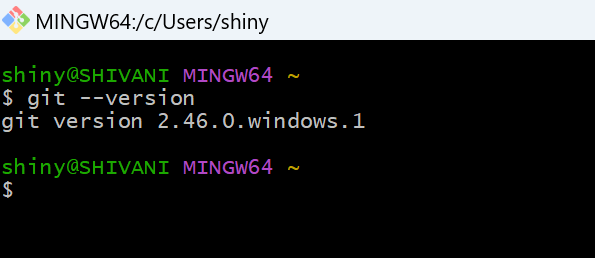
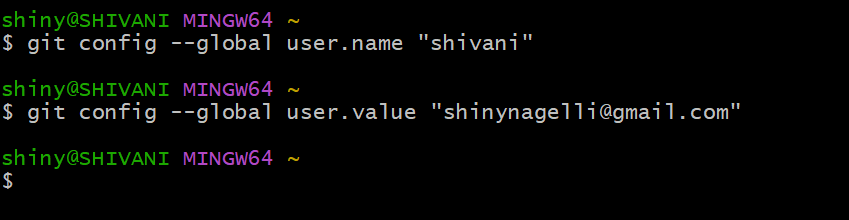
GIT COMMANDS:

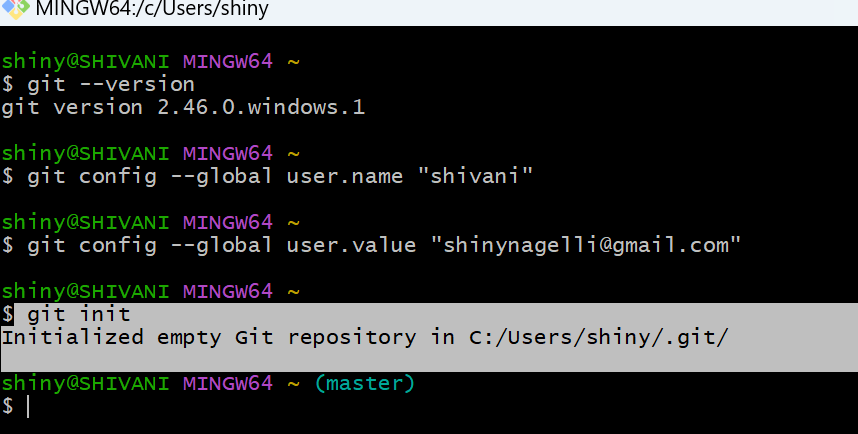
* **Git Version**: Check if Git is installed using git --version.



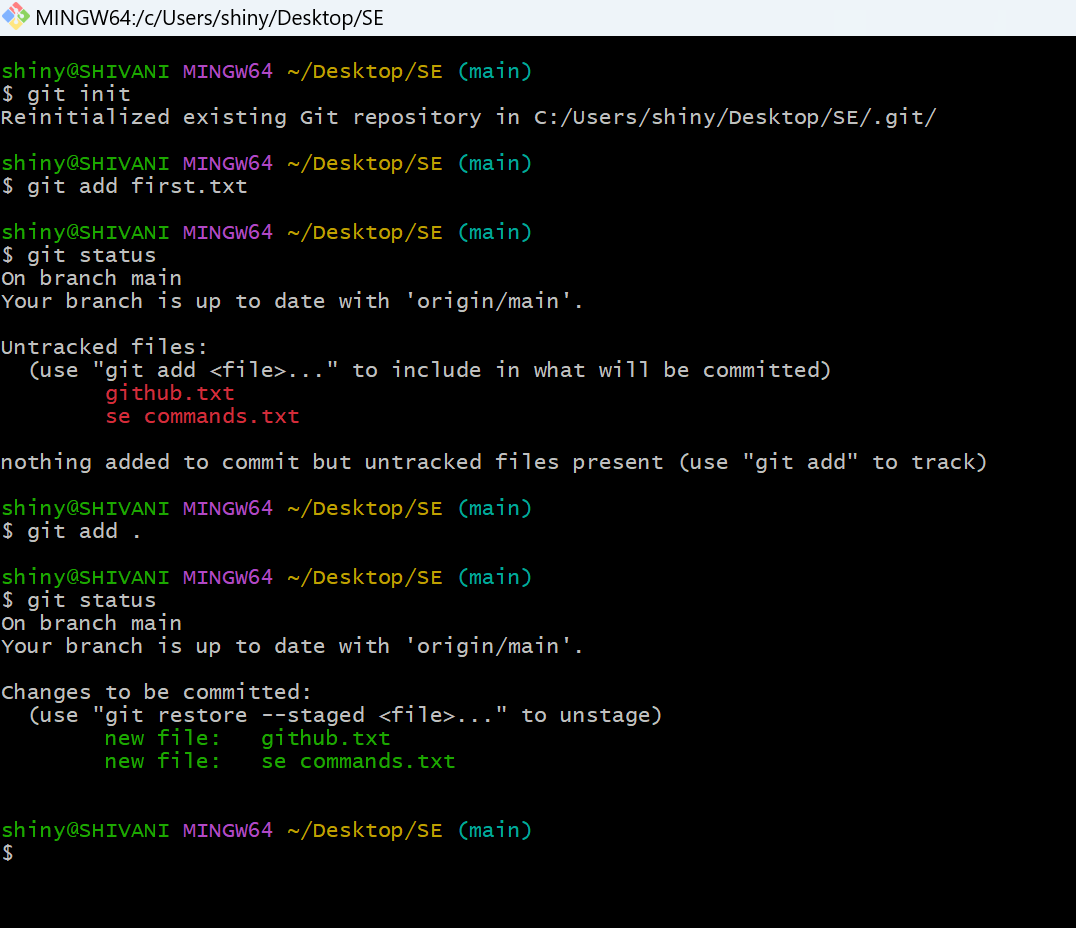
* **Git Config:** Set up your name and email using **git config.**
* Essential for setting up your user name and email, ensuring that every commit you make is associated with the correct identity in the project history.
* Example:
  + **git config --global user.name "Your Name“**
  + **git config --global user.email** [**your.email@example.com**](mailto:your.email@example.com)



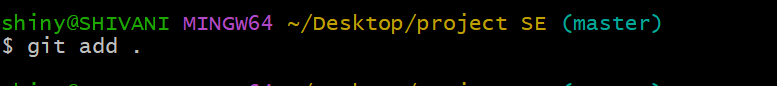
* **Git Init:** Start a new project by creating a Git repository.
* Example: git init creates an empty Git project.



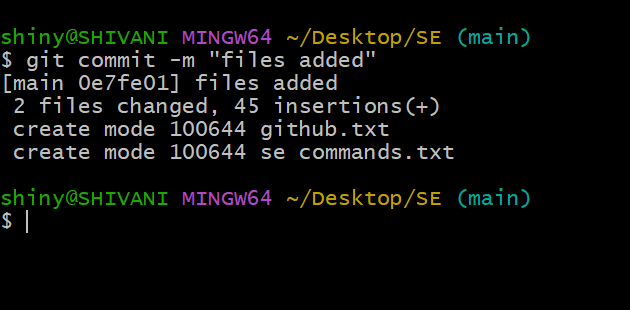
* **Git Status:** Check the current state of your files and see if any changes need to be saved.
* **Command:** git status
* **Example:** Running git status tells you if there are files to be staged or committed.



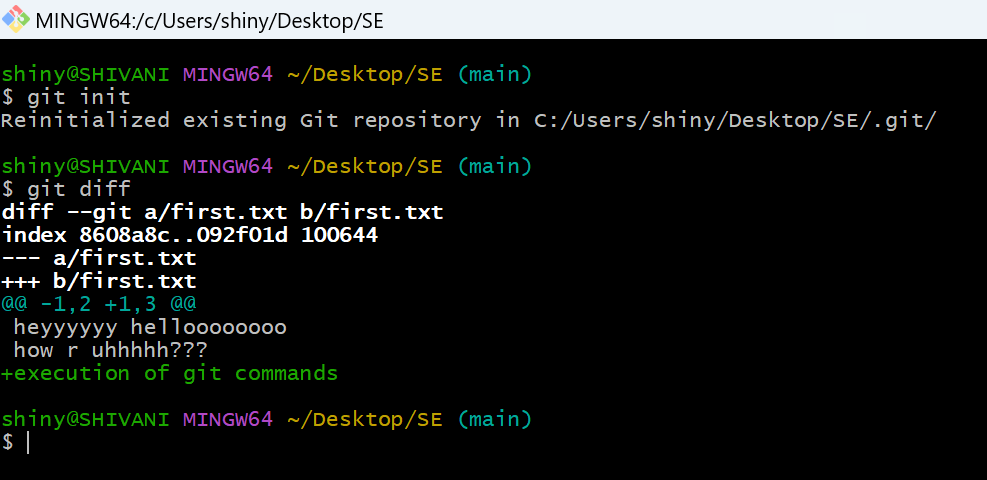
* **Git Add:** Stage changes by adding files to the staging area before saving.
* Example: **git add .** stages all changes.



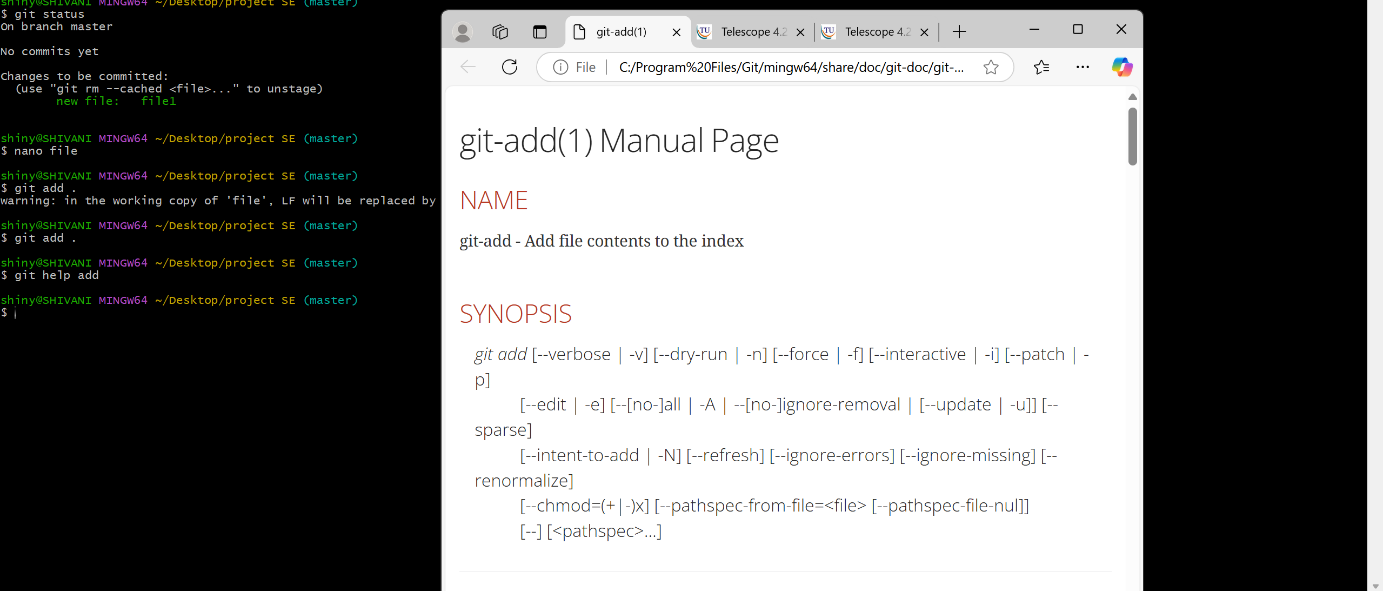
* **Git Commit:** Save your changes with a message explaining what was changed.
* Example: **git commit -m "Added new feature"**



* **Git Diff:** Compare changes between your current files and the last commit.
* Command: **git diff**
* Example: Run git diff to see the differences between your current work and the last saved version.

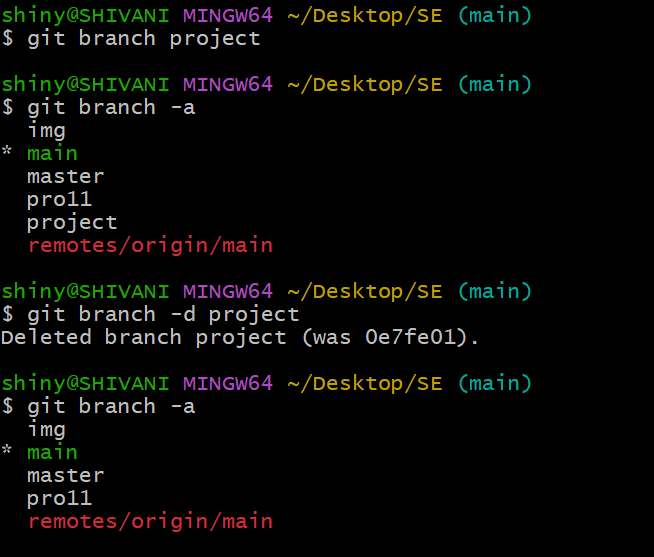
****

* **Git Help:** Get help on any Git command.
* **Command:** git help <command>
* **Example:** Use git help add to learn more about the git add command.



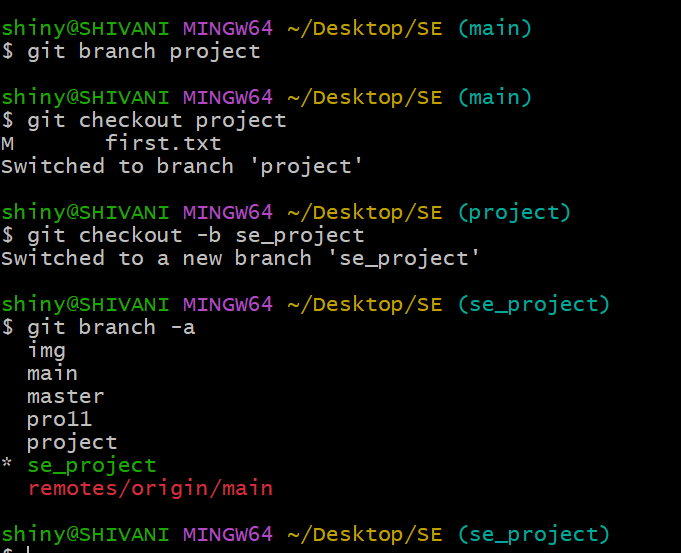
**Branch Commands:**

* **git branch <branch\_name>:** Creates a new branch.
* **git branch -a:** Lists all branches.
* **git branch -d <branch\_name>:** Deletes a branch.



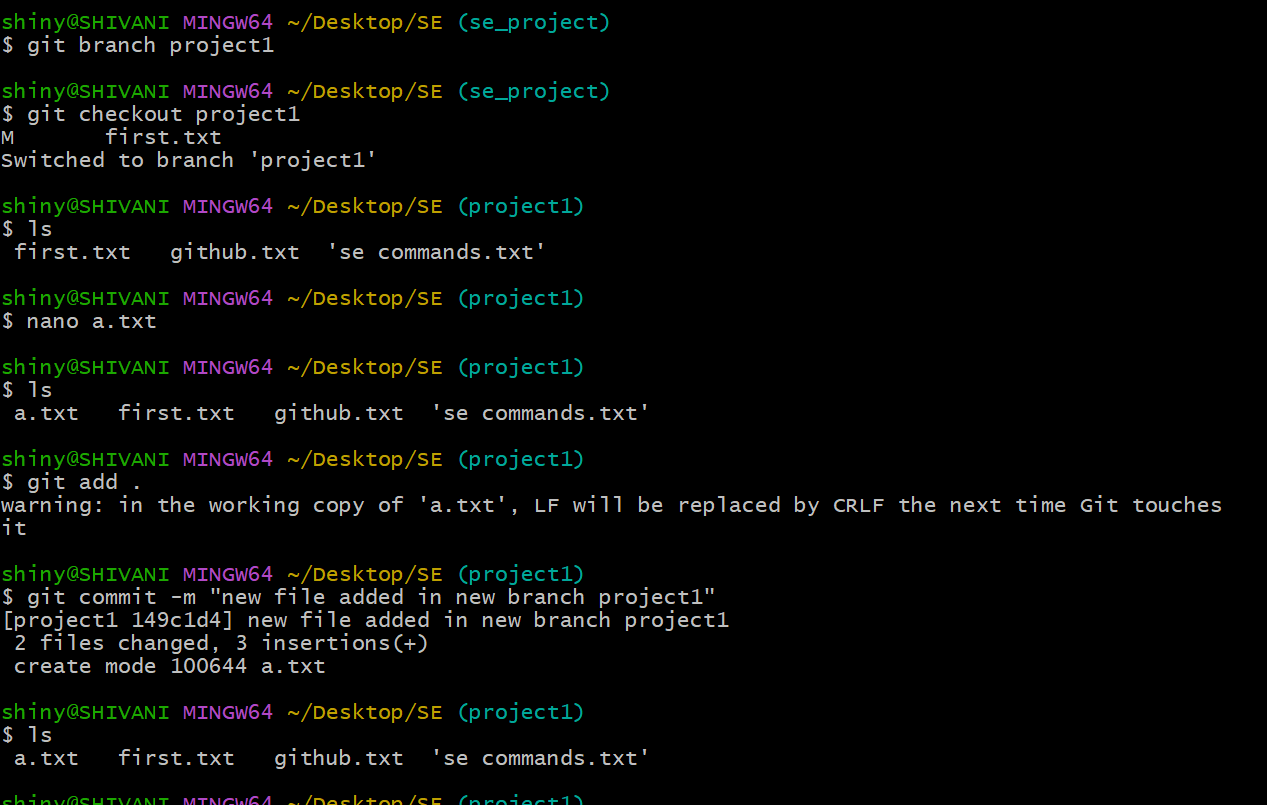
**Checkout:**

* The git checkout command allows you to switch between branches, making it easy to work on different tasks.
* **git checkout <branch\_name>:** Switch to an existing branch.
* **git checkout -b <new\_branch>:** Create and switch to a new branch in one step.



**Merge:**

* The git merge command is used to combine the changes from one branch into another.”
* This is usually done when a feature or bug fix is complete.



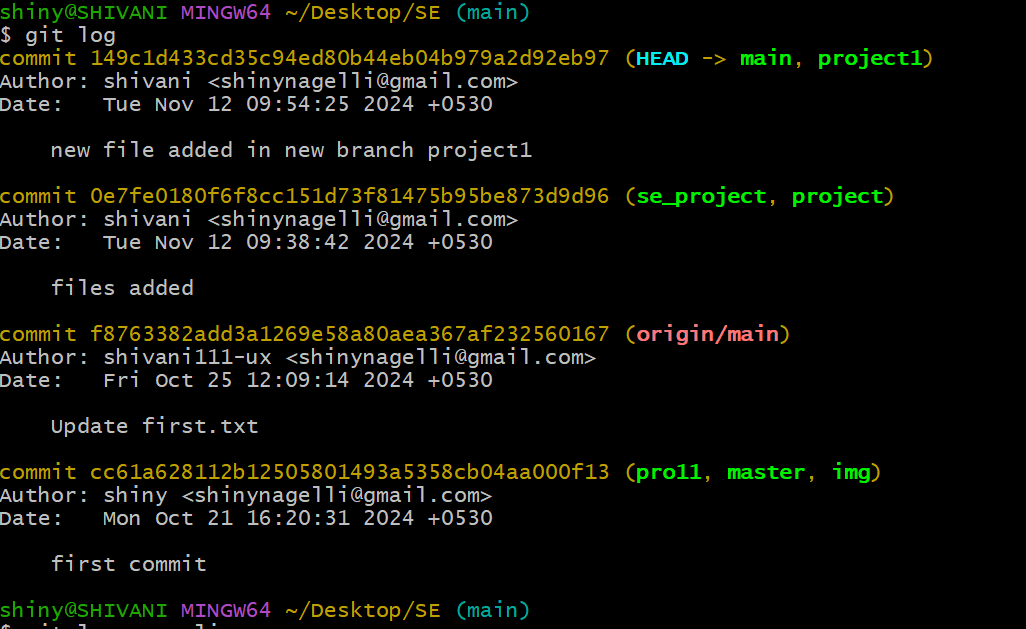


**Viewing History: Log**

“The git log command shows a detailed history of all the changes (commits) made in a branch.”

When to Use Log?

Use the log to see what changes were made, by whom, and when. This is helpful when tracking progress or debugging issues.

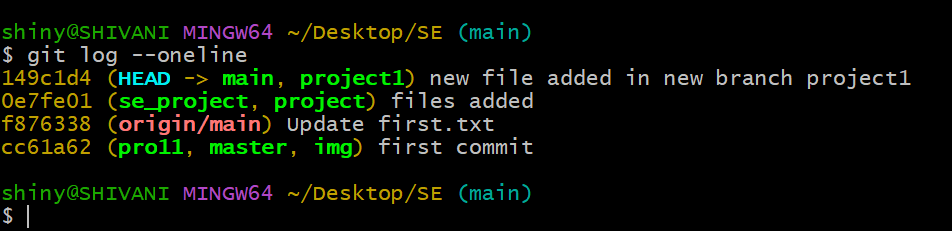
****

* Commit Hash: A unique ID for each change.
* Author: Who made the change.
* Date: When the change was made.
* Message: A brief description of the change.

**git log –oneline:** A simplified version of git log, showing the commit history in one line per commit.

Includes:

* Shortened commit hash
* Commit message

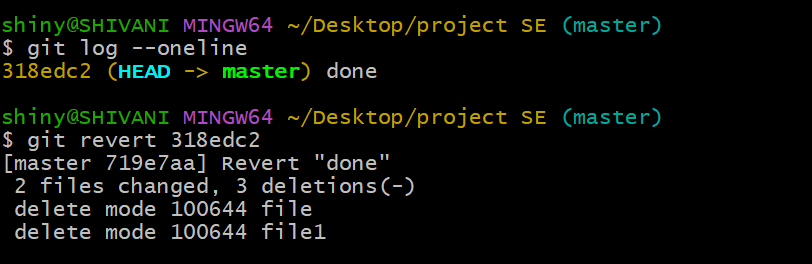


**Undoing Changes: Revert:**

* “The git revert command creates a new commit that undoes the changes made in a previous commit.”
* It doesn’t delete the commit but makes a new one that cancels it out.

**When to Use Revert?**

Use revert when a mistake has been made in a previous commit, but you want to keep a record of the change.

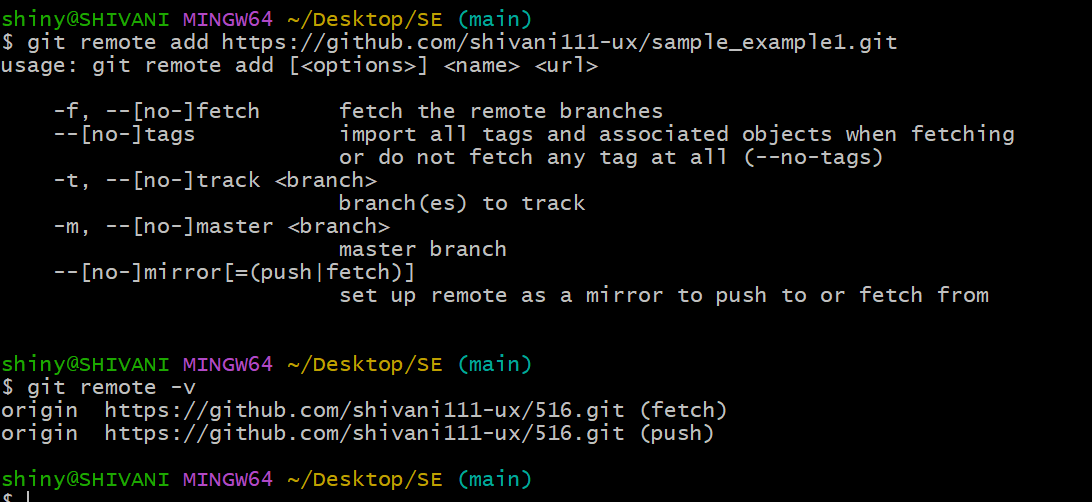


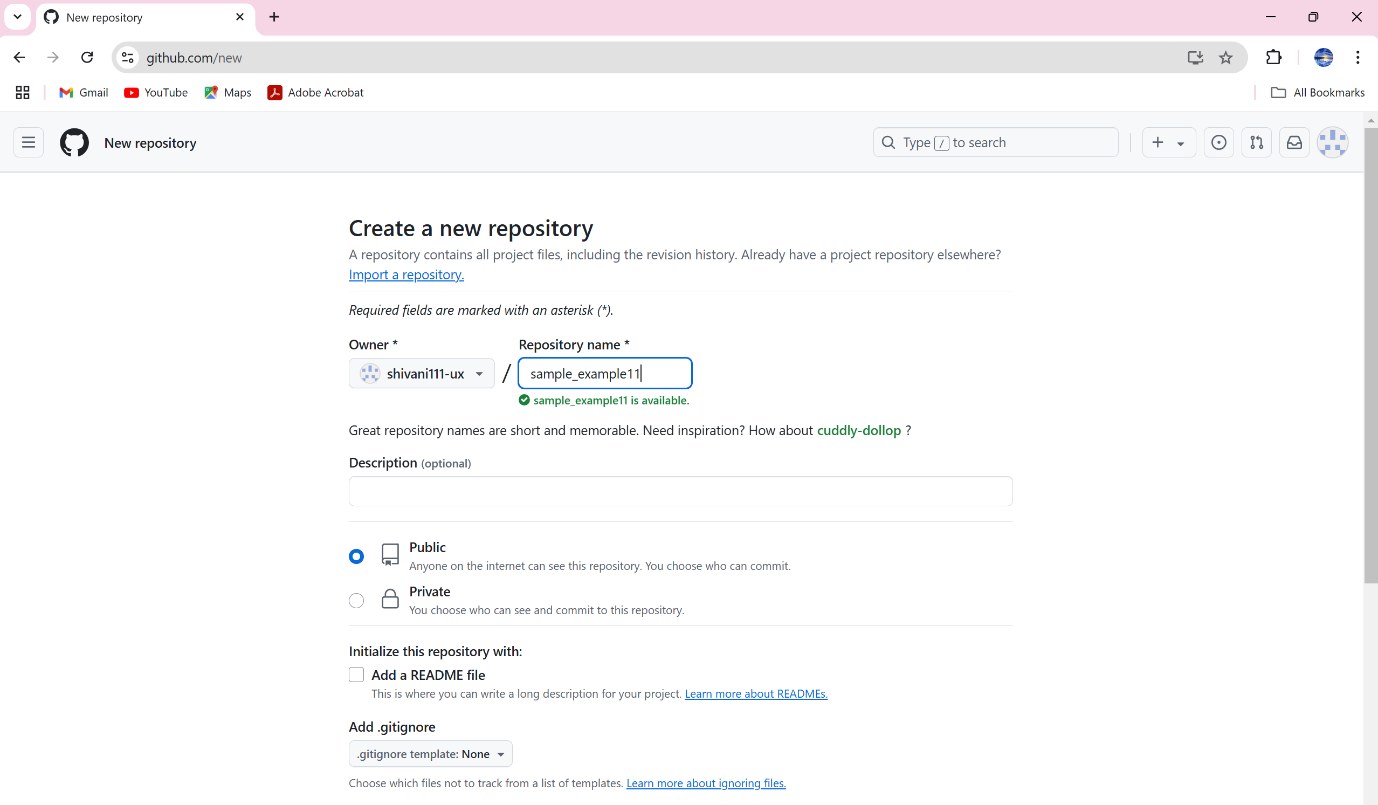
**git revert <commit\_hash>:** Reverts the changes from a specific commit without deleting the history. Where commit\_hash is the ID of the commit.

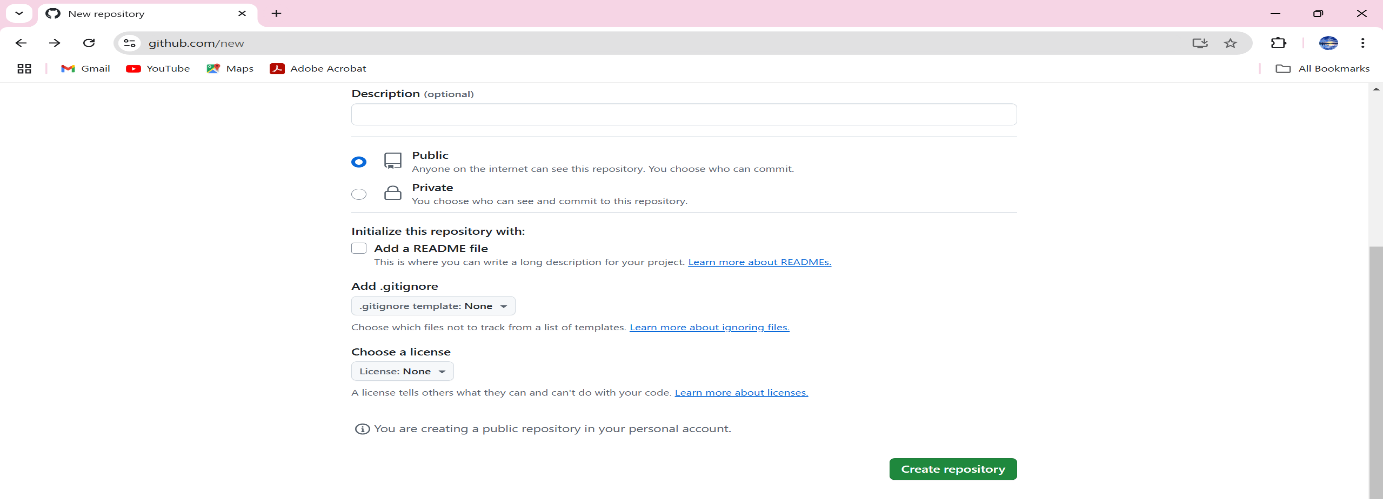
**Git Remote:**

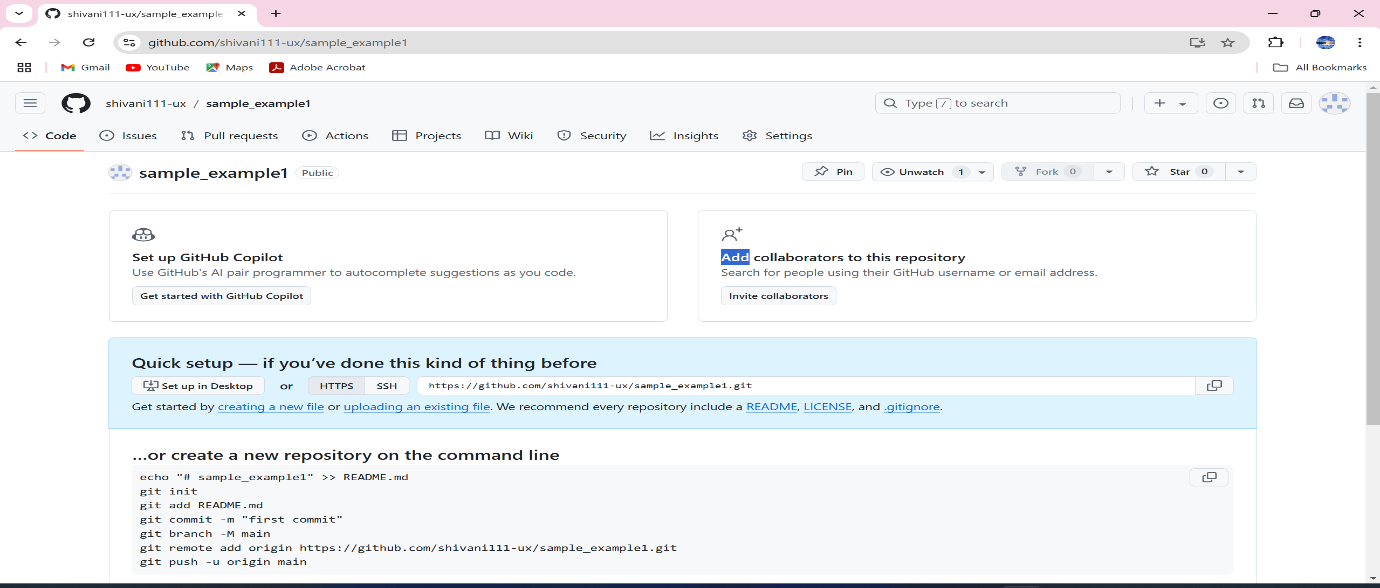
* “git remote lets you manage connections to other Git repositories stored on the internet (like GitHub or Bitbucket).”
* In Git, remote is how you save the address of your project that’s stored online.
* You use this command to add, remove, or view the places (remotes) where your project can be stored.

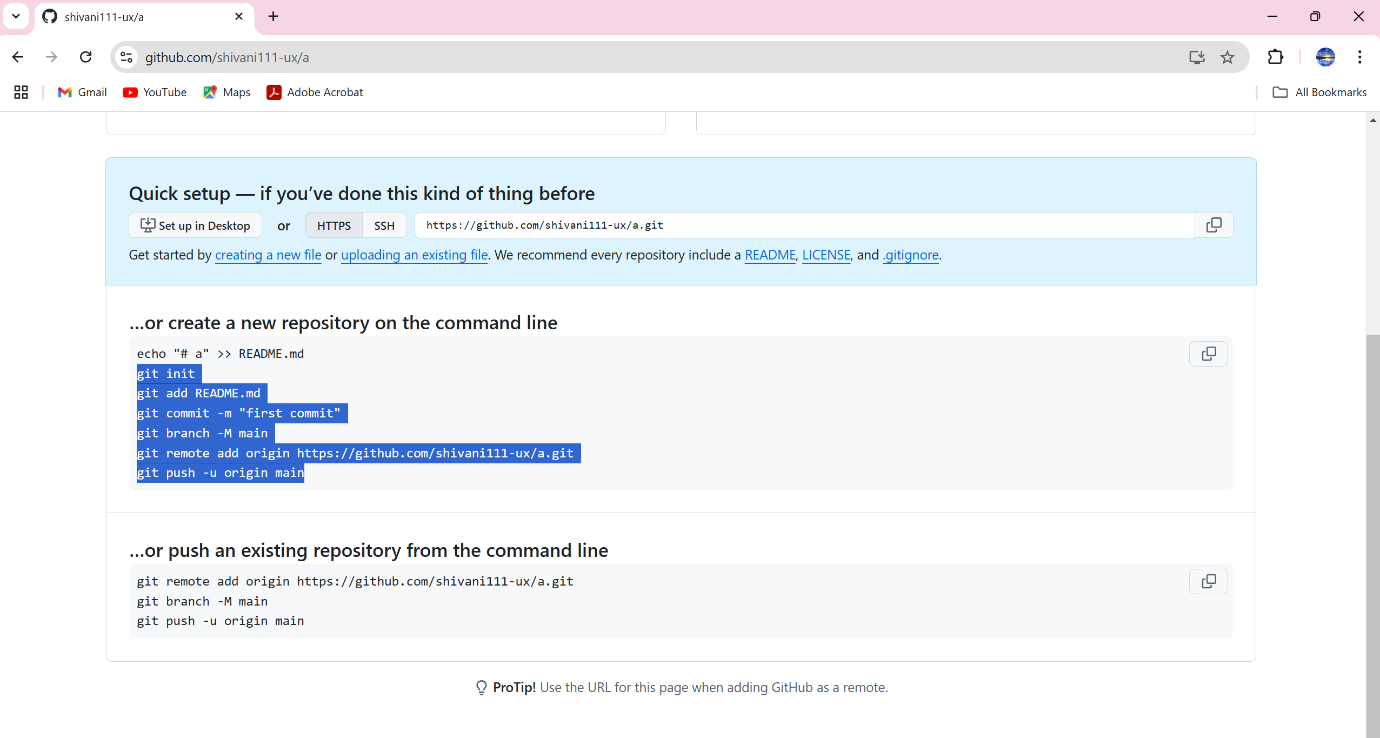
**Example**: **git remote add origin https://github.com/user/repo.git**

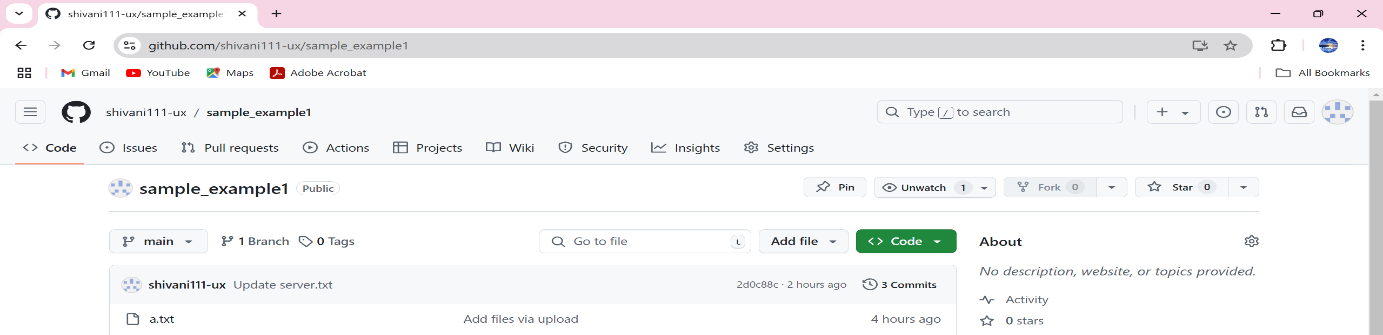




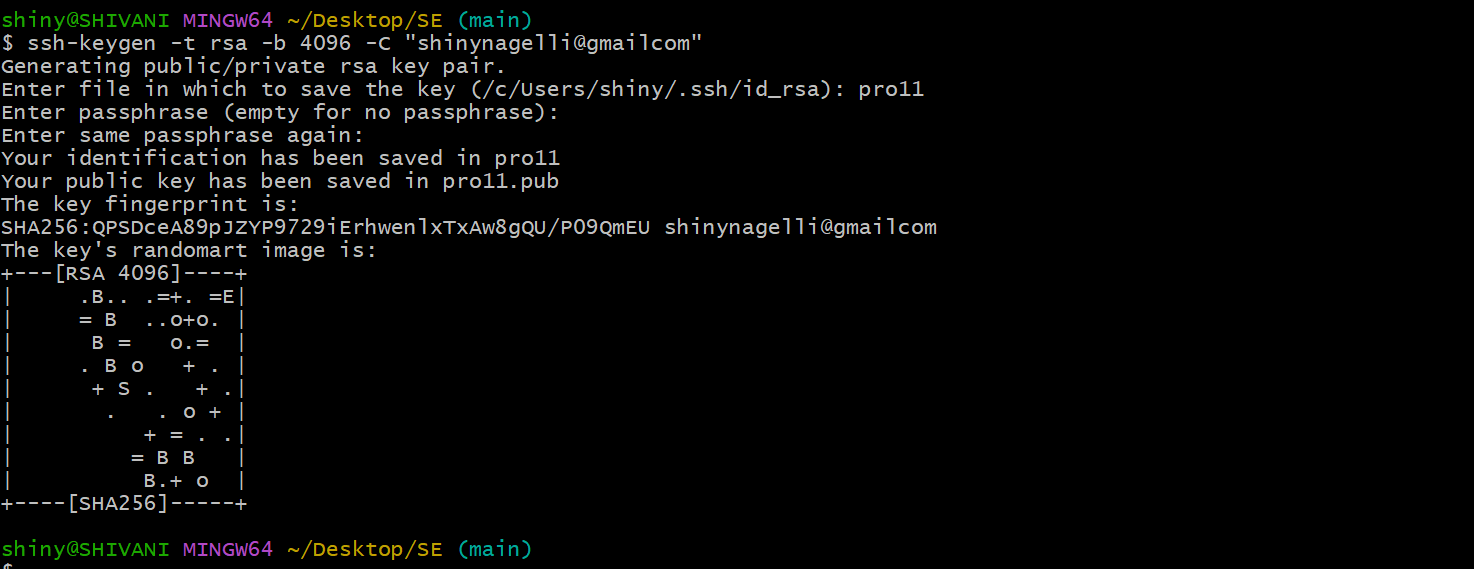


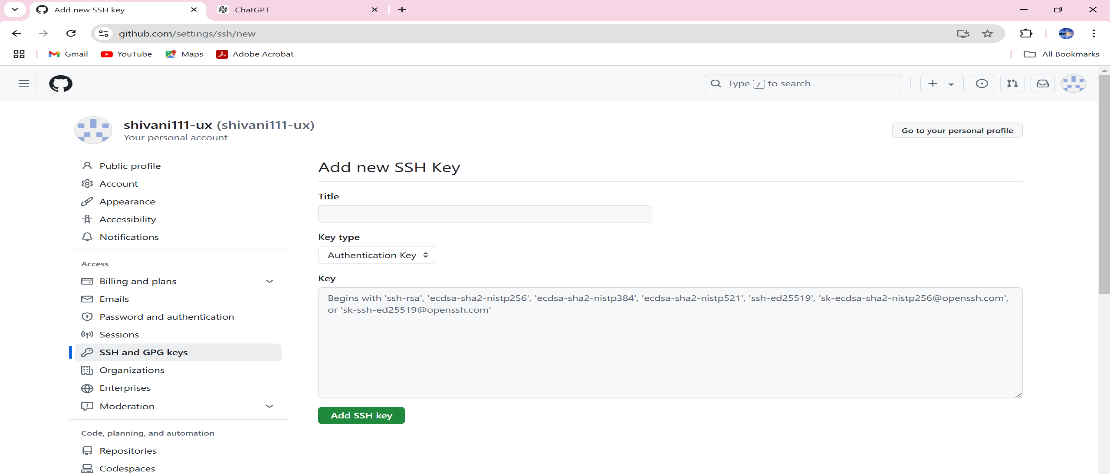


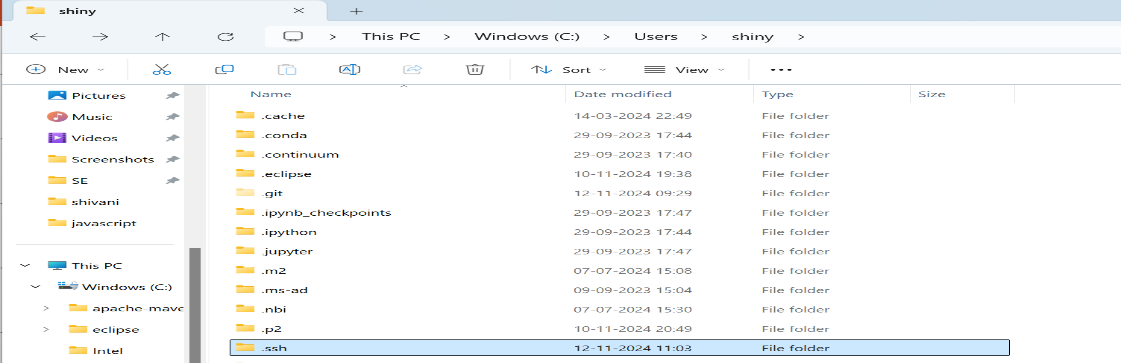


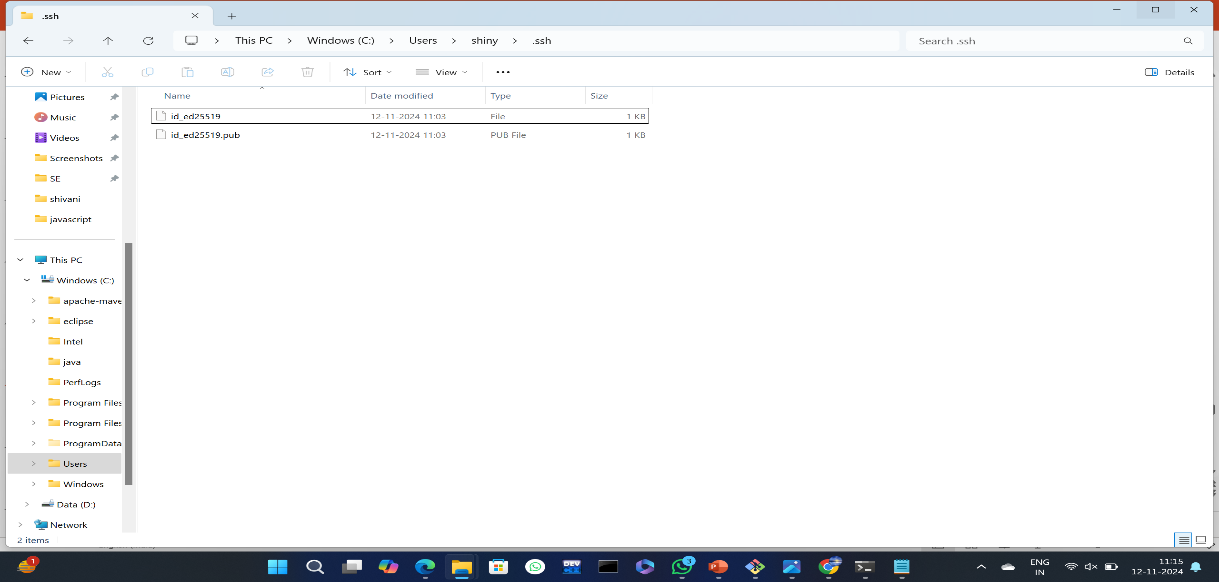


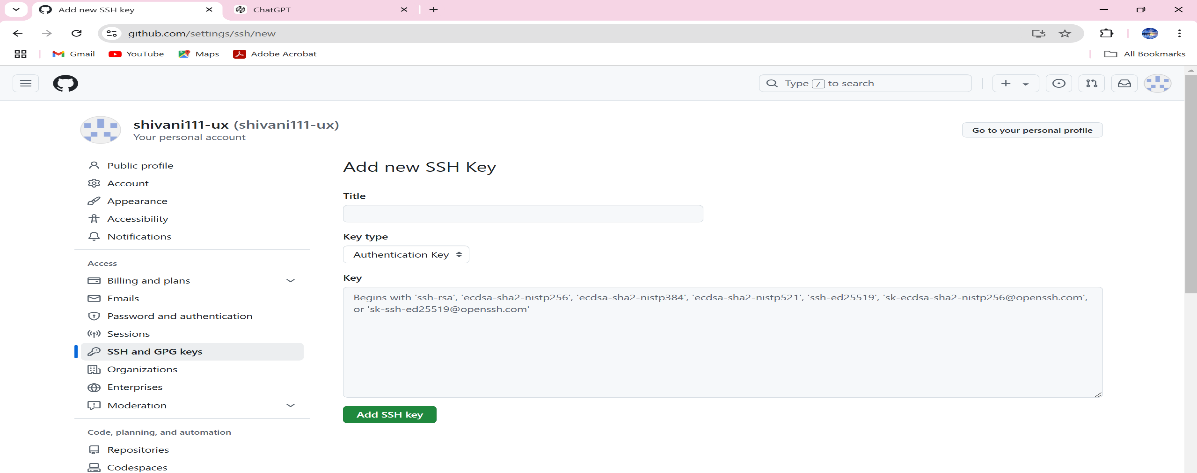
**Connect Git to GitHub using an SSH key:**

* “SSH Key: It’s a secure method to connect your computer with GitHub without using your password every time.
* The SSH key pair consists of two parts:
  + Private Key: This stays on your computer and acts as a secret key.
  + Public Key: This is shared with GitHub to verify that it’s really you when you push code.

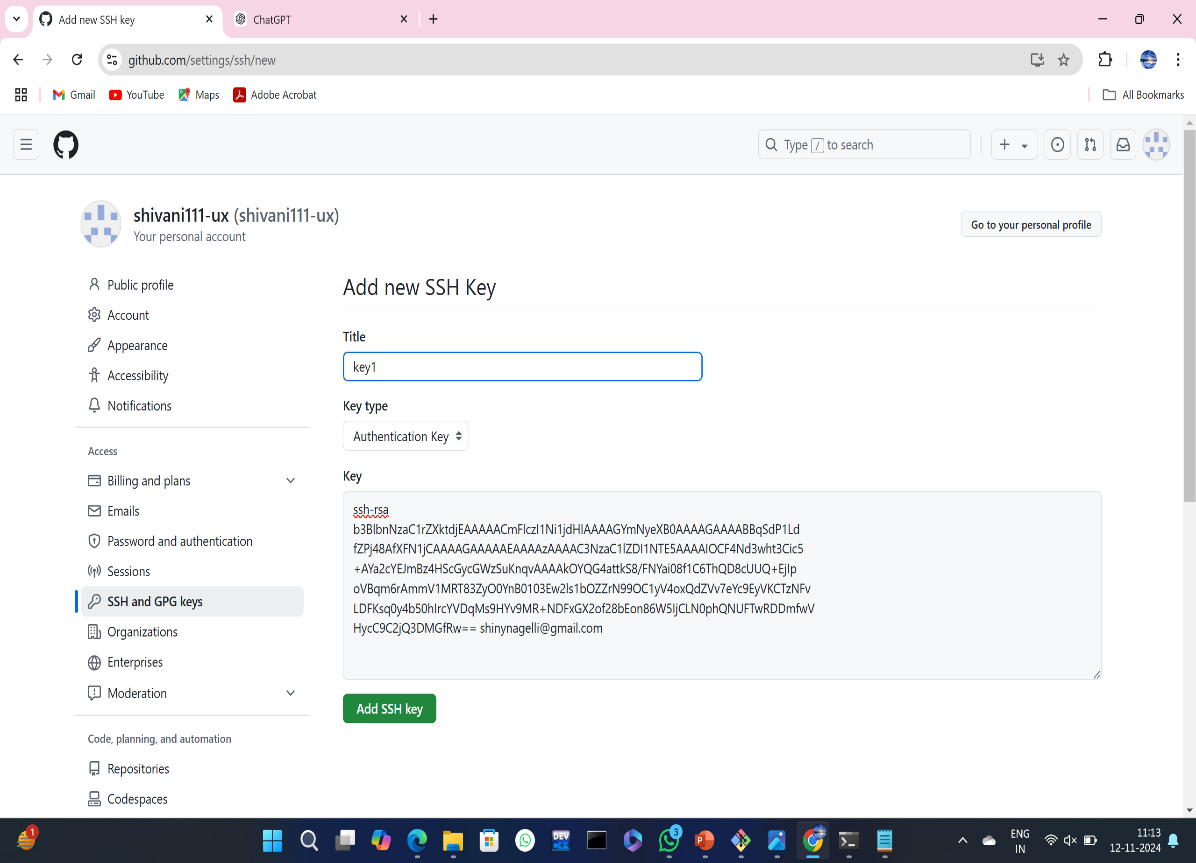








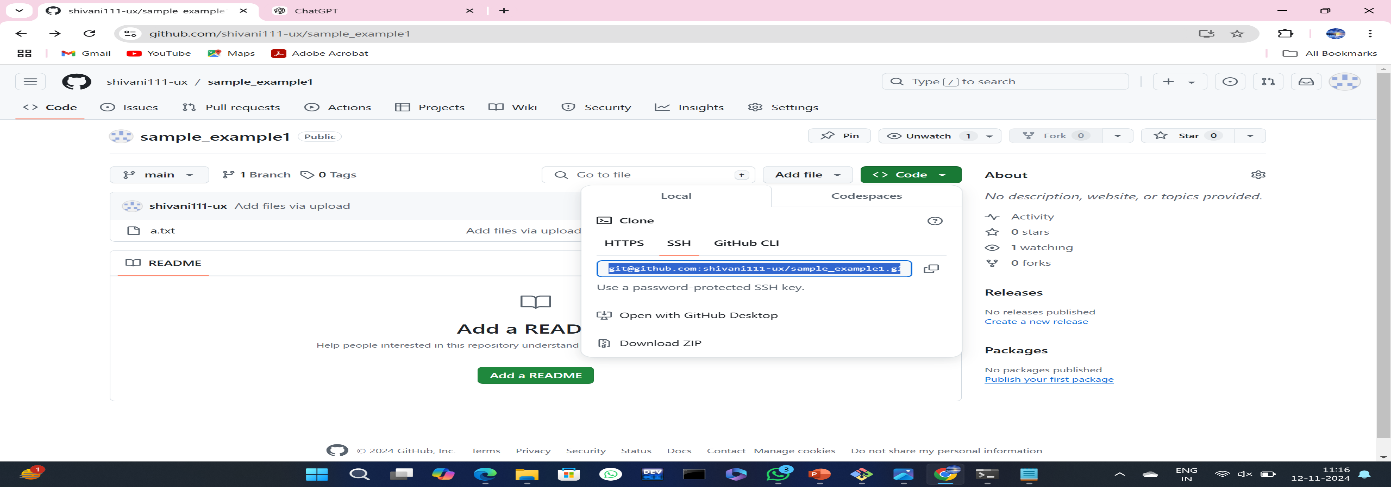
* + **Create SSH Key:** You use a command **(ssh-keygen -t rsa)** to generate a key pair on your system.
  + **Copy Public Key:** You then copy the public part of the key **(the .pub file**).
  + **Add to GitHub:** You go to GitHub settings, navigate to “**SSH and GPG keys**,” and paste the public key. This lets GitHub know it’s your computer connecting.

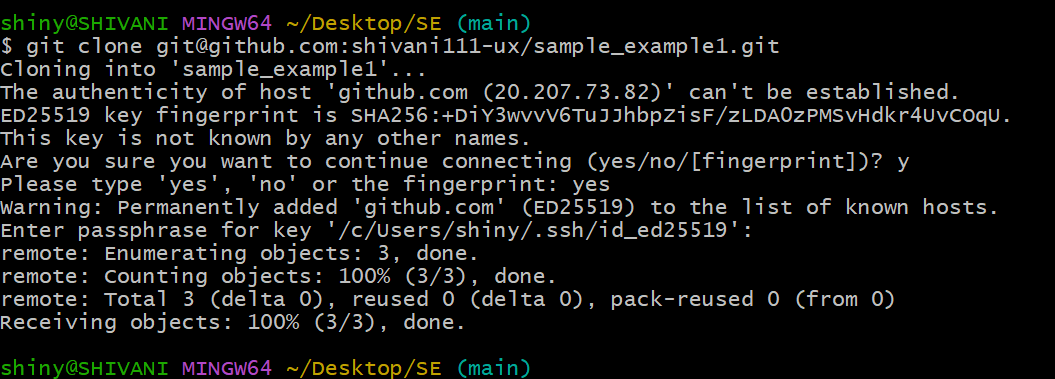


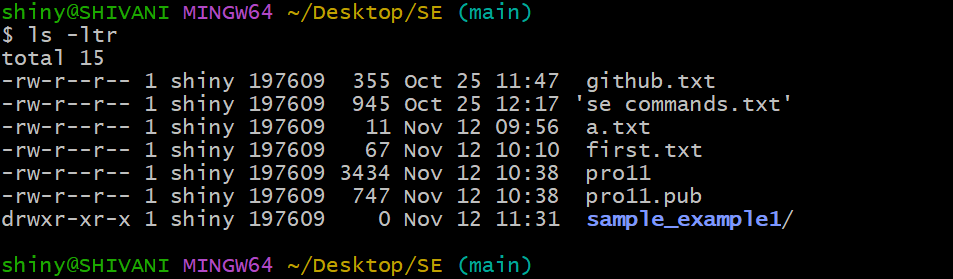


**Git Clone:**

* “git clone is used to download a copy of a project from a remote repository (like GitHub) to your own computer.”
* **Example**: **git clone** [**https://github.com/user/repo.git**](https://github.com/user/repo.git)
* When you want to start working on a project that’s stored online, use **git clone** to create a local copy on your machine.





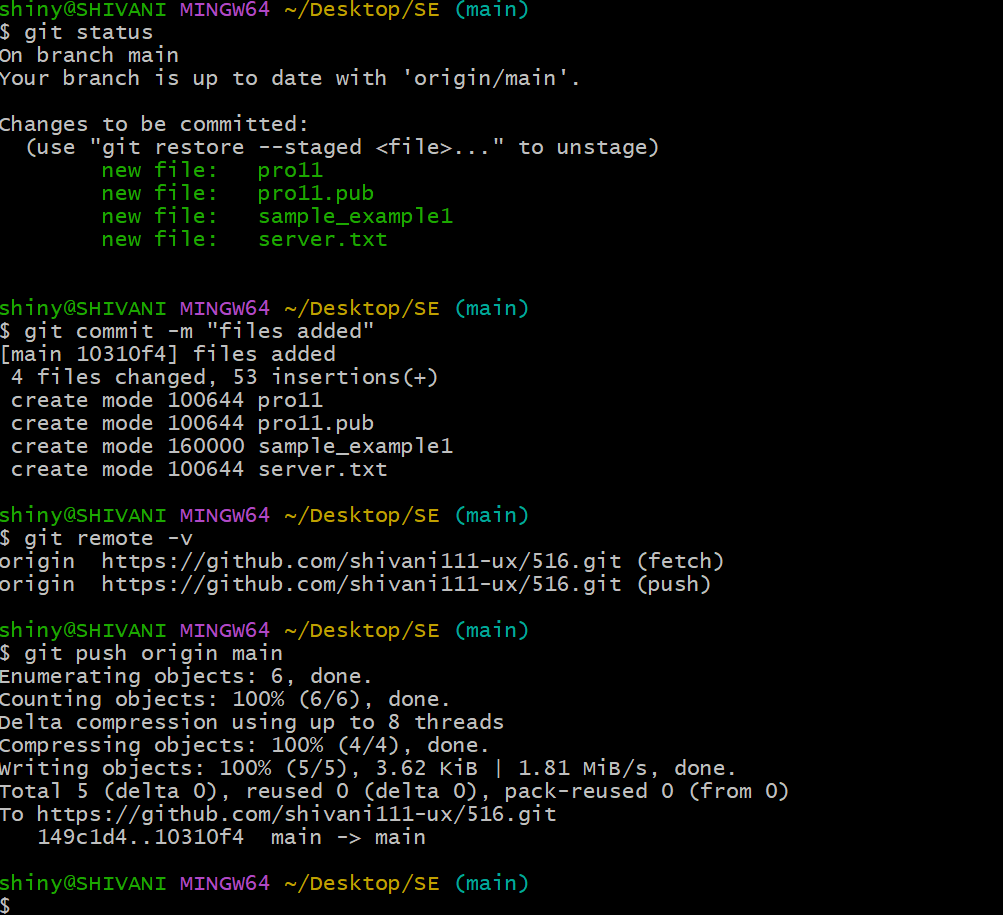


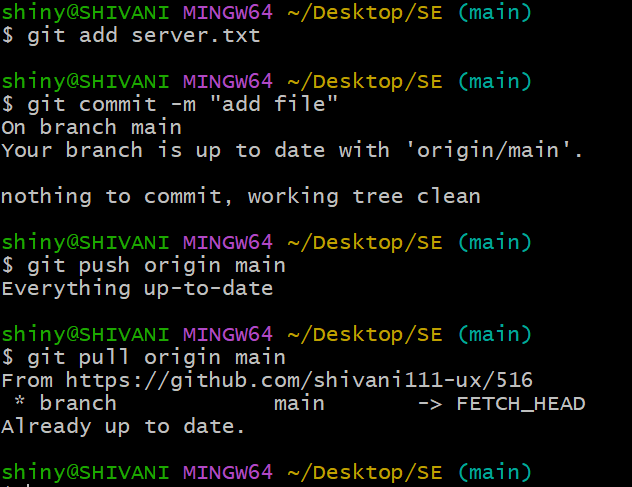
* ls –ltr will give the information of latest modified files along with date and time in the cloned GitHub.
* Move into the newly created repository folder.

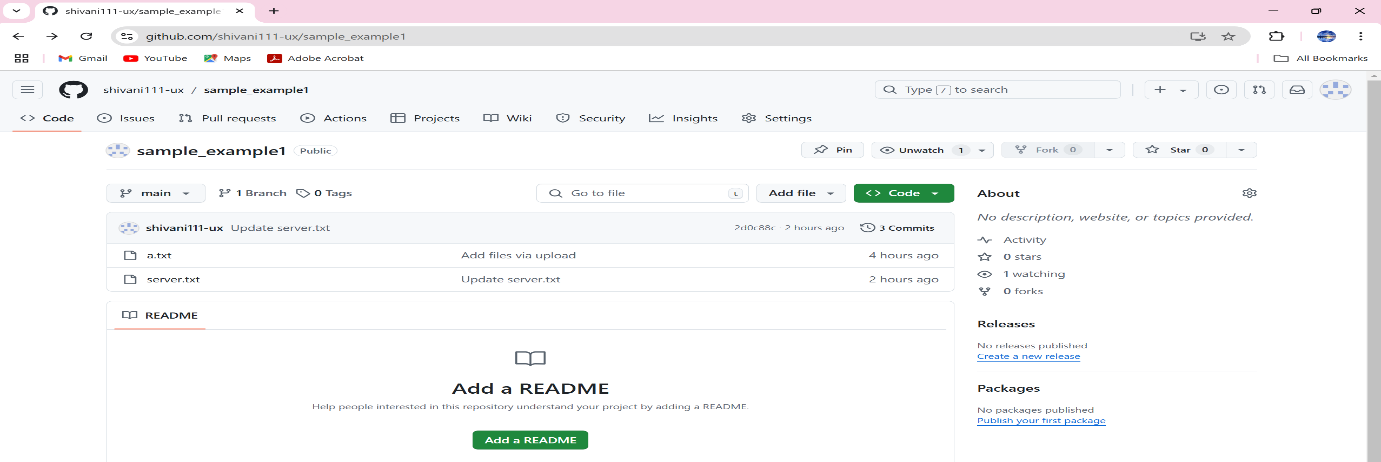
**Git Push:**

* “git push is used to send your changes from your local repository (on your computer) back to the remote repository (like GitHub). This is how you share your work with others.”
* **Example**: **git push origin master**
* After you’ve made changes on your computer, use **git push** to
* After you’ve made changes on your computer, use **git push** to upload those changes to the remote repository.



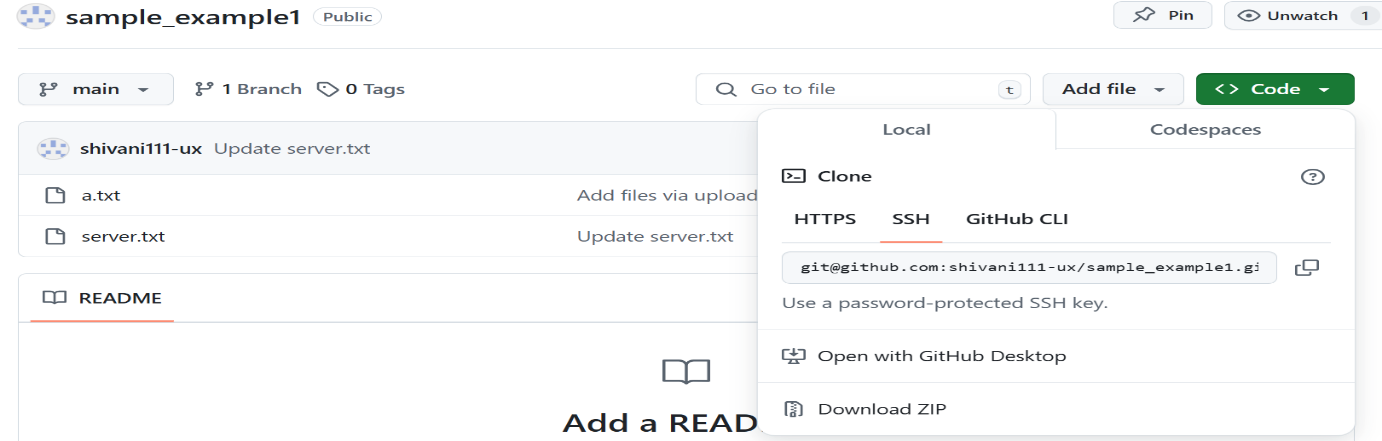


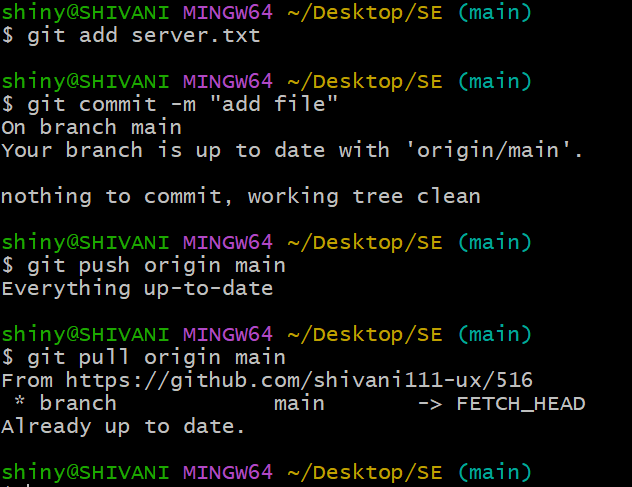


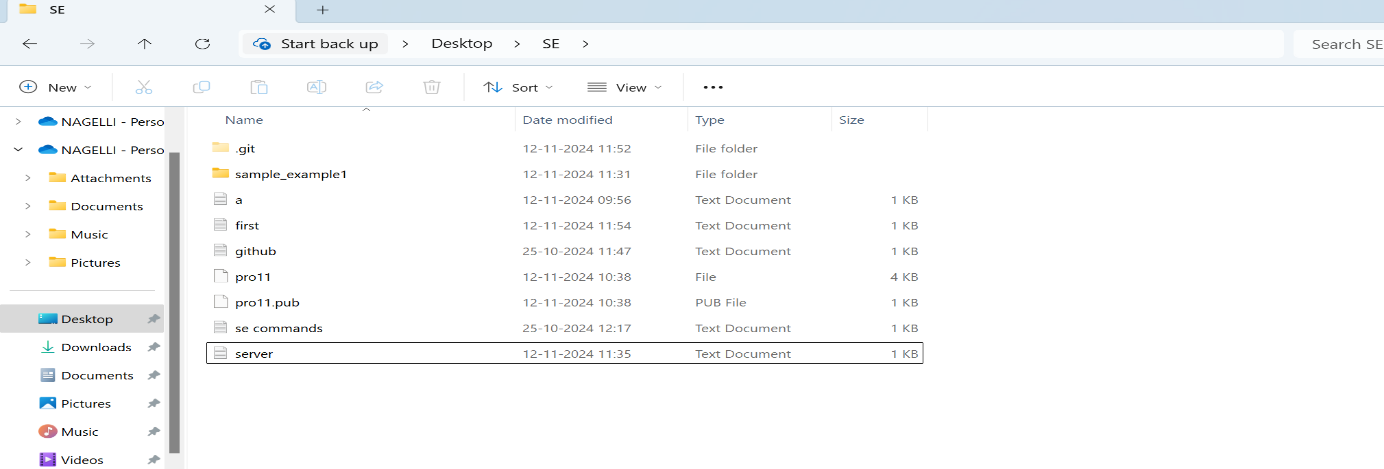


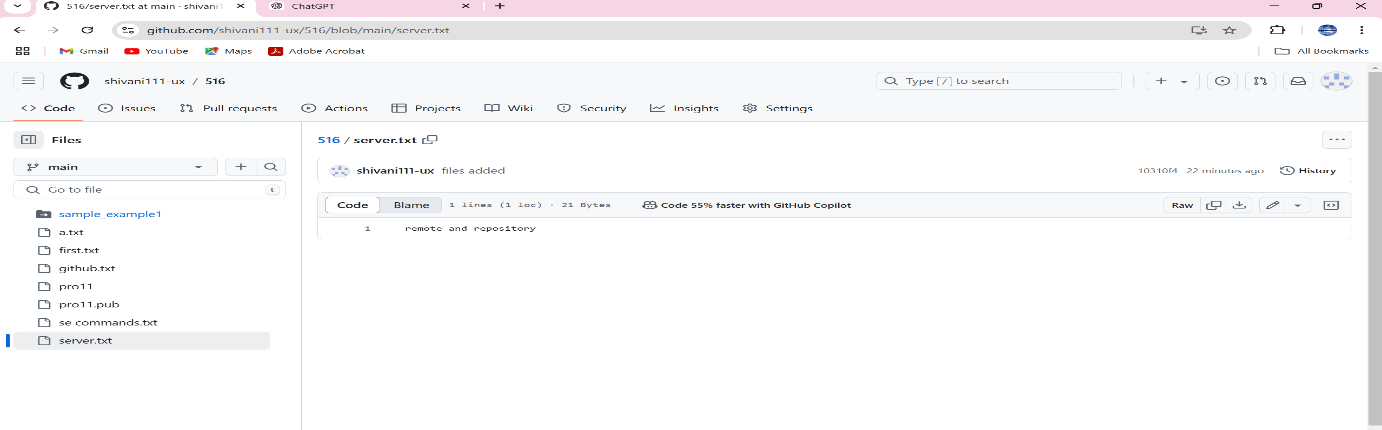
**Git Pull:**

* “git pull brings the latest updates from a remote repository to your local repository. It fetches the changes and merges them with your current work.”
* **Example**: **git pull origin master**
* If your team members made changes to the project online, you use **git pull** to update your version with their latest changes.

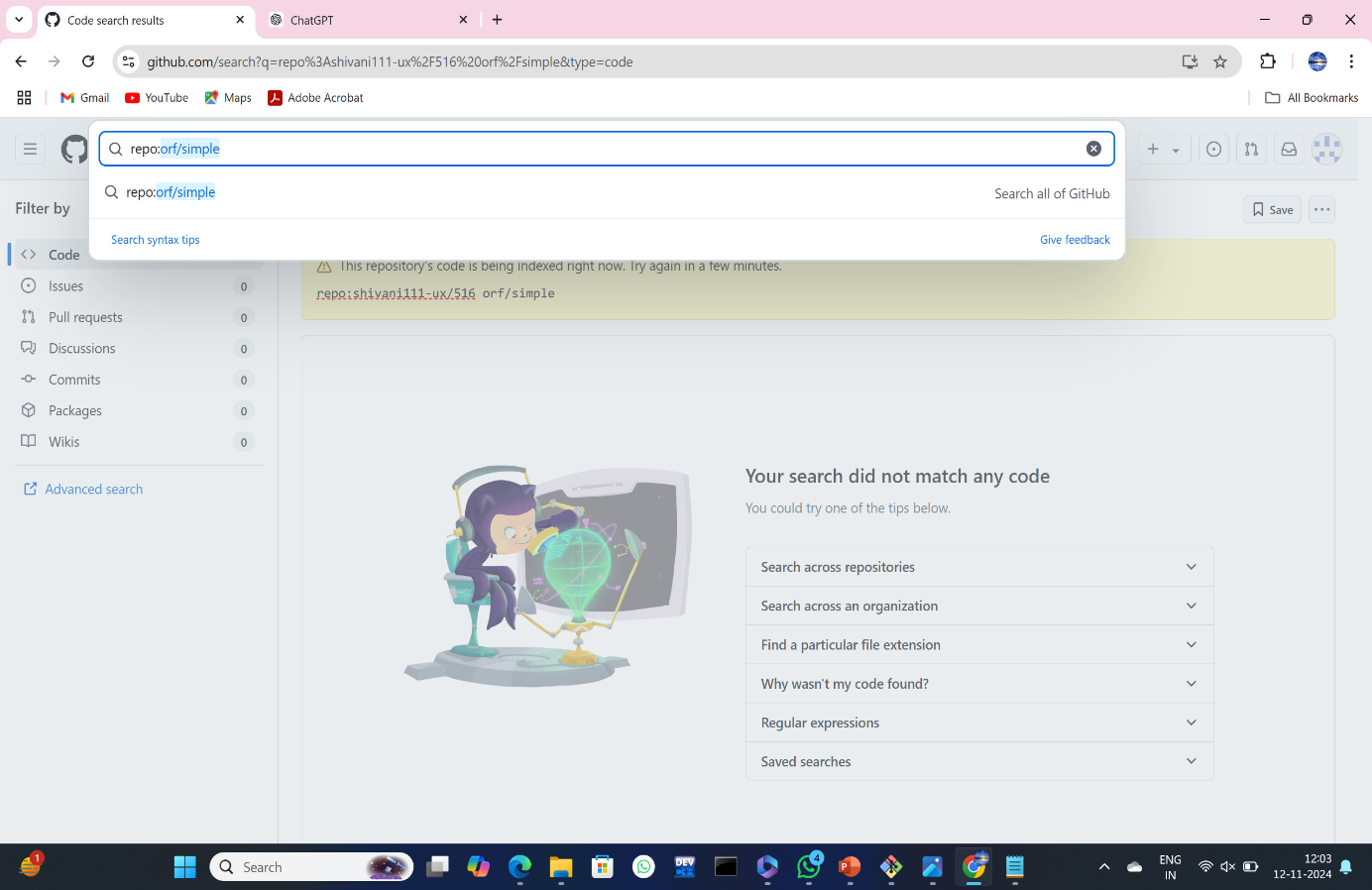


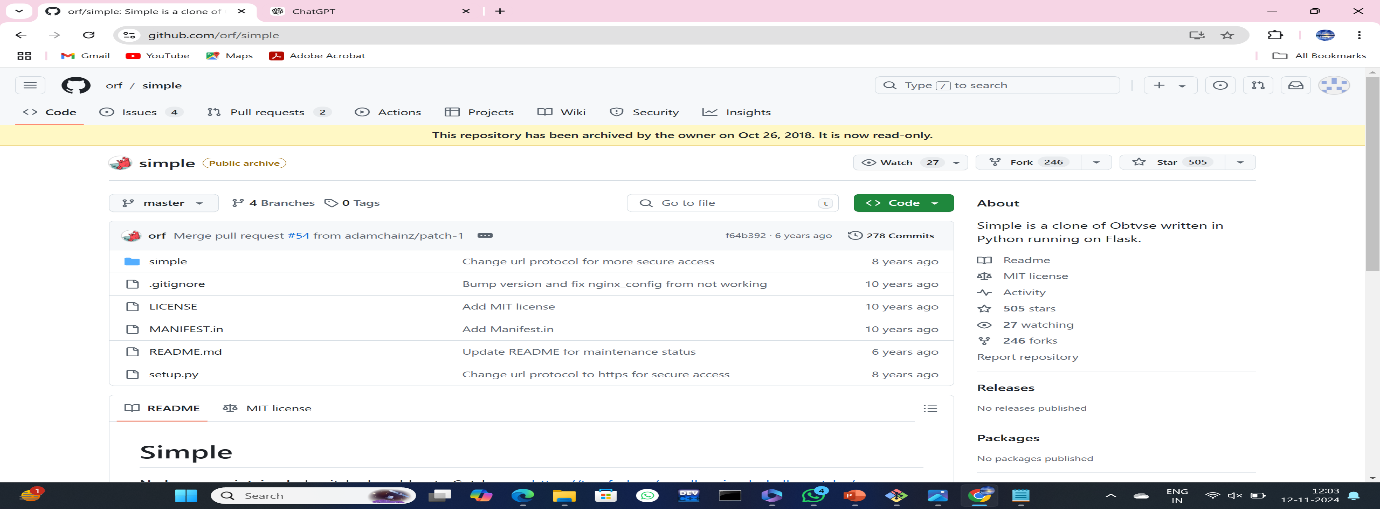


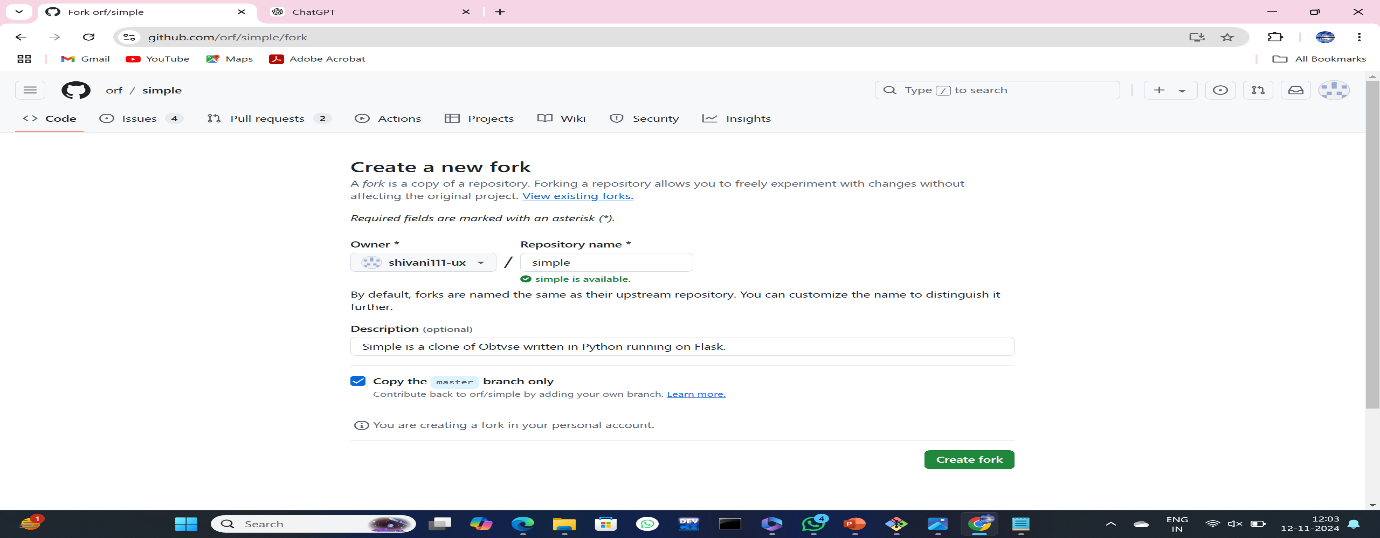


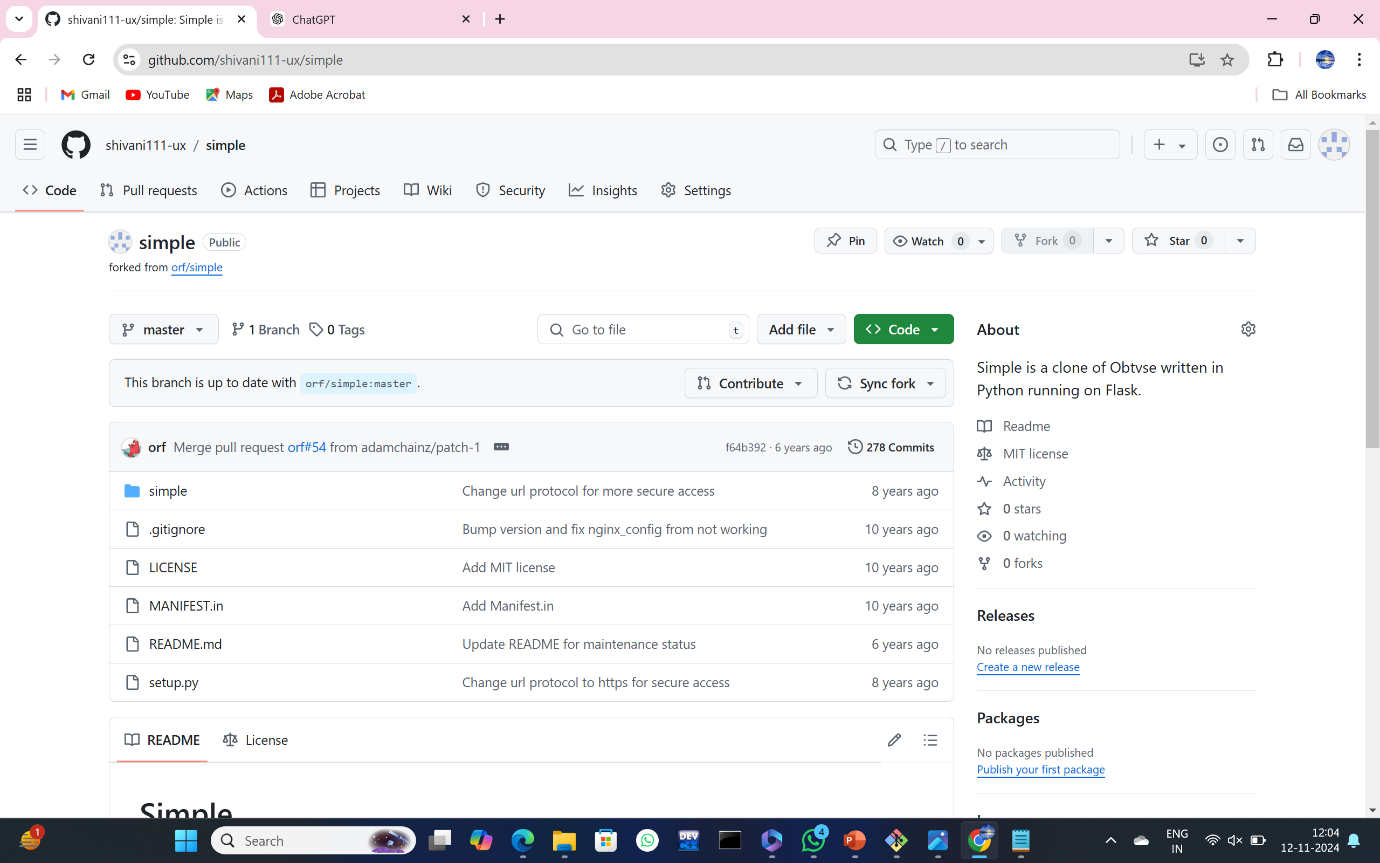


**Git Fork:**

* A fork is a personal copy of someone else’s project that you can edit and change without affecting the original project.”
* It’s usually used in open-source projects where you want to make your own changes and eventually propose them back.
* **Example:** To fork a repository:
* Go to the original repository on GitHub.
* Click the "Fork" button to make your own copy.
* 







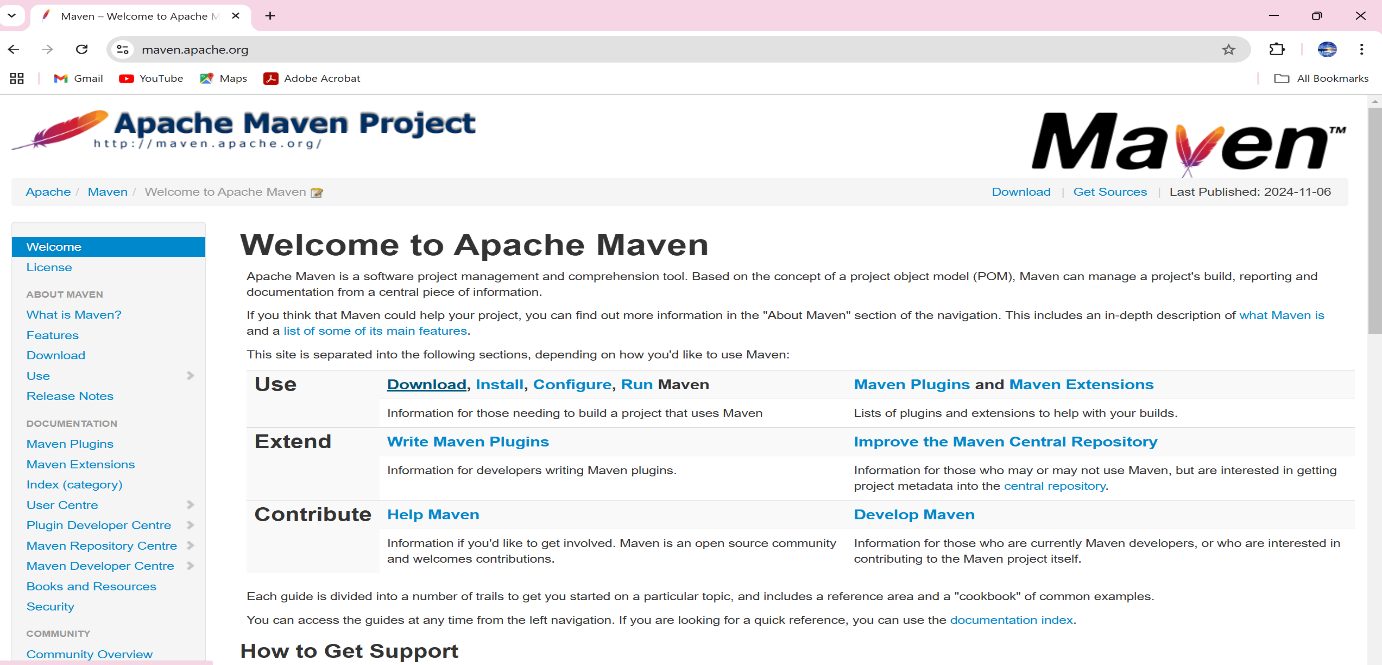
**INSTALLATION OF MAVEN:**

Step 1: Check whether maven is present in your system or not

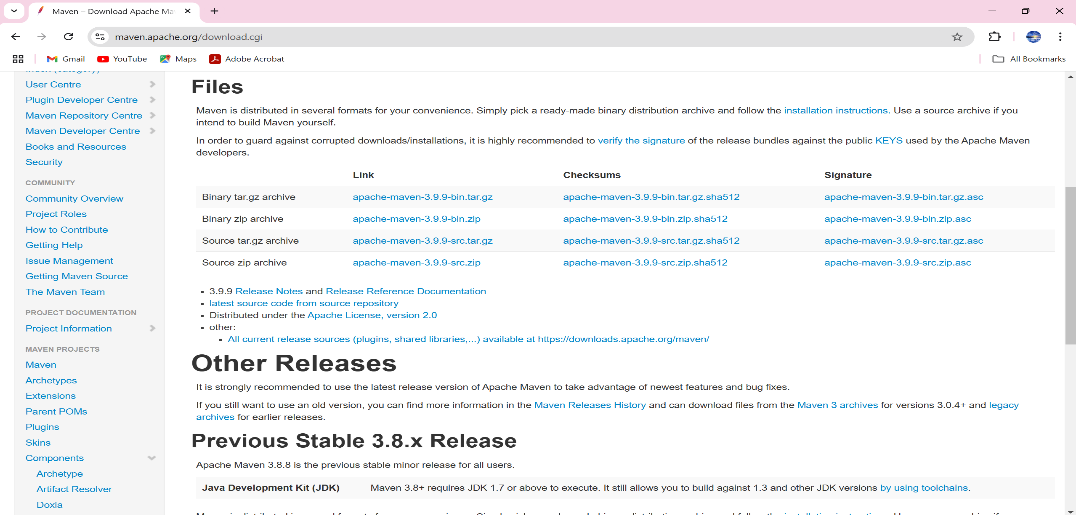
Step 2: Go to Google type download maven and click on download Apache maven.



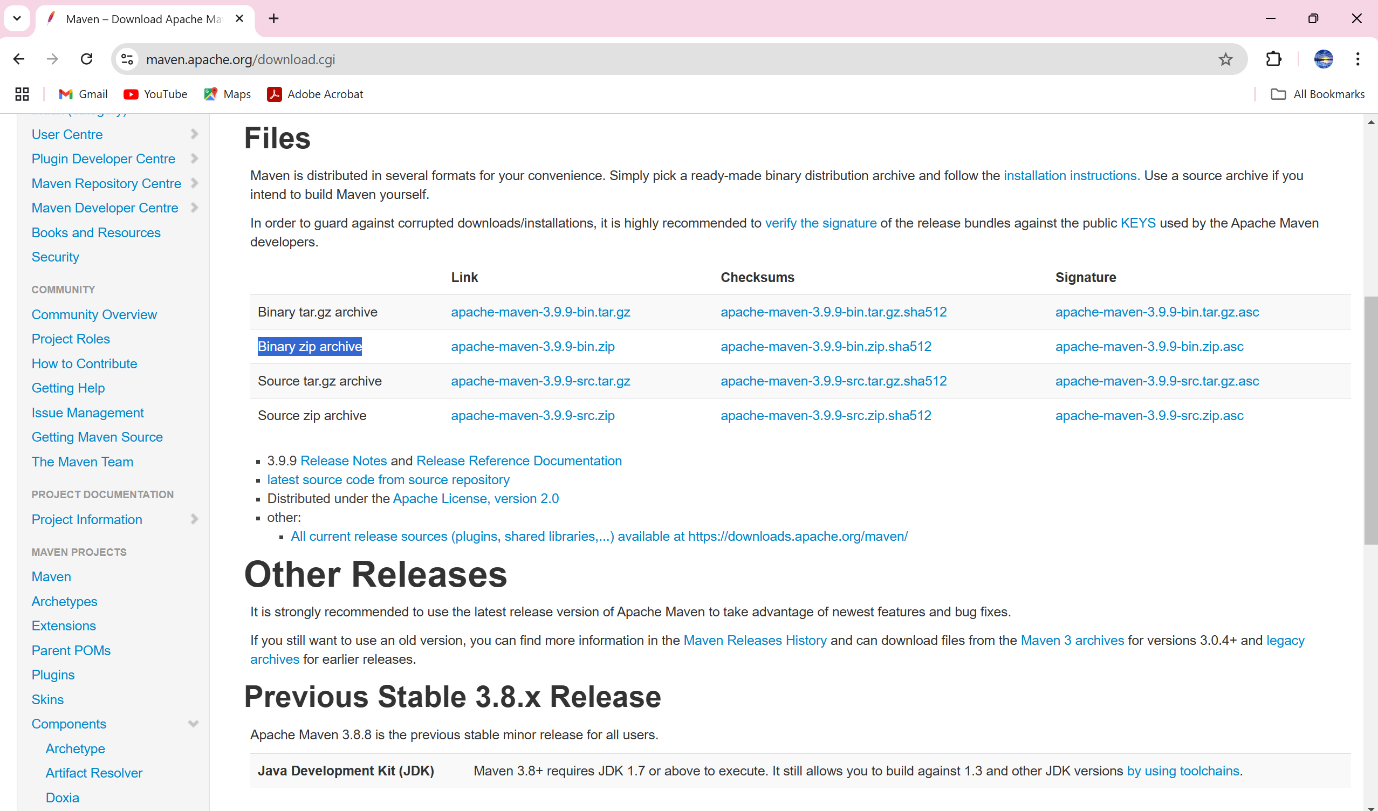
Step 3: Open the Maven Official Webpage



Step 4: Scroll down to go to downloading links.



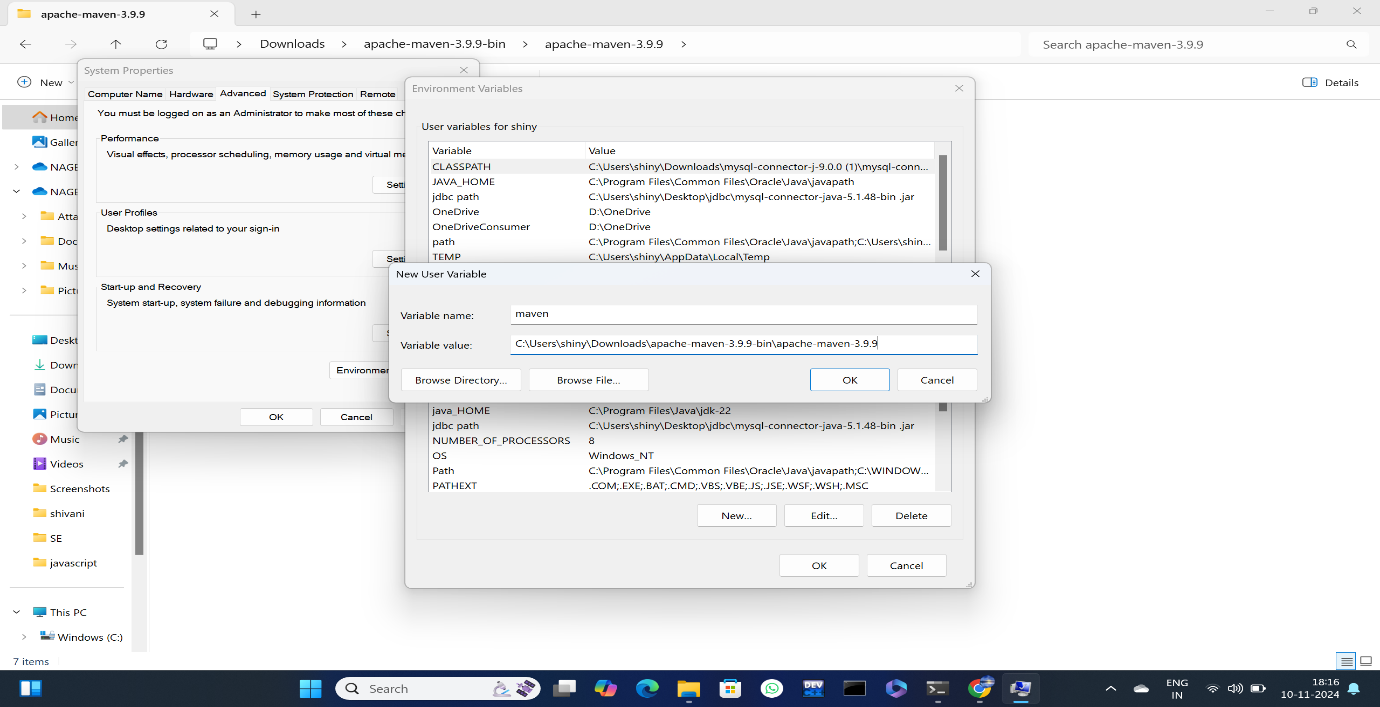
Step 5: click on apache-maven-3.3.9-bin zip.



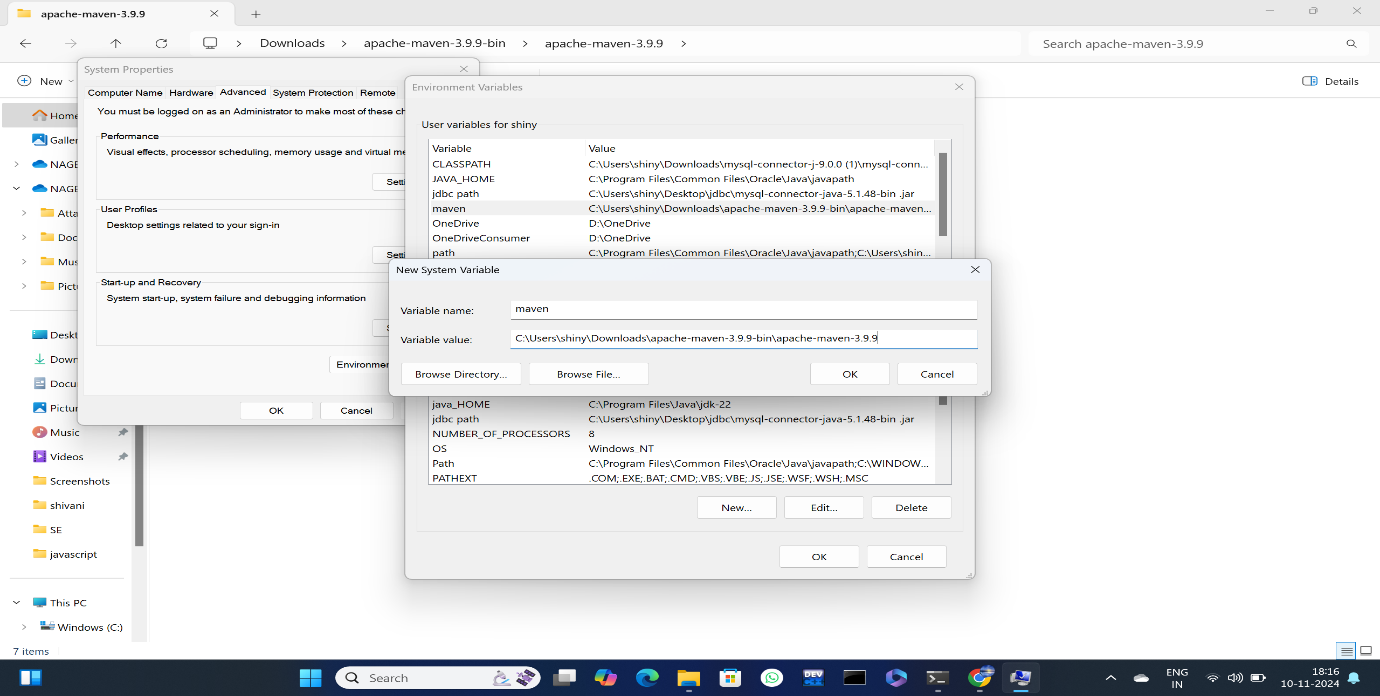
Step 6: extract zip file.

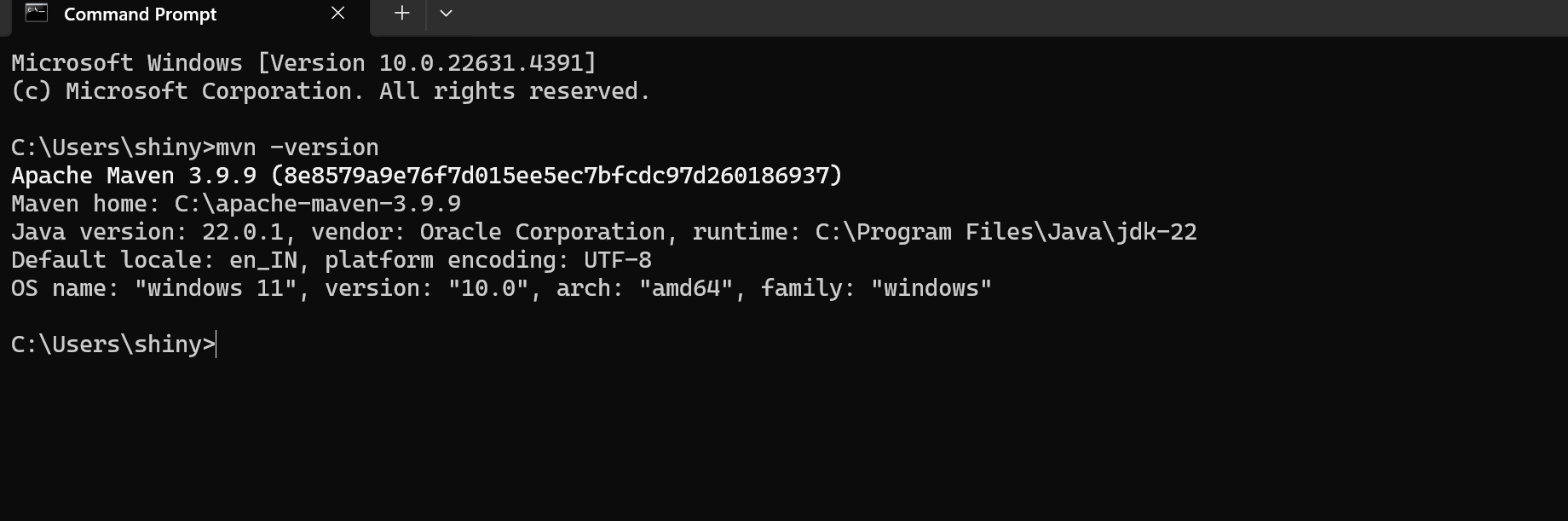


Step 7: set environment variables.



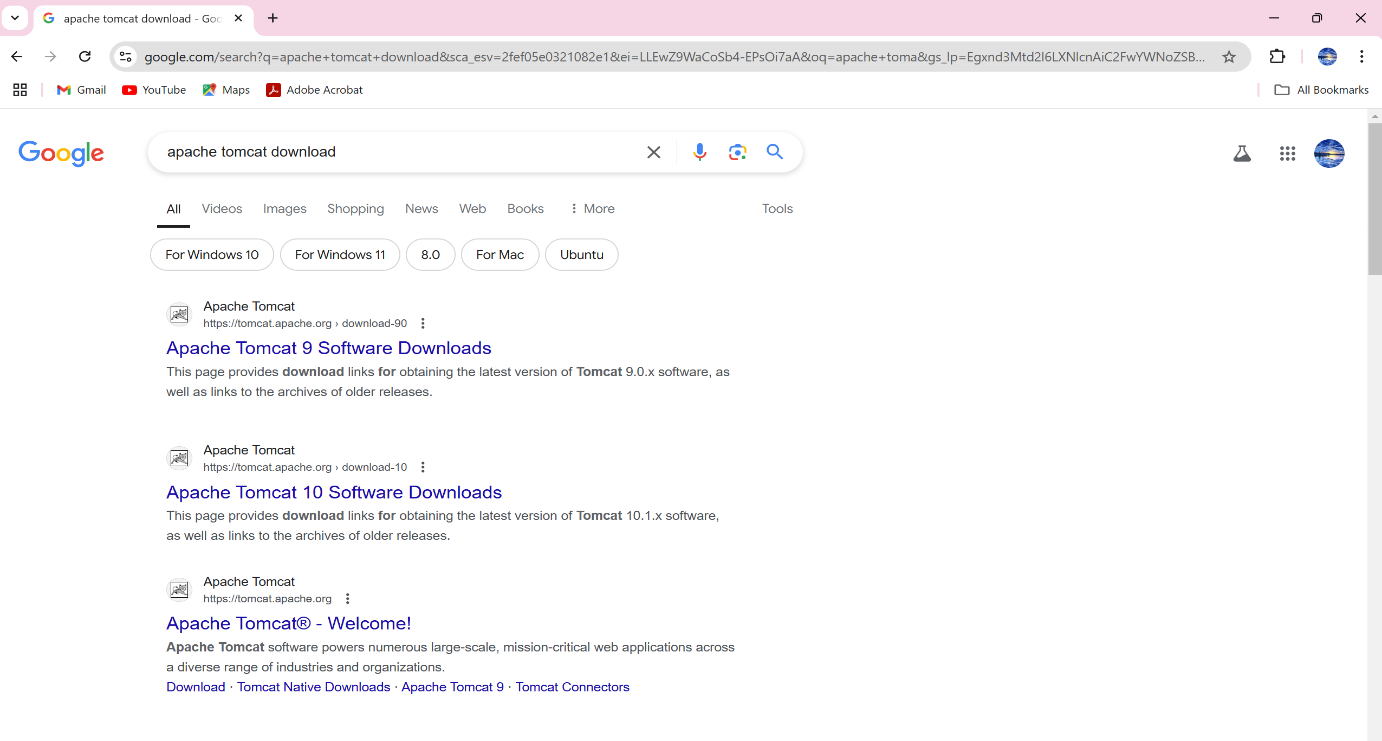
Step 8: Edit path.



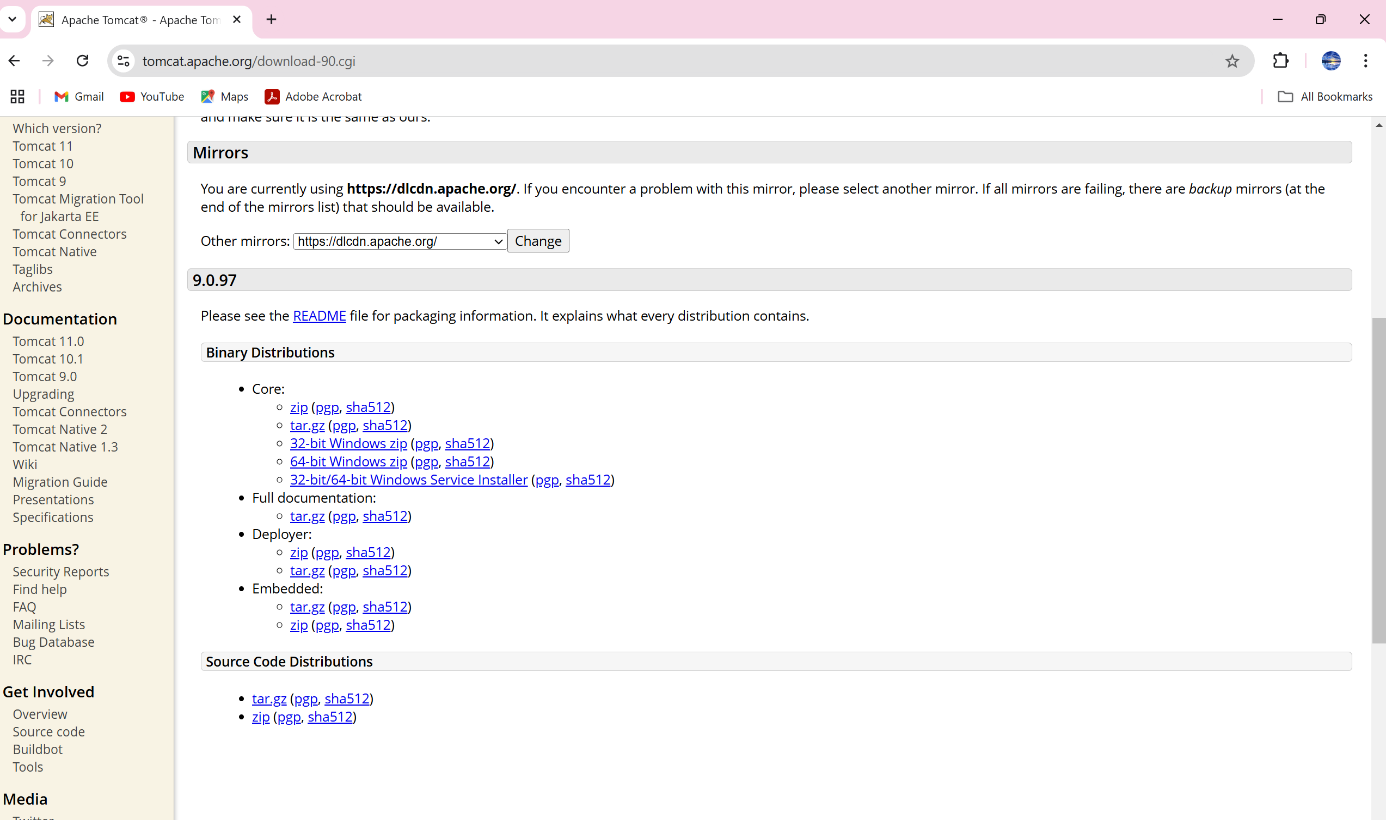


**INSTALLATION OF APACHE TOMCAT:**

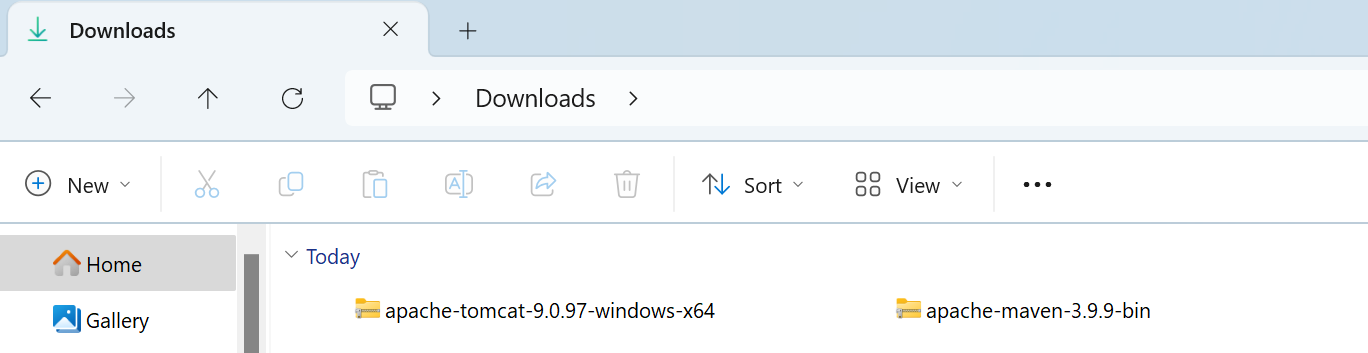
**Step1:** Download the tomcat from the first link.

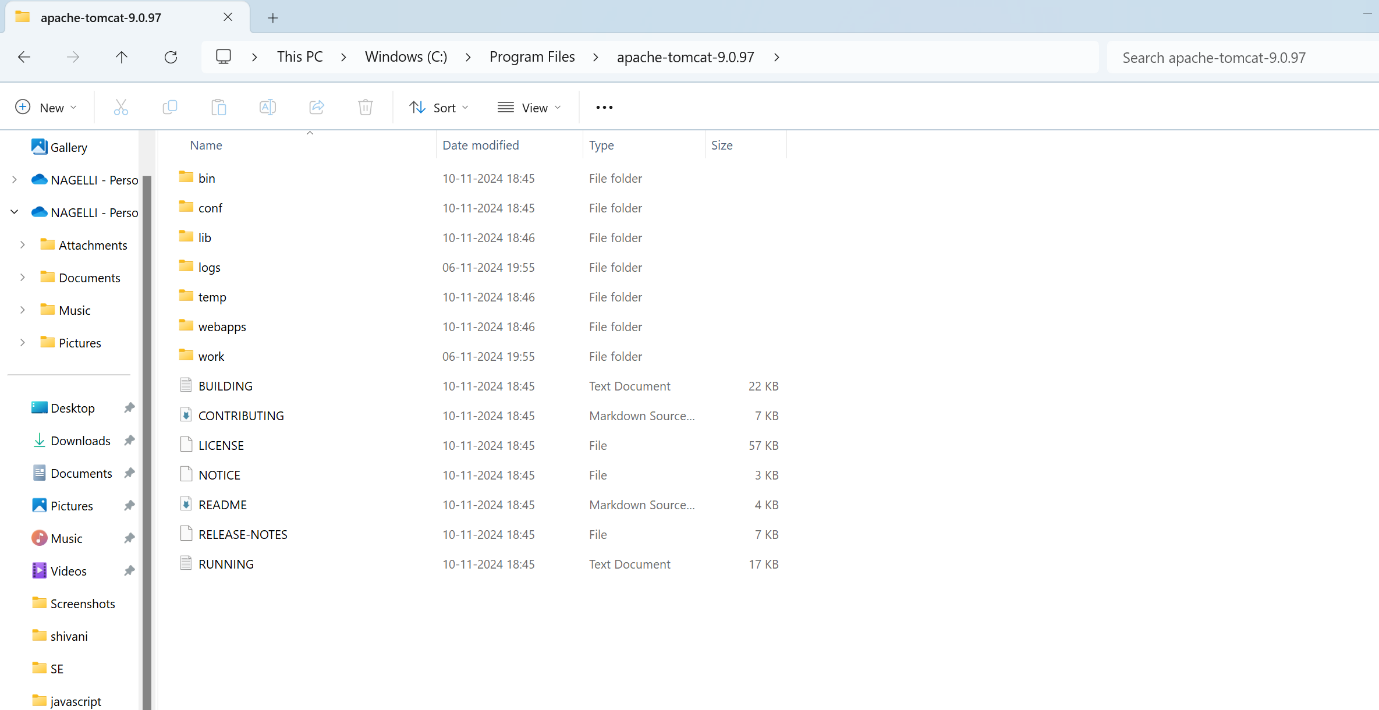
****

**Step 2:** Click on the following link 64-bit windows zip to download the zip file.



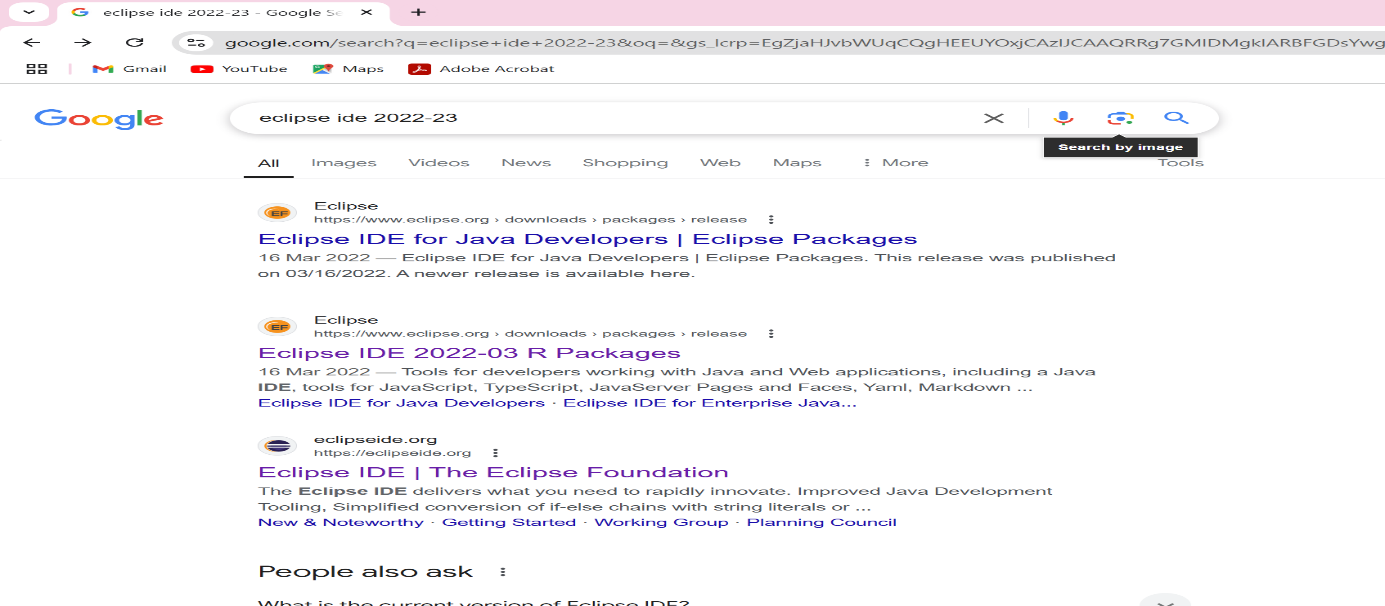
**Step 3:** We can view the downloaded file from Downloads.

****

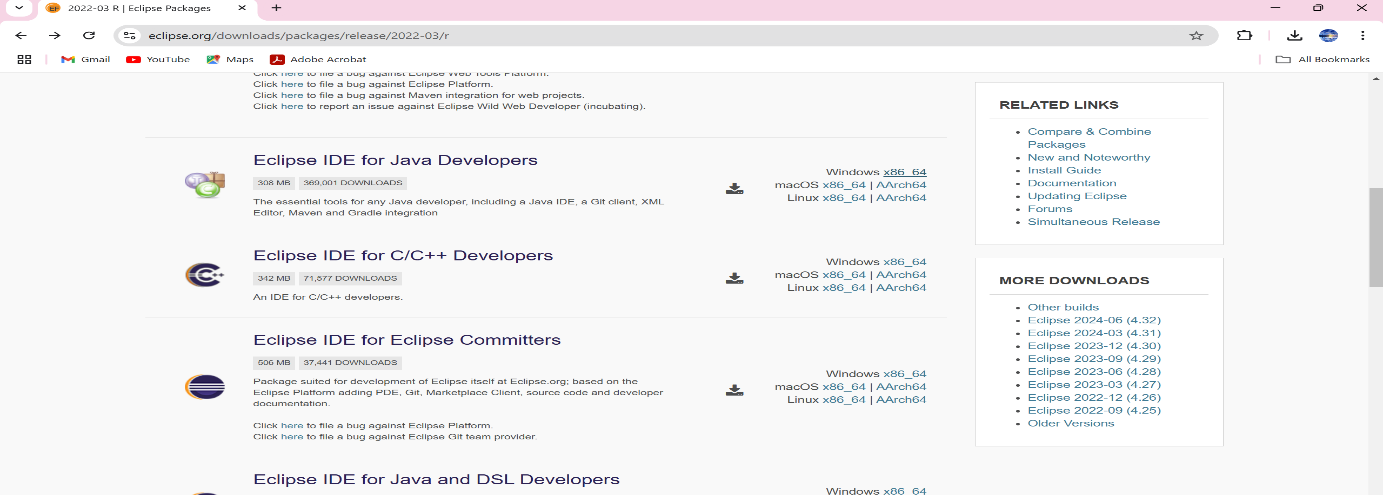
****

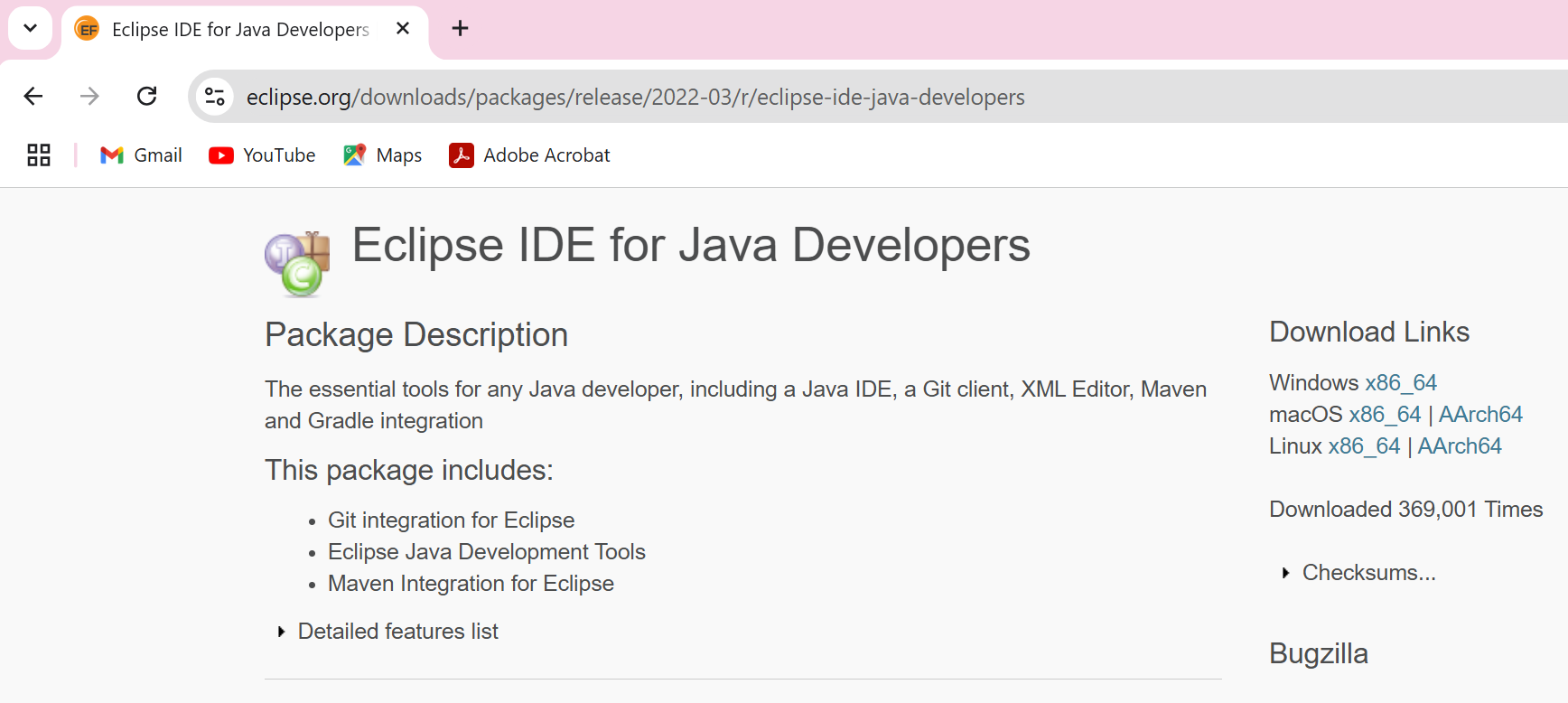
**INSTALLATION OF ECLIPSE IDE:**

**Step1:** Download the Eclipse.

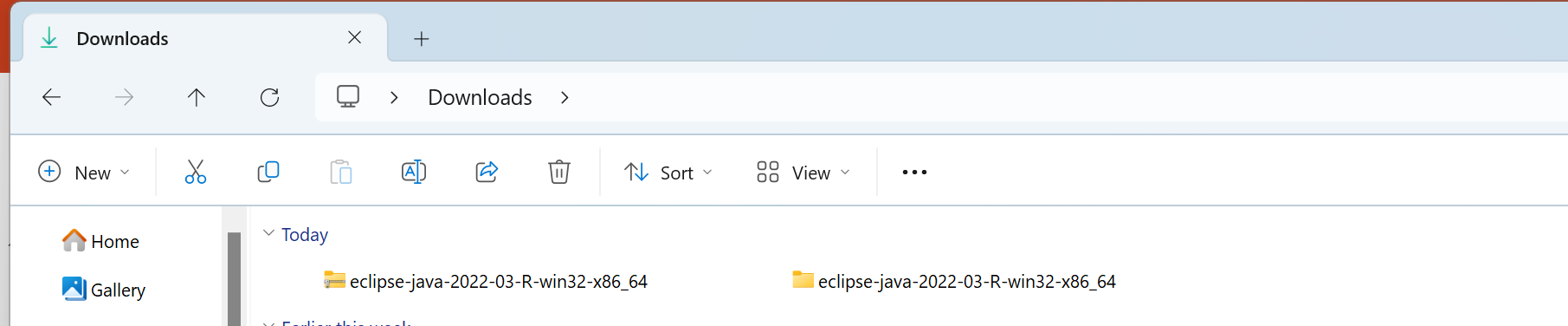
****

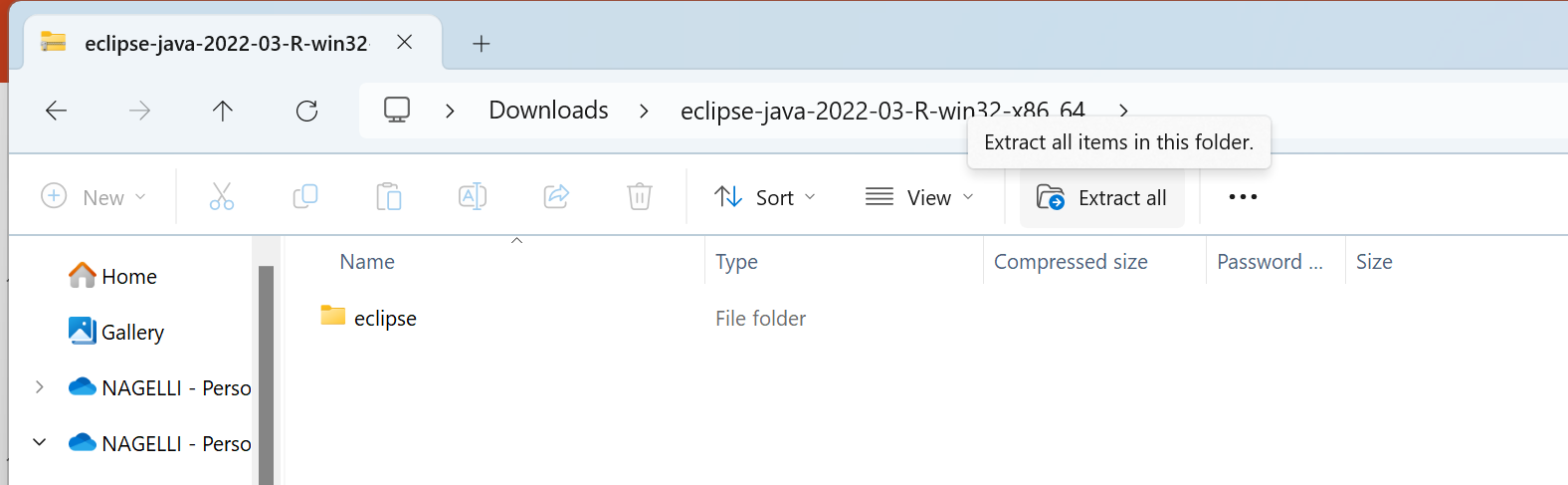
**Step 2:** Click on the windows zip to download the zip file.

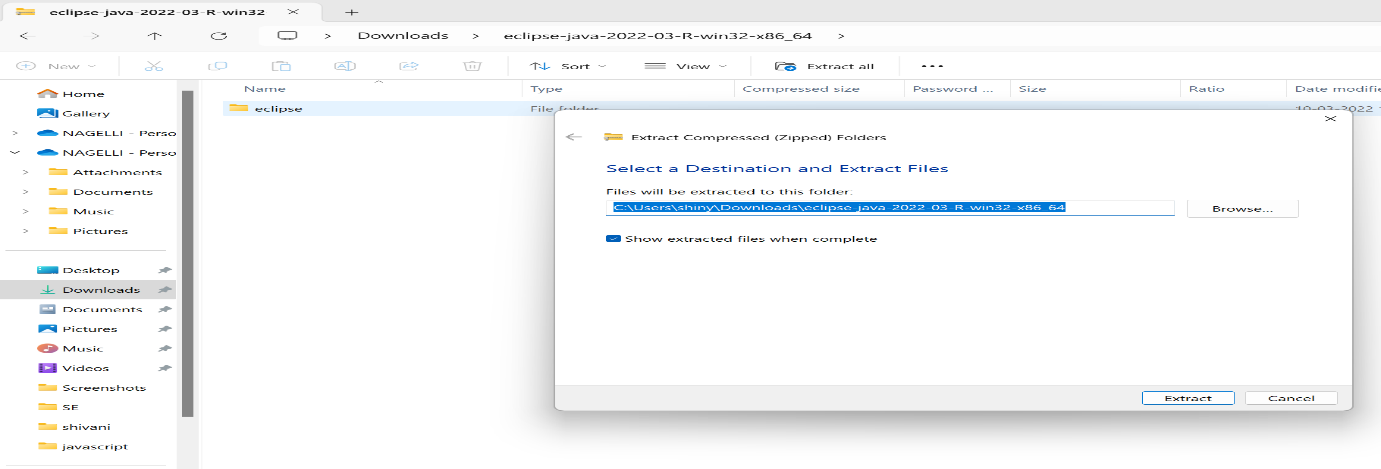
****

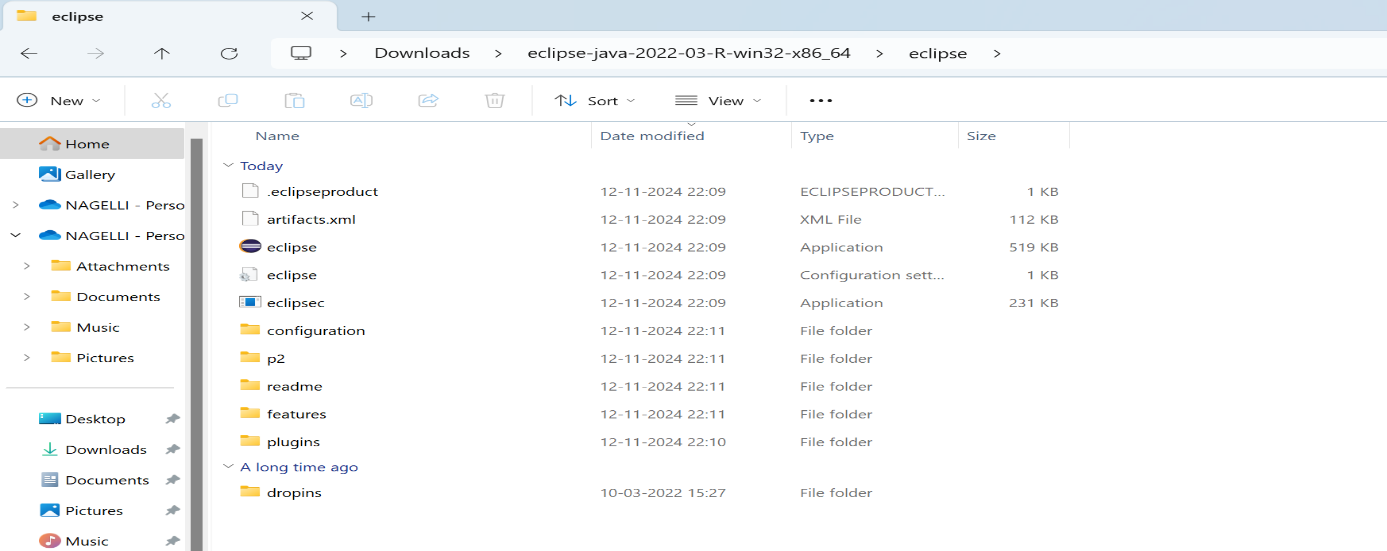
****

**Step 3:** We can view the downloaded file from Downloads.

****

****

****

****