Name : Ladva Vishal H.

En.no : 21012011043

Class : CE-IT-B

Batch : 5B-1

Sub :- Mobile Application Development

Assignment:-1

(1) Based on your understanding, Identify a recent bussiness trend that has influenced by the Android Platform. Explain how this trend impacts Android App developers and bussiness in the mobile App industry

→ In recent ~~bussiness~~ business trend, Android is the most popular mobile operating system in the world. We have approx 6.378 billion mobile users in world and mostly they are ~~user~~ using android as operating system.

→ As Per statistics. In 2022 play store have 900,000 new mobile apps. It simply means that android having very big impact in recent bussiness trend.

→ Now, this trend make big impacts on App developers because, as per statistics. mobile ~~android~~ and users are increasing day by day and their requirements also getting increasing so its very bright future for android app developer in this mobile App industry.

→ Some future App development TOPIC for developers
→ ~~A~~
→ Augmented Reality (AR)
→ IOT

→ Chatbots
→ Cloud computing
→ AI
etc...

Q.2 What is the purpose of an Inflator of layout in a Android development ? and how does it fit into the architecture of android layouts ?

→ In android development, an Inflater, specifically a layout Inflator, is a crucial component that is used to instantiate and create instance of view objects from XML layout resource files. Its primary purpose to take an XML layout file and turn it into a corresponding hierarchy of view objects, which can then be displayed on the screen.

→ Here, how Inflator fits into architecture of Android.

① .XML Layouts :- In Android, UI layouts are typically defined using XML files. These XML layouts files describe the arrangement and attributes of UI (TextViews, buttons, ImageView) within an activity or fragment. It is readable and convenient way to design UI.

(2) Activity or fragment: Activities and fragment are the building block of android app. They represent individual screens or parts of the UI, These components need to inflate the xml layout files to display their UI to user

(3) Layout Inflator: This is where the layout inflator comes into play. It's an essential part of the android framework that allows you to dynamically create instances of view objects from xml layout files.

(4) View Hierarchy: The view hierarchy created by the layout inflator forms the structure of the UI for an activity or fragment. It includes UI elements defined in the xml layout file, organized in a tree-like structure.

→ here's basic example how we use inflator in Android:

```
Val Inflator = layoutInflater
Val rootview = Inflater.Inflate(R.layout.activity_
                                main, null)
Set ~~Create~~ contentView(rootView)
```

Q.3 Explain concept of customdialouge box in Android Applications: Provide example to illustrate its use.

→ A custom Dialog Box in Android applications is a pop-up window that developers can design and customize to show specific information, receive input from users or perform actions without navigating to a new screen or activity. Custom Dialog Boxes are helpful for displaying messages, alerts forms or any custom content in controlled and visually appealing manner

1. Design Flexibility: Custom Dialog Boxes allows developers to create unique and tailored user interface

2. Contextual use :- They are typically used when you want to capture input or show information without taking the user to a different screen

3. User Interaction :- Custom Dialog Boxes can contain buttons, text fields, checkboxes or any other UI element allowing users to interact with the content inside the dialog.

→ Examples of custom Dialog Box use

① Confirmation Dialog: A common use case is asking the user for information 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 popup, prompting the user to enter their credintials.

(3) Error messages:- when there is an error, such as network issues or invalid input, a customer dialog can display an error message with details. helping the user understand and correct the problem.

(4) Date and time pickers:- You can create a custom dialog box for selecting date or time, providing a more user friendly way to input this information.

Code

```
import android.app.AlertDialog
import android.appcontent.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.setMessage("this is a custom
                    dialog box example")
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 a 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 interface and screen of an Android app they handle user interactions, display ui elements and manage UI flow

→ Example: Imagine a simple note-taking app. Each screen of the app, such as the note list, note editing and settings, can be implemented as separate activities

② Services

⇒ Role :- Services run in the background and perform long-running or background tasks without a user-interface

→ Example :- In our note-taking app, you might have service that periodically backs up notes to a cloud server without showing a user interface

③. Android Manifest File :

Role :- The Android manifest file is a configuration file 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 between activities using buttons or gestures.

② Services

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

→ This service doesn't have a user interface but operates independently to ensure data is continuously backed up

③ Android manifest File

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

→ you specify permission like 'Internet' to allow the app to access the internet for cloud 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.mynoteapp">
  <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 application? provide an example to demonstrate its significance.

→ The android manifest file impacts app development by.

① Component Declaration:- Declaring app components to define the app's structure

ex. <activity android:name=".MainActivity" />

② App Permissions:- Specifying permission for accessing device resources.

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

③ Intent-filters:- Defining how the app responds to external actions or requests

ex-> Registering to open PDF files when tapped

<activity android:name=".PdfViewerActivity">
<Intent-Filter>
<action android:name="android.intent.action.VIEW" />
<category android:name="android.intent.category.DEFAULT" />
<data android:mimeType="application/pdf" />
</Intent-Filter>
</activity>

Q.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

→ Resources in Android development are essential components that helps you create well-structured and flexible applications. they serves several purposes, such as seperating code from content adapting to different devices and simplifying localization. Here are the main types of resources and their significance

① Layout Resources.

- xml Layouts: there define the structure and appearance of your app's user interface. They help keep the UI seperate from code logic. making it easier to maintain ~~of the~~ codepts

Example: A layout xml file specifies how elements like buttons and text fields are arranged on the screen

② Drawable Resources:

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

Example:- You might have 'IC-launcher.png" for the app icon of separate versions for low, medium and high density screens.

(3) String Resources.

- text and strings: Storing text in resource files allows for easy localization and updates without modifying code.

Example:- 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.

- Example:- A color resource (primary_color) defines the primary color used in the app's UI elements

(5) Style Resources

- themes and styles: Styles define the appearance of UI elements, making it simple to apply consistent styling across app.

Examples:- You can create a custom style (AppTheme) to define fonts, colors and other visual attributes

⑥ Dimension Resources

→ Sizes and Dimensions:- Storing sizes and margins in res file makes it easy to adjust layouts for different screen sizes and orientations.

Example:- A dimension resource (margin-small) defines a consistent margin size for elements.

⑦ Raw resources:

Raw Data:- You can store non-compiled resources like audio, video or text files in the res/raw directory

Example:- Storing a JSON file in the raw folder for configuration data

⑧ Animations and Drawable Animation Resources

Animations:- You can define animations in XML resource file, making it simple to reuse and apply animations to UI elements.
Example:- A resource file (fade_in.xml) can define a fade in animations for an image view

(7) How does an Android service contribute to the functionality of a mobile application? Describe the process of developing an android service ~~write a simple service and include main points~~

→ An android service plays a crucial role in the functionality of mobile application by allowing tasks to run in the background, even when the app is not in 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 file or performing complex calculations, without causing the app to freeze.

(3) Foreground services :

→ some services can run in foreground. displaying a persistent notification to keep the user aware of ongoing tasks like navigation or

: Cheat application

(4) Inter-Component Communication:

→ Services can communicate with other app components (activities, fragments) through interfaces, allowing data exchange and coordination

→ Developing an Android Service

→ (1) Create a service class

→ Extend the 'Service' class or one of its subclasses like "IntentService" or "JobService"

→ Implement the service's functionality within the "onCreate" and "onStartCommand" methods.

(2) Declare in the manifest

→ Register your service in the AndroidManifest.xml file to make it accessible to the system and other components.

(3) Service Lifecycle:

→ understand the services lifecycle methods (onCreate, onStart, onBind, onDestroy) and override them as needed

→ service can run in three methods- Foreground, background or Bound. Choose the appropriate mode based on your app's requirement

(4) Start and stop the service

→ Start a service using 'startService(Intent)' or bind to it using 'bindService(Intent, serviceConnection, int)'
→ Stop a service when its no longer needed using 'stopService(Intent)' or 'stopSelf()'

(5) Foreground-services

→ 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)

(6) Thread management

→ when performing time consuming operations consider using worker threads or AsyncTask to prevent blocking the main UI thread

(7) communications.

→ use Intent extras, broadcast receiver or Interface to enable communication between services and other app components

(8) Clean-up and resource management

→ Ensure that you release resources, and stop the services when it no longer needed to prevent unnecessary battery drain

(9) testing

→ Thoroughly test your service to ensure it works as expected, including scenarios like app backgrounding, task interruptions, and restart

05/10/23