MOBILE APPLICATION DEVELOPMENT

MODEL PAPER - 4

SECTION - A

1 . What is Android Virtual Device (AVD)?

an Android virtual device AVD is a configuration that defines the characteristics of an Android phone tablet where OS Android TV or automatic OS device to simulate in the Android Emulator the device manager is a tool to launch from Android studio that helps to create and manage android virtual devices

2. What is an Intent in Android?

An intent in Android is a messaging object that can be used to request and action from another app component such as activities services and broadcast receiver's it serves as a bridge to enable communication between different components within an application or between different applications on the device intense can be used to start an activity start a service deliver a broadcast

3. What is an Action Bar in Android?

The action bar in Android is a user interface component that typically appears at the top of an Apps screen and provides consistent navigation brand in and actions for the user the action bar in Android is a dedicated space and the top of the screen that provide the consistent navigation and access key features and actions within an app it typically contains the app icon and title navigation tabs action icons and overflow menu the action bar helps users to access key features easily and maintain a consistent User experience across different screens and activities.

4. What is an AutoCompleteTextView in Android?

Auto complete Text view is a UI component in Android that provides suggestions to user as the type it extends from edit text and offers and auto completion feature which displays a drop down list of suggestions based on the text entered by the user this component is useful in

Pavithra.S

scenarios where uses need to select from a print define list of options such as entering an email address selecting the city or typing a search query.

5. What is Content Provider URI? Give an example.

Accountant provider you are right uniform resource identifier in Android is a unique identify that specifies the data source or location within a content provider it follows a specific format that allows applications to access and manipulate structure data in a consistent and Secure manner.

6. Why is the HTTP url connection class used in Android

HTTP url connection is a standard class provided by the Java STK and his widely used in Android for managing HDP commands this class offers of flexible and efficient way to communicate with web servers using the HTTP protocol it allows developers to send request and receive responses over the web supporting various http methods such as get post put delete etc

SECTION - B

7. What is Android? Explain the Key Features of Android OS.

Android is an open source operating system developed by Google, designed primarily for touchscreen mobile devices such as smartphones and tablets. Since its inception, Android has become the most widely used mobile operating system globally, powering billions of devices. It is known for its Bechality, extensive app ecosystem, and seamless integration with Google services. Andruid offers a robust platform for both users and developers it supports a wide range of devices including smartwatches TVs, and car infotainment systems, making it a versatile and comprehensive mobile operating system. Android's open source nature allows for vignificant custonization and innovation fostering a diverse and dynamic mobile technology landscape.

Key features of Android

1. Open Source Platform: Android is based on the Linux kernel and is released under the Apache License, allowing manufacturers and developers to modify and distribute the software Ireely.

Example: Custom ROMs like LineageOS and Paranoid Android offer alternative Android experiences with additional features and optimizations.

2. Customizable User Interface: Android allows users to customize their home screens, widgets, and themes to personalize their experience

Example: Users can add weather widgets, change wallpapers, and use custom launchers like Nova Launcher to modify the look and feel of their device

3. Google Services Integration: Android devices are tightly integrated with Google services such as Gmail, Google Maps, Google Drive, and Google Assistant.

Example: Users can access Google Maps for navigation, store files in Google Drive, and use Google Assistant for voice commands and smart home control

4. Extensive App Ecosystem: The Google Play Store offers millions of apps covering various categories, including productivity, entertainment, social networking, and gaming

Example: Popular apps like WhatsApp, Instagram, Spotify, and Netflix are readily available for download from the Play Store.

5. Multi-Device Support: Android powers a variety of devices beyond smartphones and tablets, including smartwatches (Wear OS), smart TVs (Android TV), car infotainment systems (Android Auto), and home appliances

Example: Devices like the Samsung Galaxy Watch use Wear OS, while smart TVs from brands like Sony and NVIDIA use Android TV

Pavithra.S

6. Google Assistant and Voice Commands: Android devices come with Google Assistant, a virtual assistant that can perform tasks, answer questions, and control smart home devices through voice commands.

Example: Users can say "Hey Google" to set reminders, send texts, play music, or adjust smart thermostats and lights

7. Multi-Tasking and Split-Screen Mode: Android supports multitasking and split-screen mode allowing users to run and view two apps simultaneously.

Example: Users can watch a YouTube video while browsing the web or use a note-taking app while reading an e-book.

8. Notifications and Quick Settings: Android provides a rich notification system with actionable notifications and quick settings for easy access to frequently used functions.

Example: Users can reply to messages directly from the notification shade and toggle settings like Wi-Fi, Bluetooth, and Do Not Disturb.

9. Advanced Connectivity Options: Android supports various connectivity options such as NFC. Bluetooth, Wi-Fi Direct, and USB OTG

Example: Users can make contactless payments using Google Pay (NFC), share files with nearby devices using Bluetooth, and connect USB peripherals like keyboards and flash drives using USB OTG

10 Regular Updates and Security: Google regularly releases updates to the Android operating

system by introducing new features, performance improvements, and security patches.

Pavithra.S

Example: The Google Pixel series receives timely updates directly from Google to ensure the latest features and security enhancements are available.

11. Storage: Android uses SQLite, a lightweight relational database for local data storage. This allows applications to store and manage complex data structures efficiently on the device.

Example: A note taking app can use SQLite to store and retrieve user notes

12. Connectivity: Android supports various connectivity options, including GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth (including A2DP and AVRCP profiles), WiFi, LTE, and WiMAX.

Example: Users can connect to the internet via WiFi or mobile networks, and pair their devices with Bluetooth accessories like wireless headphones.

13. Messaging: SMS and MMS: Android supports both SMS (Short Message Service) and MMS (Multimedia Messaging Service) for text and multimedia communication

Example: Users can send and receive text messages, photos, videos, and audio messages 14. Media Support: Android includes support for various media formats, including H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, and BMP

Example: Users can play music, watch videos, and view images in different formats

15. Hardware Support: Android supports various hardware sensors, including an accelerometer camera, digital compass, proximity sensor, and GPS

Example: Fitness apps can use the accelerometer to track steps, and navigation apps can use GPS for location tracking

16. Multi-Touch: Android supports multi-touch screens, allowing for intuitive gestures like pinch to-zoom and multi-finger swiping

Example: Users can zoom in on photos or maps using two finger gestures.

17. Tethering: Android supports tethering, allowing users to share their internet connection via wired (USB) or wireless (WiFi hotspot) methods.

Example: Users can turn their Android device into a portable WiFi hotspot to provide internet access to other devices.

18. Web Browser: Android's web browser is based on the open-source WebKit engine and Incorporates Chrome's V8 JavaScript engine for fast and efficient web browsing

Example: The built in browser allows users to surf the internet, access web applications, and view multimedia content

19. Google Play Store: Android users have access to the Google Play Store, a vast repository of

apps, games, movies, music, and books The Play Store offers a diverse range of content for

users to download and enjoy on their devices. Developers can publish their apps on the Play

Store to reach a global audience. Example: Apps like WhatsApp, Instagram, and Spotify are available on the Google Play Store. Users can download WhatsApp for messaging, Instagram for social networking and photo sharing, and Spotify for music streaming

20. Adaptive Battery: Adaptive Battery uses machine learning to optimize battery usage based on user habits and app usage patterns

Example: The system prioritizes battery power for frequently used apps and limits the battery consumption of less frequently used apps, extending overall battery life.

8. Explain how the intent object is used to invoke built in applications in Android

Using the intent object to invoke built in application refers to leverage in Android intent to request functionality from other application to request functionality from other applications that are pre installed on the device this allows to utilise existing apps for common task to enhance the User experience by providing seamless integration with system functionalities this functionality can include opening the web browser dialing the number launching the calendar or launching contacts app this integration enhance is the User experience by utilising the familiar system application.
To invoke built in applications an intent object is created with the specialised action and if necessary data the action defines what kind of task is to be performed and the data specifies the exact task or content to be handled when the intent is passed to the start activity method the Android system identify the appropriate application to handle the action.

. Layout XML File ('activity main.xml'):

• Defines a "RelativeLayout' with a single "Button' centered in the parent layout.

2. Main Activity Java File ('MainActivity.java'):

onCreate() Method: Initializes the activity and sets the content view to activity, main.xml
Button Click Listener: Sets up an OnClickListener for the button to handle click events

Intent Creation: Creates an Intent' with the action 'Intent ACTION VIEW' and sets the data to the contacts content URI

Start Activity: Starts the contacts application with the created intent using 'startActivity(intent)

3. Run the Application

Pavithra.S

Run the application on an Android device or emulator Click the "Open Contacts" button to launch the contacts app.

This example demonstrates how to use an Intent to invoke the built-in contacts application from an Android activity.

9. Explain WebView along with its attributes and features Give an example

WebView' is a view that displays web pages within an Android application, It is based on the Webike Web and allows integration of web content seamlessly into the apparebit can be used to show static HTML content or to lead web pages from the internet. It's particularly useful for displaying content hosted online or for creating hybrid applications that combine web and native views.

Features of WebView

1. Loading Web Content:

WebView' can load and display web pages using the "loadUrl' method.

It can load both local HTML files and web pages from the internet

2. JavaScript Support:

• WebView can execute JavaScript to run web applications that require JavaScript

3. Navigation Controls:

• It provides navigation methods such as 'goBack', 'goForward', and 'reload' to control the browsing history within the 'WebView

Pavithra.S

4. Customizable User Interface:

The look and feel of the "WebView' can be customized to match the rest of the application's design

5. Handling Links and Forms:

• WebView' can handle links and forms to enable full web page interaction within the application.

6. Zoom Controls:

It supports pinch-to-zoom and can display zoom controls for a better user experience

7. WebViewClient:

By using a 'WebViewClient', the behavior of the "WebView when navigating to new URLs. handling page loading, and managing errors can be controlled.

Explaination
MainActivity: Sets the content view to activity main.xml and initializes the WebView element.

WebSettings: Enables JavaScript for the 'WebView

WebViewClient: Ensures that links clicked within the 'WebView open within the same WebView instead of an external browser

loadUrl: Loads a specified URL. (www.google.co.in) into the "WebView onBackPressed: Overrides the back button behavior to navigate back within the WebView history if possible

10. Provide a detailed explanation on the implementation of an About Box in the Sudoku app.

The about box is an essential feature the provides uses with information about the application and about box provides user with the information about the application to enhance the over all User experience by giving inside into what the app does its purpose or any other relevant information in the context of the Sudoku app the above box will display brief description of Sudoku how it's played and its object use when the uses select the about button the Apple display a new screen activity containing the information about Sudoku the use secondary through the information and then press the back button to return to the main screen.

To create an About box that provides information about the Sudoku game when the user selects the About button, follow these steps. This process involves adding an activity and setting it up to display information about the game

1. Define a New Activity: This activity will display the About information

2. Create a Layout File for the Activity: The layout will include a ScrollView containing a

TextView for displaying the content

3. Add String Resources: These resources will contain the title and content for the About bas

4. Create the Activity Class: This class will set up the layout and handle the display of the content

5. Wire Up the About Button: In the main activity, set up a click listener for the About button to start the new activity.

6. Update AndroidManifest.xml: Declare the new activity in the manifest file so the system recognizes it.

11. Explain how to query database and fetch the records using a Cursor

12. Explain the process of creating a Custom Content Provider with example code

1. Define the Content Provider Class:

Create a Java class that extends 'ContentProvider to implement the custom content provider

Override the necessary methods such as 'query()', 'insert()", "update()', 'delete()', and 'getType() to handle data operations.

The content provider class acts as the interface for interacting with the underlying data source

2. Define URIs and MIME Types:

Define content URIs that represent the data resources within the content provider.

Content URIs follow the format content://authority/path/id and are used to access specific data or perform CRUD operations

Define MIME types for the data handled by the content provider, specifying the type of data that can be returned.

3. Implement CRUD Methods:

query(): Implement the 'query()' method to retrieve data from the content provider based on specified criteria.

insert(): Implement the insert() method to add new data to the content provider.

update(): Implement the 'update() method to modify existing data in the content provider

delete(): Implement the 'delete() method to remove data from the content provider.

getType(): Implement the 'getType() method to return the MIME type of the data associated with a content URI

4. Declare the Content Provider in the Manifest:

Register the custom content provider in the 'AndroidManifest.xml file using the <provider> tag

Specify the authority, which is a unique string to identify the content provider

Define the class name of the content provider that extends ContentProvider

Declare the permissions required to access the content provider, such as read or write permissions

5. Enforce Permissions:

Specify the necessary permissions in the 'AndroidManifest.xml' file to control access to the content provider

Define permissions like 'READ PROVIDER' or 'WRITE PROVIDER to restrict read and write operations on the data

Section-C

13. Explain the Android Application Components

An Android application is composed of several key components that work together to provide the desired functionality Android application components are essential building blocks that define different aspects of an Android app's behavior and functionality. These components are crucial for creating interactive and dynamic mobile applications. Each component has a specific role, contributing to the overall functionality of an application These components are loosely coupled and are described in the application's manifest file (AndroidManifest.xml), which also includes metadata. hardware configuration, platform requirements, external libraries, and permissions Android components can be categorized into Primary Components (Core Components) and Secondary Components (Non-Core) based on their fundamental roles and supplementary functionalities within an application.

2.2.1 Primary (or) Core Components

Core components are crucial for the basic functionality and structure of an Android application. They handle the main processes, user interactions, and essential background tasks that make the app operational.

1. Activities: Activities represent the user interface (UI) of an application. They are responsible for interacting with users and handling user interactions. Each screen in an Android app is typically implemented as an activity Each activity is typically designed to handle a specific task or function.

Example in the PhonePe application, the main screen activity is a key component that displays the user's balance, recent transactions, and various options like "Recharge." "Pay Bills" and "Transfer Money" Selecting an option opens another activity displaying relevant details and actions Chicking on "Transfer Money in the main screen opens another activity where users can enter the recipient's details and the amount to transfer. After entering the required information, the user can confirm and complete the transaction in this new activity

2. Services: A service is a component that performs long running operations in the background without a user interface They handle tasks that need to continue even when the app is not in the foreground, such as data processing, network operations, or playing music

Pavithra.S

Example: When initiating a payment transfer in Phonel'e, the app may start a background service to handle the transaction processing. This ensures continuity even if the app is minimized or the screen is locked The service can manage network communication, handle retries in case of failure, and ensure that the transaction completes successfully Additionally the service can notify the user of the transaction status via notifications.

3. Broadcast Receivers: Broadcast Receivers respond to system-wide broadcast announcements They are used to listen for and respond to broadcast messages from other apps or the system Broadcast Receivers enable applications to be notified of events like battery status changes network connectivity changes, or custom broadcasts sent by other applications

Example: A broadcast receiver may listen for network connectivity changes If the internet connertion is lost or restored, the app can respond accordingly, such as retrying a pending transaction or notifying the user

4. Content Providers: Content Providers manage a shared set of app data. They allow apps to securely share data with other apps or access data from other apps Content Providers encapsulate data storage and retrieval mechanisms, providing a standard interface for querying and updating data. They support CRUD (Create, Read Update, Delete) operations and can handle complex data structures like files, databases, or even cloud storage

Example: To access and manage user contacts for payment purposes, PhoneFe could use a content provides. This enables the app to read and write contact information stored on the device For instance, when selecting a contact to send money to the app would query the contacts content provider to retrieve and display the relevant contact details.

2.2.2 Secondary (or) Non-Core Components

Secondary (or) Non-core components enhance the primary functionalities and improve user experience, but they are not essential for the basic operation of the app. They provide additional features and optimizations.

1. Fragments: Fragments represent a behavior of a portion of the user interface in an Activity Fragments are modular sections of an activity with their own lifecycle and UI They can be dynamically added, removed, or replaced within an activity. They are reusable Ul components. that can be combined to create a multi-pane UI and are commonly used for building flexible Layouts.

Example: Imagine a Fragment as a separate section or a small window within the Phone Pe app's Payment screen. It's like having a specific area on the screen that shows payment details and options. When we open the PaymentActivity in PhonePe, we might notice a specific section labeled as "Payment Details" or "Payment Options" This section is actually the Payment Fragment. In the Payment Fragment, users can view details such as the amount to be paid, payment methods available (like UPI, credit/debit card), recipient information, and any additional options related to the payment process.

2. Intents: Intents are messaging objects used to request actions from other app components They can start an activity, start a service, or deliver a broadcast. Intents can carry data to be used by the target component. In simple terms, think of Intents as special messages that apps use to ask other parts of the app to do something specific. Apps use Intents to communicate and trigger actions within the app. They help in coordinating different parts of the app to work together seamlessly.

Example: When a user clicks on the "Transfer Money" button in the main screen, an intent is used to start the money transfer activity, passing necessary information about the transaction The Intent acts as a carrier, taking all the details about the money transfer from the main screen to the money transfer screen

3. Views and ViewGroups: Views are the basic building blocks for user interface components. and ViewGroups are containers that hold multiple views and define their layout structure Views include elements like buttons, text fields, and images, while ViewGroups include layouts like LinearLayout and RelativeLayout, which arrange the views

Example: The main screen uses TextView to display the balance. ImageView for icons, and Button for actions like "Recharge" and "Pay Bills". These elements are organized within a LinearLayout to create a structured and user-friendly interface

4. Adapters: Adapters act as a bridge between Ul components and data sources to facilitate the display of data in Ul components. Adapters are like messengers that fetch data from sources like a database and show it in Ul elements such as lists or grids. Adapters bind data to the views within these components to allow dynamic data display

Example: In the transaction history section, an adapter helps display transaction details in a scrollable list for easy viewing

5. Notifications: Notifications are used to alert users about events or updates even when the app is not in the foreground. They provide a way to keep users informed about important

information.

Example: Notifications in PhonePe serve as a valuable tool for keeping users informed about important events, such as successful payment transactions, even when the app is not actively being used. By sending notifications with transaction details, like the amount and recipient,

PhonePe ensures users stay updated on their financial activities effortlessly.

6. Loaders: Loaders load data asynchronously in activities or fragments to ensure that data

operations do not block the main UI thread. They manage the background task of loading data and provide the loaded data.
Example: When fetching transaction history from a server, a leader performs this operation is

the background to ensure a smooth and responsive user interface. Loaders play a crucial role

in maintaining a responsive user interface by handling data loading tasks in the background

thus preventing any delays or freezes in the main Ul thread 7. SharedPreferences
SharedPreferences allow an app to store key-value pairs locally os the device. They are commonly used to store small amounts of data persistently They an commonly used for saving user settings or preferences. The data is stored persistently acros user sessions

Example: User preferences such as the default payment method, notification settings, and the last used bank account are stored using SharedPreferences.

8. Widgets: Widgets are self contained UI components that can be added to the home screen to provide quick access to app functionalities without opening the app. Widgets can display dynamic data and offer interactive elements.

Example: A widget on the home screen displays the current balance and recent transactions allowing users to quickly check their financial status without opening the app

9. Intent Filters: Intent Filters define the kinds of intents an activity, service, or broadcast receiver can respond to. They specify the types of actions the component can handle, allowing the system to match intents with the appropriate component

Example: Intent Filters in PhonePe act as instructions that guide the app on how to react when specific actions such as making a payment or accessing a particular feature within the app are initiated These filters serve as guidelines that help the app understand and respond accurately to requests. It ensures that when users interact with the app, it knows precisely how to handle the intended actions

14. Describe the complete lifecycle of an Android activity and the purpose of each lifecycle method.

The below figure shows the life cycle of an activity and the various stages it goes through from when the activity is started until it ends.

1. Activity Starts-'onCreate():

Description: Called when the activity is first created. It is where static setup tasks are performed, such as creating views and binding data.

Purpose: Initialue the activity, inflate the Ul, and execute any one-time setup procedures

2. Activity is Being Created onStart():

Description: Invoked when the activity becomes visible to the user

Purpose: Make the activity visible to the user. It is visible but not in the foreground or interacting with the user

3. Activity Comes to the Foreground 'onResume():

Description: Called when the activity starts interacting with the user Purpose: Bring the activity to the foreground and make it interactive

4. Activity is Running:

State: The activity is now running, and the user can interact with it

. Another Activity Comes in Front - 'onPause()':

Description: Triggered when another activity is about to resume, partially obscuring the current activity.

Purpose: Save changes, stop animations, or perform actions that should not continue while the activity is in the background.

6. Activity is No Longer Visible 'onStop()':

Description: Called when the activity is no longer visible to the user.

Purpose: Release unnecessary resources, save data to storage, etc

7. Activity Comes to the Foreground Again 'onRestart()':

Description: Invoked when the activity has been stopped and is restarting

Purpose: Prepare the activity for restarting after being stopped.

8. Process is Killed or User Navigates Back:

'onStart()' and 'onResume()' are called again to make the activity visible and interactive.

9. Activity is Shut Down 'onDestroy()':

Description: Called before the activity is destroyed.

Purpose: Clean up resources, end threads, close databases, etc. This is the final method called before the activity is destroyed.

15. Discuss how to implement file reading and writing operations in both internal and external storage. Provide a detailed code example

16.a) Explain Linear Layout along with its features, attributes, advantages and disadvantages

A LinearLayout organizes its children into a single row or column Linear Layout organtzes Views linear direction either horizontally or vertically based on the orientation attribute. Views are placed after another in the order they are defined within the layout.

Key Features

1. Orientation:

Determines whether child views are aligned horizontally or vertically.

Controlled by the 'android:orientation' attribute

2. Weight Distribution:

Distributes extra space within a Linear Layout

Controlled by the "android:layout, weight' attribute

. Gravity:

Understanding The Android User Interface

Specifies how child views should be positioned within the LinearLayout.

• Controlled by the 'android:gravity' attribute.

4. Margins and Padding:

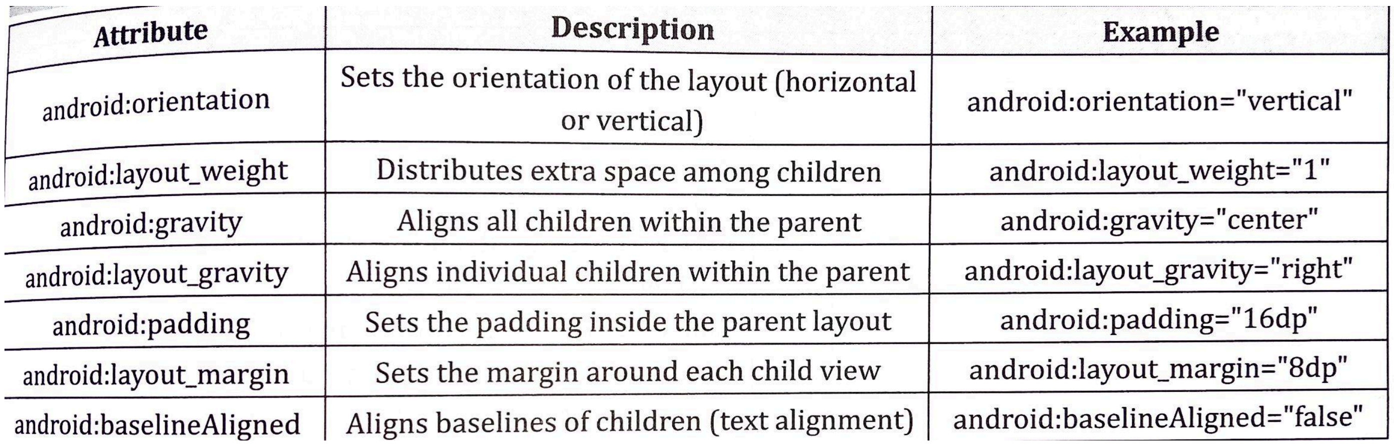
• Provides spacing around and inside child views.

• Controlled by the 'android:layout margin' and 'android:padding' attributes.

5. Baseline Alignment:

• Aligns text baselines of child views.

• Controlled by the 'android:baseline Aligned attribute.



Attribute	Description	Example
android:orientation	Sets the orientation of the layout (horizontal or vertical)	android:orientation="vertical"
android:layout_weight	Distributes extra space among children	android:layout_weight="1"
android:gravity	Aligns all children within the parent	android:gravity="center"
android:layout_gravity	Aligns individual children within the parent	android:layout_gravity="right"
android:padding	Sets the padding inside the parent layout	android:padding="16dp"
android:layout_margin	Sets the margin around each child view	android:layout_margin="8dp"
android:baselineAligned	Aligns baselines of children (text alignment)	android:baselineAligned="false"

Advantages of Linear Layout

Simplicity: Easy to use and understand, making it ideal for simple layouts.

Predictability: Provides predictable and straightforward arrangement of child views.

Control: Offers fine control over the layout with weight distribution, gravity, and margins

Disadvantages of Linear Layout

Performance: Can lead to performance issues with deep hierarchies, as each child view adds to the

rendering complexity Flexibility: Less flexible compared to other layouts like ConstraintLayout, especially for more comples Uls

Spacing: Requires careful management of spacing and alignment, especially when dealing wh different screen sizes and resolutions

b) Discuss in detail the different types of Ul notifications that can be listened to at the activity level in Android.
On kedown on key up on menu item selected on menu opened on touch event on Window focus changed on configuration changed on back pressed on save instance state on restore instance state on create options menu and on options items selected .

17. Provide a detailed example of implementing live data binding in an Android application.

18. Describe the process of creating and managing a proximity alert in an Android application.