Ganpat University | U.V. Patel College of Engineering
॥ विद्यया समाजोत्कर्षः ॥

U.V. Patel College of Engineering

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

MAD

Name: Prajapati Yash D.
Class: CE IT-B batch: 5B-5
Enrollment No.- 21012011125

Assignment-1

(1) Based on your understanding identify a recent business trend that has influenced the understanding platform. Explain through trend impact Android app developer And business in the mobile app industry.

→ A recent business trend that has significant influenced the android platform is the rise of "progressive web apps (PWA)" and the emphasis on Hybrid App development.

→ Impact on android app developer: ~~are~~

(1) Cross platform Development:
Android app developer are increasily adopting hybrid app development framework like flutter, React Native

(2) improved user Experience:
have pushed Android app developer to prioritize a better user experience.

(3) Reduced maintenance:
Maintaing seperate code bases for Android, iOS, and web apps can be resource intensive

- Impact on Business in the mobile app industry

(1) Cost Efficiency:
Developing for android is often a priority ~~due~~ have to its larger user base. By using Cross-plateform development tools and technique.

(2) Faster Time to market:
Hybrid app development allow busines launch their Android app faster

(3) User Engagement:
The enhanced user Experience from PWAs on improved Androids APP Cant lead to higher user engagement. ~~and~~

(2) What is the purpose of an inflator of layout in Android development and How does it fit into the architecture of Android layout?

→ The purpose of an inflator in Android development is to take an XML layout file convert it into its Corresponding View object in memory.

(1) XML layout files: In ~~an~~ android development you Create layout files that define the structure and appearance of your app's user interface.

(2) Activity or fragment: In your Kotlin Code, typical within an activity or fragment you need to specify which Xml layout should be inflated for the user interface.

B) Inflation: The inflator is used to take the xml layout file and inflate it which means it passes the xml and create a hierarchy of view object the memory that corresponding to the element in the xml layout.

(8) Explain the concept of a Custom Display Box in Android application. Provide example to illustrate its use.

→ A Custom dialog Box in Android application is user interface element that allow developer to create a custom, often model, dialog that appears on top of the current activity content.

Concept:

(1) Customization: Custom dialog can include various UI element like text views, button image & even custom layout.

(2) Model interaction: Custom dialog are often model memory they block the interaction with the underlying activity.

(3) usecase: Custom dialog are used for a wide range of purpose such as displaying information.

→ Example:

```
Val Customdialog = Dialog (this)
CustomDialog.setContentView (R.layout.custom_Dialog)
Val message:TextView = Custom.Dialog.findViewById
    <textView> (R.id.messageTextView)
Val okButton = CustomDialog.findView ById<Button>
                                  (R.id.okButton)
```

```
message.TextView.text = "This is a custom dialog!"
okButton.setOnClickListener {
        Custom.Dialog.dismiss()
    }
    Custom.Dialog.show()
}
```

(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 basics example of how many Cooperate to design a mobile app?

⇒ (1) Activities :-
Role: Activity represent individual Screen or UI Component in an Android App.

(2) Service:
Role: Service are background Component that perform long running Operation or handle task that don't require user interface.

(3) Android manifest file:
Role: The Android manifest.xml is like the app's blue print.

ex

```
Class manifest : AppCompatActivity()
override fun onCreate(saved instanceState: Bundle?){
Super.onCreate(saved InstanceState)
SetContentView(R.layout.activity_main)
Start service Button.setOnClick listener {
    Val service intent = Intent (this, notification Service::
                                              Class.java)
    Start.service (Service Intent)
}}
```

(5) How does the Android manifest file impact the development of an android application? Provide an example to demonstrate its significances.

→

~~Ensure~~ The android manifest file is a crucial component in the development of an android application. It serves several important purpose and its content significantly.

```
<manifest xmlns : android = "https://schema.com/
    apk /res/ android "
    package = "com.example.myapp">
<application
    android : allowBooking = "true"
    android : icon = "@ ic_icon"
    android : label = "@ string /app_name"
    android : theme = "@ style / app_theme">
<activity android : name = ".mainactivity">
 < intent filters>
< action : name = "android.intt. action. main"/>
 < category android : name = "android .intt. category>
 < / intent filter>
< 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.

→ Resources Play a fundamental role in Android development by providing a structured way to manage assets. Values, layouts and other elements used in your app.

- Types of resources and their example:

1. Layout Resources:
   - type: XML files in the 'res/Layout' directory.
   - significance: Define the structure and appearance of the app's user interface.
   - Example:

```
<Button
     android:id = "@+id/myButton"
     android: layout_width = "wrap_Content"
     android: layout_height = "wrap_Content"
     android: text = "Click here" />
```

2. Drawable Resources:
   - type: Images and drawable assets in the 'res/drawable' directory.
   - significance: store graphics, icon, and images used in your app.
   - example: 'IC_launcher.png' is the app's Launcher icon.

3. String Resource:
   - type: String defined in XML files under 'res/values'.
   - significance: store text strings, making it easier to provide translations and maintain Consistency.
   - Example:

```
<String name = "app_name"> App </String>
<string name = "welcome_message"> Welcome </String>
```

4. Color Resources:
- type: Colors defined in xml files under 'res/values'.
- Significance: Store color values, ensuring consistency in the app's design.

Example:
```
<Color name = "Primary_color"> # FF0000 </color>
```

5. Style Resources:
- type: styles defined in xml files under 'res/values'.
- significance: Define reusable style for UI Components.
- Example: 'res/values/style.xml'. defines style.
```
<Style name = "text Style".>
   <item name = "android:background">@drawable/text </Ite
</style>
```

6. Dimension Resources:
- type: Dimensions defined in xml files under 'res/values'.
- Significance: Store dimension values; ensuring a Consistant Layout.
- Example:
```
<dimen name = "margin_large"> 20dp </dimen>
```

(7) How does an android service contribute to the functionality of a mobile application? Describe the process of developing an android services.

→ 1. Background Processing: Services allow apps to perform tasks in the background without blocking the user Interface.
2. long running operations: Services are ideal for handling operations that require more times to Complete, such as playing music.

3. Inter Component Communication: Services enable Components like activities, broadcast receivers and other Services to Communicate with eachother efficiently.

4. foreground Services: Android Services can run in the foreground, even when the app isn't in the foreground. This is useful for features that require ongoing user interactions.

- Process of Developing an android Services:

(1) Define service Class: Create a new Java or Kotlin class that extends the 'service' class. override method like, oncreate(), onStartCommand(), ondestroy() to define behaviour of service.

(2) Configure Service in manifest: Declare your service in the android manifest.xml file to inform the android System about its existence and Configuration.

(3) Start or Bind the service: Decide whether you want to start or Bind your service.

(4) Implement service logic: In service class, implement the Specific logic your service needs to perform its task.

(5) Handle Lifecycle: Release resources when they are no longer needed and Consider using 'stopservice()'.

(6) Interact with other Components: user appropriate mechanisms like intents, broadcasts. or callbacks to facilitate Communication.

(7) Foreground Service: If your service needs to ~~[illegible]~~ in the foreground, 'Start Foreground()'.

(8) Testing: Throughly test your service to ensure it functions as expected, including handling various scenarios like network failures.

[illegible] 19/10/23