**LAB 2**

1. **Git-HOL**

## ****1. What is**** .gitignore****?****

A .gitignore file tells Git which files or folders it **should not track**.  
This is useful for **ignoring unwanted files** that

Are automatically generated (e.g., log files, build outputs)

Contain sensitive info (e.g., API keys in config files)

Are large and not needed in version control

Even if the file exists in your project folder, if it’s listed in .gitignore, Git won’t show it in git status, won’t add it to commits, and won’t push it to the remote repo.

## ****2. How to ignore unwanted files using**** .gitignore

### ****Step 1 – Create**** .gitignore ****file****

In the root of your project:

touch .gitignore

### ****Step 2 – Add patterns to**** .gitignore

Inside .gitignore, write patterns for the files/folders you want Git to ignore.

#### ****Examples****

# Ignore all .log files

\*.log

# Ignore the log folder

log/

# Ignore all .tmp files in any folder

\*.tmp

# Ignore a specific file

secret.txt

# Ignore build directory

/build/

### ****Step 3 – Apply changes****

If the files were already tracked before adding to .gitignore,  
you must remove them from the index first:

git rm --cached <file\_name>

Example:

git rm --cached debug.log

### ****Step 4 – Commit the**** .gitignore ****file****

git add .gitignore

git commit -m "Added .gitignore to ignore unwanted files"

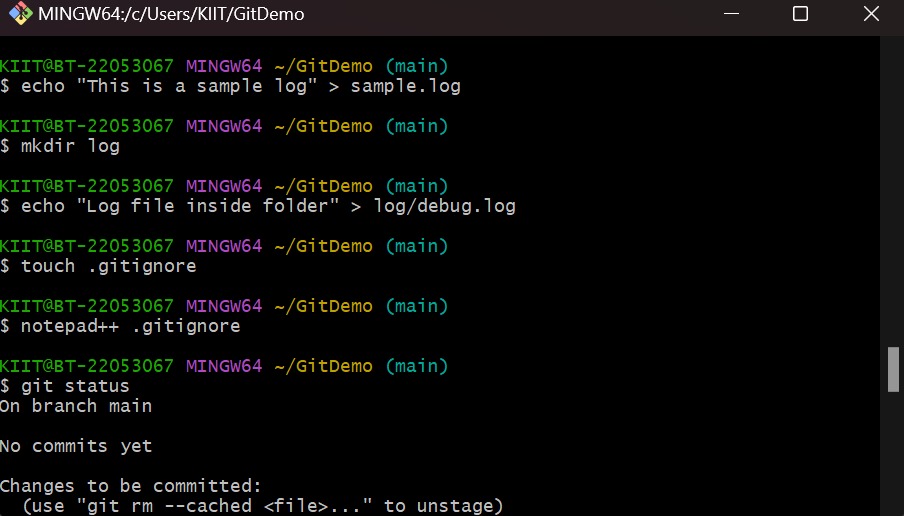
**Key Notes**

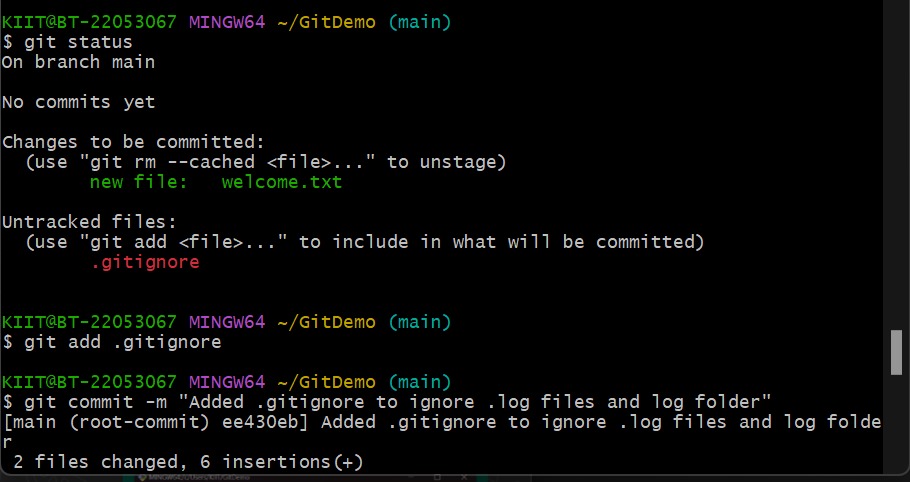
.gitignore only affects **untracked** files. If a file is already in Git history, ignoring it won’t remove it — you must untrack it first.

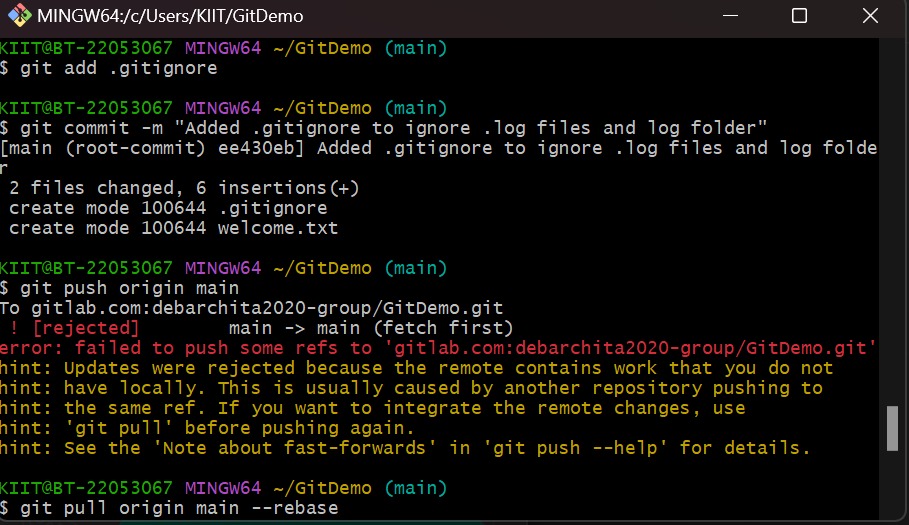
We can create **global** .gitignore rules for all repositories on your system:

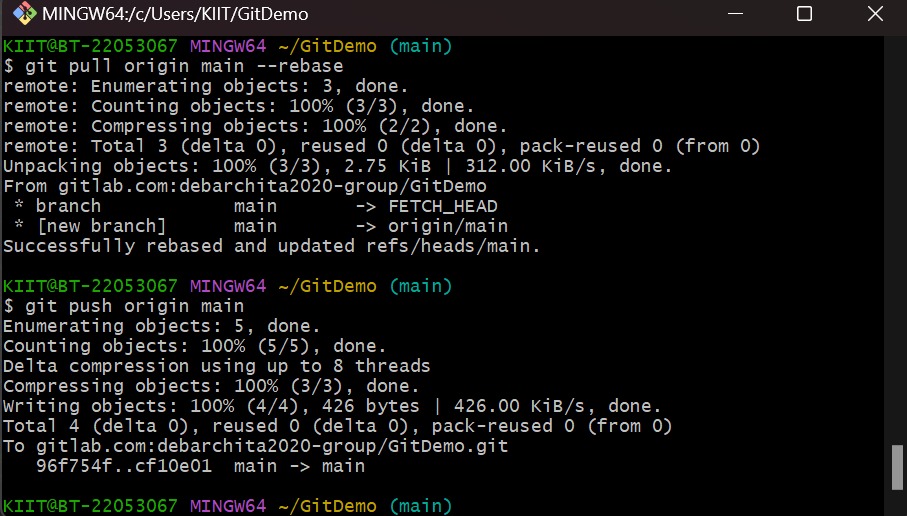
git config --global core.excludesfile ~/.gitignore\_global

**Output:**

****

****

****

****