## **GIT HANDSON-2**

Gitlab Proejct : <https://gitlab.com/yogithaaah-group/gitdemo>

## **Objectives**

**What is .gitignore?**

When working with Git (a popular version control system), you may have **files and folders you don’t want to track**, like log files, build artifacts, temporary files, or system-specific data. The .gitignore file is a special text file that **tells Git to ignore specific files and folders** in your project, so they aren’t tracked or included in commits.

**How to Use .gitignore to Ignore Files**

1. **Create/Edit the .gitignore file**  
   In your project’s root directory, create a plain text file named .gitignore. If it already exists, just open it.
2. **List the files or folders you want to ignore**  
   Write patterns, one per line.
3. **Save the file**  
   Git will now ignore any files and folders that match the patterns specified in .gitignore.

Estimated time to complete this lab: **20 minutes.**

Create a **“.log”** file and a **log folder** in the working directory of Git. Update the **.gitignore** file in such a way that on committing, these files (.log extensions and log folders) are ignored.

Verify if the git status reflects the same about working directory, local repository and git repository.

**Step - 1**: Open Git bash and navigate to your repo

A black screen with white text

AI-generated content may be incorrect.

**Step - 2**: Create a new file named **.gitignore** in the root of your repository:

A black background with white and purple text

AI-generated content may be incorrect.

**Step – 3**: Open the .gitignore file in any Notepad and add the following line:

A black background with yellow text

AI-generated content may be incorrect.

A screenshot of a computer

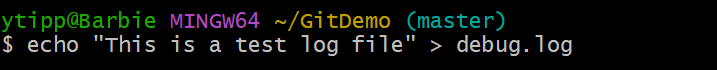
AI-generated content may be incorrect.

**Step – 4**: Now check the .gitignore file from GitBash, it should display the content entered in Notepad++

A yellow text on a black background

AI-generated content may be incorrect.

**Step – 5**: Now, create a sample log file to test the .gitignore:



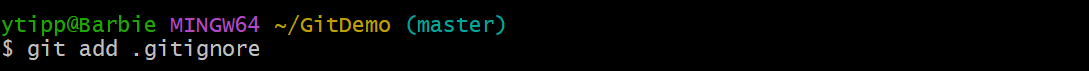
**Step – 6**: Check the Git status to confirm that debug.log is not being tracked:

A screen shot of a computer

AI-generated content may be incorrect.

We donot see a debug.log file because that is being ignored by the Git.

**Step – 7**: Add the .gitignore file to Git:



**Step – 8**: Commit the .gitignore file:

A computer screen with white text

AI-generated content may be incorrect.

**Step – 9**: Push to GitLab and Verify

A computer screen with white text

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.

**Step – 10**: Final Project in GtitLab

A screenshot of a computer

AI-generated content may be incorrect.