Name: Shrirang Zend **Batch: T2-3** **Roll Number: 2201125**

Experiment 3: To Perform various Git operations on local and remote repositories using Git cheat sheet.

THEORY:

Introduction to Git

Git is a distributed version control system used for tracking changes in source code. It allows multiple developers to work on a project simultaneously while keeping track of changes and enabling collaboration through remote repositories like GitHub, GitLab, and Bitbucket.

Configuring Git

Before using Git for the first time, it is necessary to configure the user's identity. The following commands set up the user's name and email, which will be associated with all commits:

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

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

The `--global` flag ensures that the configuration applies to all repositories on the system.

Initializing a Git Repository

A Git repository must be initialized before tracking changes. This is done using the `git init` command:

git init

Executing this command creates a hidden `.git` directory within the project folder, which stores all version control information.

Checking the Status of a Repository

To check the current state of the repository, including untracked and modified files, use:

git status

This command provides an overview of changes that need to be staged, committed, or pushed.

Adding Files to the Staging Area

Before committing changes, files must be added to the staging area. This can be done using:

git add <file_name> # Adds a specific file

git add . # Adds all modified and new files

The staging area acts as an intermediate step before committing changes.

Committing Changes

A commit captures the current state of the repository and saves it locally. Each commit requires a message that describes the changes made:

git commit -m "Descriptive commit message"

Commits are local and do not affect the remote repository until they are pushed.

Connecting to a Remote Repository

To link the local repository with a remote repository (e.g., GitHub), use:

git remote add origin <repository_URL>

For example:

git remote add origin https://github.com/username/repository.git

To verify that the remote repository has been added, use:

git remote -v

Pushing Changes to a Remote Repository

To upload commits to a remote repository:

git push origin main

- `origin` refers to the remote repository.
- `main` refers to the branch being pushed.

For the first push, use:

git push -u origin main

The `-u` flag sets `origin main` as the default upstream branch, allowing future pushes to be done with `git push` alone.

Pulling Changes from a Remote Repository

To retrieve and merge updates from the remote repository:

git pull origin main

This command ensures the local repository is up-to-date with the remote repository.

Cloning an Existing Repository

To create a local copy of an existing remote repository:

git clone <repository_URL>

For example:

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

This command downloads the repository and sets up a connection to the remote repository.

Branching and Merging

Git allows working with multiple branches to develop new features without affecting the main codebase.

Creating a new branch:

git branch new-branch

Switching to the new branch:

git checkout new-branch

Merging a branch into the main branch:

git merge new-branch

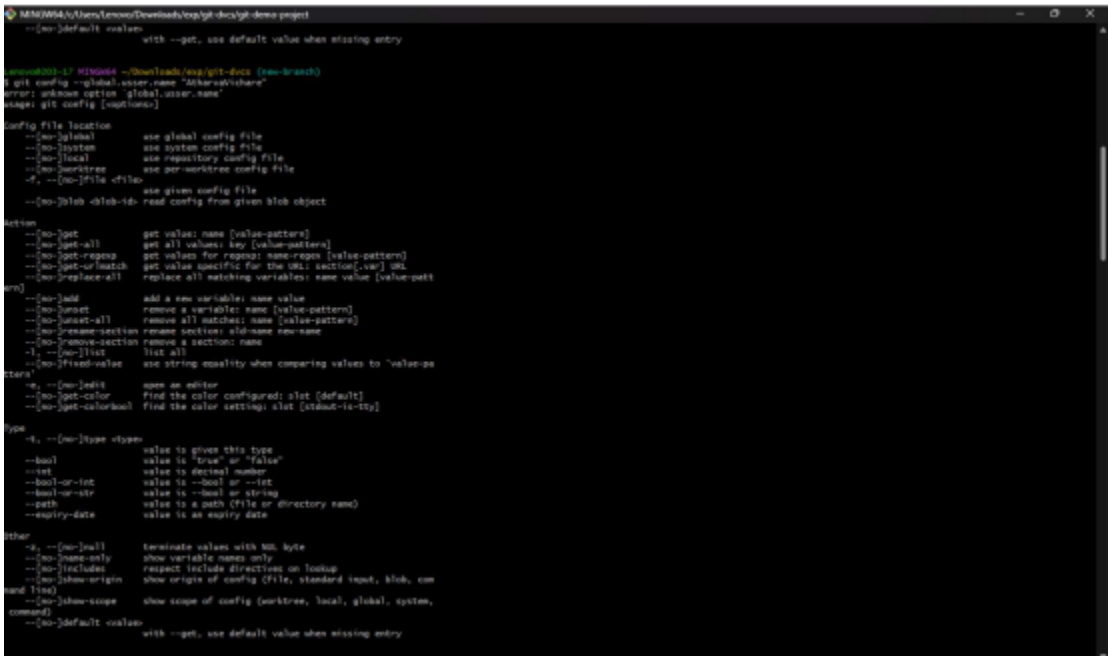
Deleting a branch:

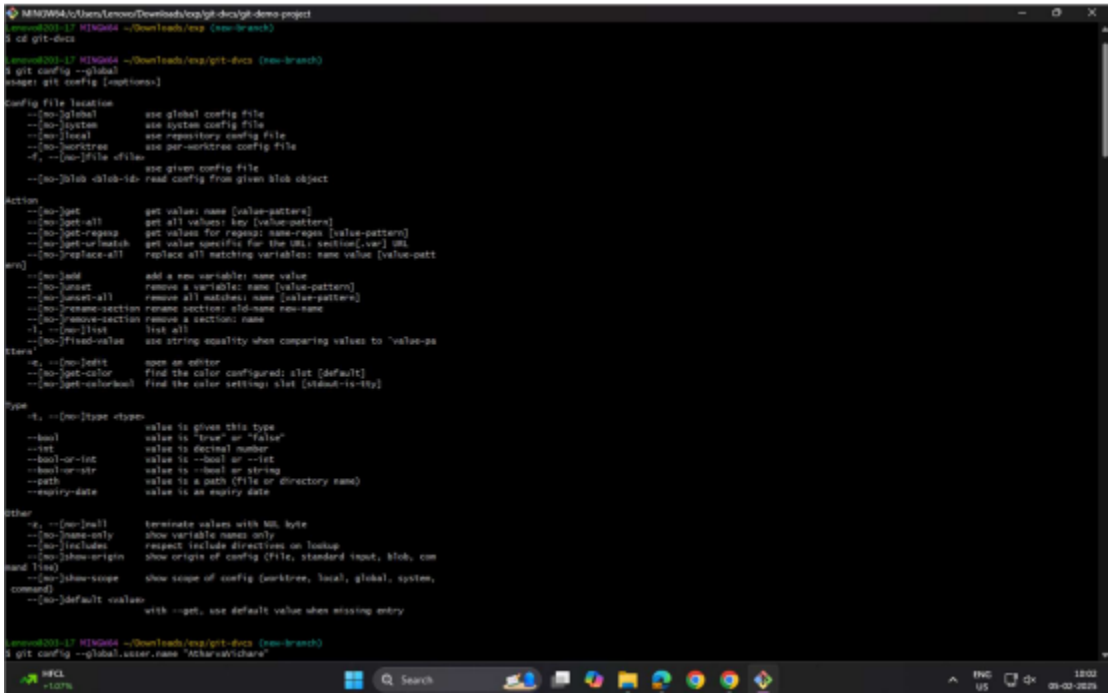
git branch -d new-branch

Branches help in parallel development and version control management.

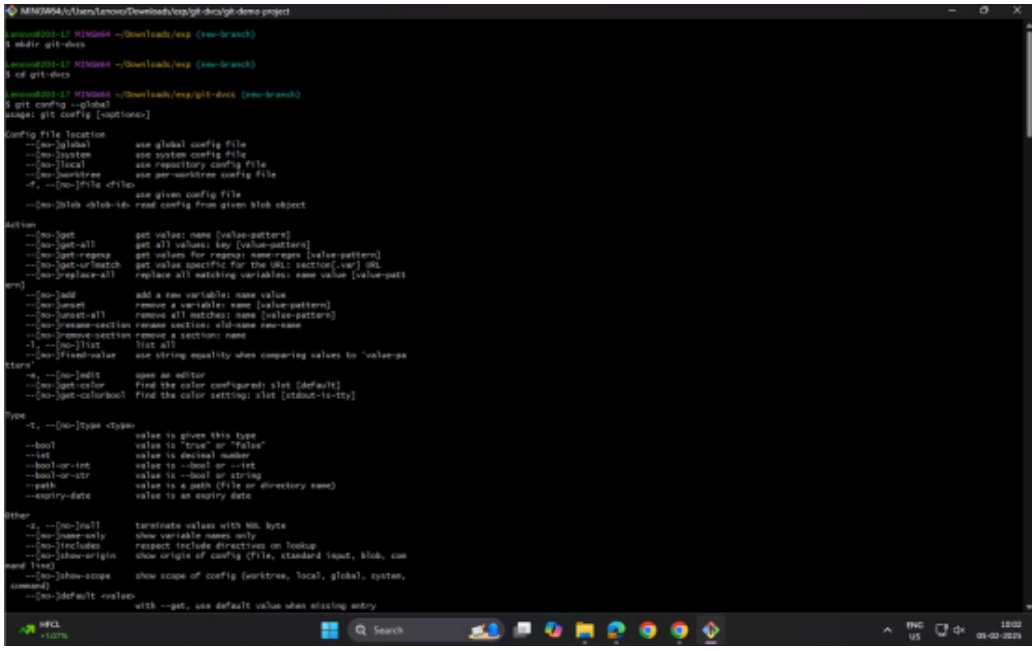
Name: Shrirang Zend **Batch: T2-3** **Roll Number: 2201125**

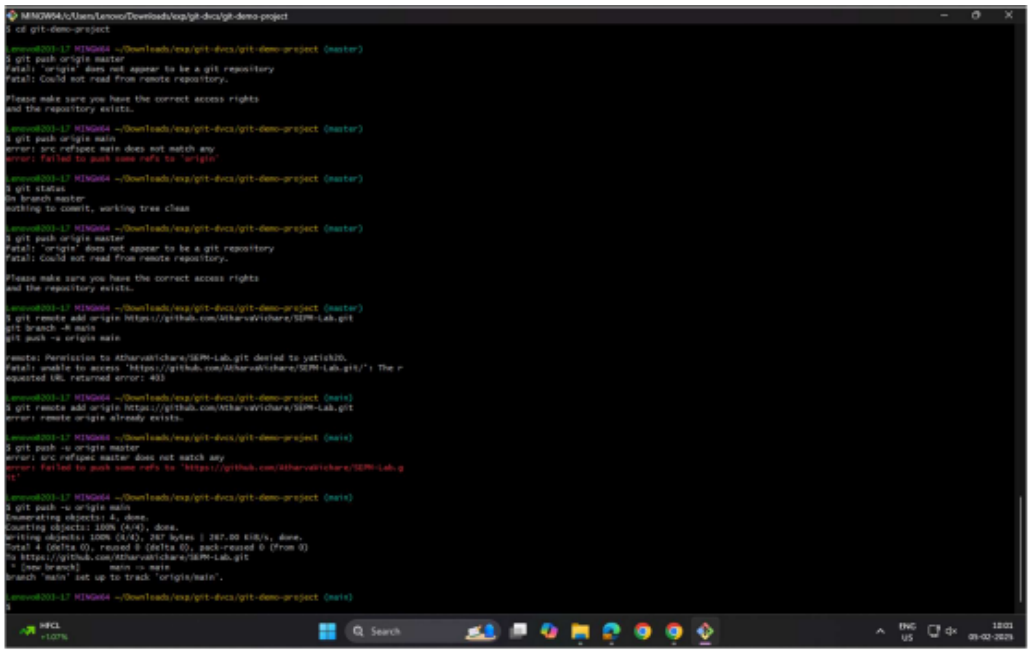
Implementation:





Name: Shrirang Zend **Batch: T2-3** **Roll Number: 2201125**





.

Conclusion:

Successfully implemented various Git operations on local and remote repositories using the Git cheat sheet.