# MC MIDS QUESTIONS AND BLANK STATEMENTS

- Once you provide a resource in your application, you can apply it by referencing its resource ID. All resource IDs are defined in your project's R class, which the aapt tool automatically generates
- Look at the tag for the ConstraintLayout, and notice that it says androidx.constraintlayout.widget.ConstraintLayout instead of just ConstraintLayout like the TextView. This is because ConstraintLayout is part of Android Jetpack, which contains libraries of code which offers additional functionality on top of the core Android platform. Jetpack has useful functionality you can take advantage of to make building apps easier. You'll
recognize this UI component is part of Jetpack because it starts with "androidx"
- The xmlns stands for XML namespace, and each line defines a schema, or vocabulary for attributes related to those words. The android: namespace, for example, marks attributes that are defined by the Android system. All of the attributes in the layout XML begins with one of those namespaces.
- Each layout file must contain exactly one root element, which must be a View or ViewGroup object. Once you've defined the root element, you can add additional layout objects or widgets as child elements to gradually build a View hierarchy that defines your layout.
- To define a view's position in ConstraintLayout, you must add at least one horizontal and one vertical constraint for the view.
- Each constraint represents a connection or alignment to another view, the parent layout, or an invisible guideline.
- Each constraint defines the view's position along either the vertical or horizontal axis; so each view must have a minimum of one constraint for each axis, but often more are necessary.
- If a view has no constraints when you run your layout on a device, it is drawn at position [0,0].
- For your app to be able to use activities, you must declare the activities, and certain of their attributes, in the manifest.

1. **What is a “version control system”?**
   Version control systems are a category of software tools that helps in recording changes made to files by keeping a track of modifications done in the code.
2. **Benefits of git?**
   1) Keep track of changes to code
   2) Different developer

3) Code Synchronizing between different groups
4) Same code available to all members
5) Different people working on different files
6) Test changes without losing original
7) Revert back to old versions

3. **Name some VCS's?**

   a) GitLab b)BitBucket c) Visual studio d)GitHub

4. **What are merge conflicts?**

   Conflicts generally arise when two people have changed the same lines in a file, or if one developer deleted a file while another developer was modifying it.

5. **How to Resolve Merge Conflicts in Git?**

   There are a few steps that could reduce the steps needed to resolve merge conflicts in Git.

   1. The easiest way to resolve a conflicted file is to open it and make any necessary changes
   2. After editing the file, we can use the git add a command to stage the new merged content
   3. The final step is to create a new commit with the help of the git commit command

   Git will create a new merge commit to finalize the merge

6. **What is Branching?**

   When you're working on a project, you're going to have a bunch of different features or ideas in progress at any given time – some of which are ready to go, and others which are not. Branching exists to help you manage this workflow. When you create a branch in your project, you're creating an environment where you can try out new ideas. Changes you make on a branch don't affect the main branch, so you're free to experiment and commit changes, safe in the knowledge that your branch won't be merged until it's ready to be reviewed by someone you're collaborating with.

7. **What is git stagging area?**

   Staging area is files that are going to be a part of the next commit, which lets git know what changes in the file are going to occur for the next commit. The repository contains all of a project's commits.

8. **Git Commands and their roles (Short Questions and Fill the blanks)**
   **git init =>** initialize empty git repository
   **git status=>** The git status command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven't, and which files aren't being tracked by Git.
   **Git add =>** The git add command adds new or changed files in your working directory to the Git staging area.
   **Git add . =>** adds all modified and new (untracked) files in the current directory and all subdirectories to the staging area (a.k.a. the index), thus preparing them to be included in the next git commit .
   **Git - -rm cached =>** The Git rm –cached flag removes a file from the staging area. The files from the working directory will remain intact. This means that you'll still have a copy of the file locally. The file will be removed from the index tracking your Git project.
   **Git Clone=>** Cloning a repository pulls down a full copy of all the repository data that GitHub.com has at that point in time, including all versions of every file and folder for the project. You can push your changes to the remote repository on GitHub.com, or pull other people's changes from GitHub.com
   e.g,(git clone https://github.com/haqnawaz99/20220322_A)
   **git push =>** The git push command is used to upload local repository content to a remote repository. Pushing is how you transfer commits from your local repository to a remote repo.
   **Git pull =>** The git pull command is used to fetch and download content from a remote repository and immediately update the local repository to match that content
   **Git commit =>** Changes made with message to track changes
   (git commit -m "Message ")
   **Git log =>** Details of commits
   **Git branch =>**( sari branches show krta hai or jis branch pr hein usse green color me show krdeta he)
   **Git checkout -b newbranch =>** git checkout -b new branch bnata hai or us branch pr shift b krdeta hai or git checkout bus branch pr shift krta hai
   **Git merge newbranch =>** Merge different branches

**Git branch -D newbranch =>** use to delete branch

-------------------------------------------------------------------------------------------------------------------------------

9. **Write down 5 steps of projects creation.**
   - Start a new new android studio project
   - Select empty activity
   - Type name
   - Type package name

- Select path for saving project
- Select language
- Select minimum API level

11. **What are intent filters?**
    App activities, services, and broadcast receivers are activated by intents. An intent is a message defined by an Intent object that describes an action to perform, including the data to be acted upon, the category of component that should perform the action, and other instructions.
    An app component can have any number of intent filters (defined with the <intent - filter> element), each one describing a different capability of that component.

12. **What is AAPT ?**

AAPT2 (Android Asset Packaging Tool) is a build tool that Android Studio and Android Gradle Plugin use to compile and package your app's resources. AAPT2 parses, indexes, and compiles the resources into a binary format that is optimized for the Android platform

13. **What is Gradle? Write its properties**
    Gradle is an open-source build automation tool focused on flexibility and performance. Gradle build scripts are written using a Groovy or Kotlin DSL Domain - Specific Languages
    **Highly customizable —** Gradle is modeled in a way that is customizable and extensible in the most fundamental ways.
    **Fast —** Gradle completes tasks quickly by reusing outputs from previous executions, processing only inputs that changed, and executing tasks in parallel.
    **Powerful** — Gradle is the official build tool for Android, and comes with support for many popular languages and technologies.
14. **Why Gradle Required?**
    Every Android project needs a Gradle for generating an apk from the .java and .xml files in the project. Simply put, a gradle takes all the source files (java and XML) and applies appropriate tools, e.g., converts the java files into dex files and compresses all of them into a single file known as apk that is actually used.
15. **What are the types of build.gradle scripts?**
    There are two types of build.gradle scripts:
    **Top-level build.gradle**: It is located in the root project directory and its main function is to define the build configurations that will be applied to all the modules in the project.
    **Module-level build.gradle:** Located in the project/module directory of the project this Gradle script is where all the dependencies are defined and where the SDK versions are declared. This script has many functions in the project which include additional build types and override settings in the main/app manifest or top-level build.gradle file.

16. **What is View and ViewGroups ?**

    **View:** A View usually draws something the user can see and interact with

    The View objects are usually called "widgets" and can be one of many subclasses, such as Button or TextView

    **ViewGroups:** A ViewGroup is Invisible container

    The ViewGroup objects are usually called "layouts" can be one of many types that provide a different layout structure, such as . LinearLayout or ConstraintLayout

17. **What are Layout Parameters?**

    XML layout attributes named layout_something define layout parameters for the View that are appropriate for the ViewGroup in which it resides Every ViewGroup class implements a nested class that extends ViewGroup.LayoutParams. This subclass contains property types that define the size and position for each child view, as appropriate for the view group.

18. **What is match_Parent Attribute?**

    MATCH_PARENT means that the view wants to be as big as its parent, minus the parent's padding

19. **What is wrap_content Attribute?**

    WRAP_CONTENT means that the view wants to be just large enough to fit its own internal content, taking its own padding into account.

20. **What is match_constraint attribute?**

    It means, that the view will take up as much space as the Constraints allow it to take. So, if I constrain a view to be between the left and right edge of the screen, then the view will expand its width to be equal to the width of the screen (if no margins are set).

21. **What is Barrier?**

    A Barrier references multiple widgets as input, and creates a virtual guideline based on the most extreme widget on the specified side. For example, a left barrier will align to the left of all the referenced views

22. **What is dp (density independent pixels)? Why it is used?**

    Density-independent pixels, written as dp (pronounced as “dips”) are flexible units that scale to have uniform dimensions on any screen. They provide a flexible way to accommodate a design across platforms.

    ➔ Material UIs use dp to display elements consistently on screens with different intensities.

23. **What are Vector assets?**

    Vector Assets in Android Studio helps to add material icons and import Scalable Vector Graphics and Adobe Photoshop Document files into your project as vector drawable resources.

**24. What is the difference between native and hybrid applications?**

To put it plainly, the main difference between hybrid and native apps is this:
Hybrid apps are developed across all platforms, while native apps are developed for specific operating system.

**25. What is an activity?**

An activity is a single, focused thing that the user can do. Almost all activities interact with the user, so the Activity class takes care of creating a window for you in which you can place your UI with setContentView(View).

- An Activity is an application component
- Represents one window, one hierarchy of views
- Typically fills the screen, but can be embedded in other Activity or a appear as floating window
- Java class, typically one Activity in one file

**26. Write functions of activity?**

- Handles user interactions, such as button clicks, text entry, or login verification
- Can start other activities in the same or other apps
- Has a life cycle—is created, started, runs, is paused, resumed, stopped, and destroyed

**27. What is the difference between task and back track?**

A task is a collection of activities that users interact with when trying to do something in your app. These activities are arranged in a stack—the back stack—in the order in which each activity is opened. For example, an email app might have one activity to show a list of new messages. When the user selects a message, a new activity opens to view that message. This new activity is added to the back stack. Then, if the user presses or gestures Back, that new activity is finished and popped off the stack.

Best of luck 😊