Experiment 3

Mahi Palimkar

231071036

SY BTech Computer engineering

**Aim**: Version Control Systems: Introduction to git, basic git commands (clone, commit, push, pull, branch, merge) and working with GitHub or GitLab**.**

# Theory:

## What is git, and why is it called version control?

Git is an open source distributed version control system that helps software teams create projects of all sizes with efficiency, speed, and asynchronicity.

It is a [distributed version control system](https://about.gitlab.com/topics/version-control) that enables software development teams to have multiple local copies of the project's codebase independent of each other. These copies, or branches, can be created, merged, and deleted quickly, empowering teams to experiment, with little compute cost, before merging into the main branch (sometimes referred to as the [master branch](https://about.gitlab.com/blog/2021/03/10/new-git-default-branch-name/)). Git is known for its speed, [workflow compatibility](https://about.gitlab.com/blog/2020/04/07/15-git-tips-improve-workflow/), and open-source foundation.

Git is called a version control system because it tracks and manages changes to files over time, allowing you to maintain a history of your project. It lets you revert to previous versions if needed and enables multiple people to collaborate on the same project without overwriting each other's work. Git helps merge changes from different contributors and ensures that everyone is working with the latest version of the files.

## Explain the purpose of the git clone command.

The git clone command is used to create a copy of a specific repository or branch within a repository.

When you clone a repository, you don't get one file, as you may in other centralized version control systems. By cloning with Git, you get the entire repository – all files, all branches, and all commits.

Cloning a repository is typically only done once, at the beginning of your interaction with a project. Once a repository already exists on a remote, like on GitHub, then you would clone that repository so you could interact with it locally. Once you have cloned a repository, you won't need to clone it again to do regular development.

The ability to work with the entire repository means that all developers can work more freely. Without being limited by which files you can work on, you can work on a feature branch to make changes safely.

## What is the difference between git add and git commit?

The difference between git add and git commit lies in their roles within the Git workflow:

* **git add**: This command stages changes by adding them to the staging area (also called the index). It prepares the changes you want to include in your next commit. You can think of it as selecting the specific changes you want to save in your project history.
* **git commit**: This command takes the changes that have been staged (using git add) and permanently records them in the repository's history. Each commit is a snapshot of your project at that moment, along with a message describing the changes.

In summary, git add prepares changes, while git commit saves those changes to the repository.

## How does the git push command work?

The git push command is used to transfer commits from your local repository to a remote repository. This is essential for sharing your work with others or deploying your changes to a shared codebase.

* **Transfer Commits**: When you push, you send the changes you’ve committed in your local repository to a corresponding branch in the remote repository.
* **Remote Repository**: The remote is usually referred to as origin, and it points to the remote repository's URL where the code is hosted.
* **Branch Updates**: By pushing to a branch, you update that branch in the remote repository to reflect the commits you’ve made locally.

## Basic Example:

git push origin main (the command for this)

* **origin**: The default name for the remote repository.
* **main**: The branch you are pushing to (commonly main or master).

## What Happens During git push:

1. **Synchronization**: Your local branch and the remote branch are synchronized. The remote branch will now include all the commits from your local branch.
2. **Collaboration**: Other collaborators can pull the latest changes from the remote repository to their local repositories using git pull.

## Use Cases:

* **Sharing Work**: After you’ve made some commits locally, you use git push to share those changes with others by pushing them to the remote repository.
* **Deploying Code**: In some workflows, git push is used to deploy code directly to a server (e.g., through continuous integration/deployment systems).

## How to make a new branch using the command line?

To create a new branch in Git using the command line, follow these steps:

**Check Current Branch**: Use the git branch command to list all branches and identify the current branch you are working on.

**Command**: git branch

**Create a New Branch**: Use the git branch command followed by the desired name for the new branch to create it. This command creates the branch but does not switch to it.

**Command**: git branch <new-branch-name>

**Switch to the New Branch**: Use the git checkout command to switch to the newly created branch.

**Command**: git checkout <new-branch-name>

**Create and Switch in One Step**: Alternatively, you can combine the creation and switching to the new branch into a single command using git checkout -b.

**Command**: git checkout -b <new-branch-name>

**Push the New Branch to Remote**: To make the new branch available to others, push it to the remote repository using the git push command with the -u option to set up tracking.

**Command**: git push -u origin feature-xyz

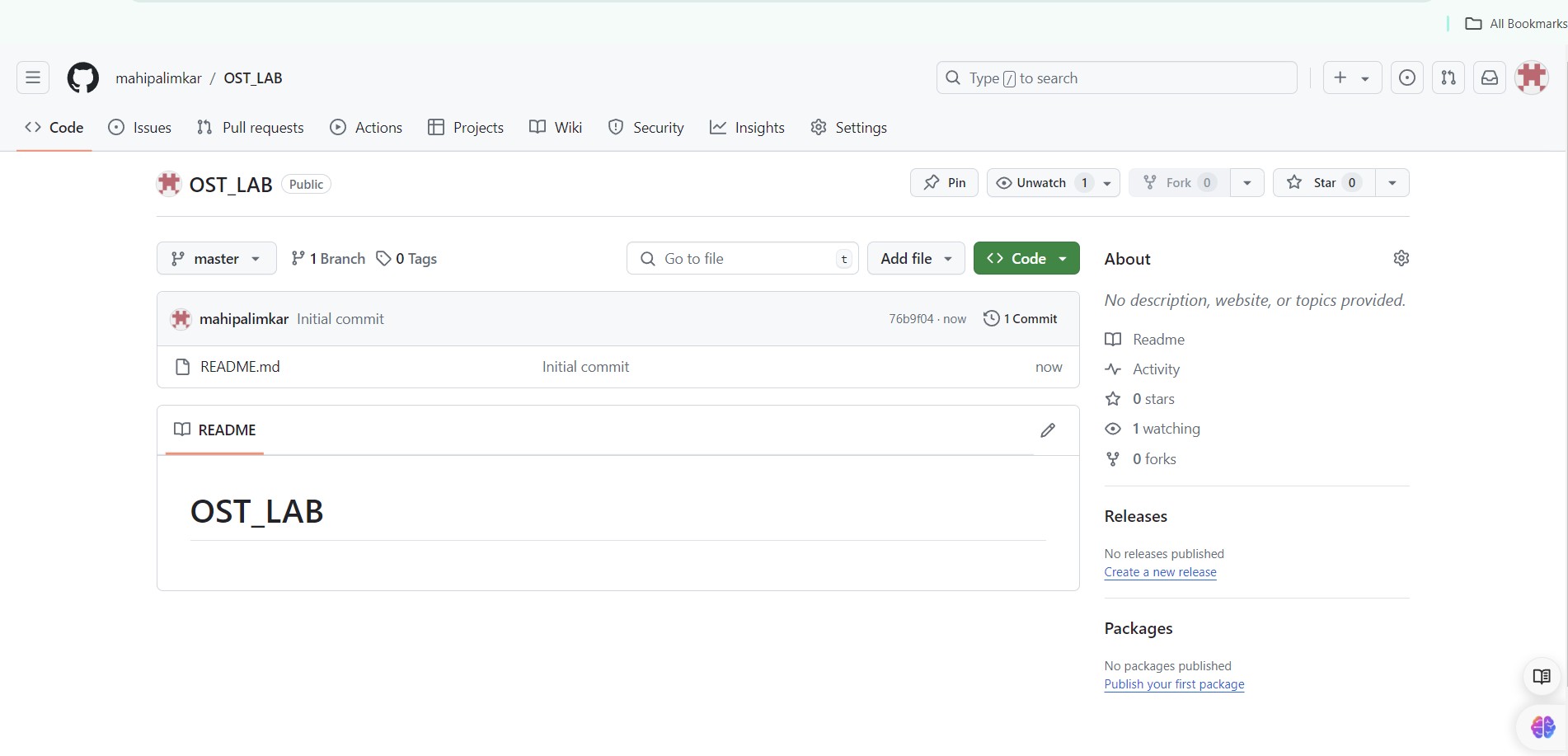
## Explain how the git merge command works. Overview of git merge

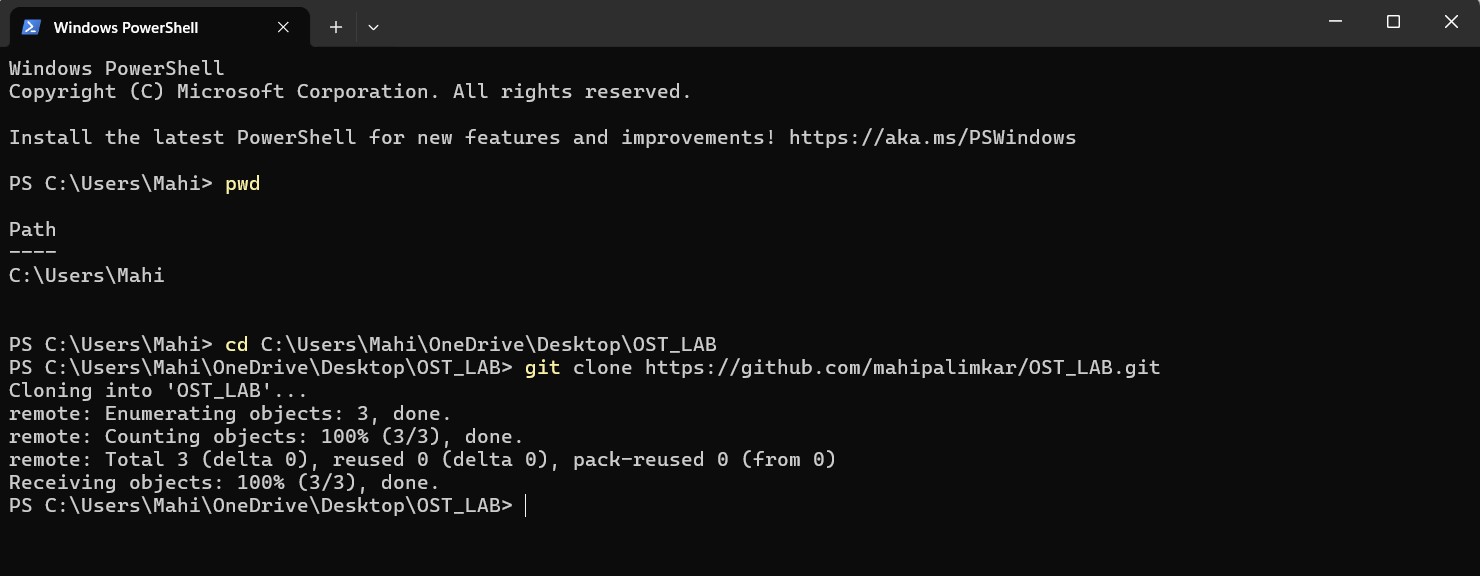
1. **Purpose**: The git merge command combines the changes from a source branch into the target branch, which is typically the branch you are currently on. This allows you to incorporate updates from different branches into a unified codebase.

## How It Works:

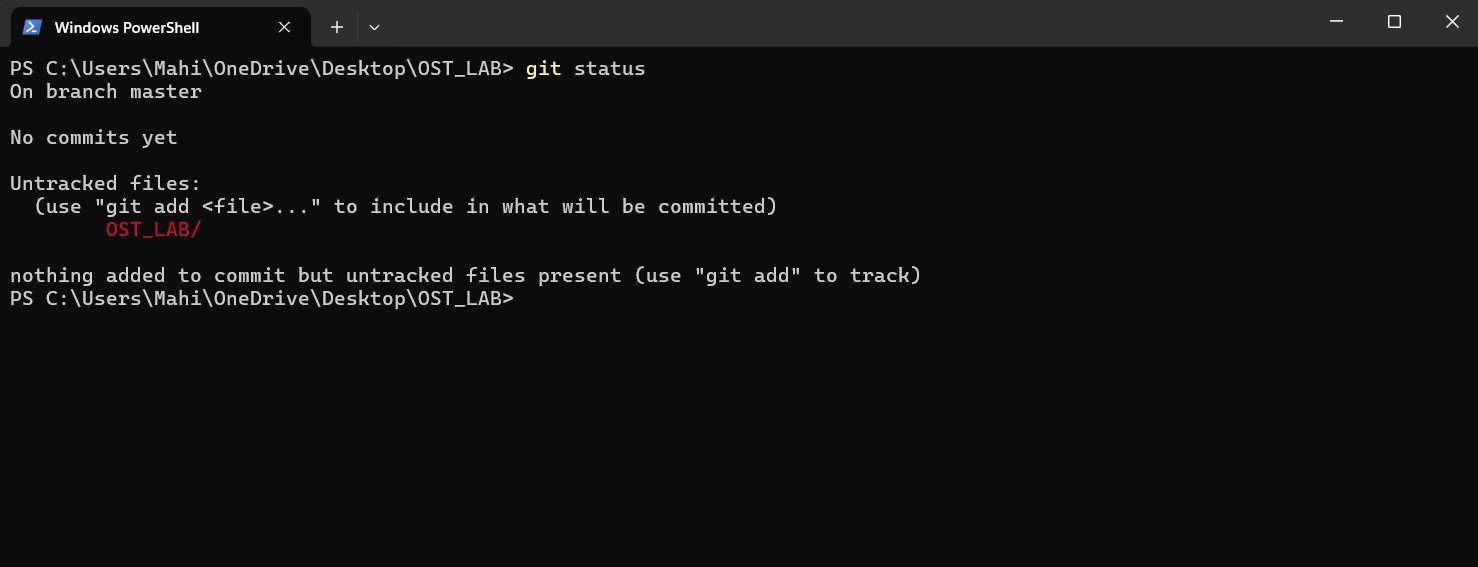
* + **Merge Commit**: When you merge, Git tries to automatically combine the changes. If there are no conflicts, Git creates a new commit called a "merge commit" that records the integration of the branches.
  + **Fast-Forward Merge**: If the target branch has not diverged from the source branch, Git performs a fast-forward merge. This means Git simply moves the target branch pointer forward to the latest commit of the source branch without creating a merge commit.
  + **Three-Way Merge**: If the branches have diverged, Git performs a three-way merge. It uses the latest common ancestor of both branches, along with the tips of both branches, to create a new commit that reconciles the changes.

# Implementation Screenshots:

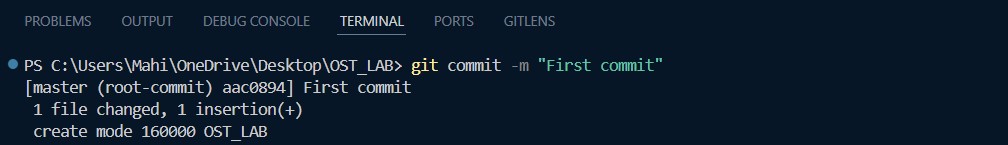
1. We first create a repository on github.
2. Then we clone the repository from the remote system to the local system.



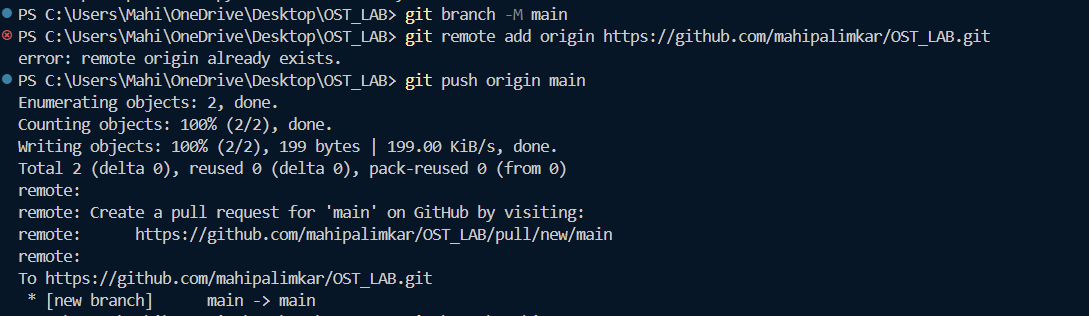
1. Now that the repo is on our local system, we can add a file in it.

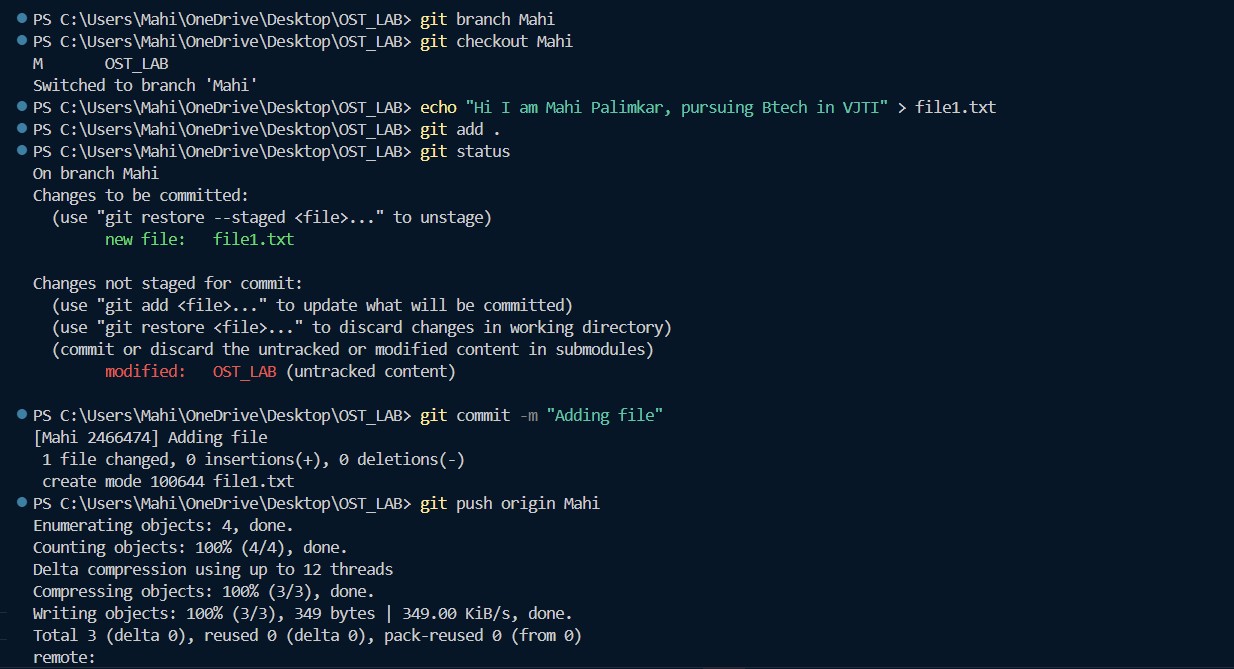
I have decided to enter a simple calculator program in the file in the main branch. This is before I git add .

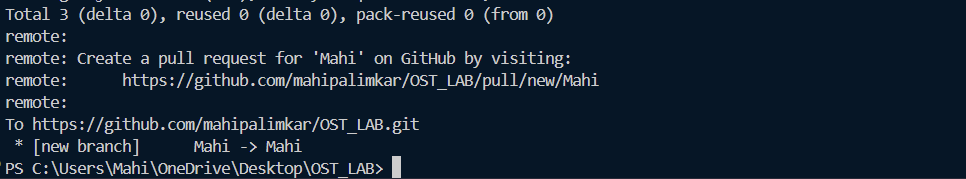
Now I have added the file and also committed it



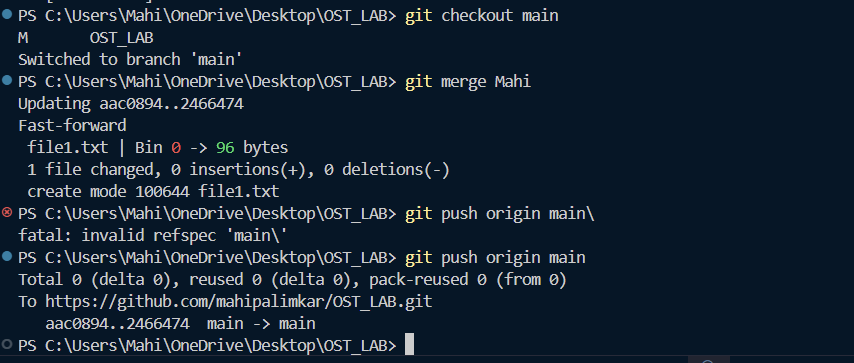
Now navigate to the main branch from the master branch and push the file there. I also created a new branch called Mahi and added, committed and pushed a simple txt file in it.

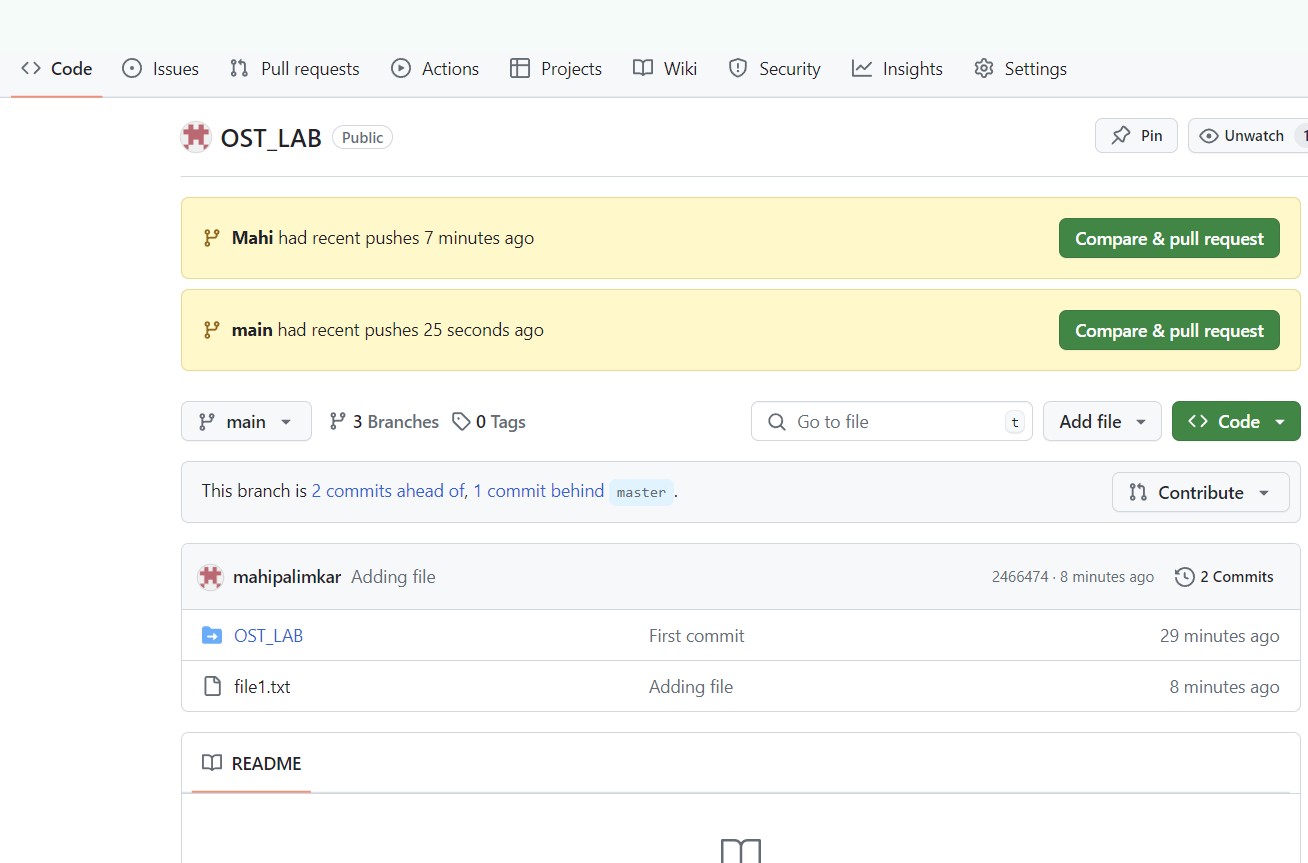




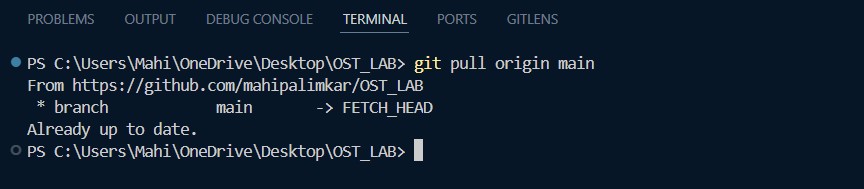


Now to merge both the branches.





Now we pull from the repo



Conclusion:

In this assignment, I learnt how to implement basic Git commands. I initialized a repo on github, and cloned it. I created a calculator file in the main branch, added it for staging, then committed and pushed it on github. Then I created another branch called Mahi, added a simple txt file to it and then merged the two of them. I created a pull request too. Overall, I learnt how version control works using Git and github in this exercise. It helped me gain insights about how to effieciently work on github and other open source softwares.