**Explain git ignore**

gitignore is a **special file** in a Git repository that tells Git **which files or folders it should ignore**.

Ignored files:

* Won’t be tracked by Git.
* Won’t appear in git status.
* Won’t be pushed to remote repositories (GitHub, GitLab, etc.).

Commonly ignored files include:

* Temporary files (\*.tmp)
* Logs (\*.log)
* Build output folders (/build, /dist)
* IDE-specific files (.vscode/, .idea/)
* OS-generated files (Thumbs.db, .DS\_Store)

**Explain how to ignore unwanted files using git ignore**

**Step 1 – Create/Edit .gitignore file**

In your project folder:

notepad .gitignore

**Step 2 – Add Patterns for Files/Folders to Ignore**

Example .gitignore content:

# Ignore all .log files

\*.log

# Ignore log folder

log/

# Ignore build folder

build/

# Ignore temporary files

\*.tmp

**Step 3 – Save and Verify**

* Run:

git status

You should not see those ignored files listed as untracked.

**Implement git ignore command to ignore unwanted files and folders**

**Step-by-Step Instructions**

**Step 1 – Initialize Git Repository**

Navigate to your project folder and initialize a Git repository (if not already done):

cd path/to/your/project

git init

**Step 2 – Create Files and Folders to be Ignored**

Create a .log file and a log folder to simulate unwanted files:

echo "This is a log file" > debug.log

mkdir log

echo "Error log content" > log/error.txt

**Step 3 – Create or Edit .gitignore File**

Open or create the .gitignore file and add rules to ignore .log files and the log folder:

notepad++ .gitignore

# Inside .gitignore:

\*.log

log/

**Step 4 – Verify Ignored Files**

Check the Git status to ensure the .log file and log folder are ignored:

git status

**Step 5 – Add and Commit .gitignore**

Add .gitignore to the repository and commit changes:

git add .gitignore

git commit -m "Add .gitignore to ignore .log files and log folder"

**Step 6 – Push to Remote Repository**

Push the changes to your remote repository (GitHub or GitLab):

git remote add origin https://github.com/your-username/your-repo.git

git branch -M main

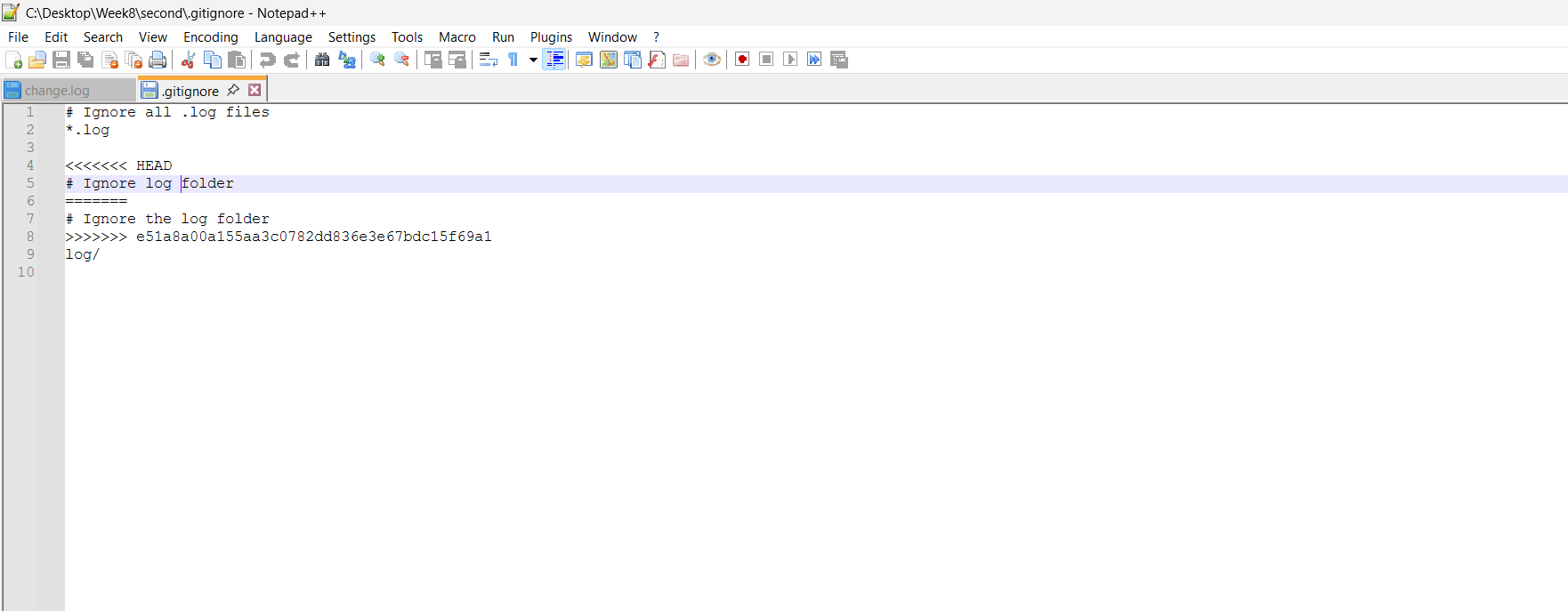
git push -u origin main

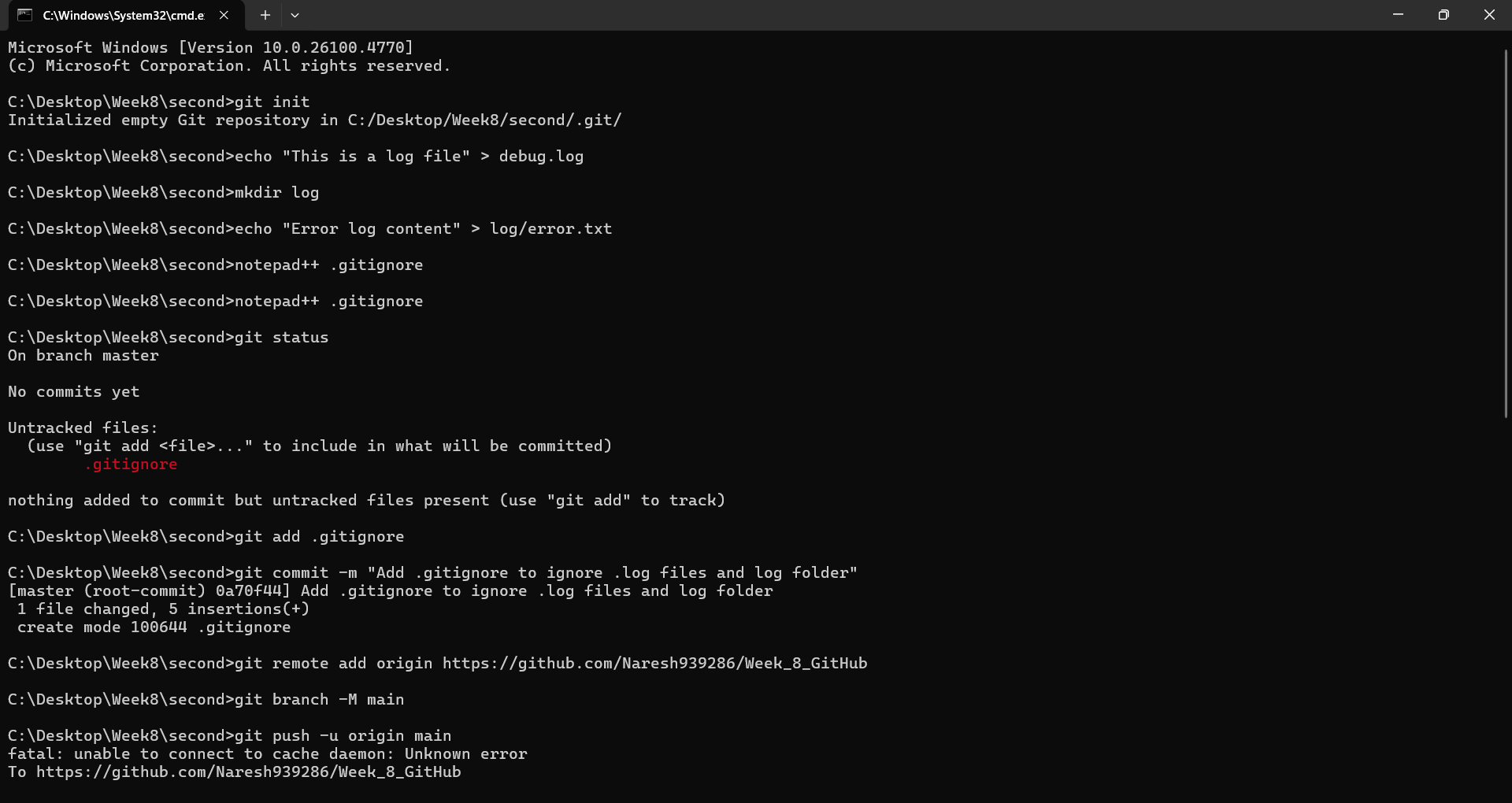
**Step 7 – Final Verification**

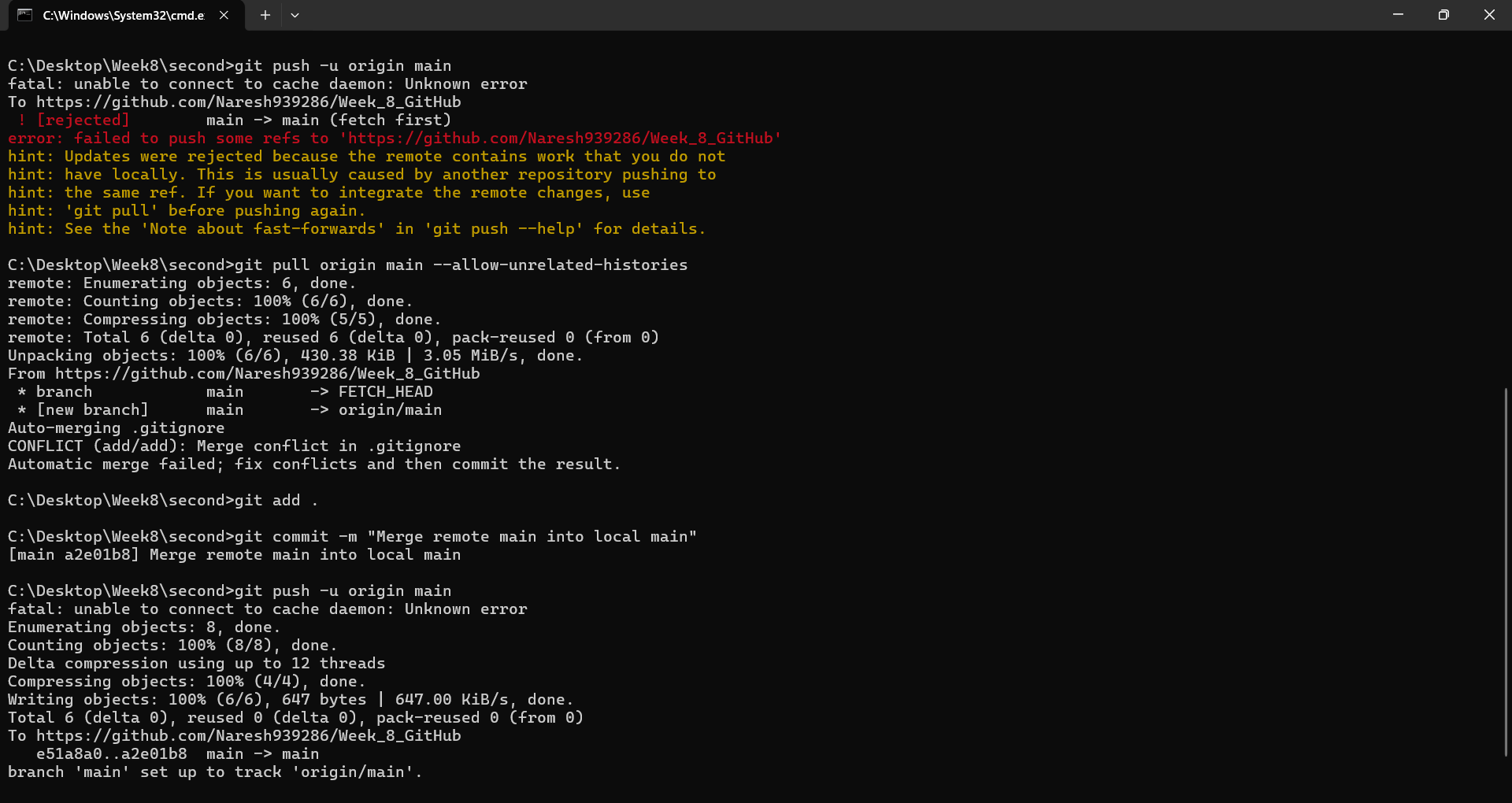
Run the status command again to confirm no unwanted files are being tracked:

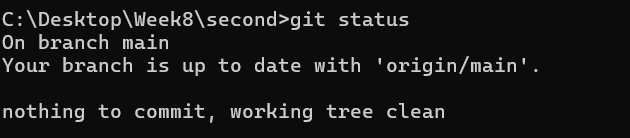
git status

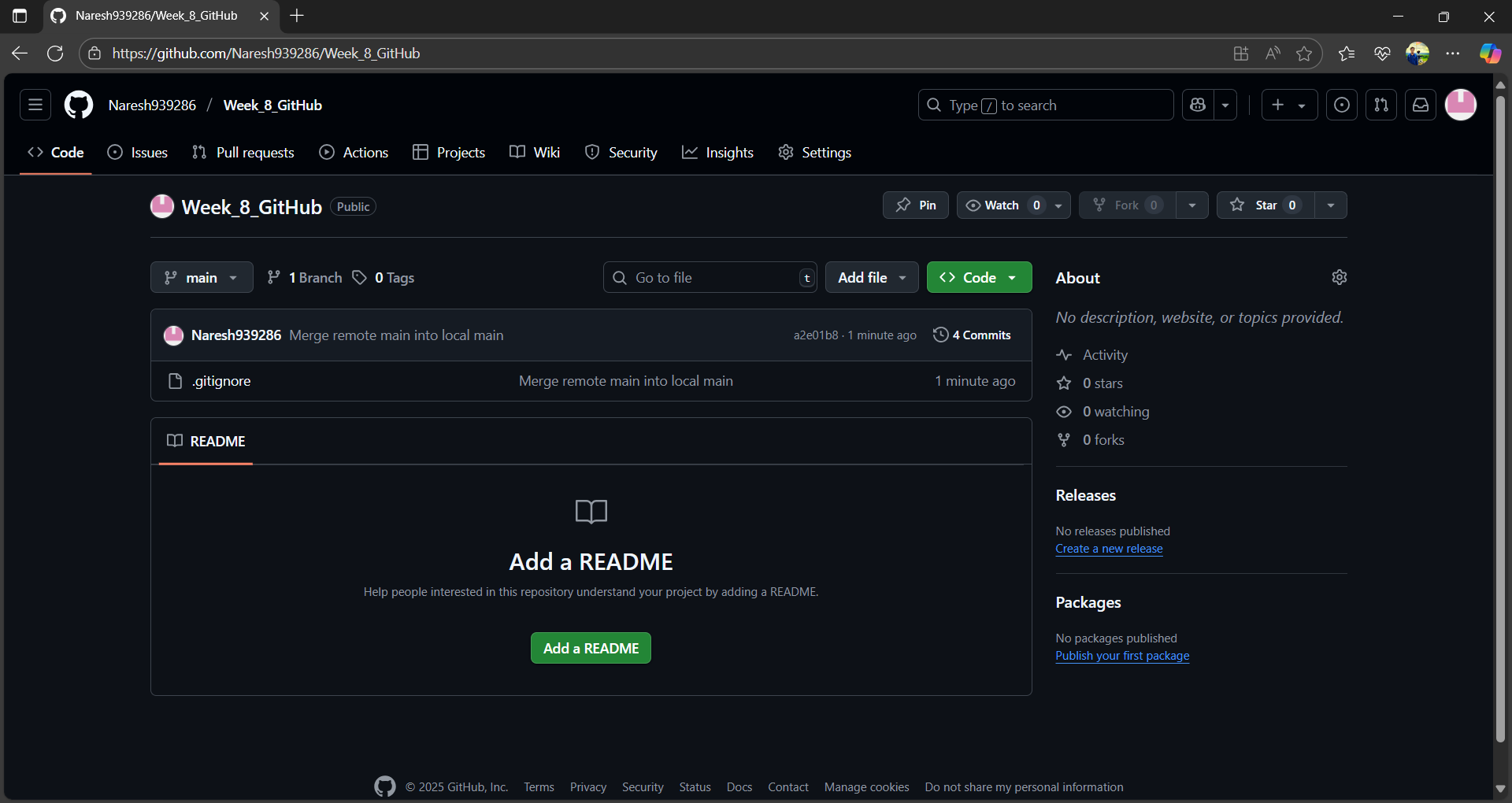
**Output :-**

****

****

****

****

****