**A1**. Demonstrate and Create project in local and remote repository using GitBash and GitHub and apply  init, status, log, add, commit, push, config, clone and reset commands on repository.

Git **init** command:

Description

The git init command is used to initialize a new Git repository. This creates a .git subdirectory in your project folder that contains all the necessary metadata for version control.

Syntax : git init

Example : mkdir my\_project

cd my\_project

git init

Git **status** command:

Description

The git status command shows the current state of the working directory and staging area. It displays changes that are staged for the next commit, changes that have been made but not yet staged, and files that are not being tracked by Git.

Syntax : git status

Example : git status

Git **log** command:

Description

The git log command displays the commit history for the current repository. It shows the commits, along with their commit IDs, authors, dates, and commit messages.

Syntax : git log

Example : git log

Git **add** command:

Description

The git add command adds changes in your working directory to the staging area, preparing them for a commit. You can add individual files or all modified files.

Syntax : git add <file\_name>

Example : git add index.html

Git **commit** command:

Description

The git commit command records the staged changes to the repository with a commit message. This is where you save the state of the repository.

Syntax : git commit -m "Your commit message"

Example : git commit -m "Added new feature for user authentication"

Git **push** command:

Description

The git push command uploads local repository content to a remote repository, like GitHub. You need to have set up a remote repository for this command to work.

Syntax : git push <remote\_name> <branch\_name>

Example : git push origin master

Git **config** command:

Description

The git config command is used to set configuration options for Git, such as user details (name, email), editor, and more.

Syntax : git config --global user.name "Your Name"

git config --global user.email "youremail@example.com"

Example : git config --global user.name "John"

git config --global user.email "john@example.com"

Git **clone** command:

Description

The git clone command creates a copy of an existing remote repository (from GitHub or other hosting services) on your local machine

Syntax : git clone <repository\_url>

Example : git clone https://github.com/username/repository.git

Git **Reset** command:

Description

The git reset command undoes changes by resetting the current HEAD to a specified state. This can be used to unstage files or undo commits.

Syntax : git reset <commit\_id>

Example : git reset --hard HEAD~1

**Git user configuration details**

Git relies on user configuration settings to associate commits with a specific identity. This is important because Git uses the name and email address to record who made changes to a repository. These details are stored in Git's configuration files, and they are used whenever you commit changes to a repository. There are three primary places where you can configure Git user information:

1. Global Configuration: Applies to all repositories on your system.
2. Local Configuration: Specific to a single repository.
3. System Configuration: Applies to all users on the system (rarely used directly by most developers)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Configuration** | **Execution** | **Viva** | **Total** | **Verified By** |
|  |  |  |  |  |