**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

Initializes a new Git repository in your current directory. It creates a .git folder that tracks all changes in the project

Syntax : git init

Example : git init

Git **status** command:

Description

Shows the current status of the working directory and staging area. It displays which files are staged, unstaged, or untracked.

Syntax : git status

Example : git status

Git **log** command:

Description

Displays the commit history of the current branch.

Syntax : git log

Example : git log

Git **add** command:

Description

Adds file changes in the working directory to the staging area for the next commit.

Syntax : git add <file-name> git add .

Example : git add index.html

git add .

Git **commit** command:

Description

Records the changes in the staging area into the repository with a descriptive message.

Syntax : git commit -m "commit message"

Example : git commit -m "Initial commit"

Git **push** command:

Description

Uploads your local repository commits to a remote repository like GitHub

Syntax : git push <remote> <branch>

Example : git push origin main

Git **config** command:

Description

Used to set Git configuration values like user name, email, editor, etc. Sets the user name and email globally for all repos.

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

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

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

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

Git **clone** command:

Description

Creates a local copy of a remote repository. Clones the remote repository repo.git to your local machine.

Syntax : git clone <repository-url>

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

Git **Reset** command:

Description

Undoes changes by resetting the current HEAD to a specified state. Can reset staging area and working directory depending on options.

Syntax : git reset --soft <commit>

git reset --mixed <commit>

git reset --hard <commit>

Example : git reset --hard HEAD~1

**Git user configuration details**

**Step 1: Configure Git User**

git config --global user.name " **HarshithK-froxcy**"

git config --global user.email "froxcy31@gmail.com"

**Step 2: Create a new local project**

mkdir myproject

cd myproject

git init

**Step 3: Create a file**

echo "Hello, Git!" > readme.txt

**Step 4: Check status and add file**

git status

git add readme.txt

**Step 5: Commit changes**

git commit -m "hello"

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