**PURBANCHAL UNIVERSITY**

****

**KHWOPA ENGINEERING COLLEGE**

**LIBALI-08, BHAKTAPUR**

LAB REPORT ON .NET

LAB NO. 01

**SUBMITTED BY: SUBMITTED TO:**

Name: Dipesh Shrestha Department of Computer Engineering

Roll No. : 770310

Group: A

Submission: 2082/01/07

**Theory:**

1. **Git:**

Git is a distributed version control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

Git is a **distributed version control system (DVCS)** designed to help multiple developers collaborate on a project without conflicts or data loss. It enables efficient tracking of changes, version management, and merging of contributions from different developers.

**Key Purposes of Git as a DVCS:**

1. Distributed System:

Unlike centralized version control systems (e.g., SVN), Git does not rely on a single central repository. Each developer has a complete copy of the repository, including the full history, allowing them to work offline and independently.

1. Collaboration Without Overwrites:

Developers work on separate branches, preventing direct conflicts. Changes are merged strategically using Git’s merging and rebasing features.

1. Version Control & History Tracking:

Every change is recorded with a commit message, making it easy to track who changed what and when. Previous versions of the project can be restored if needed.

1. Branching & Merging:

Developers can create isolated branches for new features or bug fixes. Once tested, changes can be merged into the main branch without disrupting others’ work.

1. Conflict Resolution:

If multiple developers edit the same file, Git detects conflicts and allows manual resolution before finalizing changes.

1. Collaboration via Remote Repositories:

Platforms like GitHub, GitLab, and Bitbucket enable developers to push and pull changes to a shared repository. Teams can work asynchronously and from different locations.

Git's primary features include:

* Version Control: Git keeps a history of changes made to the codebase, allowing developers to revert to previous versions if needed.
* Branching and Merging: Developers can create branches to work on new features or fixes independently and merge them back into the main branch once they are ready.
* Distributed Workflow: Git allows each developer to have a complete copy of the repository, enabling offline work and reducing dependency on a central server.

1. **GitHub**

GitHub is a web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features. 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.

**Key Features of GitHub for Collaboration**

1. Pull Requests (PRs) – Code Review & Merging:

Allows developers to propose changes, review them, and merge them into the main project.

1. Issues – Bug Tracking & Feature Requests:

Acts as a built-in ticketing system for reporting bugs, requesting features, and discussing improvements.

1. Project Management Tools (GitHub Projects & Discussions):

Helps teams organize tasks and workflows directly within GitHub.

1. Actions & CI/CD – Automating Workflows

Automates testing, deployment, and other development tasks.

1. Forking & Open Source Contributions

Encourages open-source collaboration by allowing anyone to copy a repository, modify it, and contribute back.

It builds on Git's capabilities and offers:

* Repository Hosting: GitHub stores your code in repositories, making it accessible from anywhere.
* Collaboration Tools: GitHub allows multiple developers to work on the same project, review each other's code, and suggest improvements through pull requests.
* Project Management: GitHub provides tools for tracking issues, managing tasks, and organizing projects.

**General Git and GitHub Commands:**

|  |  |
| --- | --- |
| Commands | Functions |
| git init | Initializes a new Git repository in the current directory. |
| git add . | Stages all changed and new files for the next commit. |
| git commit -m “message” | Commits staged changes with a descriptive message. |
| git status | Shows the current status of the working directory and staging area. |
| git log | Displays a history of commits in the repository. |
| git add file | Stages a specific file for the next commit. |
| git pull origin branch\_name | Fetches and merges updates from a remote repository. |
| git push origin branch\_name | Pushes a specific branch to the remote repository. |
| git remote -v | Lists all remote repositories linked to the project. |
| git merge branch\_name | Merges the specified branch into the current branch. |
| git branch | Lists all local branches in the repository. |
| git branch branch\_name | Creates a new branch. |
| git checkout branch\_name | Switches to a different branch. |
| git checkout -b branch\_name | Creates and switches to a new branch. |
| git clone repo\_url | Copies (clones) a remote repository to your local machine. |
| git remote add origin repo\_link | Add the local repository with the github repository specified by the url link. |

**Lab Works**

1. First set the global username and email of the GitHub.

A screenshot of a computer screen

AI-generated content may be incorrect.

1. Create a folder and files inside it 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

A screen shot of a computer

AI-generated content may be incorrect.

As we can see a new file is created but it is not saved in the local repository so we use add command to do so.

A screenshot of a computer program

AI-generated content may be incorrect.

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

A screen shot of a computer

AI-generated content may be incorrect.

1. To connect the local and github repository we use follwing command



1. Now to create and change the current branch of the repository

A screen shot of a computer

AI-generated content may be incorrect.

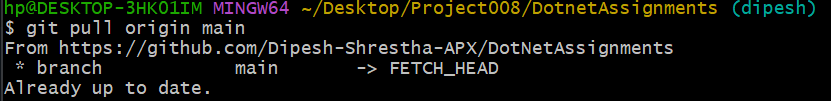
1. Now to save the commited files in the github repository through specific branch

A screenshot of a computer

AI-generated content may be incorrect.

Here develop is the branch through which files are pushed

1. Similary to pull the files form github to local repository.



1. To clone the public github repository in the local repository.

A screen shot of a computer

AI-generated content may be incorrect.

**Conclusion:**

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