity android:name = ".MainActivity">
      <intent-filter>
        <action android:name = "android.intent.action.MAIN" />
        <category android:name = "android.intent.category.LAUNCHER" />
      </intent-filter>
```

DEEP XEROX

</activity>
<uses-permission android:name=".android.permission.INTERNET
<uses-sdk android:minSdkVersion="21" android:targetSdk
Version="33" />
</application>
</manifest>

→ This file defines the main activity, INTERNET permission, and minimum / target SDK versions.

- Significance:

→ The Android Manifest file's importance lies in its ability to control how apps interact with Android systems and other apps. It ensures security, compatibility, and performance.

Que-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 example to clarify your points.

•) Role of resources in Android development:

→ Resources in Android development are vital for separating code and data, adapting to diverse device configurations, and supporting multilingual apps.

•) Types of resources:

1) Drawable: Images and UI graphics
2) Layouts: Screen structure definitions
3) strings: Text content for UI.
4) styles: UI appearance definitions
5) Colors: Color schemes.
6) Menus: Menu structure definitions.
7) Animations: Animation instructions.
8) Raw resources: Unprocessed binary data.

GANPAT UNIVERSITY
॥ विद्यया समाजोत्कर्षः ॥
U V PATEL COLLEGE OF ENGINEERING

U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

1) The significance of using resources in Android development includes:

1. Maintainability: Resources separate content from code, making updates and maintenance easier without changing the core code.
2. Adaptability: Resources enable apps to adjust to different device configurations, enhancing the user experience on various screens and languages.
3. Localization: String resources simplify app translation, allowing it to be available in multiple languages without altering the code.
4. Consistency: Resource-based styles and colors ensure a consistent visual design, enhancing the app's overall look and feel.
5. Efficiency: Resources are compiled efficiently, improving app runtime performance and reducing memory usage.

Ex

2) strings:

```
<string name = "greeting_morning"> Good morning ! </string>
<string name = "greeting_afternoon"> Good afternoon ! </string>
<string name = "greeting_evening"> Good evening ! </string>
```

3) activity_main:

```
<TextView
    android:id = "@+id/textview"
    android:layout_width = "wrap_content"
    android:layout_height = "wrap_content"
    android:text = "@string/greeting" />
```

DEEP XEROX
Page No.

•) MainActivity.kt.

```
class MainActivity : AppCompatActivity() {
    private lateinit var textView : TextView

    override fun onCreate(savedInstanceState : Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        textView = findViewById(R.id.textView)
        val hourOfDay = calendar.getInstance().get(
                        Calendar.HOUR_OF_DAY)
        val greeting = when {
            hourOfDay < 12 → getString(R.string.greeting_morning)
            hourOfDay < 18 → getString(R.string.greeting_afternoon)
            else → getString(R.string.greeting_evening)
        }
        textView.text = greeting
    }
}
```

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

→ An Android service is a component that runs in the background to perform long-running operations or to provide functionality for other applications. services can perform many kinds of tasks, such as:

- Playing music in the background
- Downloading files
- uploading data to a server
- syncing data with a cloud service
- performing background processing
- communicating with other devices.

GANPAT UNIVERSITY
U V PATEL COLLEGE OF ENGINEERING
U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

Process of developing an Android service, you must implement the following method:

→ onStartCommand():
This method is called when the service is started. It should contain the code that you want your service to execute.

→ OnBind():
This method is called when a client binds to the service. It should return an IBinder object that the client can use to interact with the service.

→ OnUnBind():
This method is called when a client unbinds from the service.

→ You can also implement other methods in your service subclass, such as onCreate(), OnDestroy(), and onTaskRemoved().

- To start your service, you can use the startService() method.
- To bind to your service, you can use the bindService() method.

→ Android services contribute to the functionality of mobile applications in a number of ways.

→ For example, a music player app may use a service to play music in the background while the user is using other apps or even when the screen is off. A photo editing app may use a service to upload edited photos to a cloud storage service.

DEEP XEROX

Page No.

~~★~~ alarm service: startV, stopV, DestroyV

Ex:
```
if (intent != null)
    player = MediaPlayer.create(this, R.raw.alarm)
    Player.start()

    return START_STICKY
}

override fun onDestroy() {
    Player.stop()
    super.onDestroy()
}
```

5/10/23