Ganpat University
॥ विद्यया समाजोत्कर्षः ॥
U.V. Patel College of Engineering
MEHSANA DISTRICT EDUCATION FOUNDATION SANCHALIT
U. V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA - 384 012 DIST. MEHSANA. (N.G.)

Assignment - 01 (MAD)

Q-1 Based on your own understanding. Identify a recent business trend that has influenced the other android platform. Explain how this trend impacts android app developers and business in the mobile app industry.

→ One significant trend in the android app industry was the increasing emphasis on user privacy & data security.

⇒ Impact On App developers:

1. Enhanced permission and consent:

→ Developers had to be more transparent about the data their apps collects and request explicit user content. This meant redesigning permi-ssion dialogues & ensuring that user understood why certain data was being collected

2. Limitations on Advertising:

→ For Apps relying on advertising revenue, changed in ad tracking due to privacy concerns affected their monetization strategies. Developers needed to adapt these changes, possibly exploring automatic monetization models

3. Data Handling & Storage:

→ Developers had to review how they handled and store user data, implementing data protection measures. This could lead to increased develop-ment time & costs.

→ Impacts On Business:

1. Competence cost:
→ Business operating in the android app industry needed to allocate resources for competence with stricter data privacy regulations. This could include legal and technical measures to ensure data protection

2. Monetization Changes:
→ Businesses relying on the user data for advertising and personalized content to maintain their revenue streams. They needed to find new ways to perform user and generalize income.

3 Reputation Management:
→ Privacy breaches or mishandling of user data could result in revenue reputations damage. Building & maintaining trust with users become even more critical.

Q-2 What's purpose of an inflator or layout in Android development and how does it fit into the architecture of Android layouts?

→

In Android App Development, think of the 'inflator' like a magical toolkit, it helps to turn your design plans to actual buttons, text boxes & other things you see on your phone screen

→ Purpose of layout inflator

1) Dynamic UI inflation: layout inflator is used to create instances of android view objects from xml layout resource files at runtime

2) Reusability: It promotes the reusability of UI components by defining their structure and appearance in xml layout files, making it easier to instantiate & populate them in different parts of an App

3) Separation of Concerns: layout inflator helps maintain clear separation betn the UI design & the code that manipulates and interacts with these UI elements.

→ Architecture of Android layout

1) xml layout files: Developers design the layout & structure of UI elements in xml layout resource file.

2) Activity / Fragement: In Java or Kotlin code of an android activity or fragment developers use the layout inflator to "inflate" or parse the xml layout files creating a hierarchy of view objects. This is typically done within "onCreate()" method

3) View Hierarchy: The result of inflating the layout xml is used as hierarchy of view objects, with the root view being top level layout

4) Data Binding & Event Handling: Developers often bind data to these views using data binding libraries or handle user interactions by attaching event listener

5) Rendering on the screen: The android system is responsible for rendering this hierarchy of the views on the device screen according to the layout specification defined in xml file

Q-3 Explain the concept of custom Dialog Box in Android application provide examples to illustrate its use

→ A custom Dialog Box in Android applications is a pop-up window that developers can design and customise to show information, receive input from user or perform action without navigating to a new screen. Custom Dialog Box allows developers to appealing manner

1) Design flexibility: Custom Dialog Box is used to create unique and tailored user interface

2) Contextual Use: They are typically used when they want to capture user input or show info without taking the user to different screen

3) User interaction: Custom Dialog Boxes can contain buttons, textfields, checkboxes or any other UI element, allowing user to interact with the content inside the dialog.

→ Examples of Custom Dialog Box User

1) Confirmation Dialog: A common use case is asking for confirmation before performing a critical action.

2) Login or Registration Dialog: Instead of navigating to a separate screen for login or registration, a custom dialog box can pop up, prompting the user to enter their credentials.

3) Error Message: When there's an error, such as network issues or invalid input, a custom dialog can display an error message with details helping the user understand & correct the problem.

4) Date & time Picker: You can create a custom dialog box for selecting dates or time, providing a more user friendly way to input this information.

code :

```
import android.app.AlertDialog
import android.content.DialogInterface
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity

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

        val builder = AlertDialog.Builder(this)
        builder.setTitle("Custom Dialog example")
        builder.setMessages("This is a custom dialog
                             boxes")
        builder.setPositiveButton("OK") { dialog, which -> }
        val dialog = builder.create()
        dialog.show()
    }
}
```

Q-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 example how they cooperate to design an mobile app?

→ Activities, services and android manifest file are essential components in the android app architecture each with distinct role that contribute to the functionality and behaviour of an app

1) Activities :-

→ Role : Activities represent the user inter-face and screen of an Android app they handle user interactions display all elements and manage the UI flow Provides essential information about the app to the Android system it declares the app's components permission, and other settings.

→ Example : In the manifest file, you define which activities are part of your app Specify permissions and declare services your app uses.

→ How they Cooperate :

1) Activities

→ The app starts with an activity showing a list of notes.

→ When the user taps on a note, another activity opens to display and edit the note's content

→ User can navigate betⁿ activities using buttons or gestures

2) Services

→ While the user is using the app, a service runs in the background to Periodically Solve the user's notes to cloud storage

3) Android Manifest file :

→ In the manifest file, you declare the activities & services used in your app

→ you specify permission like "INTERNET" to allow the app to access the internet for backups

→ The manifest file also defines which activity to start when the app launches.

```
<manifest xmlns: android = http://schemas.android.com/apk/res/ package = "com.example.mynotes app">
<application>
<activity android:name = ".mainActivity">
<intent-filter>
<action android:name = "android.intent.action.MAIN">
<category android:name = "android.intent.category.LAUNCHER"/>
```

Q-5 How does the android manifest file impact the development of an android app? provides an example to demonstrate its significance

→

1. Component Declaration : Declaring app components to define the apps structure

eg : <activity android:name = ".mainActivity"/>

2. App permission : Specifying permission for accessing device resources

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

3. Intent filters : Defining how the App responds to external actions or requests.

eg: Registering to open pdf files when tapped

```
<activity android:name=".PdfViewActivity">
    <intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <action android:mimeType="application/pdf" />
    </intent-filter>
    </activity>
```

Q-6 What is the role of resources in Android development. Discuss various types of resources & their significance in creating well-structured applications provide examples to clarify your points

→ Resources in android development are essential components that helps you create well-structured & flexible application. They serves several purposes, such as separating code from content developing to different devices, and simplifying localization. Here are the main types of resources and their significance

1) layout Resources

xml layout : These defines the structure and appearance of your application user interface They help keeps the UI separate from code logic, making it easier to maintain and adapt

Example: A layout xml file, specifies how element like button and texte fields are arranged on the screen

2) Drawable Resources

→ images and icons : Drawable resources store images, icons and other graphics used in your app different versions can be provided for different screen densities

eg: You might have 'ic_launcher.png' for the app icon & separate versions for low, medium and high density screens

3) String Resources:

- Text and string : Storing text in resource files allows for the easy localisation & updates without modifying code.

Eg: A String resource (app_name) contains the app's name, which can be changed for different languages

4) color Resources:

- Colors: By defining colors in resources you can maintain a consistent color scheme across your app and easily switch themes

→ Eg: A color resource define the primary color used in app's UI elements.

5) Style Resources:
- Themes & Styles: Style define the appearance of UI elements, making it simple to apply consistent styling across app

→ Eg: You can create a custom style to define fonts, colors and other visual attributes

6) Dimension Resource:
- Sizes and Dimensions: Storing sizes and margins in res file makes it easy to adjust layout for different screen sizes and orientations

Eg: A dimension resource defines a consistent margin size for elements

7) Raw Resources:
- Raw Data: You can store non-compiled resources like audio, video and text files in the 'res/raw' directory

8) Animation & Drawable Animation Resources

→ Animation : You can define animations in xml resource file, making it simple to reuse and apply animations to UI elements

Eg: A resource file can define a fade in animation for an image view

Q-7 How does an Android Service contribute to the functionality of a mobile app ? Describing the process of developing an android service. Write in simple language and include main points.

→

An android service plays a crucial role in the functionality of a mobile app by using tasks to run in the background, even when the app is not actively in use.

- Contribution of Android Service :

1) Background Processing : Services run tasks in background ensuring the essential functions like music playback, location tracking, or data syncing can continue without disrupting the user interface.

2) long running operations : Services are ideal for operations that take a long time to complete such as downloading large files or performing compler such as calculations, without using the app to freeze

3) Foreground Services: Some services can run in forger displaying a peersistent notification to keep the user aware of engaging tasks like navigation or chat application.

4) inter-component communication: Services can communicate with other app component through interfaces, allowing data exchange & coordination

→ Developing an android service:

1) Create a service class
→ Extend the 'Service' class one of its subclasses like 'Intent Service' or 'Job Service'.

→ Implement the services functionality within the 'On Create' & 'On Start Command' methods

2) Declare the manifest
→ Register your service in the Android manifest. xml file to make it accessible to the system & other components

3) Service Lifecycle
→ Understand the service's lifecycle methods & override them as needed
→ Service can run in three methods: foreground, background or bound choose the appropriate mode based on you app's requirements

4) Start and Stop the Service

→ Start a service using 'Start Service' or bind to it using 'bind Service'.
→ Stop a service when it's no longer needed using 'stop service' or 'stop self()'.

5) foreground Service

→ To create a foreground Service, provide a notification that informs the user about ongoing tasks.

→ Use Startforeground() to start a service in the foreground mode

6) Thread management

→ When performing time-consuming operation, consider using a threads or AsyncTask to prevent blocking the main UI thread

7) Communication

→ Use intent extras, broadcast receivers, or interfaces to enable communication b/w Services and other app components

8) Clean Up & Resource Management

→ Ensure that you release resources and stop the service when it is no longer

no longer needed to prevent unnecessary battery drain

9) Testing:

→ ~~Th~~ Thoroughly test your service to ensure it works as expected, including scenerios like application background, task interuptions, and restarts

5/10/25