MAD Assignment-1 21012011076

Ganpat University | U.V. Patel College of Engineering

U.V. Patel College of Engineering

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

(1) As per my knowledge, one notisable trend in the mobile app industry that was influcing the android platform was the rise of progresive web app. PWAS are web applications that often app-like experiences directly through web browsers.

→ Impact on Android app developer:-

→ 1) Cross-platform compatibility:-
PWAS are designed to work seamlessly more engaging user experience

→ 2) Progresive Enhoncement:-
Developers needed to adupt progressive ~~end~~ enhancement strategies to ensure that Android apps remained competivie by offering progressive and responce user experience similar to PWAS.

(2) In Android development, an Inflater refer to the layout inflater, which plays a crucial role in creating a user interface from XML layout files

→ 1) XML layout files:-
In Android, UI components are often defined using XML layout files.

→ 2) Layout Inflation:-
When your Android app runs. It need to ~~tea~~ turn XML layout files into actual view objects that can be displayed on the screen.

→ 3) Layout Inflater:
It is responsible for reading the Layout files and Instantiating the corresponding view objects in memory.

→ 4) Dynamic UI Creation:
Layout inflation is particularly valuable when you need to create UI elements dynamically, for example, in responce to user inrection.

→ 5) Binding Data:
Once the view objects are created, they can be further Customized and data can be bound to them.

③ A custom Diploy Box in Andoid is a pop up window that developers Can design and Customize to display informetiv, options or actions to the user.
It's a versatile UI element for Showing alerte inputs forms, Confirmation dialogs, or any ther customs. intoactins that doesn't the Standard layout of an Activity.
Ex: when you need to provide a tailoxd user experience or other ~~depe~~ spcific input from the user without to a new screen, like alerting warring asking for input at like gander, name, etc info. for Rating etc.

(4) In Android app development, Activities, services, & the android manifest file work togethe to create the structure & functionality of an app.

→ 1) Activities:-
Activities represent indivisual screens or user interfaces in an Android app.
Ex:- Imagine a simple emaile app with two activities : one for composing emails & another for viewing the inbox.

→ 2) Services:-
Services are background components thats perform long-running ~~or~~ operations without a user interface.
Ex:- In our email app example, a servarce could be used to periodically check for new emails in the background update the in box.

→ 3) Android manifest file:-
The Android manifest XMl file is a configuration file that definess esscatial information about the app its components and required permissions.
Ex:- In the manifest file, you specify which activities & services the app contains, their properties, & any permission required, among other things.

(5) The android manifest file is a critical component
android app development it serves several significa
purposes that impact the development and functionlity
of an Android application.

Ex:-

1) Component Declaration :-
The Android manifest file is where
you delare all the components of your android
application, including activities, services, broadcast recivers
and Contect providers.

```
→ <application...>
      <activity android:name = ".main Activity">
       <intent - filter>
         <action android: name="android. intent. action. MAIN"/>
        <category android: name="android. intent. category. LAUNCHER"/>
      </intent_ filter>
     </activity>
    <service android: name = ".My Service"/>
    <receiver android: name = ".My Receiver"/>
  <Provider
      android : name = ".My Content Provider"
      android: authorities = "Com. example. my app. Provider"/>
</application>
```

2) Application Configuration:-

```
→ <Application
       android: icon = "@drawable /app_icon"
       android : label = "@ string /app_ name"
       android: theme= "@ style /app theme"
       android: allowBackup = "true"
       android: versionCode = "1"
       android: VersionName = "1.0">
```

(6) Resources are the additional files & Static Content that your code uses, such as bitmaps layout definitions user interface string, animation instructions, & more.

⇒ Resources types overview.

→ 1) Animation resources:-
Define pre-determined animations fame animations are saved in res/drawable/ and accessed from the R.drawable class.

→ 2) Color State list resource:-
Define a Color resource that changes bassed on the view state. saved in res/color/ and accessed from the R.color Class.

→ 3) Drawable resources:-
Define various graphics with bitmaps or XML.

→ 4) layout resources:-
Define the layout for your application UI.

→ 5) Menu resources:-
Define the contents of your application menus.

-> 6) Style resources!- Define the look and format for UI element.

-> 7) font resources!- Define the families so inculude custom font in XML.

(7)
=>1) Background processing !-
Services allow apps to perform tasks in the background without blocking the user interface.

=>2) long running operation!-
Services are ideal for healing operations that require more time to complete such as playing music.

=>3) Inter Component Communication !-
Services enable components like activities brodcast receivers and other services to communicat with each other efficiently.

=>4) foreground Service!-
Android services can run in the foreground, even when the app isn't in the foreground.

=> Process of Developing an android service !-

-> 1) Defines the service class !-
Create a new Java or Kotlin class that extend the 'Service' class.

CH

21012011076

Ganpat University | U.V. Patel College of Engineering

U.V. Patel College of Engineering

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

→ 2) Configure Service in manifest :-
Declare your service in the Android manifest.xml file to inform the android system ~~on~~ about its exsit and configuration.
<service android: nume = ".My service"/>

→ 3) Start or Bind the serviece:-
Declare whether you want to start your service or bind it to other components.

→ 4) Implement service logic :-
In service class, implement the specific logic your service need to perform its task.

→ 5) Handle lifecycle :-
Release resource when they no longer needed and consider using 'stopself()' or 'stopservice ()'.

→ 6) Interect with other components:-
use appropriate mechanisume like intents broadcast or callback to facilitate communication.

→ 7) foreground services (optional):

If your service needs to run in the foreground 'Start foreground()'.

[illegible]