**2 . GIT – HOL**

**Using .gitignore to Ignore Unwanted Files in Git**

**What is .gitignore and Its Purpose in Git**

.gitignore is a plain text file used by Git to determine which files or directories to ignore in a project. When a file or folder is listed in .gitignore, Git will not track changes to those files, nor will it include them in commits. This is useful for excluding temporary files, logs, build outputs, or sensitive information that should not be part of the version control system. Using .gitignore helps keep the repository clean and focused only on relevant source files.

**How to Ignore Unwanted Files and Folders Using .gitignore**

To ignore files or folders, you add patterns to the .gitignore file placed at the root of your repository. For example:

* To ignore all files with the .log extension, add:

\*.log

* To ignore a specific folder and its contents (e.g., a folder named log), add:

log/

Once these rules are added, Git will automatically exclude matching files and folders from being tracked or committed.

**Introduction**

In this lab, we learn how to use the .gitignore file to exclude unwanted files and folders from being tracked by Git. This helps keep the repository clean by ignoring temporary or log files.

**Prerequisites**

* Git installed and configured
* Notepad++ set as the default editor for Git
* Local Git repository with a remote repository on GitLab

**Steps Performed**

1. **Navigate to the local Git repository folder**  
   Used the command:  
   cd ~/GitDemo
2. **Created a sample .log file and a log folder with a file inside**  
   Commands used:

bash

Copy code

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

mkdir log

echo "Log folder file" > log/test.log

1. **Checked current Git status**  
   Ran: git status  
   Result showed sample.log and log/ as untracked files.
2. **Created or updated the .gitignore file to ignore .log files and the log folder**  
   Edited .gitignore to add the lines:

bash

Copy code

\*.log

log/

1. **Verified that .log files and the log folder are ignored**  
   Ran: git status  
   Result showed these files/folders were no longer listed as untracked.
2. **Added and committed the .gitignore file**  
   Commands:

sql

Copy code

git add .gitignore

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

1. **Pushed changes to remote GitLab repository**  
   Command:  
   git push origin main
2. **Result / Output**

The .log files and the entire log folder were successfully ignored by Git as expected. The repository remains clean without tracking unwanted log files.

