Thursday, 18 April 2024

Basics of Git

What is Git?

Git is a distributed version control system used for tracking changes in files and coordinating work among multiple contributors.

It allows users to collaborate on projects efficiently by managing revisions, facilitating branching and merging, and enabling remote collaboration.

Why Use Git?

Git provides a structured and organized way to manage code changes, allowing developers to track modifications, revert to previous versions, and collaborate seamlessly.

It offers features such as branching and merging, which enable parallel development and experimentation without disrupting the main codebase.

Git's distributed nature allows for offline work and decentralized collaboration, making it suitable for both individual and team projects.

Logical Separation in Git:

1. Workspace (Working Directory):

The workspace refers to the directory on your local machine where you create, edit, and organize your project files.

It contains the current state of your project, including modified files and new additions.

2. Staging Area (Index):

The staging area acts as an intermediate step between the workspace and the repository.

It serves as a holding area where changes to files are prepared (staged) before they are committed to the repository.

Files in the staging area are ready to be included in the next commit.

3. Repository (Local and Remote):

The repository is where Git stores the complete history of a project, including all versions of files and their associated metadata.

It consists of two components: the local repository stored on your machine and the remote repository hosted on a server (e.g., GitHub, Bitbucket).

The local repository contains all commits, branches, and tags related to the project, while the remote repository allows for collaboration and synchronization with other contributors.

Common Git Commands:

git init:

Initializes a new Git repository in the current directory, creating a .git subdirectory to store versioning information.

git status:

Displays the current status of the repository, showing which files are modified, untracked, or staged for commit.

git add:

Adds files or changes in the workspace to the staging area, preparing them for the next commit.

Syntax: git add <file> (add a specific file), git add . (add all changes), git add -p (interactively add changes).

git commit:

Records the staged changes in the repository, creating a new commit with a unique identifier (SHA-1 hash) and a commit message describing the changes.

Syntax: git commit -m "Commit message"

git remote add:

Adds a remote repository as a named alias, allowing you to reference it by a convenient name (e.g., "origin").

Syntax: git remote add <name> <url>

git push:

Pushes commits from the local repository to a remote repository, synchronizing changes with the server.

Syntax: git push <remote> <branch>

These Git commands form the foundation of version control workflow, allowing developers to manage project history, collaborate with others, and maintain code integrity effectively.

How to upload a file or folder into git ?

1. Navigate inside the folder where the files are present.
2. Initialise a git repo with command : *git init*
3. Add the files to staging area using command : *git add filename*
4. Commit the files in staging are using command : *git commit -m "message"*
5. Add the remote to central github repo*: git remote add <remote-name> <url>*
6. Push local repo to remote : *git push <remote-name> <branch-name>*