Assignment Module 4 – Android Studio and Android App Structure

1. Explain the Android app structure in detail, including the purpose of the AndroidManifest.xml file, Gradle, and various directories (e.g., res, src).

➢ AndroidManifest.xml :

- An AndroidManifest.xml file is an XML document that contains information about an Android app, such as its components, permissions, and hardware and software requirements
- The file is required for every app project and is located at the root of the project source set.

➢ Gradle

- Gradle file is used to define global configuration and settings for the entire project.
- It is also used to implement dependencies and plugins on top level.
- By default, all configurations and settings inside our project-level build.
- Gradle should be inherited by all the modules in the project, and build.

➢ Directories

- src/: Contains app source code and test files.
    - main/java: Java/Kotlin code for app logic.
    - main/AndroidManifest.xml: App manifest file.

- res/: Stores resources like UI layouts, images, and constants.
    - drawable/: Images and icons.
    - layout/: XML files for UI design.
    - values/: Strings, colors, dimensions, and styles.
    - mipmap/: Launcher icons.

2. Write an essay on the different layout types in Android (Linear Layout, Relative Layout, Constraint Layout). Compare their usage and performance.

1. Linear Layout

- Linear Layout is the most basic layout in android studio, that aligns all the children sequentially either in a horizontal manner or a vertical manner by specifying the **android:orientation** attribute.
- If one applies **android:orientation=”vertical”** then elements will be arranged one after another in a vertical manner
- If you apply **android:orientation=”horizontal”** then elements will be arranged one after another in a horizontal manner.

2. Relative Layout

- The Relative Layout is used to arrange the Child Views in a proper order which means arranging the child objects relative to each other.
- There are so many properties that are supported by relative layouts. some of the most used properties are listed below
    - layout_alignParentTop
    - layout_alignParentBottom
    - layout_alignParentRight
    - layout_alignParentLeft
    - layout_centerHorozontal
    - layout_centerVertical
    - layout_above
    - layout_below

3. Explain the differences between Fragment and Activity. How does Android handle fragment lifecycle differently from activity lifecycle?

- **Activity**:
  - An **Activity** is a full screen or page in your app. It's like one complete window where you show content to the user.
  - Each Activity handles one screen, like a login page, a profile page, or a settings page.
  - Activities are independent and run on their own.

- **Fragment**:
  - A **Fragment** is like a small piece of a screen inside an Activity. You can think of it like a part or section of the screen (for example, a menu or a list).
  - Fragments can't exist by themselves; they need to be inside an Activity.
  - You can combine multiple Fragments inside one Activity to create a more complex screen.

- **How Android handles the lifecycle of Fragments vs. Activities:**

  - Both **Fragments** and **Activities** have their own life cycles, but the Fragment lifecycle is connected to the Activity's lifecycle.

    - **Activity Lifecycle**:
      - The Activity goes through several stages, like starting, running, pausing, or stopping. For example:
        - **onCreate()**: When the Activity starts.
        - **onPause()**: When the Activity is about to go into the background.
        - **onStop()**: When the Activity is no longer visible to the user.
        - **onDestroy()**: When the Activity is completely destroyed.

    - **Fragment Lifecycle**:
      - A Fragment has its own lifecycle but is mostly tied to the Activity's lifecycle. For example:

        - **onAttach()**: When the Fragment is attached to an Activity.
        - **onCreateView()**: When the Fragment's view is created.
        - **onPause()**: When the Fragment is paused, like when the Activity is paused.

- **onDetach()**: When the Fragment is removed from the Activity.

➢ **Key Differences in Lifecycle:**

- When an **Activity** is paused, stopped, or destroyed, the **Fragment** inside it is also paused, stopped, or destroyed automatically.
- However, a Fragment also has extra lifecycle methods for managing its own UI, like onCreateView() (creating the Fragment's layout) and onDestroyView() (cleaning up the Fragment's layout).

➢ **In simple terms:**

- **Activity** is like a whole page, and **Fragment** is like a section or piece of that page.
- Both have lifecycles, but the Fragment's lifecycle depends on the Activity. When the Activity is paused or stopped, the Fragment goes through similar stages.