You will learn below topics while following this document:

* **Git and Gitub Basics -** Page#1
* **Git download steps –** Page#3
* **Create an account on GitHub -** Page#12
* **Push the project to github –** Page#13

**Git and Gitub Basics**

**1. What is Git?**

✅ **Git is a version control system (VCS)** used to track changes in source code during software development.  
✅ It allows multiple developers to collaborate, track changes, and revert to previous versions when needed.  
✅ Git is **installed locally** on your machine and can work **without** an internet connection.  
✅ Commands like git init, git add, git commit, and git push are used to manage code.

🔹 **Example:**  
When you modify a file and commit it using Git, the changes are saved in the local repository.

**2. What is GitHub?**

✅ **GitHub is a cloud-based platform that hosts Git repositories** for collaboration and sharing.  
✅ It allows multiple developers to work on the same project using features like **pull requests, issues, branches, and actions**.  
✅ GitHub provides a **remote backup** for Git repositories, making it easy to share code.  
✅ You can access GitHub via a **web interface** or Git commands.

🔹 **Example:**  
When you push a Git repository to GitHub, it becomes available for others to view, clone, and contribute to.

**Key Differences Between Git & GitHub**

| **Feature** | **Git** | **GitHub** |
| --- | --- | --- |
| **Definition** | A version control system | A cloud-based hosting service for Git repositories |
| **Usage** | Tracks changes in files locally | Stores Git repositories remotely for collaboration |
| **Internet Required?** | ❌ No (works offline) | ✅ Yes (to sync changes) |
| **Installation Required?** | ✅ Yes (installed on local machine) | ❌ No (access via browser) |
| **Commands Used?** | git init, git add, git commit | git push, git pull, git clone (along with Git commands) |
| **Collaboration** | Limited to local repositories | Enables team collaboration via pull requests & issue tracking |

**How They Work Together?**

1. **Git** is installed on your computer to track changes.
2. **GitHub** is used to store and share those changes with a remote team.
3. Developers **push** their code to GitHub and **pull** updates when collaborating.

**Example Workflow:**

sh

git init # Initialize a Git repository

git add . # Stage all changes

git commit -m "Initial commit" # Save changes locally

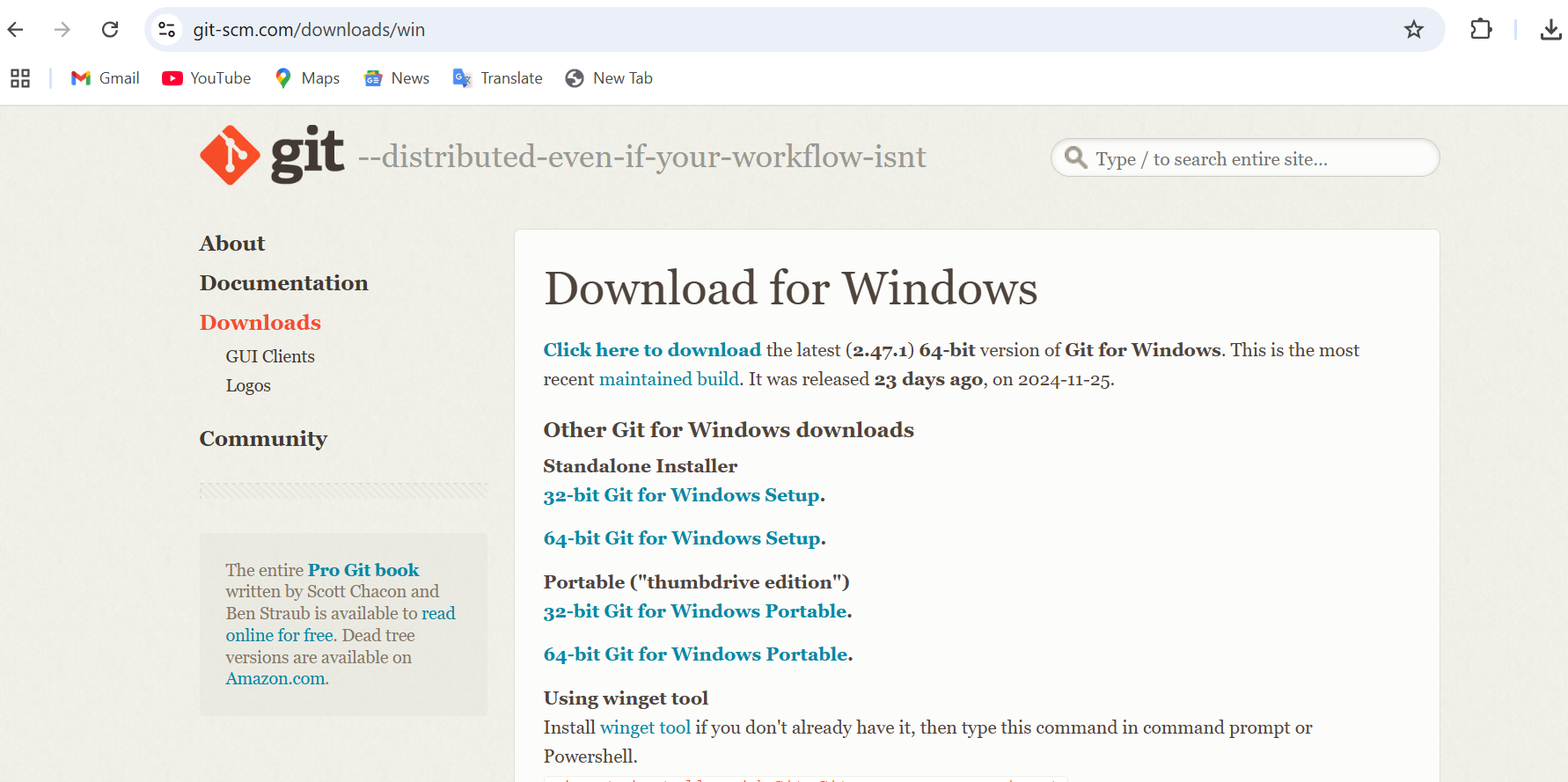
git remote add origin https://github.com/your-repo.git # Link to GitHub

git push -u origin main # Push code to GitHub

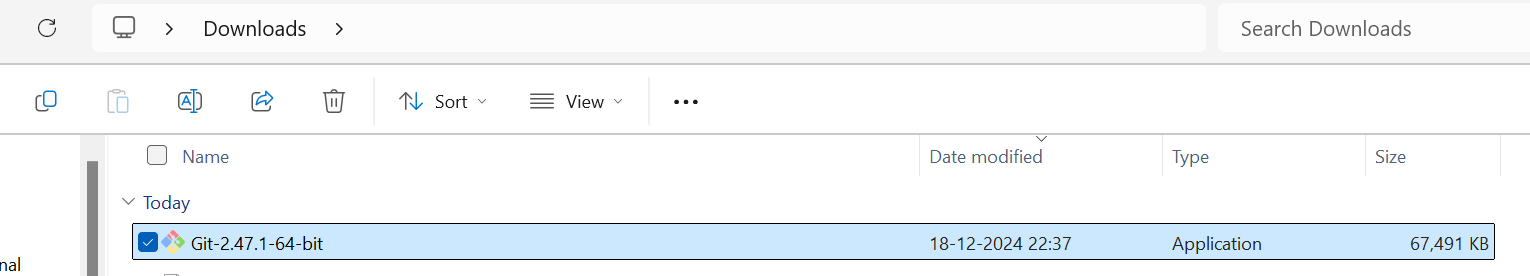
**Git download steps**

1. Download git from below link (download as per your OS)

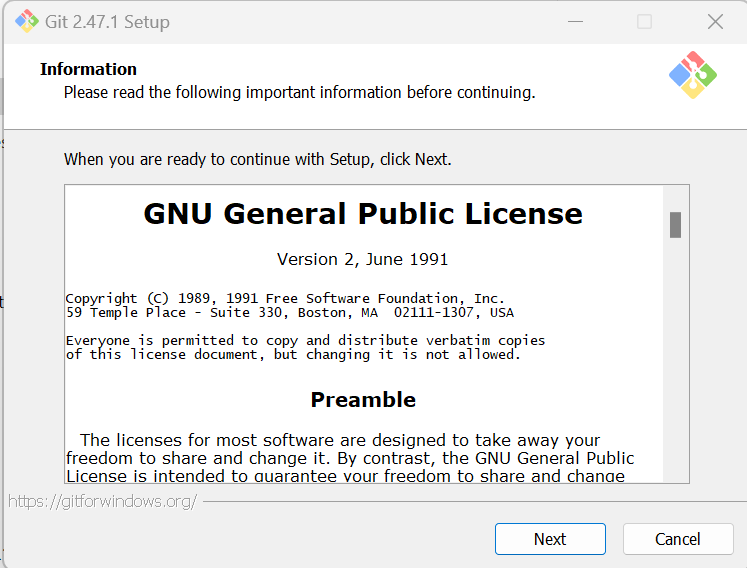
<https://git-scm.com/downloads/win>

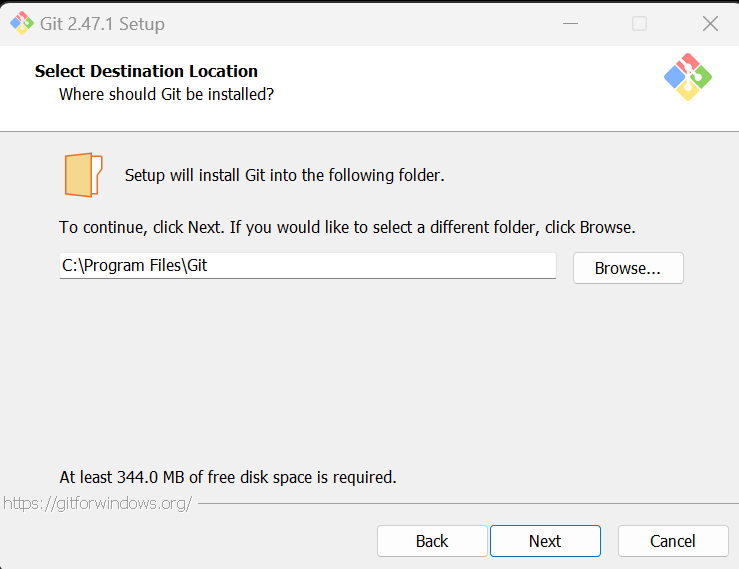


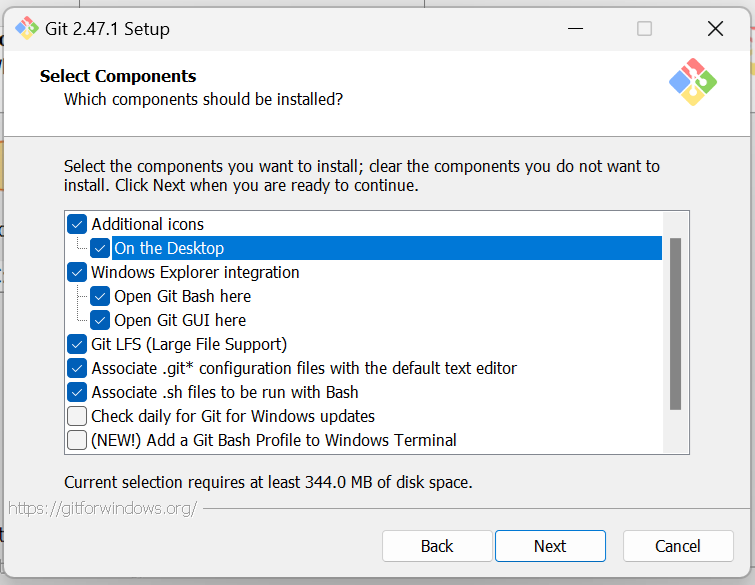
1. Click on git exe file

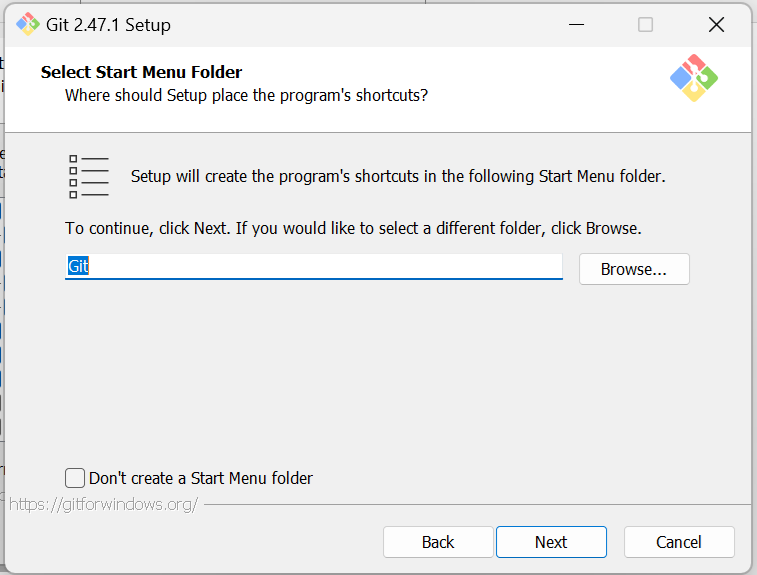


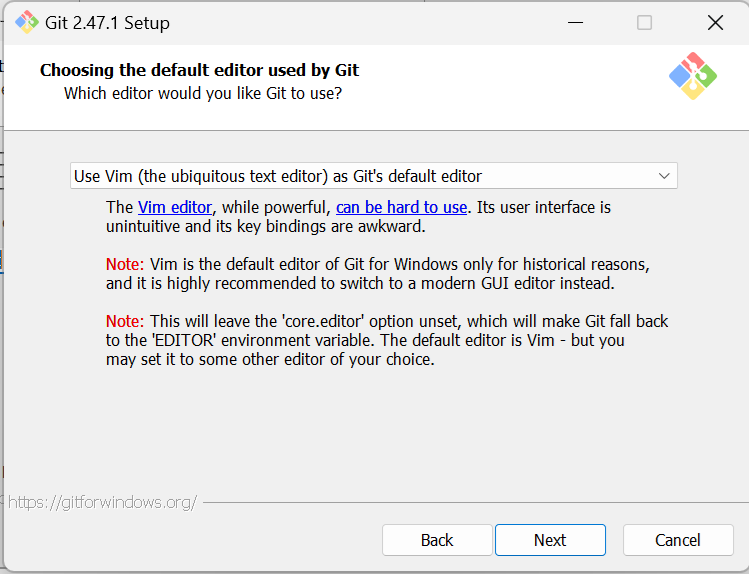
1. Follow the below instructions/screenshot for git installation.

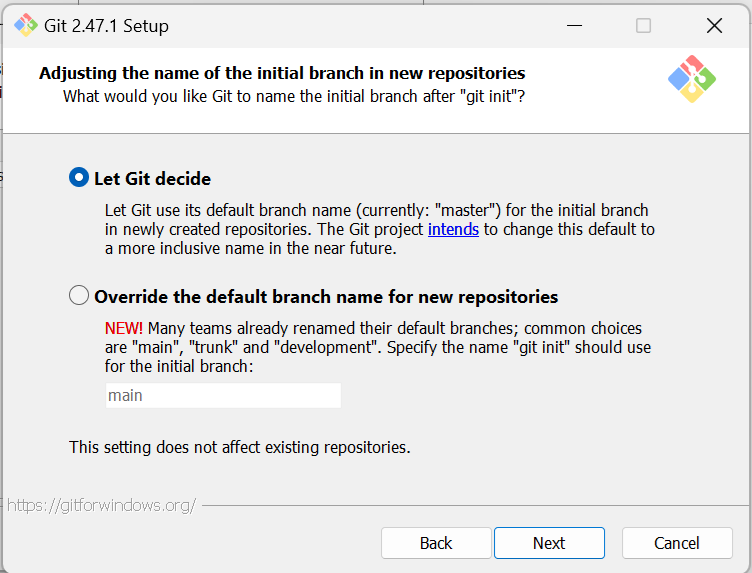


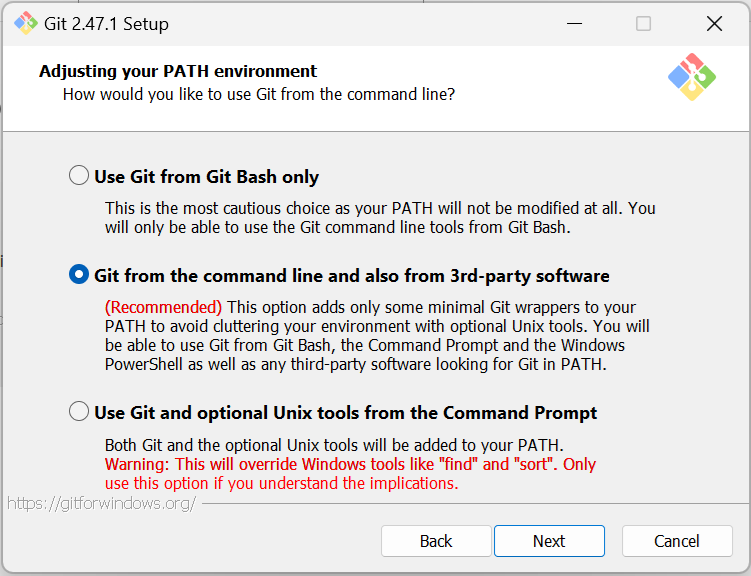


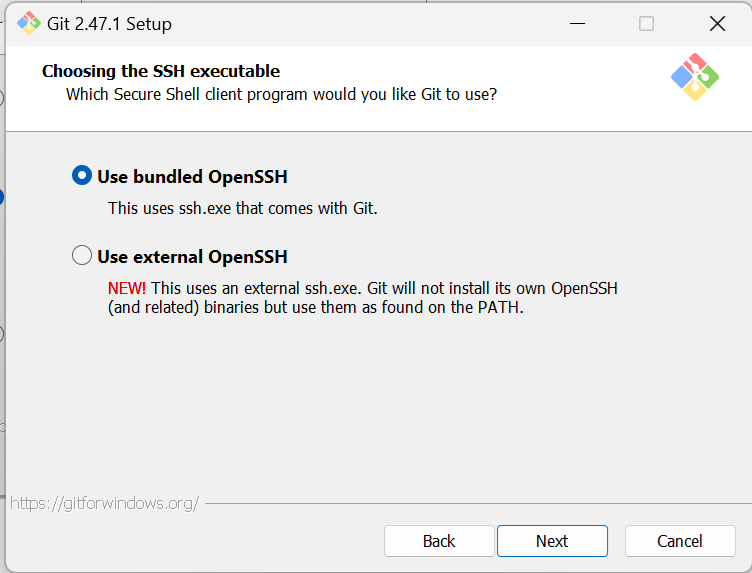


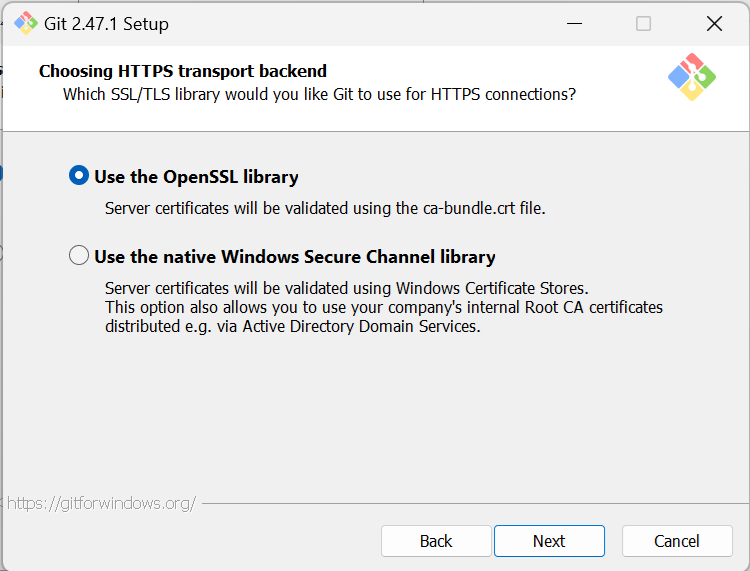


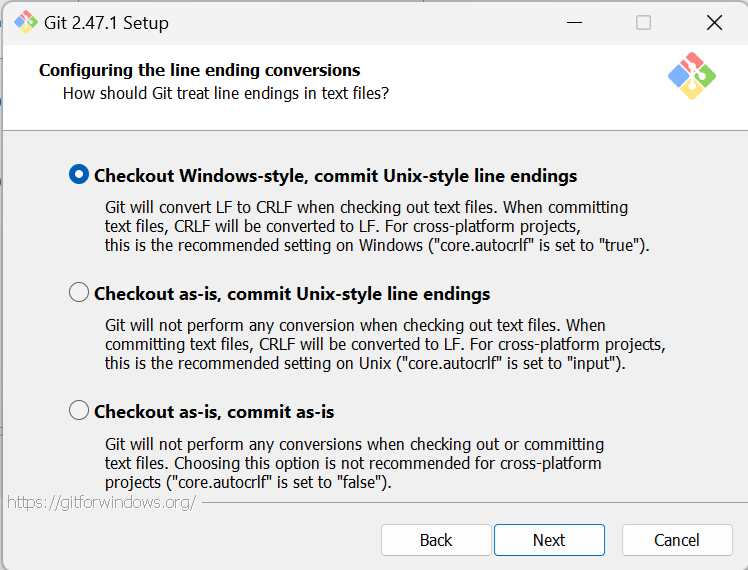


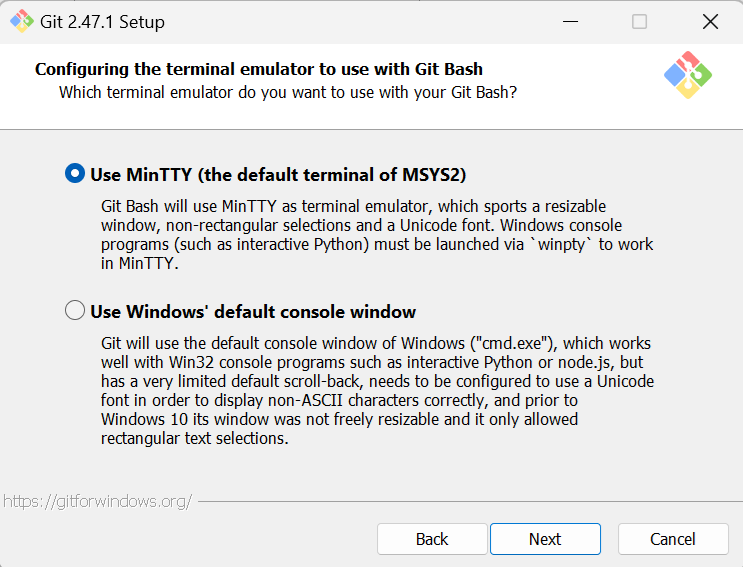


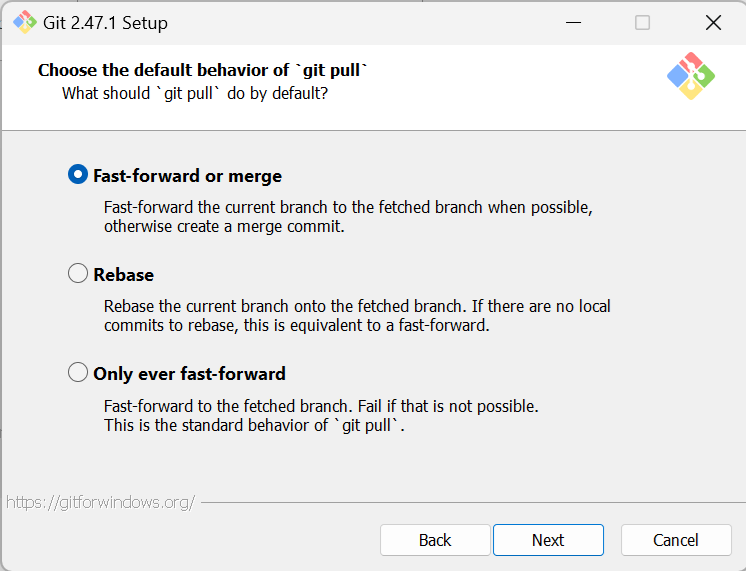


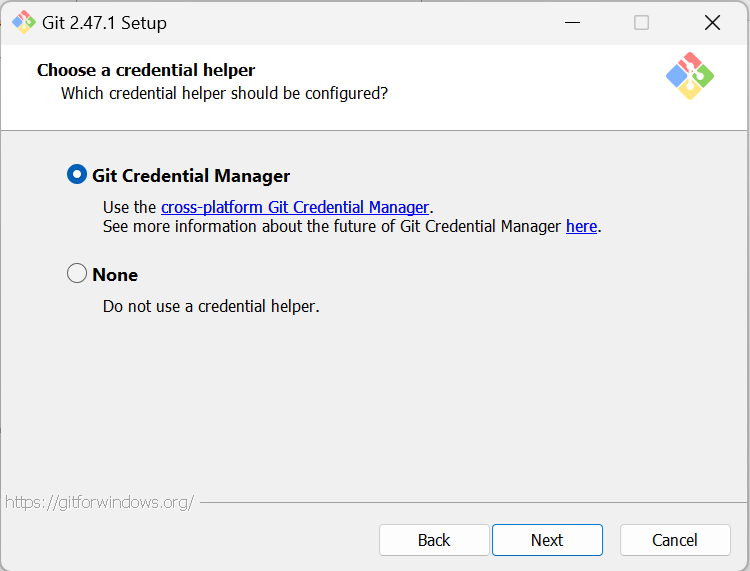


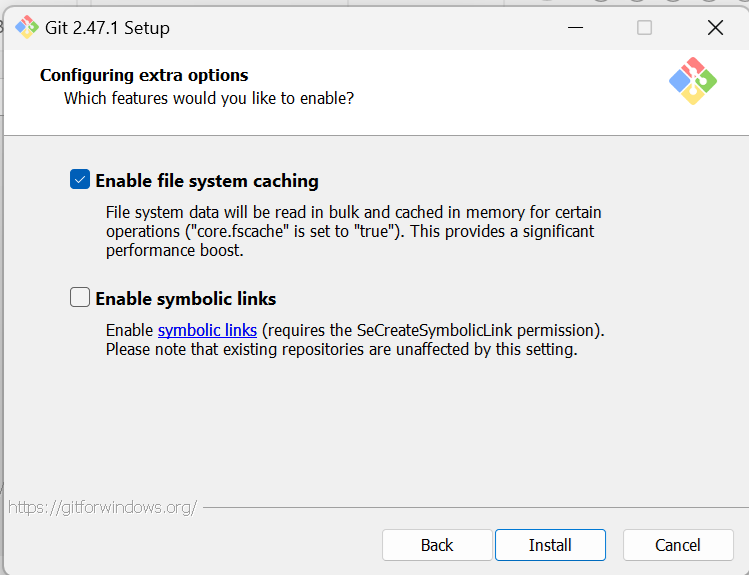


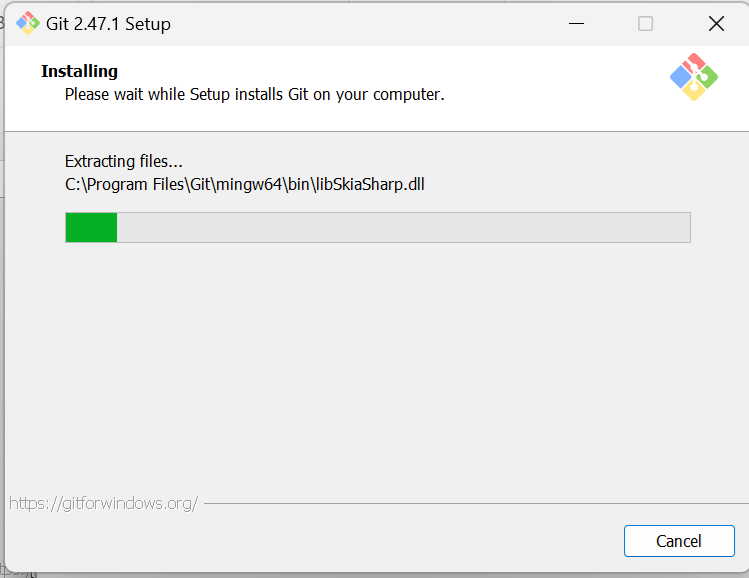


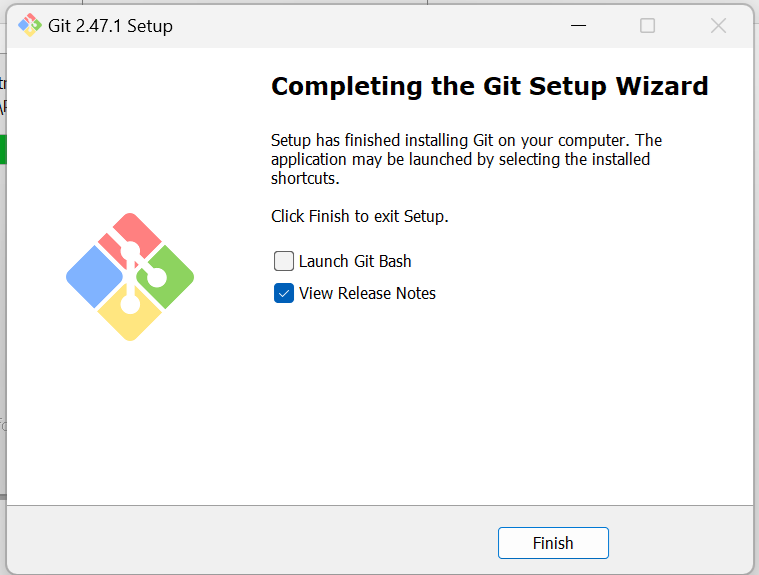












1. Launch git bash from your desktop icon



**Create an account on GitHub:**

**Step 1: Go to GitHub**

1. Open a web browser and go to [GitHub's official website](https://github.com/).

**Step 2: Sign Up**

1. Click on **"Sign up"** in the top right corner.
2. Enter your **email address**.
3. Choose a **username** (must be unique).
4. Create a **strong password**.
5. Click **"Continue"**.

**Step 3: Verify Your Account**

1. GitHub might ask you to solve a **CAPTCHA** to verify you're not a bot.
2. Click **"Create account"**.
3. Check your **email inbox** for a verification code from GitHub.
4. Enter the **verification code** and click **"Verify"**.

**Step 4: Choose Preferences**

1. GitHub may ask about your experience level (you can skip or select options).
2. Choose whether you want a **free plan** or a **paid plan** (select **free** if you're just getting started).
3. Click **"Continue"**.

**Step 5: Explore GitHub**

1. Once your account is created, you can set up your **profile**, add a profile picture, and explore repositories.
2. To start using GitHub, click on **"New Repository"** to create your first project.

**Step 6: Set Up Git Locally (details given above)**

1. Download and install **Git** from [git-scm.com](https://git-scm.com/).
2. Configure your Git username and email:

sh

CopyEdit

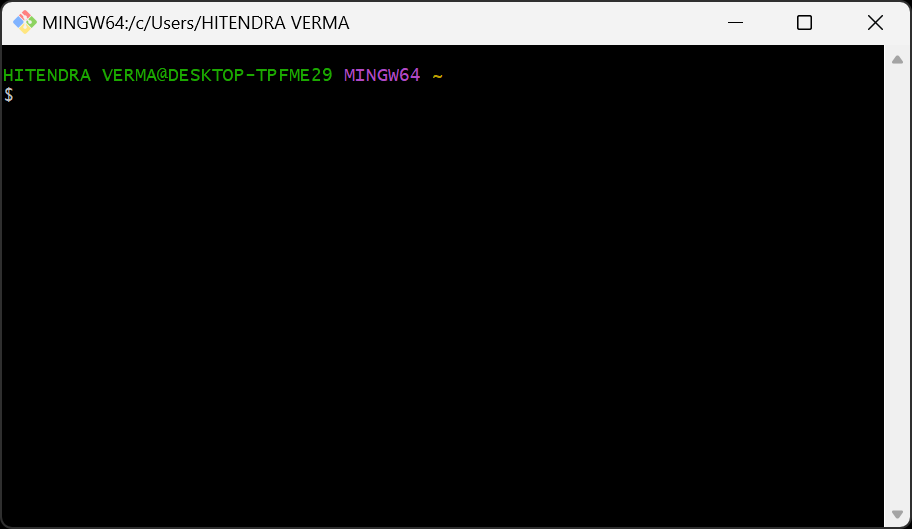
git config --global user.name "YourUsername"

git config --global user.email "youremail@example.com"

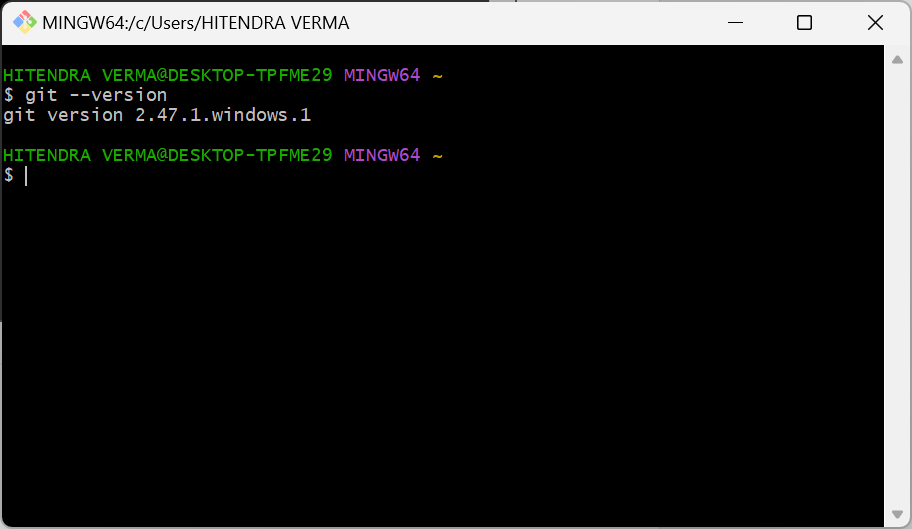
1. You can now clone repositories, commit changes, and push them to GitHub.

**Push the project to github**

**Launch gitbash from your desktop**



You can verify the git version using below command



Now navigate to your project directory

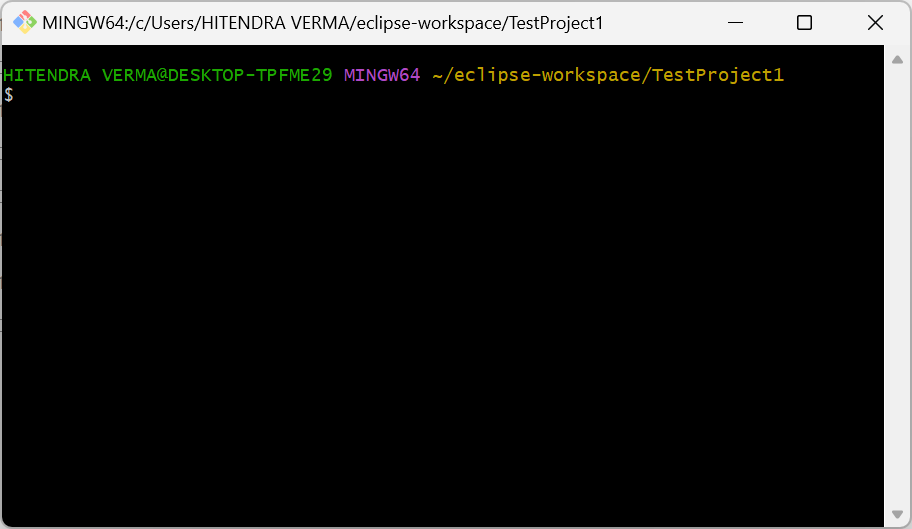
If your project is in C:\Users\YourName\Projects\MyProject, use:

cd /c/Users/YourName/Projects/MyProject

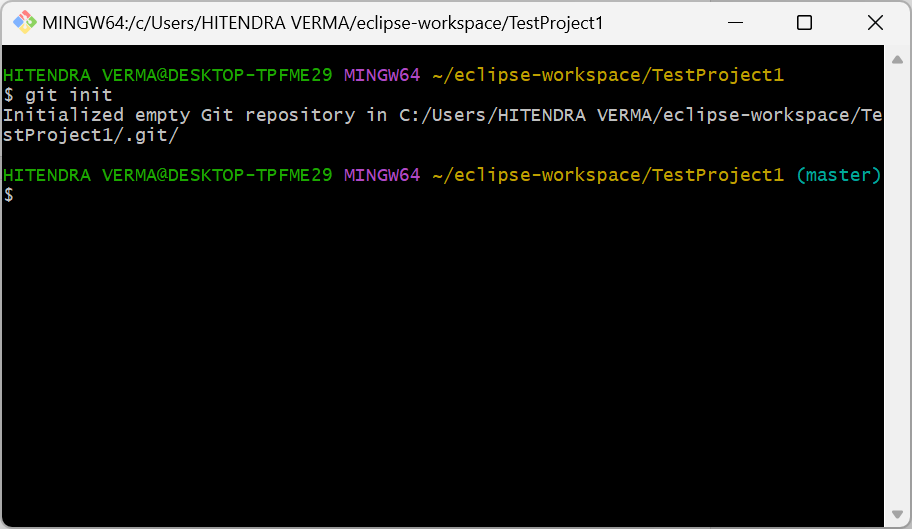
Or

**Shortcut: Open Git Bash Directly in Your Project Folder**

* In **Windows Explorer**, navigate to your project folder.
* **Right-click inside the folder** → Select **"Git Bash Here"**.



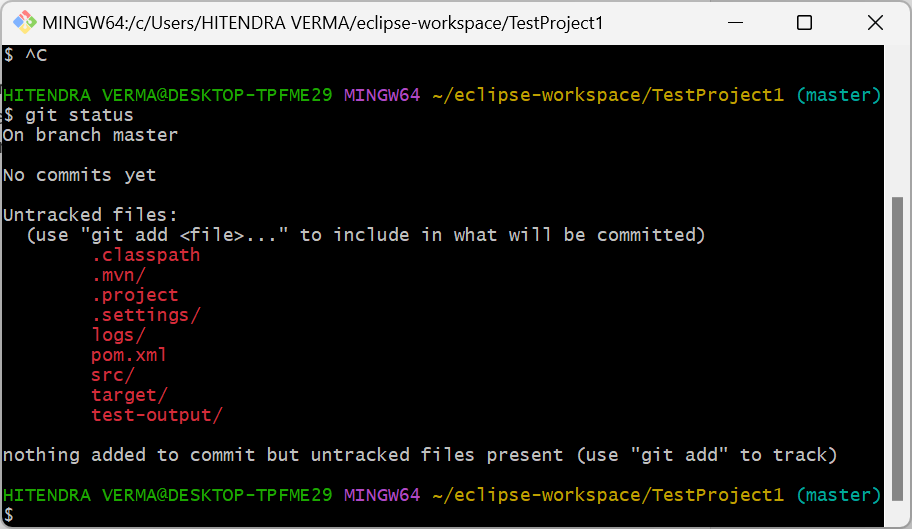
Initialize your project using **git init** command



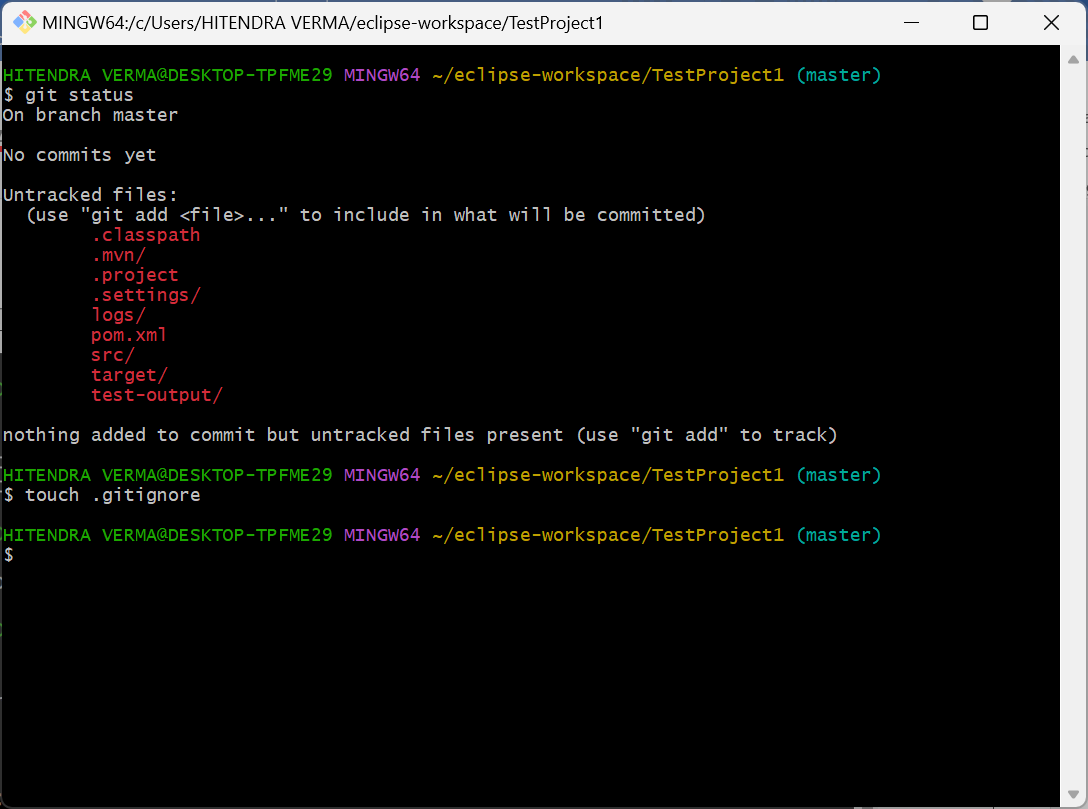
**What Does This Mean?**

* Git has **initialized a new empty repository** in your project folder.
* A hidden .git folder has been created inside TestProject1/.
* This .git folder stores all version control data (commits, branches, logs, etc.).

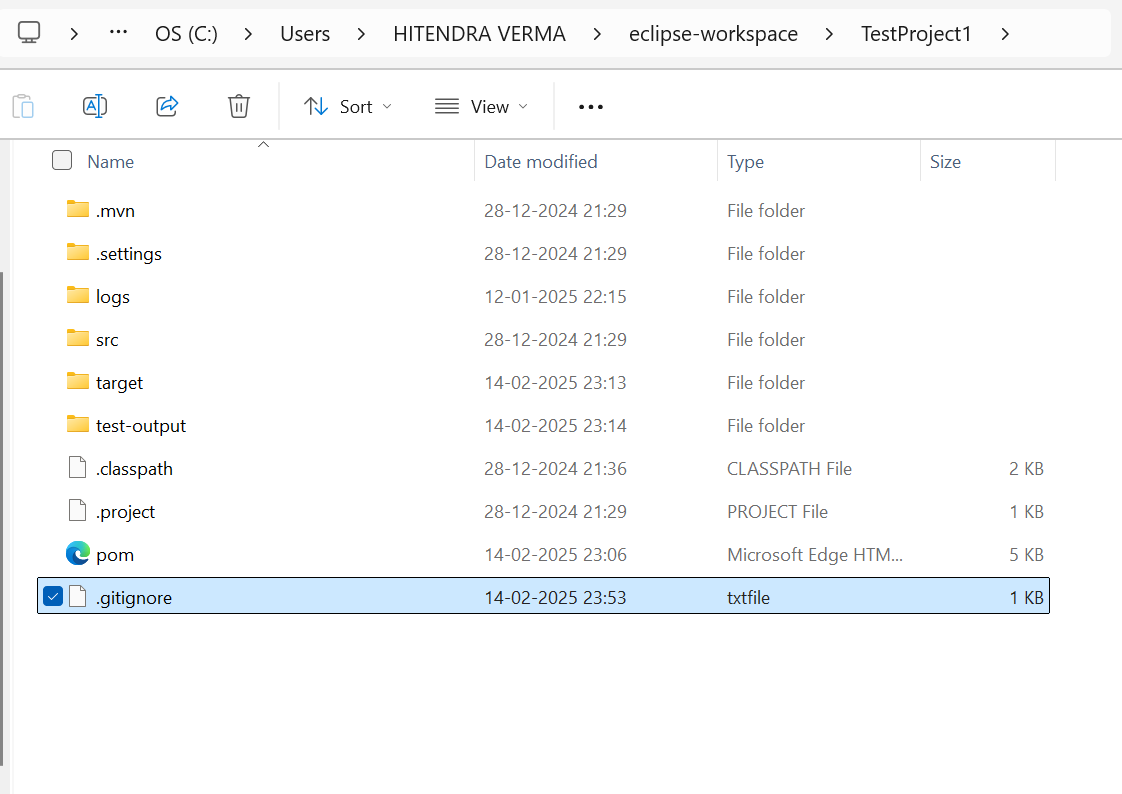
Now do git status



touch .gitignore



.gitignore file will be created in project directory



Right click and open the file in notepad/wordpad

Here’s a .gitignore file tailored for your Selenium Java framework. It ensures unnecessary files (such as logs, compiled binaries, IDE settings, and reports) are ignored by Git.

# Ignore build and dependency files

/target/

\*.class

\*.jar

\*.war

\*.ear

# Ignore IDE-specific files

.idea/

\*.iml

.vscode/

\*.suo

\*.ntvs\*

\*.njsproj

\*.sln

\*.sw?

# Ignore Maven-specific files

.mvn/

mvnw

mvnw.cmd

# Ignore log and output files

\*.log

/logs/

\*.out

\*.err

# Ignore test reports and screenshots

/test-output/

screenshots/

allure-results/

reports/

ExtentReport/

\*.html

# Ignore generated Selenium WebDriver files

chromedriver.log

geckodriver.log

selenium-server.log

# Ignore compiled and temporary files

/bin/

\*.swp

\*.DS\_Store

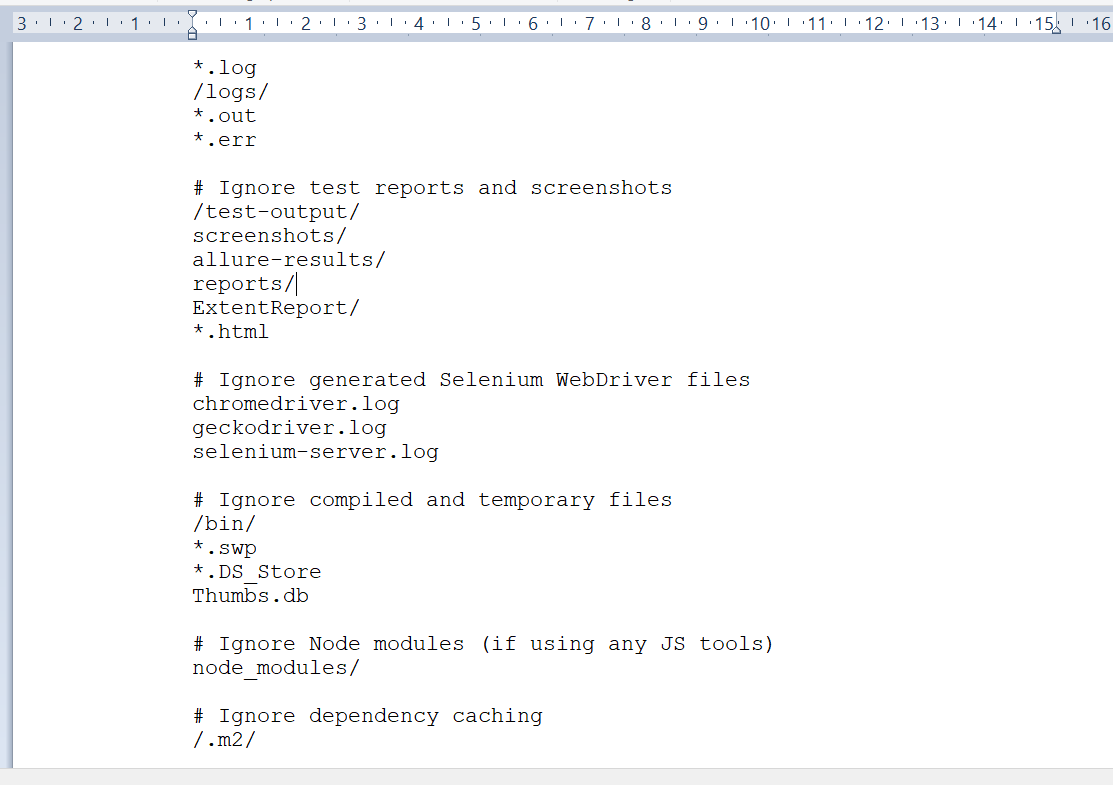
Thumbs.db

# Ignore Node modules (if using any JS tools)

node\_modules/

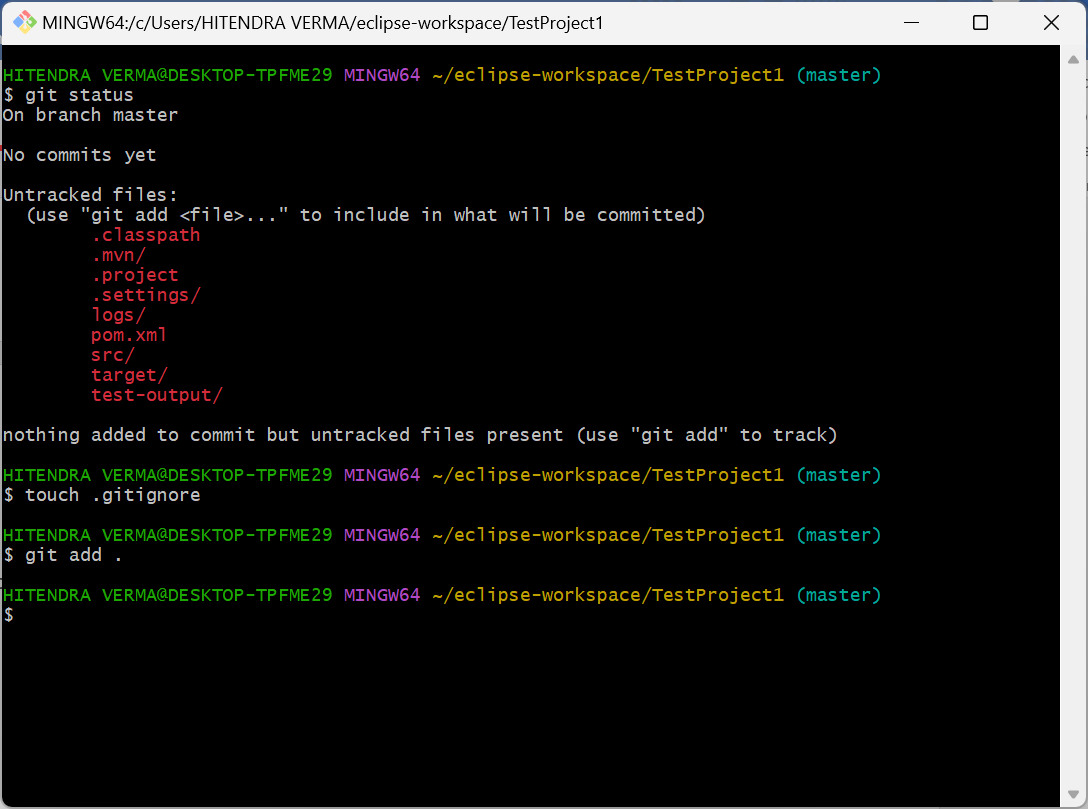
# Ignore dependency caching

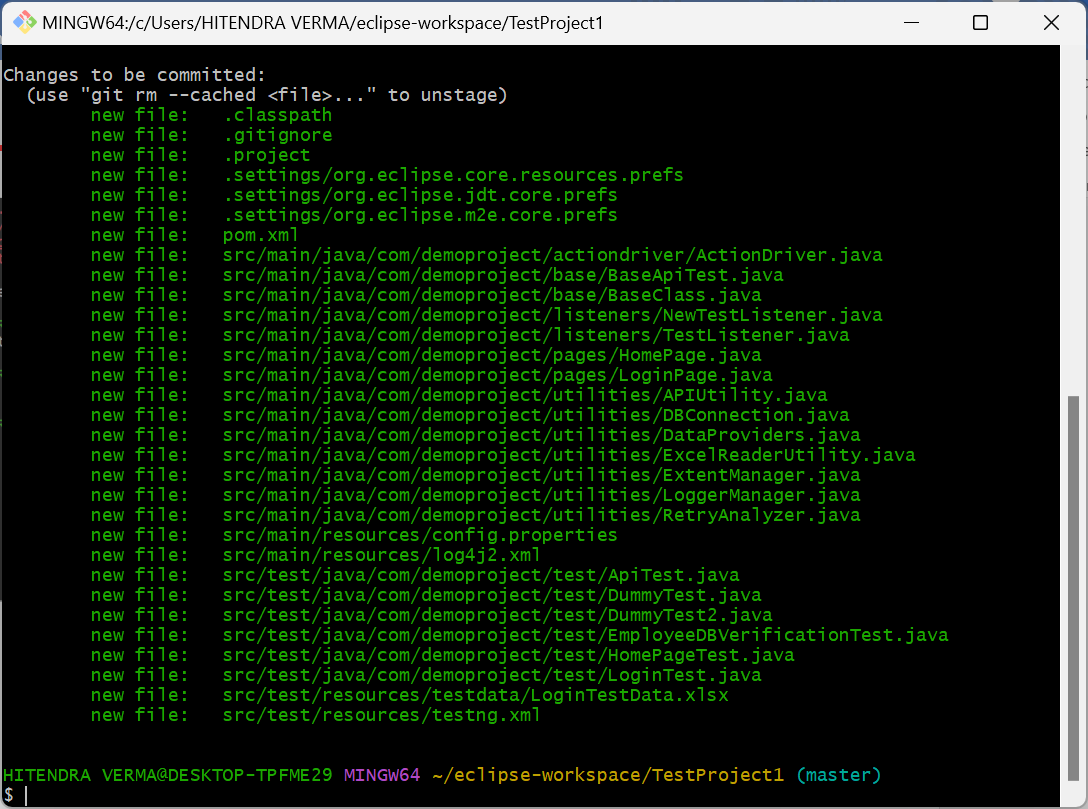
/.m2/



Save the file

Now run command git add .

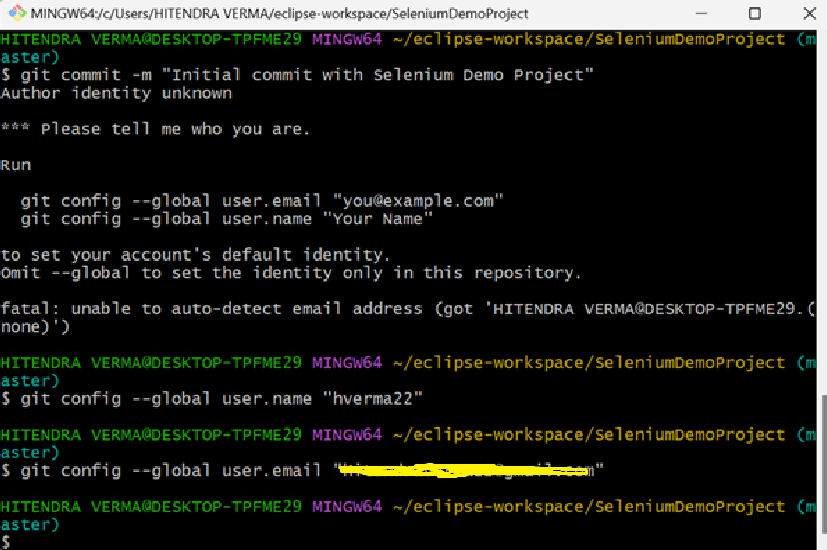


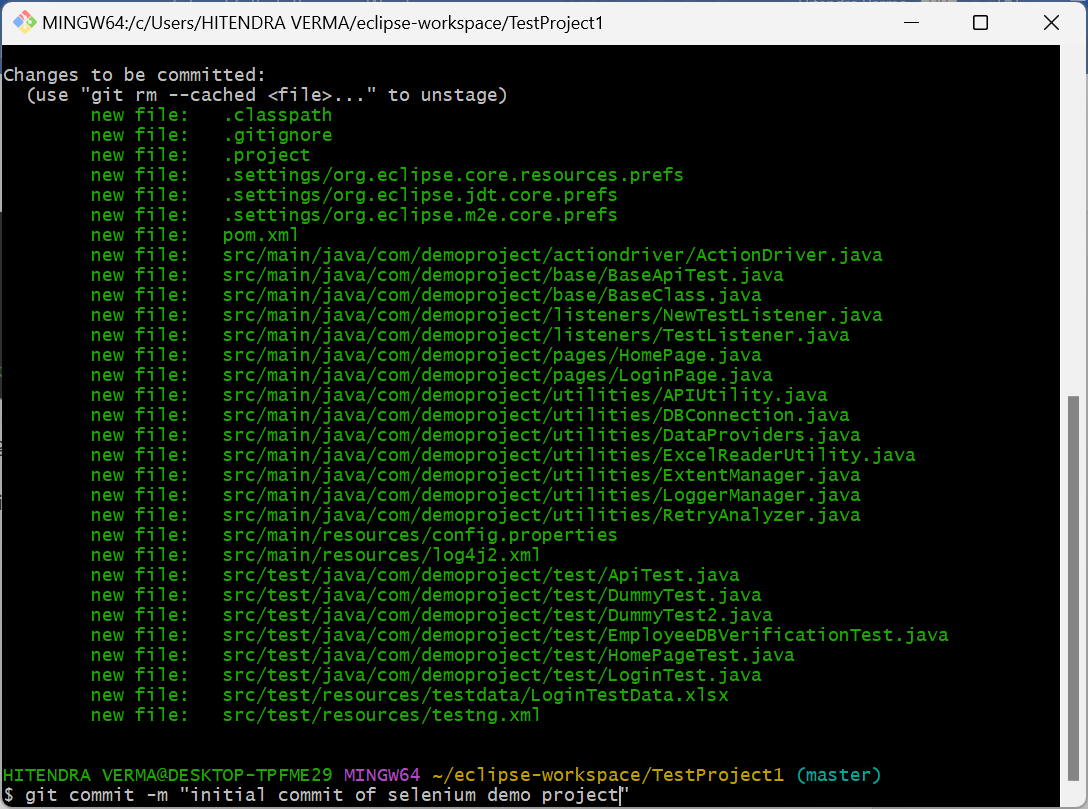


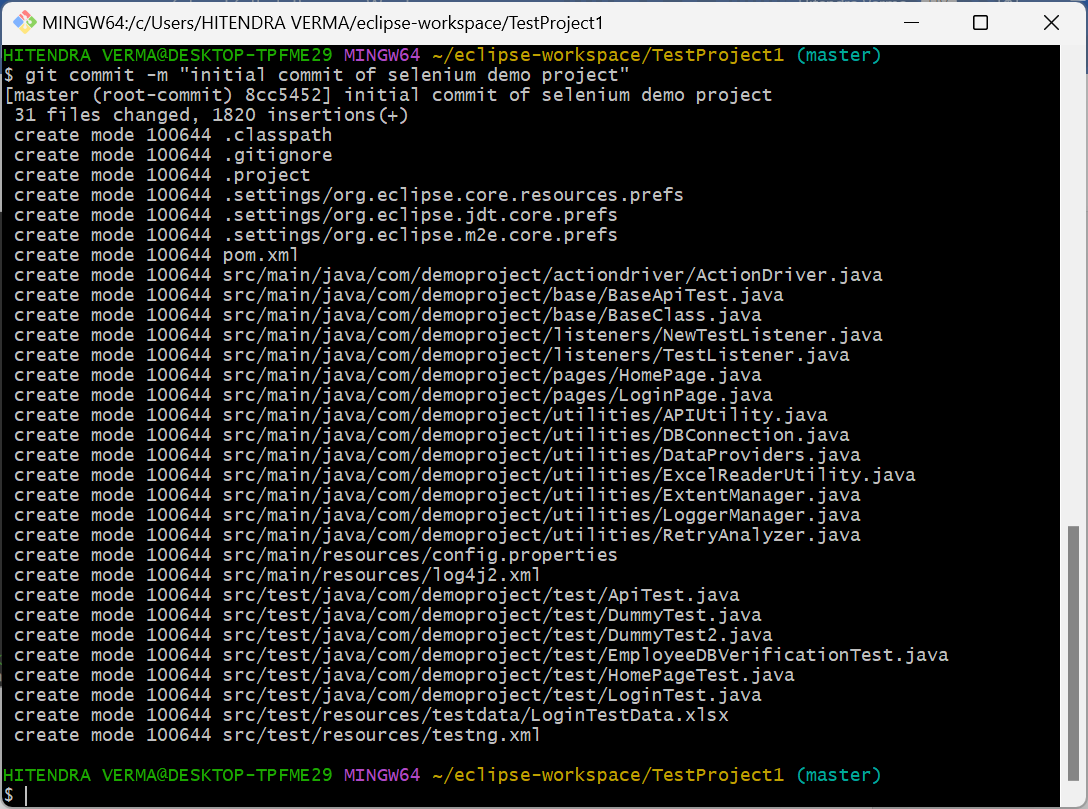
Commit the changes

git commit -m “Initial commit with Selenium Demo Project”

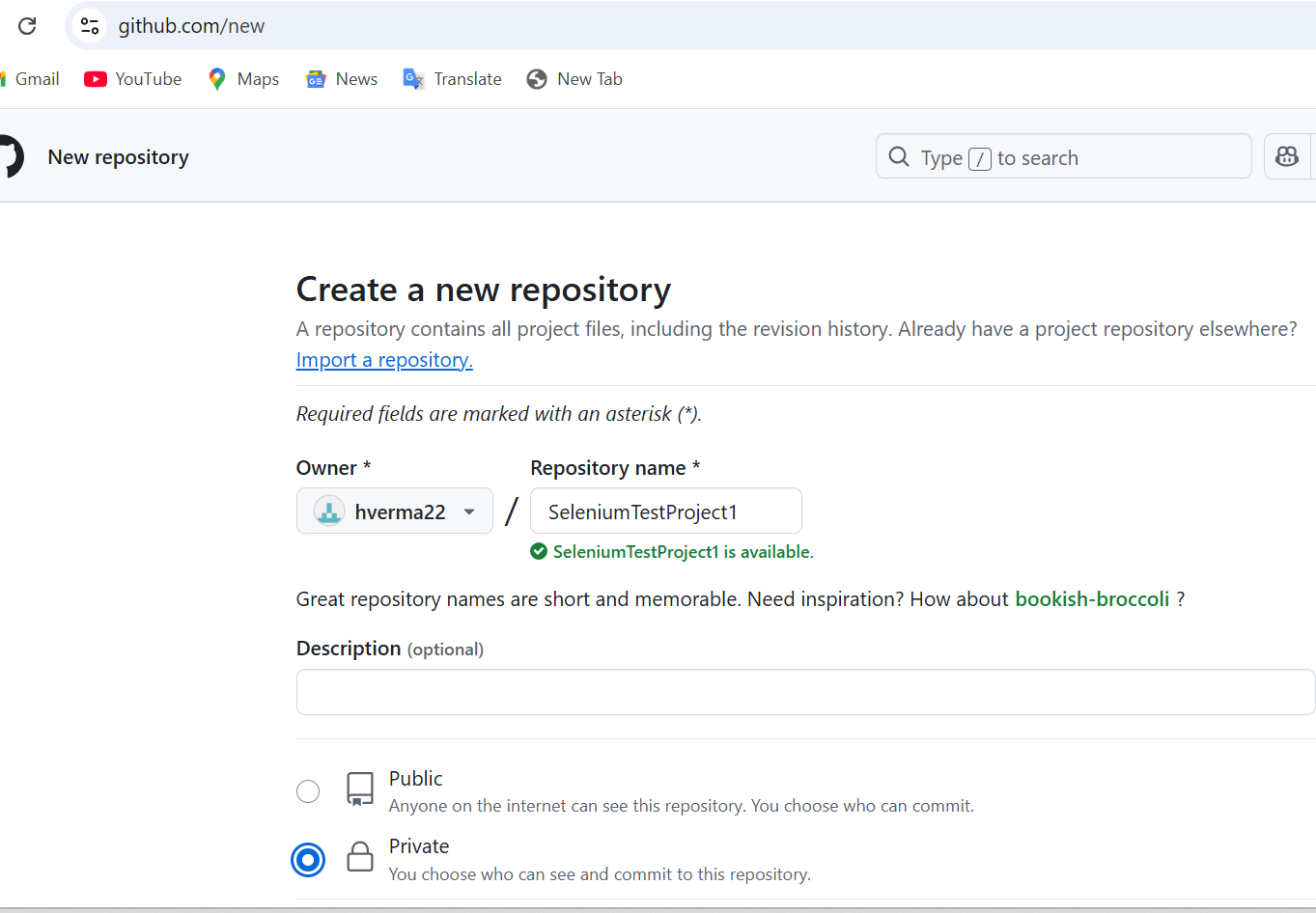
Do initial commit you may need to provide your github user and email id associated with github (create your github account if not done so)

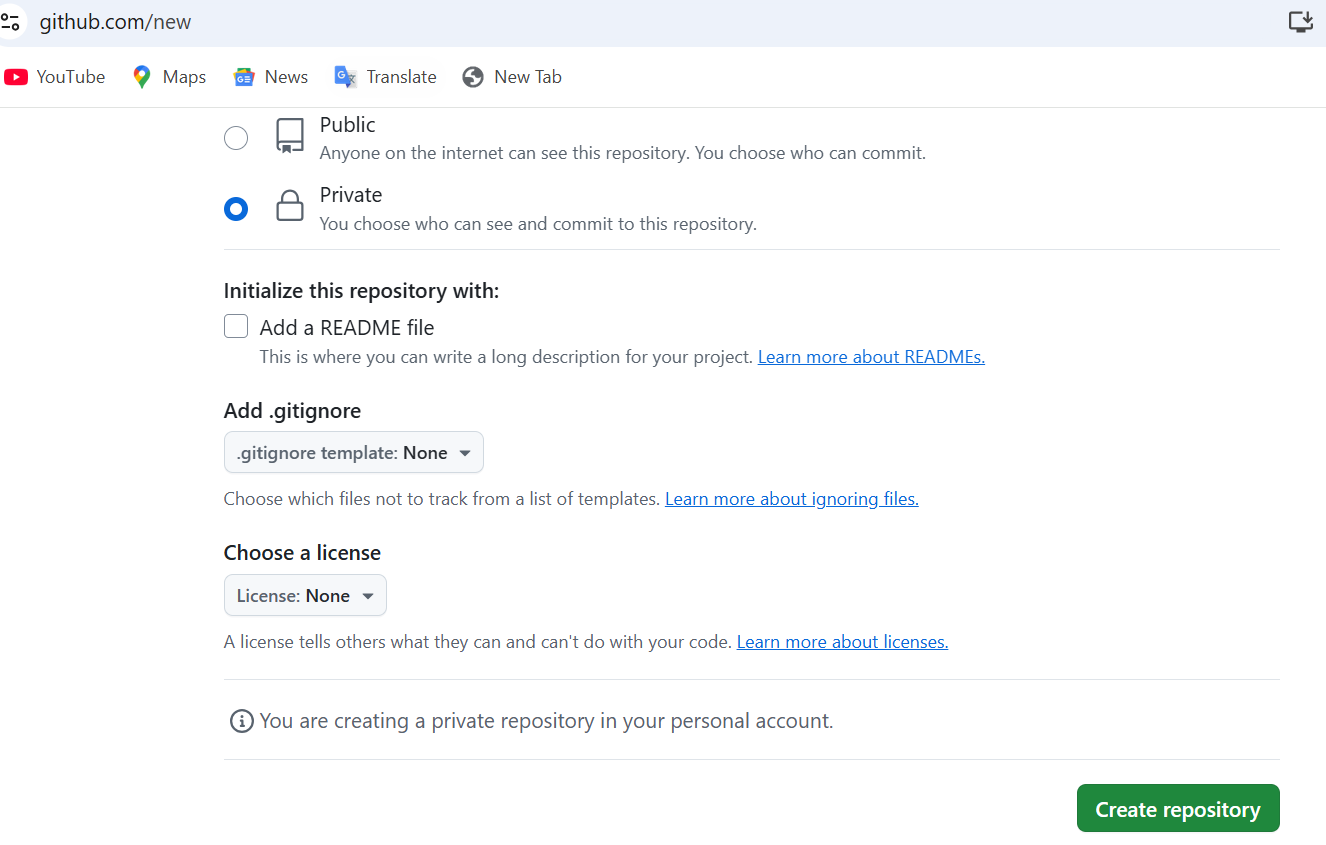


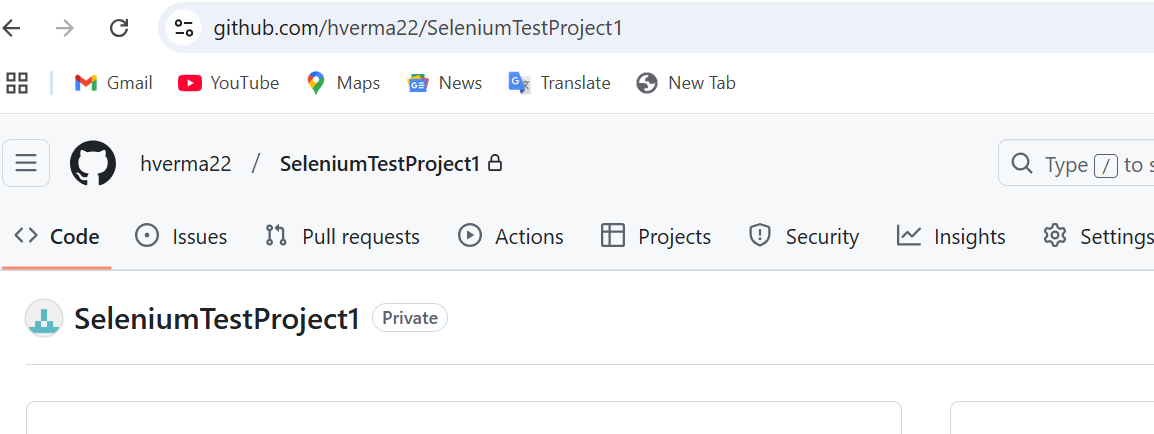




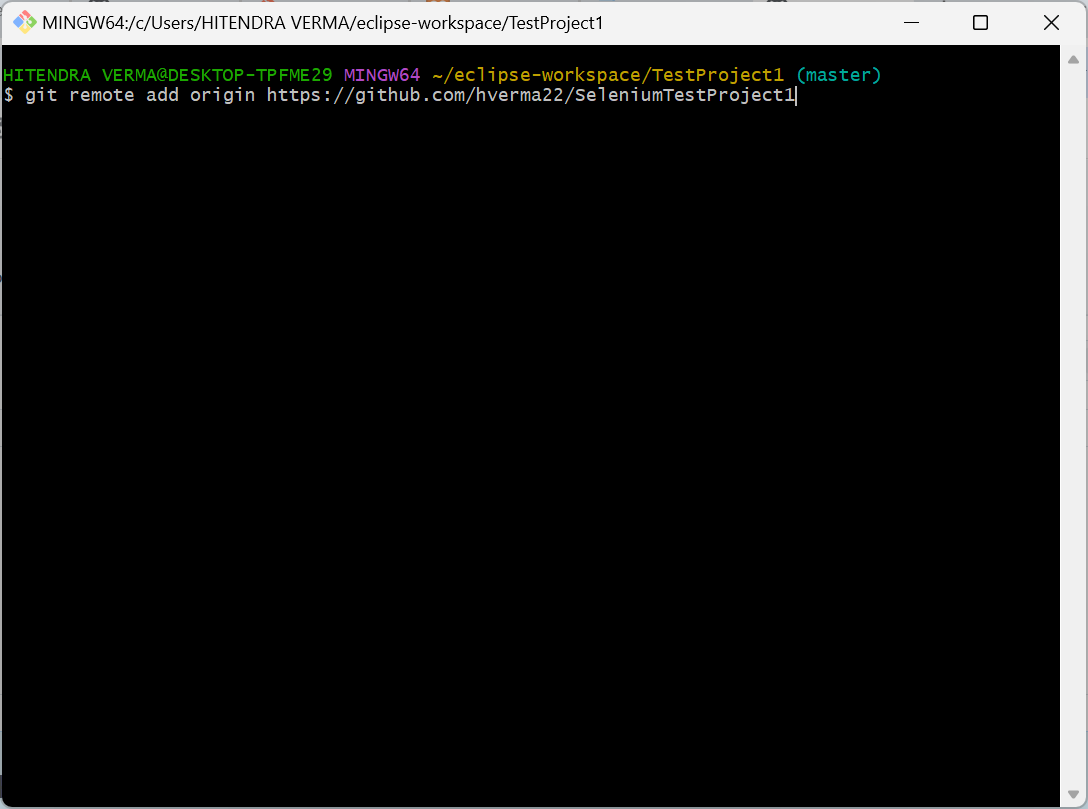
Before pushing the changes make sure you have created your git repository on github

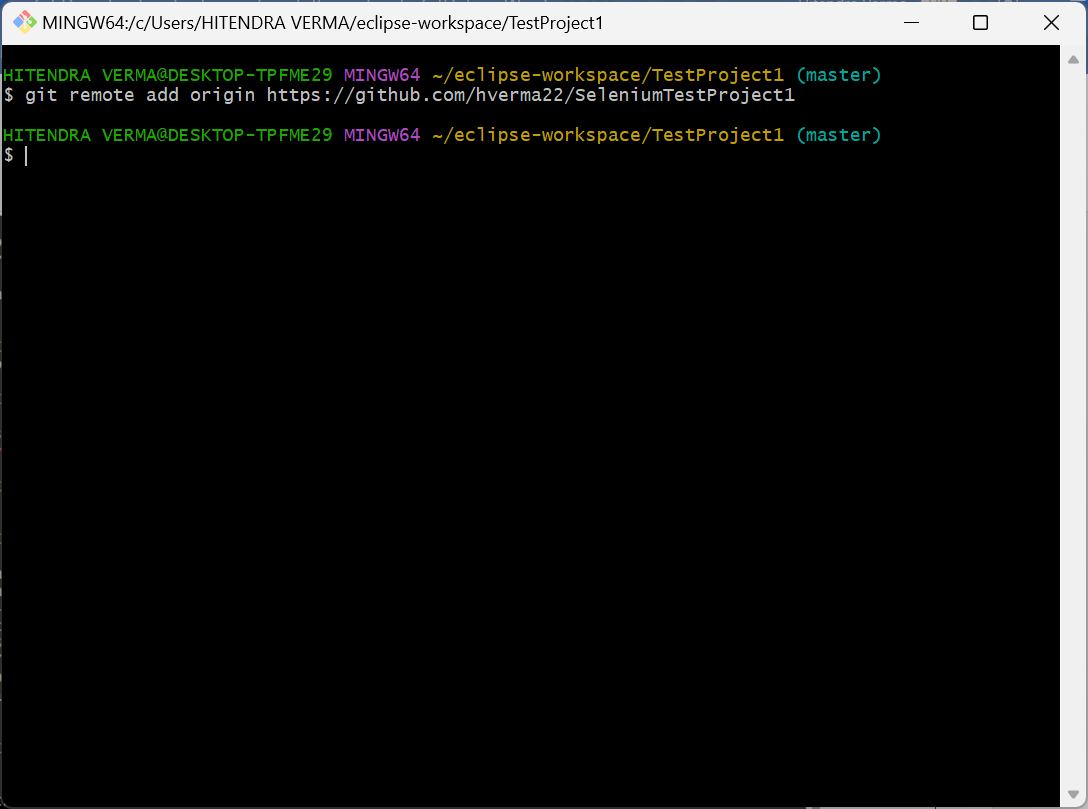




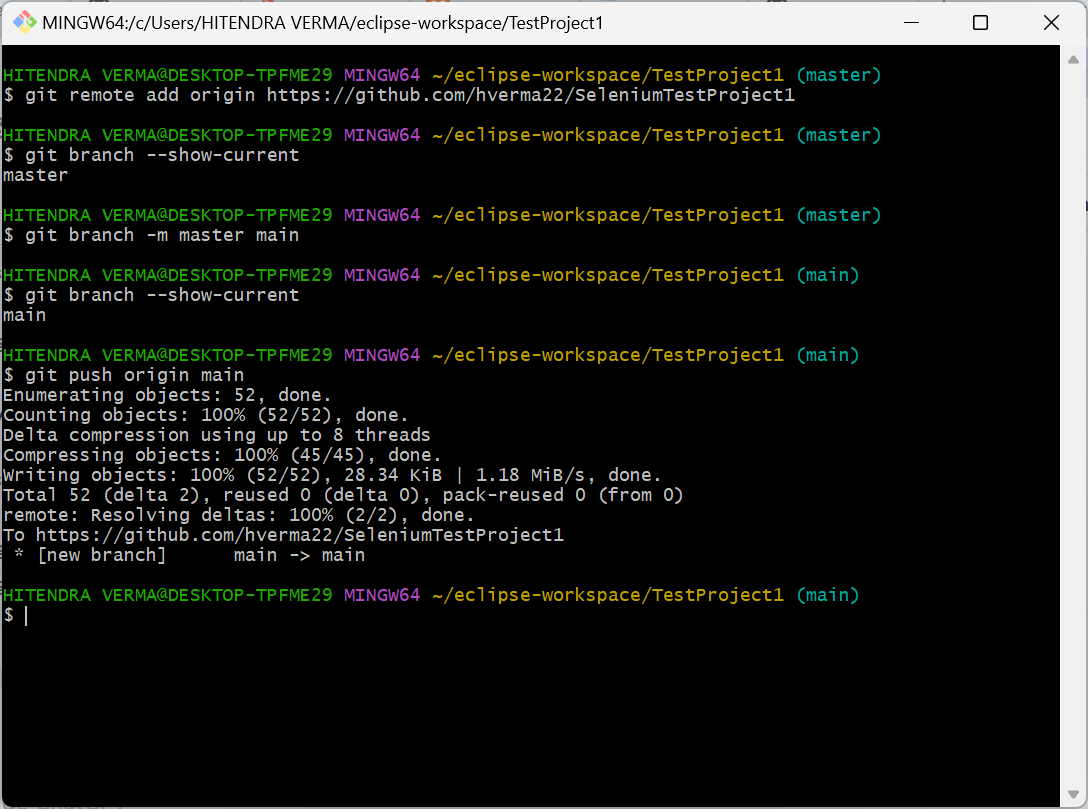


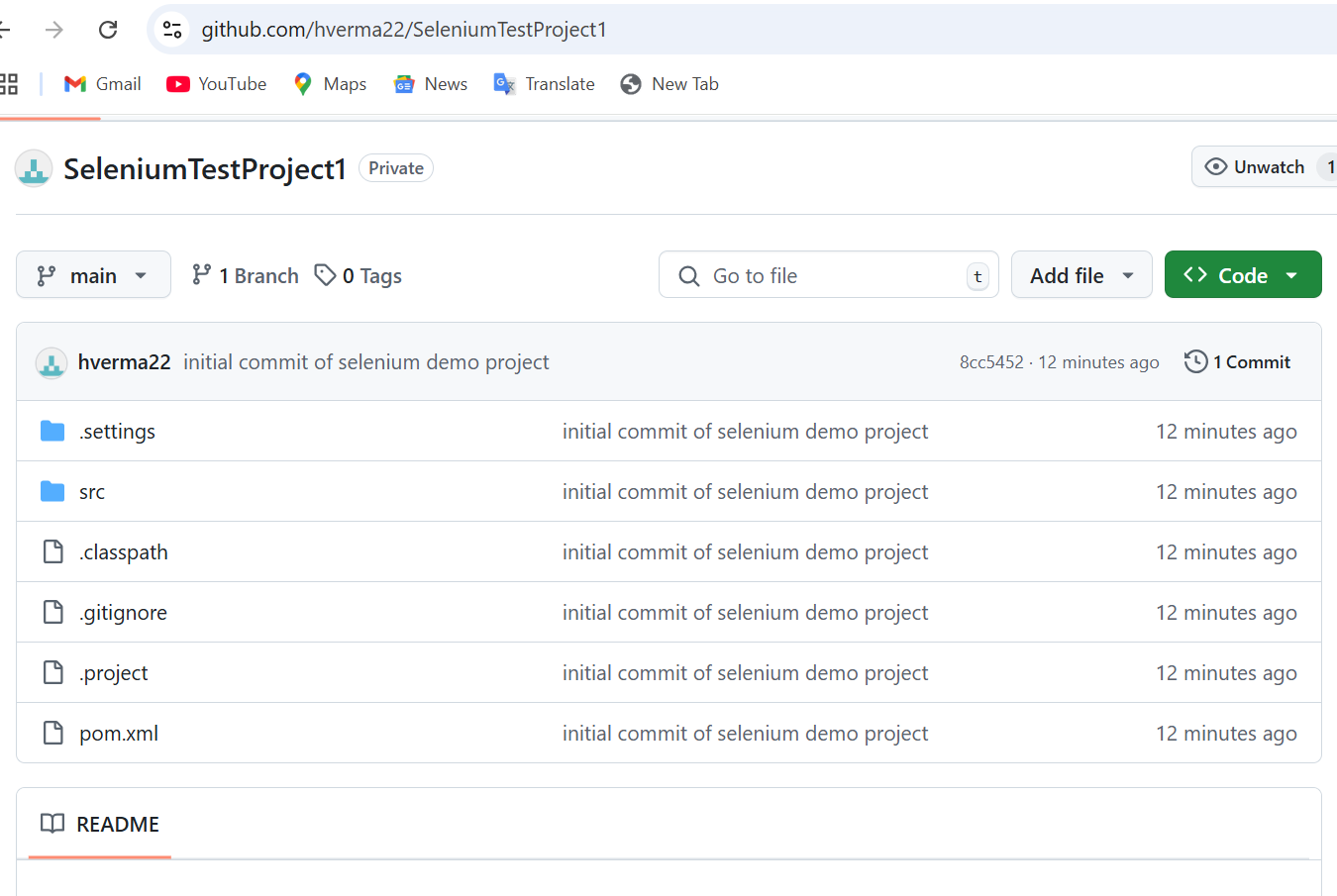
Link to github repository – **git remote add origin <github project link>**





Push to main branch





In eclipse if cylindrical symbol is not displayed then:

