

Experiment 03

Aim: To Perform various GIT operations on local and Remote repositories

Theory:

The command `mkdir git` creates a new directory (folder) named "git" in the current working directory. This command is used to make a new directory in a Unix-like operating system.

The command `cd git` is used to change the current working directory to the directory named "git." After executing this command, any subsequent commands or file operations will occur within the "git" directory. "cd" stands for "change directory."

The `git config --global user.name` and `git config --global user.email` commands are used to set your global Git username and email address, respectively. They are part of the configuration settings in Git and are associated with the commits you make.

If you want to check your configuration settings, you can use the `git config --list` command to list all the settings Git can find at that point `git commit -am "commit message"` stages and commits all changes in tracked files with a commit message in a single command.

The command `nano index.html` opens the Nano text editor for the file named "index.html." Nano is a simple command-line text editor that allows you to view and edit files directly in the terminal.

The command `touch teststatus` creates an empty file named "teststatus" in the current directory. The touch command is commonly used to update the timestamps of a file or create an empty file if it doesn't exist.

`git checkout -- teststatus`: Discards changes to the file "teststatus" in the working directory. This reverts the file to the state it has in the last commit.

The `git add` command is used to stage changes in the working directory for the next commit in Git. It prepares modifications, additions, or deletions to be included in the upcoming commit.

The `git log` command is used to display the commit history of a Git repository. It shows a chronological list of commits, including commit hashes, author information, timestamps, and commit messages.

The command `git log --oneline` displays a simplified and concise one-line representation of the commit history in a Git repository, showing only the commit SHA-1 hash and the commit message.

The `git clone` command is used to create a copy of a Git repository. When you run this command, it duplicates the entire repository, including its files, commit history, and branches, and downloads it to your local machine. This is often the initial step when you want to work with a project hosted on a remote Git repository.

The `git pull` command is used to fetch and integrate changes from a remote repository into the current branch of your local repository. It combines two actions: it fetches the changes from the remote repository, and then it automatically merges those changes into your local branch. This is a convenient way to update your local repository with the latest changes from the remote repository.

The `git push` command is used to upload or push the local changes in your Git repository to a remote repository. It updates the remote repository with the latest changes made in your local branch, making them accessible to others who share the same remote repository.

The `git fetch` command is used to retrieve changes from a remote repository. It fetches any new branches or changes made in the remote repository since your last interaction. However, it does not automatically merge these changes into your local branches. After using `git fetch`, you can inspect the changes and decide whether to integrate them using `git merge` or `git rebase`.

Output:

```
MINOR@kali:~/Desktop/git-demo-project$
MINOR@kali:~/Desktop/git-demo-project$ git config --global user.name "dhanishth"
MINOR@kali:~/Desktop/git-demo-project$ git config --global user.email "dhanishth40@gmail.com"
MINOR@kali:~/Desktop/git-demo-project$ git config --global --list
user.name=dhanishth
user.email=dhanishth40@gmail.com
MINOR@kali:~/Desktop/git-demo-project$ cat ~/.gitconfig
[user]
  name = dhanishth
  email = dhanishth40@gmail.com
MINOR@kali:~/Desktop/git-demo-project$ mkdir git-demo-project
MINOR@kali:~/Desktop/git-demo-project$ cd git-demo-project/
MINOR@kali:~/Desktop/git-demo-project$ git init
Initialized empty Git repository in C:/Users/L1/Desktop/git-demo-project/.git/
MINOR@kali:~/Desktop/git-demo-project$ ls -la
.
..
.git/
MINOR@kali:~/Desktop/git-demo-project$ ls -la
total 4
drwxr-xr-x 1 L1 18720 0 Jan 29 11:00 ./
drwxr-xr-x 1 L1 18720 0 Jan 29 11:00 ../
drwxr-xr-x 1 L1 18720 0 Jan 29 11:00 .git/
MINOR@kali:~/Desktop/git-demo-project$ git init
Initialized existing Git repository in C:/Users/L1/Desktop/git-demo-project/.git/
MINOR@kali:~/Desktop/git-demo-project$ git add
nothing specified, nothing added.
hint: Make sure you want to say "git add ."
hint: Use the message with the commit
hint: "git commit -m 'add myPathname'"
MINOR@kali:~/Desktop/git-demo-project$ git add .
MINOR@kali:~/Desktop/git-demo-project$ git status
On branch master
no commits yet
nothing to commit (create/copy files and use "git add" to track)
MINOR@kali:~/Desktop/git-demo-project$ git commit -m "First Commit"
[initial commit]
nothing to commit (create/copy files and use "git add" to track)
```

```
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ nano index.html
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  index.html

nothing added to commit but untracked files present (use "git add" to track)
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git commit -m "First Commit"
[master (root-commit) 109d350] First Commit
1 file changed, 1 insertion(+)
create mode 100644 index.html
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git log
commit 109d350dc7b71a1a1c3f07a22a01993c5fc32 (HEAD -> master)
Author: dhanistha <dhanistha4@gmail.com>
Date:   Mon Jan 29 11:12:48 2023 -0530

    First Commit
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git log --oneline
109d35d (HEAD -> master) First Commit
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git log --oneline index.html
109d35d (HEAD -> master) First Commit
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ |
```

```
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git clone https://github.com/dhanisthajewani1910/python-april23.git
Cloning into 'python-april23'...
remote: Enumerating objects 15, done.
remote: Counting objects 100% (15/15), done.
remote: Compressing objects 100% (15/15), done.
remote: Total 15 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects 100% (15/15), done.
Resolving deltas 100% (3/3), done.
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ ls
index.html  python-april23/
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git remote add origin https://github.com/dhanisthajewani1910/python-april23.git
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git remote show origin
* remote origin
Fetch URL: https://github.com/dhanisthajewani1910/python-april23.git
Push URL: https://github.com/dhanisthajewani1910/python-april23.git
HEAD branch: main
Remote branch:
main new (next fetch will store in remotes/origin)
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git pull
remote: Enumerating objects 15, done.
remote: Counting objects 100% (15/15), done.
remote: Compressing objects 100% (15/15), done.
remote: Total 15 (delta 0), reused 5 (delta 0), pack-reused 0 (from 0)
Unpacking objects 100% (15/15), 3.93 KiB | 731.00 KiB/s, done.
From https://github.com/dhanisthajewani1910/python-april23
 * [new branch]      main -> origin/main
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=origin/<branch> master
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ git fetch
1150201-006 WIN64@ ~/desktop/git-dcvs/git-demo-project (master)
$ |
```

Conclusion:

Thus, we have successfully studied and performed various GIT operations on local and Remote repositories.