PURBANCHAL UNIVERSITY

KHWOPA ENGINEERING COLLEGE

LIBALI-08, BHAKTAPUR

LAB REPORT ON .NET

LAB NO. 01

SUBMITTED BY:

Name: Nekesh Koju

Roll No. : 770321

SUBMITTED TO:

Department of Computer Engineering

Submission: 2081/12/10

Theory:

1. **Git:**
   Git is a distributed version control system used for tracking the changes in the source code during software development. It allows multiple developers to collaborate efficiently by managing different version of project. Git enables branching, merging and reverting changes, making code management easier. It is widely used open-source and commercial projects. Popular platform like GitHub, GitLab, and Bitbucket provide remote repositories for Git-based collaboration.

2. **GitHub**
   GitHub is a web-based platform for version control and collaboration using Git. It allows developers to store, manage, and share code repositories efficiently. GitHub supports features like branching, pull requests, issue tracking, and CI/CD integration. It is widely used for open-source and private projects, enabling seamless teamwork. GitHub also provides cloud-based hosting, making it accessible from anywhere.

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

First set the global username and email of the GitHub.

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

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.

```
PS D:\Dotnetlab\lab1> git init
Reinitialized existing Git repository in D:/Dotnetlab/lab1/.git/
PS D:\Dotnetlab\lab1> git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
PS D:\Dotnetlab\lab1> git add.
git: 'add.' is not a git command. See 'git --help'.

The most similar command is
        add
PS D:\Dotnetlab\lab1> git add .
PS D:\Dotnetlab\lab1> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   test.py
        new file:   test.txt
```

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

```
PS D:\Dotnetlab\lab1> git commit -m "Initial commit"
[master (root-commit) bb23b22] Initial commit
 2 files changed, 2 insertions(+)
 create mode 100644 test.py
 create mode 100644 test.txt
```

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

```
PS D:\Dotnetlab\lab1> git status
On branch master
nothing to commit, working tree clean
PS D:\Dotnetlab\lab1> 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.py
        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS D:\Dotnetlab\lab1> git add test.py
PS D:\Dotnetlab\lab1> git add test.txt
PS D:\Dotnetlab\lab1> git commit -m "commit"
[master 0bf9f42] commit
 2 files changed, 3 insertions(+), 1 deletion(-)
```

After changing the contents in the file **"test.py & test.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.

```
PS D:\Dotnetlab\lab1> git remote add origin https://github.com/nekeshkoju07/test.git
```

Now push the files in the repository created.

```
PS D:\Dotnetlab\lab1> git push origin master
To https://github.com/nekeshkoju07/test.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/nekeshkoju07/test.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

To encounter this error message "git push –force origin " command is used

```
PS D:\Dotnetlab\lab1> git push --force origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 597 bytes | 597.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/nekeshkoju07/test.git
 + 00f7b78...0bf9f42 master -> master (forced update)
```

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

```
PS D:\Dotnetlab\lab1> git branch developer
PS D:\Dotnetlab\lab1> git branch
  developer
* master
```

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

```
PS D:\Dotnetlab\lab1> git  checkout developer
Switched to branch 'developer'
PS D:\Dotnetlab\lab1> git add .
PS D:\Dotnetlab\lab1> git status
On branch developer
nothing to commit, working tree clean
PS D:\Dotnetlab\lab1> git status
On branch developer
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 D:\Dotnetlab\lab1> git add .
PS D:\Dotnetlab\lab1> git status
On branch developer
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   test.txt

PS D:\Dotnetlab\lab1> git commit -m "chanes in the developer branch"
[developer fd22feb] chanes in the developer branch
 1 file changed, 1 insertion(+), 1 deletion(-)
```

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

```
PS D:\Dotnetlab\lab1> git switch master
Switched to branch 'master'
```

```
PS D:\Dotnetlab\lab1> git switch developer
Switched to branch 'developer'
PS D:\Dotnetlab\lab1> git push -u origin developer
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 608 bytes | 608.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'developer' on GitHub by visiting:
remote:      https://github.com/nekeshkoju07/test/pull/new/developer
remote:
To https://github.com/nekeshkoju07/test.git
 * [new branch]      developer -> developer
branch 'developer' set up to track 'origin/developer'.
```

Merging the branches such that the changes in the new branch or new features added in the new branch is added to the main code base.

```
PS D:\Dotnetlab\lab1> git merge developer
Updating 0bf9f42..a76d0ad
Fast-forward
 check.txt | 1 +
 test.txt  | 2 +-
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 check.txt
```

To check the commits performed in the past

```
PS D:\Dotnetlab\lab1> git log
commit a76d0ad76f34afd88812f13c6e9f80ed9cdf1abc (HEAD -> master, origin/developer, developer)
Author: nekeshkoju07 <nekeshkoju@gmail.com>
Date:   Fri Mar 21 21:42:30 2025 +0545

Author: nekeshkoju07 <nekeshkoju@gmail.com>
Date:   Fri Mar 21 21:42:30 2025 +0545

    new file created in developer branch

commit fd22feb637beec101c4989fd78e48962b9f73487
Author: nekeshkoju07 <nekeshkoju@gmail.com>
Date:   Fri Mar 21 21:40:31 2025 +0545

    chanes in the developer branch

commit 0bf9f42ee3c5f2d8f8dbf5adc46cd442ea89bbfc (origin/master)
Author: nekeshkoju07 <nekeshkoju@gmail.com>
Date:   Fri Mar 21 21:31:01 2025 +0545

    commit

commit bb23b22e845c2b96af9eaffb0434db45c396a17c
Author: nekeshkoju07 <nekeshkoju@gmail.com>
Date:   Fri Mar 21 21:25:45 2025 +0545

    Initial commit
```

Conclusion:

In this lab, we learn about the basics of the Git and GitHub. We perform initialization, branching, merging, pushing and commit.

Link to the repo: https://github.com/nekeshkoju07/test.git