Experiment 02 - Version Control

| Roll No. | 37 |
| --- | --- |
| Name | Mikil Lalwani |
| Class | D15-A |
| Subject | DevOps Lab |
| LO Mapped | LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements  LO2: To obtain complete knowledge of the “version control system” to effectively track changes augmented with Git and GitHub |
|  |  |

**Aim**: To understand Version Control System / Source Code Management, install git and create a GitHub account.

**Introduction**:

The purpose of version control is to allow software teams to track changes to the code while enhancing communication and collaboration between team members. Version control facilitates a continuous, simple way to develop software.

Source code acts as a single source of truth and a collection of a product’s knowledge, history, and solutions. Version control (or code revision control) serves as a safety net to protect the source code from irreparable harm, giving the development team the freedom to experiment without fear of causing damage or creating code conflicts.

Suppose developers code concurrently and make incompatible changes. In that case, version control identifies the problem areas so that team members can quickly revert changes to a previous version, compare changes, or identify who committed the problem code through the revision history. With version control systems, a software team can solve an issue before progressing further into a project. Software teams can analyze earlier versions through code reviews to understand how a solution evolved.

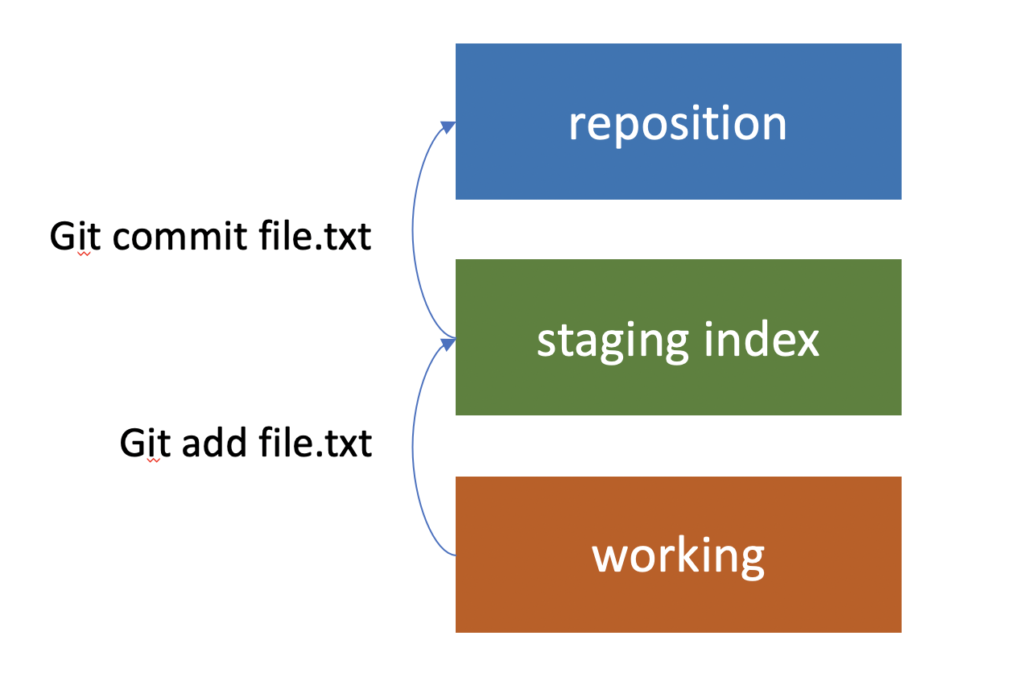
Depending on a team's specific needs, a version control system can be local, centralized, or distributed. A local version control system stores files within a local system. Centralized version control stores changes in a single server. A distributed version control system involves cloning a Git repository.

**Git**:

Git uses a three-tree architecture.

It has the repository and the working copy, but in between is another tree, which is the staging index. When we made our first commit, we didn’t just perform a commit. First, we used the add command. We added, then we committed. It was a two-step process (added our files to the staging index, and then from there we committed to the repository).

Git Concepts and Architecture-



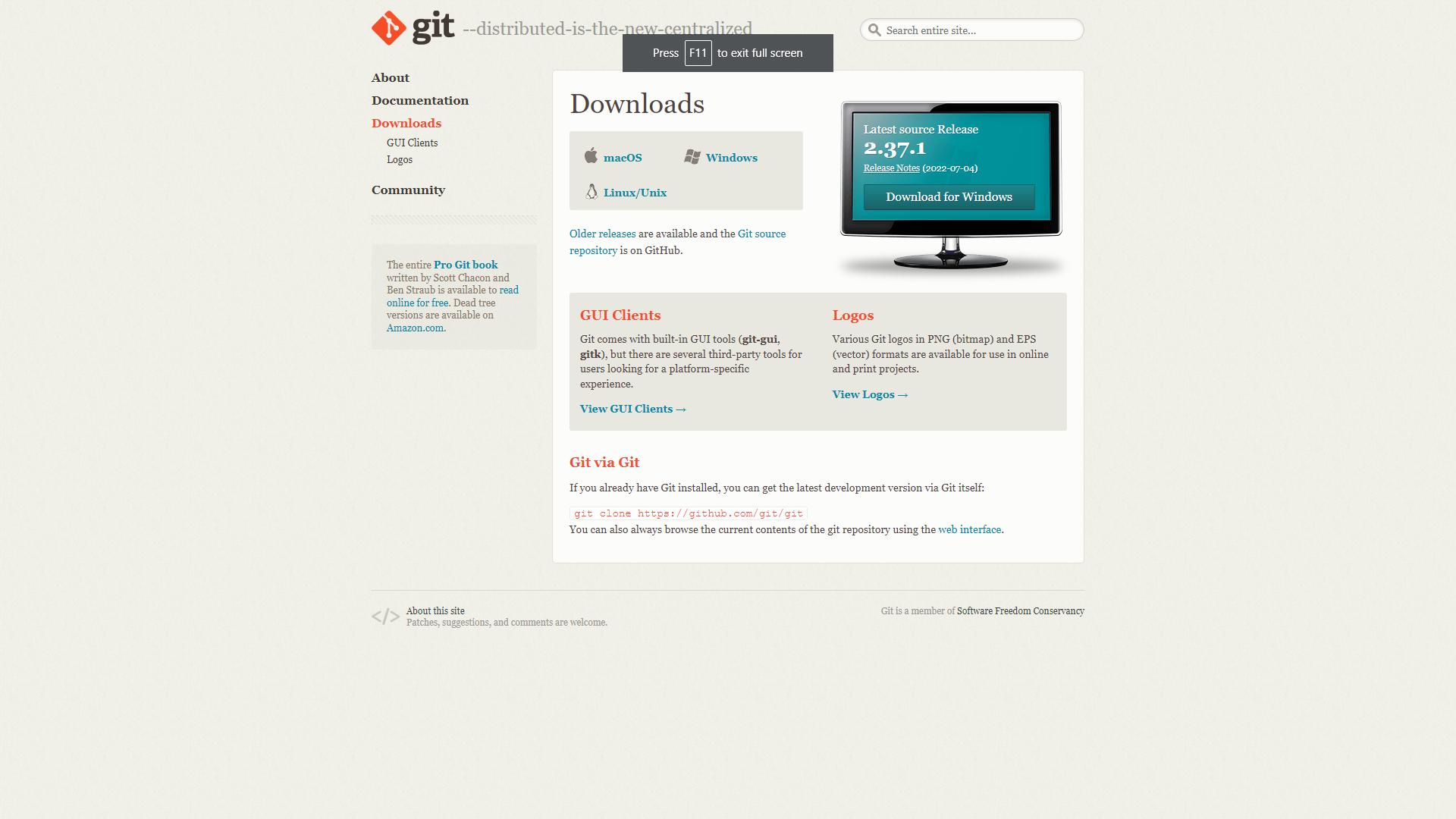
It is possible to just commit directly to the repository and skip adding them to the staging index. But it’s important to understand that this is part of the architecture of Git. It’s a really nice feature because we can make changes to ten different files in our working copy, and then selectively commit five of the changes as one set. That’s why it’s called staging. We have the chance to add the changes that we want to the staging, and then get them ready before we commit them. And we can check out changes from the repository the same way.

It’s also possible to pull them from the repository into the staging index, and then from the staging index to the working directory, but most of the time, that’s not what we do. Usually, we go ahead and pull them straight from the repository, down into the working directory. In the process, the staging index will also be updated too.

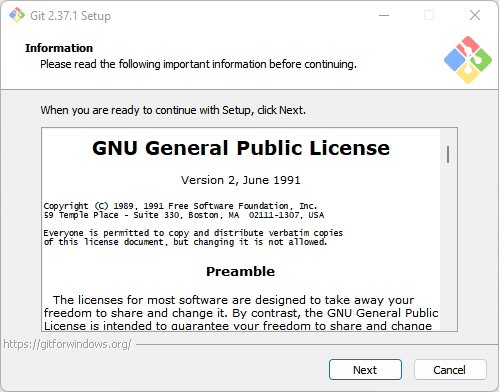
As we’re working with Git, it’s useful to keep these three different trees in mind. There’s our working directory, which contains changes that may not be tracked by Git yet, there’s the staging index, which contains changes that we’re about to commit into the repository, and then there’s the repository, and that’s what’s actually being tracked by Git.

**Installation**:

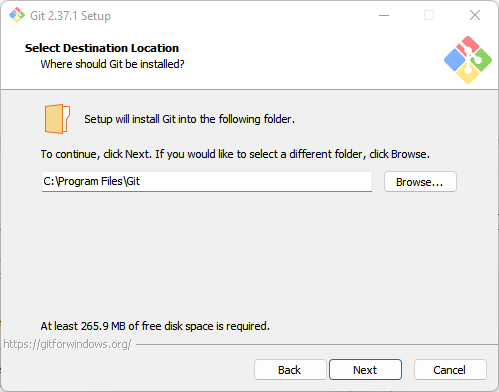
1. Search git download on browser and open 1st link and download the version of git relevant for your OS.

**

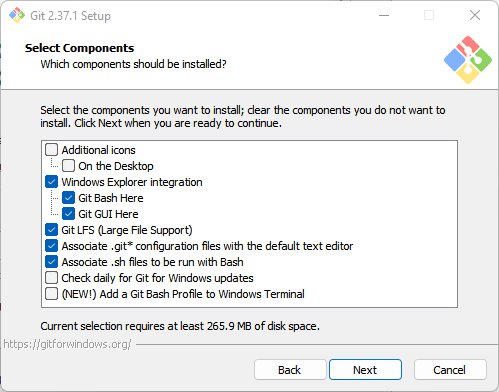
1. Run the setup and click on next.

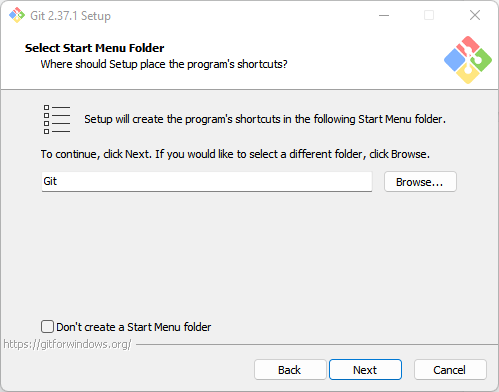
**

1. Specify the location where you want to install Git and click next.

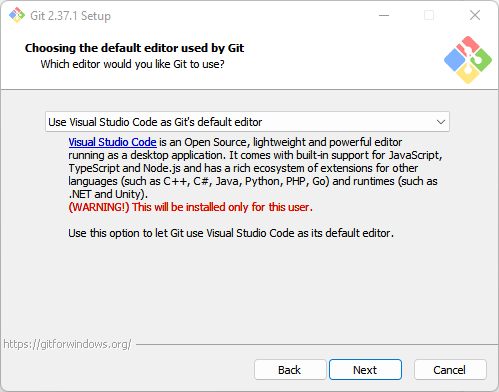
**

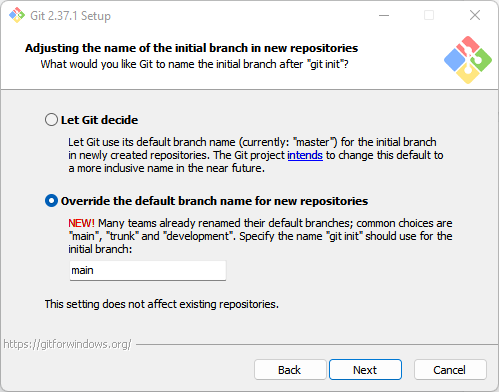
3. Click on the next till you install button.

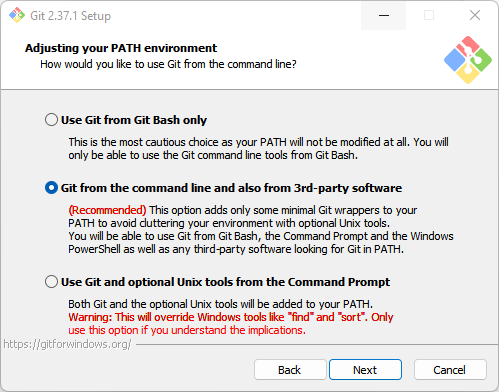
**

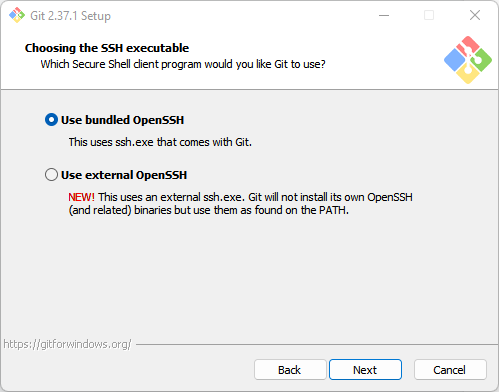
**

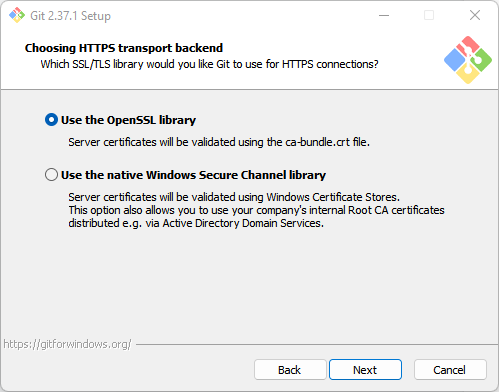
1. We can make VSCode our default editor.

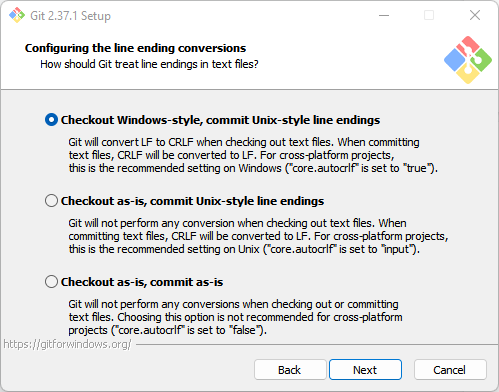
**

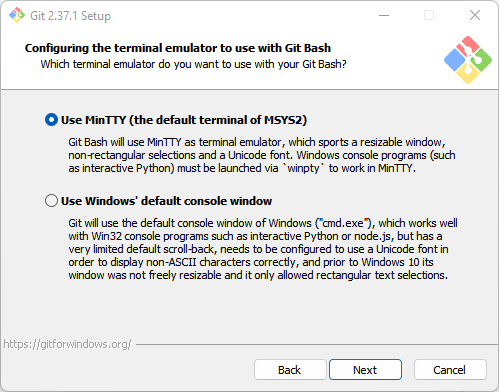
**

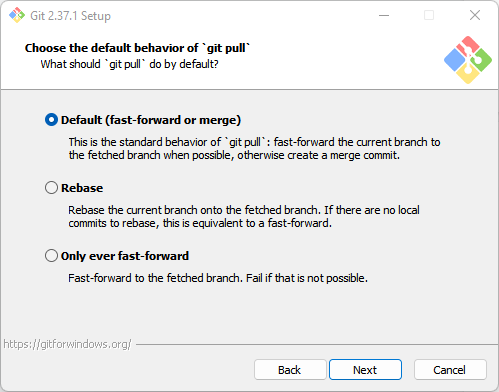
**

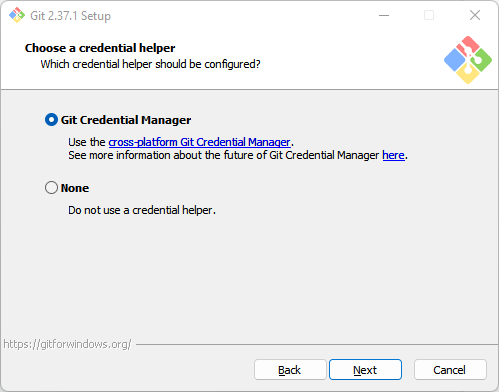
**

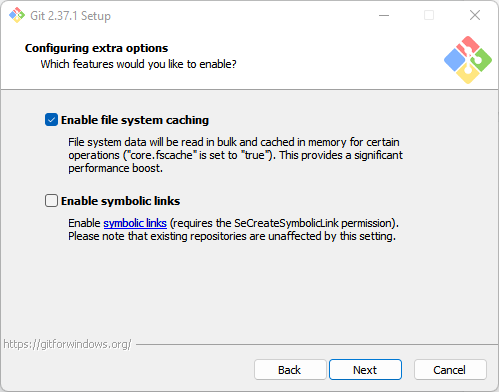
**

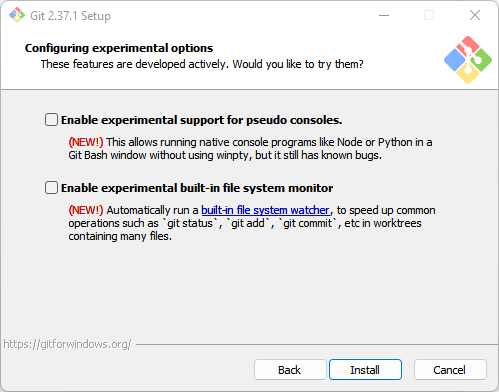
**

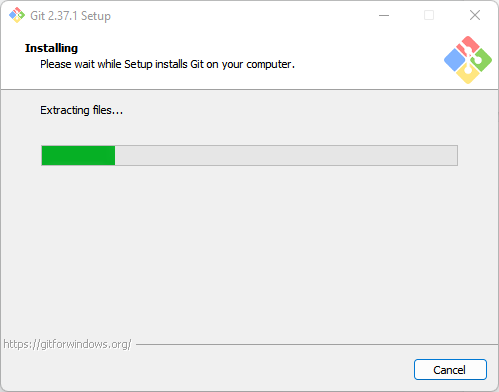
**

**

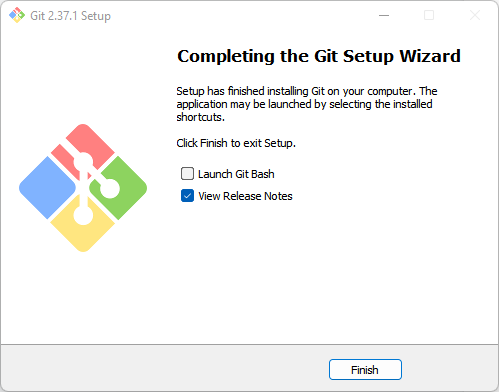
**

**

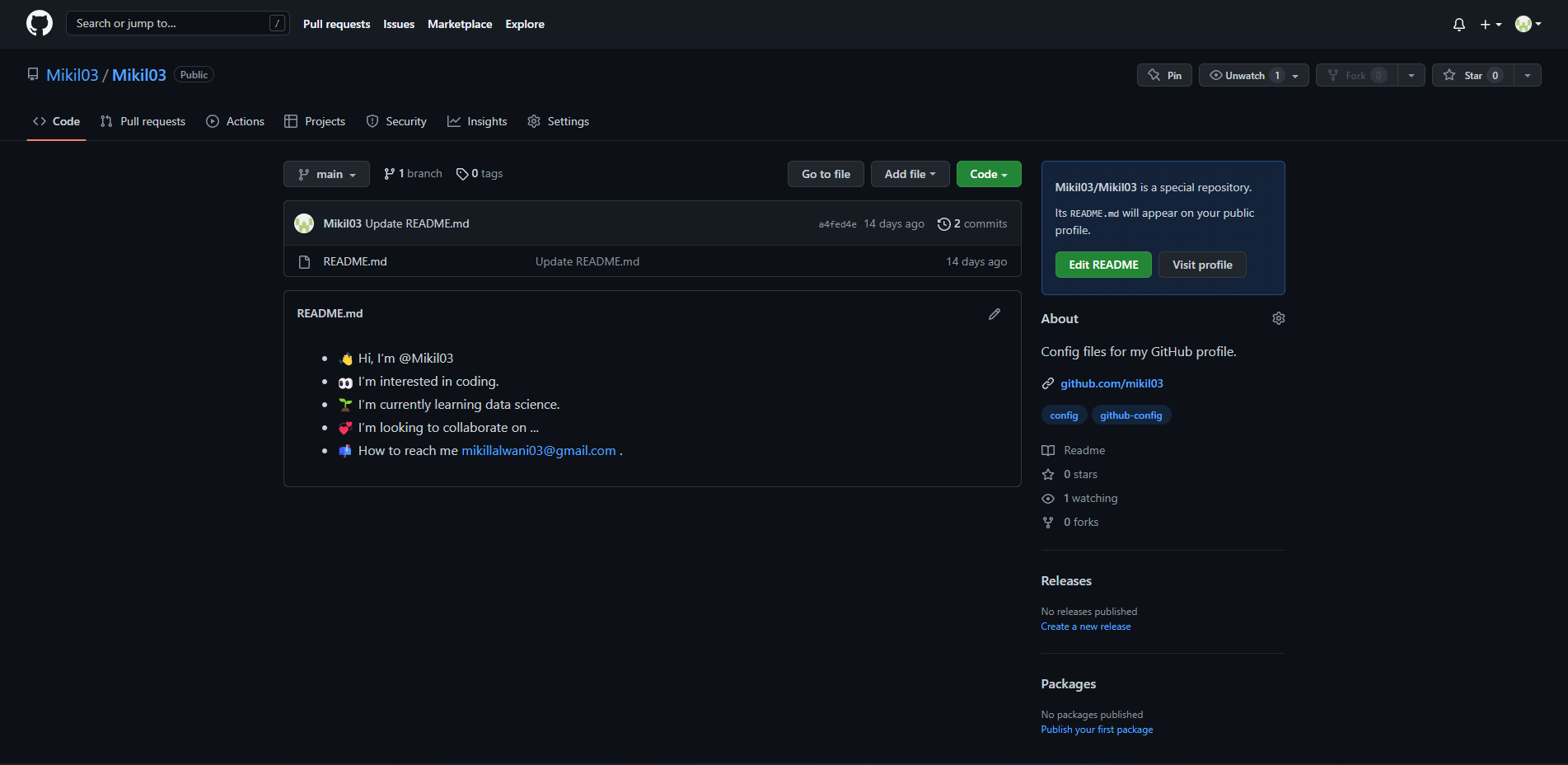
**

**

5. Click on finish and now you can start using git.

**

**Github**:

**

**Conclusion**

We have successfully installed git and created our GitHub account.