**Git (Global Information Tracker)**

Git is a powerful tool used to manage project source code effectively. It is a Version Control System (VCS) that tracks changes in code, helps collaborate with other developers, and maintains a history of modifications.

Why Use Git?

1. Version Control: Maintain multiple versions of the same file, track changes, and understand why modifications were made.
2. Collaboration: Enable multiple developers to work on the same project without overwriting each other’s changes.

Key Features of Git:

* Collaboration: Multiple developers can simultaneously work on code.
* History Maintenance: Keeps a record of every version of files.
* Prevents Overwriting: Ensures changes by different developers don’t conflict.
* Backup: Works as a backup system for project code.
* Speed: Faster compared to other version control tools.

Types of Version Control Systems (VCS):

1. Centralized VCS (CVCS):
   * Example: SVN (Subversion Control System)
   * Developers connect to a central server to work on the project.
   * Disadvantages:
     + Vulnerable to data loss if the server is destroyed.
     + Data can be copied directly by anyone accessing the server.
2. Decentralized/Distributed VCS (DVCS):
   * Example: Git
   * Developers create local repositories and can sync with remote repositories (e.g., GitHub).
   * Git allows cloning code from remote repositories for local development.

Key Terminologies:

1. Repository: A storage space for project files. GitHub can have multiple repositories, each corresponding to a project.
2. Local Repository: A copy of the project repository stored on your local machine.
3. Remote Repository: A version of the repository hosted on platforms like GitHub or GitLab.
4. Cloning: Downloading a remote repository to a local repository.
5. Fork: Copying a project from one GitHub account to another.
6. Push: Sending files from a local repository to a remote repository.
7. Pull: Fetching files from a remote repository to a local repository.

Git Protocols:

1. HTTPS: Secure protocol for Git operations.
2. SSH: Provides a secure way to connect to repositories using unique keys.

Basic Git Commands:

1. Checking Git Installation:
2. git --version
3. Configuring Git:
4. git config --global user.name "Your Name"
5. git config --global user.email "youremail@example.com"
6. git config --global --list # Check configuration
7. Initializing a Repository:
8. git init
9. Adding Files to Git:
   * Add a specific file:
   * git add filename.py
   * Add all files:
   * git add -all
10. Committing Changes:
11. git commit -m "Your commit message"
12. Checking the Status:
13. git status
14. Changing Branches:
15. git init -b branchname
16. Removing Files from Git:
17. git rm --cached filename
18. Viewing Differences Before Committing:
19. git diff

Creating an SSH Key:

1. Create an SSH Folder:
2. mkdir ssh
3. cd ssh
4. Generate SSH Key:
5. ssh-keygen -o
6. Retrieve and Add Key to GitHub:
7. cat ~/.ssh/id\_ed25519.pub
   * Copy the generated key and add it to your GitHub account.

Git is an essential tool for version control and collaboration. It is widely used in software development to manage code efficiently and ensure team collaboration without conflicts. Platforms like GitHub provide a remote repository system to enhance the functionality of Git.