PURBANCHAL UNIVERSITY

KHWOPA ENGINEERING COLLEGE

LIBALI-08, BHAKTAPUR

LAB REPORT ON .NET

LAB NO. 01

SUBMITTED BY:

Name: Jenish Prajapati

Roll No. : 770314

Group: A

SUBMITTED TO:

Department of Computer Engineering

Submission: 2081/12/09

Theory:

1. **Git:**
   Git is a distributed version control system used to track changes in source code during software development. It allows multiple developers to collaborate on a project by enabling them to work on different parts of the code simultaneously. Git tracks the history of changes, making it easy to revert to previous versions of the code, merge updates from different contributors, and manage branches for different features or bug fixes. Popular platforms like GitHub and GitLab are built on Git, allowing developers to host and share their repositories online.

2. **GitHub**
   GitHub is a web-based platform used for version control and collaboration, allowing developers to store, manage, and track changes to their code. It uses Git, a distributed version control system, to enable multiple contributors to work on a project simultaneously. GitHub also provides features like issue tracking, pull requests, and project management tools, making it popular for open-source software development and team collaboration.

General Git and GitHub Commands:

Git Configuration

git config --global user.name "Your Name"
This command sets the global username for the Git commits.
git config --global user.email "your_email@example.com"
This command sets the global email associated with Git commits.

Initializing

git init
initializes a new Git repository in the current directory.

Staging and Commits

git add .
It stages all changes and new files for commit.
git commit -m "Your commit message"
Saves the staged changes with a descriptive message.

Branching and Merging

git branch
Lists all the branches in the repository.
git branch <branch_name>
Creates a new branch for separate development.
git checkout <branch_name> / Git switch <branch_name>
Switches to the specified branch

git merge <branch_name>
Merges changes from the specified branch into the current branch.

Pushing and Pulling

git push -u origin <branch_name>
Uploads the local changes to the remote repository.
git pull origin <branch_name>
Fetches and merge the latest changes from the remote repository.

Status and Logs

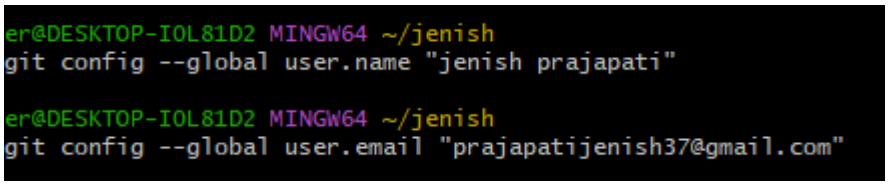
git status
Show the current state of the files in the working directory (modified, staged or untracked).
git log
Displays the commit history of the repository.

GitHub Specific

git remote add origin <repo_url>
Links the local repository to a remote repository on GitHub.

Lab Works

Set the global username and email of the GitHub.



Create a folder and files as per the desire so that we can identify the changes inside the file using the version control (Git).

```
 C:\Users\Acer\Desktop\dotnet> git init
itialized empty Git repository in C:/Users/Acer/Desktop/dotnet/.git/
 C:\Users\Acer\Desktop\dotnet> git status
 branch master

 commits yet

tracked files:
(use "git add <file>..." to include in what will be committed)
      test.py
      test.txt

thing added to commit but untracked files present (use "git add" to track)
 C:\Users\Acer\Desktop\dotnet> git add
thing specified, nothing added.
nt: Maybe you wanted to say 'git add .'?
nt: Disable this message with "git config set advice.addEmptyPathspec false"
 C:\Users\Acer\Desktop\dotnet> git status
 branch master

 commits yet

tracked files:
      test.py
      test.txt
```

On creating the new files, initially the files are in the untracked stage so sent the untracked files to the staging stage. To do so first initialize the directory and staged the files.

Now commit the files such that the files are stored in the local repository.

```
S C:\Users\Acer\Desktop\dotnet> git commit -m "initial commi
"
master (root-commit) 9c7a953] initial commit
2 files changed, 1 insertion(+)
create mode 100644 test.py
create mode 100644 test.txt
S C:\Users\Acer\Desktop\dotnet>
```

Make certain changes inside the file to see the changes in the file status.

```
PS C:\Users\Acer\Desktop\dotnet> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working
directory)
        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit
-a")
PS C:\Users\Acer\Desktop\dotnet>
```

After changing the contents in the file **“text.txt”** add the file and commit it.

All of these files are saved in the local repository. Now to add these files in the remote repository create the repository in the GitHub and copy the url of the repo and use the following code.

```
S C:\Users\Acer\Desktop\dotnet> git remote
dd origin https://github.com/jenishpraz/tes
.git
S C:\Users\Acer\Desktop\dotnet>
```

Now push the files in the repository created.

```
PS C:\Users\Acer\Desktop\dotnet> git push or
igin master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (7/7), 526 bytes | 263
.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-
reused 0 (from 0)
To https://github.com/jenishpraz/test.git
 * [new branch]      master -> master
PS C:\Users\Acer\Desktop\dotnet>
```

Now creating branches, allowing the work on different version of a project without affecting the main codebase.

```
S C:\Users\Acer\Desktop\dotnet> git checkou
 developer
witched to branch 'developer'
S C:\Users\Acer\Desktop\dotnet>
```

Moving on to the recently created branch to modify the contents in the file without affecting the main codebase.

```
PS C:\Users\Acer\Desktop\dotnet> git branch

  * developer
    master
PS C:\Users\Acer\Desktop\dotnet> git add .
PS C:\Users\Acer\Desktop\dotnet> git status
On branch developer
Changes to be committed:
  (use "git restore --staged <file>..." to u
nstage)
        new file:   hell.py

PS C:\Users\Acer\Desktop\dotnet>
```

To change the branch, we use the command *"git switch main"*. To make sure the branch is visible to other users of the repository push the branch in the GitHub.

```
PS C:\Users\Acer\Desktop\dotnet> git push -u
 origin developer
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 312 bytes | 156
.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-
reused 0 (from 0)
To https://github.com/jenishpraz/test.git
   b501226..48a7643  developer -> developer
branch 'developer' set up to track 'origin/d
eveloper'.
PS C:\Users\Acer\Desktop\dotnet>
```

Merging branches ensures that the changes or new features added in a new branch are integrated into the main codebase.

```
PS C:\Users\Acer\Desktop\dotnet> git merge d
eveloper
Already up to date.
```

Use the "***git log***" command to view **past commits and their details**.

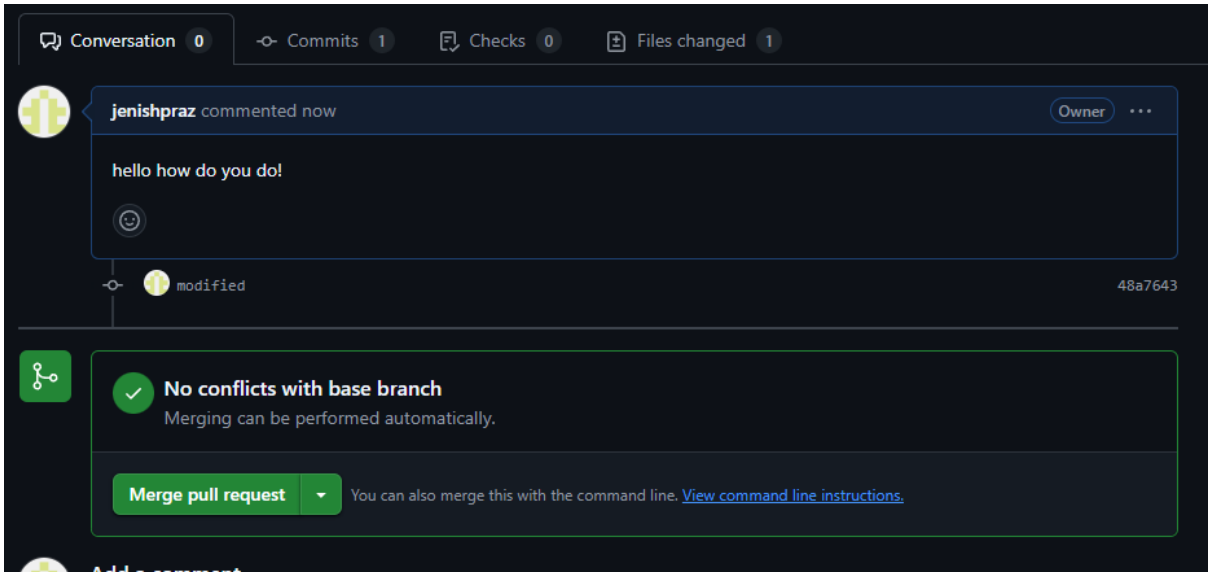
```
PS C:\Users\Acer\Desktop\dotnet> git log
commit b501226de62a97f39b5b93ec0ca930ca64a82
1f1 (HEAD -> developer, origin/master, origi
n/developer, master)
Author: jenish prajapati <prajapatijenish37@
gmail.com>
Date:   Sat Mar 22 22:29:07 2025 +0545

    initial commit

commit 9c7a95391a2bc68b0da3cf40c5ac4e0deccfe
591
Author: jenish prajapati <prajapatijenish37@
gmail.com>
Date:   Sat Mar 22 22:21:26 2025 +0545

    initial commit
PS C:\Users\Acer\Desktop\dotnet>
```

Merging the branch in the GitHub (Web)



Conclusion:

In this lab, we explored the fundamentals of Git and GitHub, including initialization, branching, merging, pushing, and committing. These concepts are essential for effective version control and collaborative software development.