Name :- Kothari Pratik Rajkumar

Enrollment No :- 21012011041

Class :- 5CEIT- A

Batch :- 5A - 3

Subject :- Mobile Application Devlopment (MAD)

Department :- U.V. Patel College of Engineering.

college :- Ganpat University.

Assignment : 1

Assignment : 1

ue :- 1 Based on your understanding identify a recent business trend that Has influenced the Android platform. Explain How this trend impacts Android app developers & business in the mobile app industry.

The recent trend of heightened privacy and data security concerns on the Android platform is reshaping the landscape for both Android app developers and business in the mobile app industry. This trend is driven by evolving Privacy regulations, increased user awareness & a growing emphasis on protecting user data.

For android app developers, this trend means adapting to stricter privacy requirements imposed by platform like Google. Developers need to rethink their data collection Practices, ensuring that they request user Permissions transparently and only collect data necessary for their app's core functionality. This can lead to changes in How apps function and the data they collect.

Scanned with CamScanner

Que :- 2. What is the purpose of an Inflator of Layout in Android devlopment, and How does it fit into the architecture of Android layout ?

→ The Layout Inflater in Android serves the Perpose of dynamically Converting XML Layout files into functional view objects during the app's runtime. It fits into the Android Architecture by enabling developers to define the Structure and appearance of the user interface in XML files. These XML files (templates) are then transformed into tengible View objects by the Inflater as required, whether within Activities, Fragments or when designing custom UI components. This dynamic Process is fundamenteel for building adaptive and efficient Android user Interfaces.

Que :- 3 Explain the concept of a Custom Dialog Box in Android applications. Provide Examples to illustrate its use.

→ A custome Dialog Box in Android applications allows developers to create unique pop-up windows with custome layouts and functionality unlike standard dialog, custom dialogs are tailored to the app's specific design and needs.

Ex. For Instance, Imagine developing to-do list app you can create dialog to let users select due dates for their tasks. This custom dialog would Have a personalized layout with a date picker and buttons for confirming or canceling the selection. The code would initialize this dialog, set its content view to the custom layout and define the button click actions for confirmation and cancellation. when needed, you would display this custom dialog to enhace the user experience and ensure it aligns with your app's design.

ue :- 4 How do activities, services & the android Manifest file work together to make an Android app? Can you describe their main roles and provide basic Example of How they cooperate to design a mobile app?

→ Activities, Services & the Android Manifest file work together to create the core structure and functionality of Android app. Each plays a distinct role in the app's architecture:

1. Activities:

Main role: Activities represent the user interface and interaction points of an Android app. They provide screens or windows where users can intreact with the app's content.

2. Services:

main role: Services run in the background and Perform tasks that don't require a user interface. They are used for long-running operations such as downloading files, playing music or Processing data.

3. Android Manifest file:

main role: The Android Manifest file is crucial configration file that defines essential information about the app. It specifies the app's components, Permissions & other important Settings.

⟹ Cooperation Example :-

```
<!-- AndroidManifest.XML -->
<manifest --->
    <application ---->
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.
                                      action.MAIN"/>
            </intent-filter>
        </activity>
        <activity android:name=".SettingsActivity"/
        <service android:name=".WeatherUpdateservice"/

    <!-- Permissions and other app settings --!>
    ...
    </application>
</manifest>
```

Que :- 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 (AndroidManifest.XML) has a significant impact on the development of an Android application. It defines the entire structure and behavior of the app, including its component permissions, intent filters, configurations and more.

without a properly configured Manifest file, the Android System would not know how to intract with Your app, and the app's functionality and user experience would be compromised. it is a central and essentia Part of Android app development.

⇒ Example of Significance:

```
<!-- AndroidManifest.xml --!>
<manifest ...>
   <application...>
      <activity android! name = ".MainActivity">
         <intent-filter>
            <action android! name = "android.intent.action.Main'
         </intent-filter>
         <intent-filter>
            <action android! name = " android.intent.action.View
            <data android! mimeType = " image /* "/>
         <intent-filter>
      </activity>
      <activity android! name = ".UploadActivity" />
      <service android! name = ".PhotoSyncService"/>
     <uses-Permission android! name "android.Permission.CAM
   </application>
</manifest>
```

Scanned with CamScanner

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

→ Resources in Android development are essential assets that provide various types of content used in Android applications. they play a crucial role in separating content from code facilitating localization & ensuring a consistent user experience across different devices and configurations.

Role of Resources:
Separation of content and code
Device configuration Handling.

— Type of Resources and significance:

(a) Layouts: Define the Structure and appearence of UI components in XML files. These layouts are crucial for creating app screens.

Ex.
```
<LinearLayout ...>
    <EditText ../>
    <EditText .../>
    <Button ... />
</LinearLayout>
```

(b) Strings : Store text and string literals in separate resource files. This simplifies localization & make it easier to update app text.

Ex

```
<resources>
    <string name = "app_title">My APP</string>
</resources>
```

(c) Images & Icons ! Drawable :- Store image, icons & other graphics in drawable resource folder. Diffrent folder cater to different screen densities

Ex.

```
drawable - mdpi/
    ic_launcher.png
drawable - hdpi
    ic_launche.png
```

(d) Colors and Styles : Define colors, Styles & thems in resource files to maintain a consistent look and feel across the app.

Ex.
```
<resource>
    <color name = "Primary_color">#3498db</color>
</resorce>
```

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 Contributes to the functionality of a mobile application by performing tasks in the background without requiring direct user interaction. It enhances an app's capabilities in several ways, such as:

- Back Ground Tasks
- Data Processing
- Location Updates
- Scheduled Operations
- Fore Ground Services

→ Devloping an Android Service involves the following Steps:

Step 1: Create a Java/kotlin class that extends the service class. This class defines the logic and behavior of your service.

Step 2: Implement and override essential life cycle method in your service class.

Step 3: Define the functionality your service should perform within the on Start Command () method.

Step 4 : Declare your service in the AndroidManifest.xml file to inform the Android system about your service and its properties.

Step 5 : Initiate your service by sending an Intent that specifies the service class you want to start.

Step 6 : Service can communicate with other app componants or the User interface using mechanisms like Broadcast receiver, Result Receiver or Messager as needed.

Step 7 : Properly manage the termination of your service to it's no longer required to release system resource.

Step 8 : Thoroughly test your service to ensure it functions correctly.

Step 9 : Optimize your service for efficiency, manage background execution limits, handle errors, gracefully, and optimize power consumption to ensure a positive user experience.

m
5/10/23