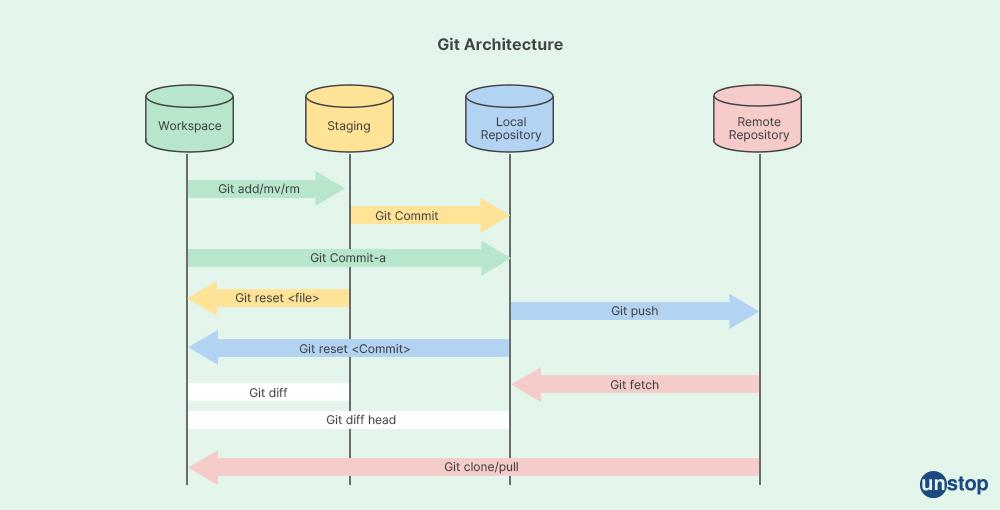
**Explain the Architecture of Git.**

Git, a distributed version control system, has a unique architecture that sets it apart from other version control systems. Its architecture can be understood by breaking down its key components and concepts:



### **1. Repository**

A Git repository is where all the project's files and the entire history of changes are stored. It has two main states:

* **Local Repository**: Stored on your local machine.
* **Remote Repository**: Stored on a remote server, such as GitHub or GitLab.

### **2. Working Directory**

This is the actual directory (folder) on your file system where you create and edit files. It reflects the content of the repository at a particular point in time (a specific commit).

### **3. Staging Area (Index)**

The staging area is a file (usually located in the Git directory) that stores information about what will go into your next commit. Think of it as a preparation area where you can group changes before creating a commit.

### **4. Git Directory (Repository)**

This is the directory where Git stores all the metadata and object database for your project. It includes subdirectories like objects, refs, and configuration files.

### **5. Commits**

Commits are snapshots of your repository at a specific point in time. Each commit is identified by a unique SHA-1 hash. A commit stores:

* A pointer to its parent commit(s).
* A tree object that represents the state of the working directory.
* Metadata such as the author, committer, and commit message.

### **6. Branches**

Branches are pointers to commits. The default branch is usually named main or master. Branches allow for parallel development. Each branch is essentially a separate line of development.

### **7. Tags**

Tags are references that point to specific points in Git history. They are often used to mark release points (e.g., v1.0, v2.0).

### **8. Objects**

Git's object model consists of four types of objects:

* **Blobs**: Store file data.
* **Trees**: Store directory structures and blob references.
* **Commits**: Store metadata about changes, including pointers to tree objects.
* **Annotated Tags**: Store tag data, including pointers to commits.

### **9. References (Refs)**

References are pointers to commits (or other objects). Common types of refs include:

* **Heads**: Represent branches.
* **Tags**: Represent specific points in history.
* **Remotes**: Track branches in remote repositories.

### **10. Remote Repositories**

These are versions of your project hosted on the internet or network, allowing multiple developers to collaborate. Common commands related to remotes include:

* git fetch: Download objects and refs from another repository.
* git pull: Fetch from a remote repository and merge.
* git push: Update the remote repository with your local changes.