1.Git-HOL

Step 1: Check Git Installation

Open Git Bash and run:

git –version

Step 2: Configure Git User Details

Set your name:

git config --global user.name "Sidharth K"

Set your email:

git config --global user.email "iamsidharthkarthikeyan@gmail.com"

Verify configuration:

git config --global --list

Step 3: Add Notepad++ to PATH

1. Find your Notepad++ install path C:\Program Files\Notepad++\notepad++.exe
2. Add to Environment Variables → Path (User Variables) → Add the above folder path.
3. Verify: notepad++

Step 4: Make Notepad++ Default Git Editor

git config --global core.editor "notepad++ -multiInst -nosession"

Verify:

git config --global -e

Step 5: Create Local Repository

mkdir GitDemo

cd GitDemo

git init

Step 6: Create a File

dir

type welcome.txt

Step 7: Check Git Status

git status

Step 8: Add File to Staging

git add welcome.txt

git status

Step 9: Commit Changes

git commit -m "Initial commit - added welcome.txt"

Step 10: Link Remote Repo

git remote add origin https://github.com/sid110305/GitDemo.git

Step 11: Push to Remote

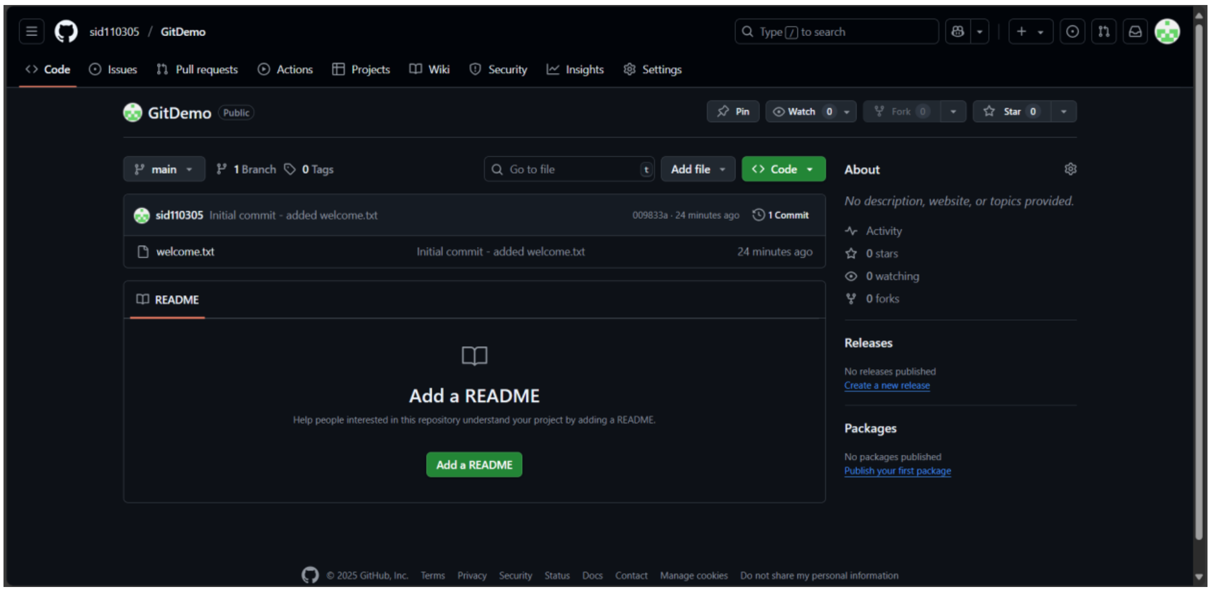
git branch -M main

git push -u origin main

Step 12: Pull from Remote (Optional Test)

git pull origin main

OUTPUT:



sid110305 Initial commit - added welcome.txt 009833a · 24 minutes ago 1 Commit

welcome.txt Initial commit - added welcome.txt 24 minutes ago

Help people interested in this repository understand your project by adding a README.

About

No description, website, or topics provided.

Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

2.Git-HOL-Ignoring Files in Git (Git Bash)

Step 1: Ensure Git is initialized

```
git init
```

Step 2: Create unwanted files and folder

```
echo "This is a log file" > error.log
mkdir log
echo "This is inside the log folder" > log/info.txt
```

Step 3: Create .gitignore file

```
echo "*.log" > .gitignore
echo "log/" >> .gitignore
```

Step 4: Stage .gitignore (ignored files won’t be staged)

```
git add .gitignore
```

Step 5: Commit changes

```
git commit -m "Added .gitignore to exclude .log files and log folder"
```

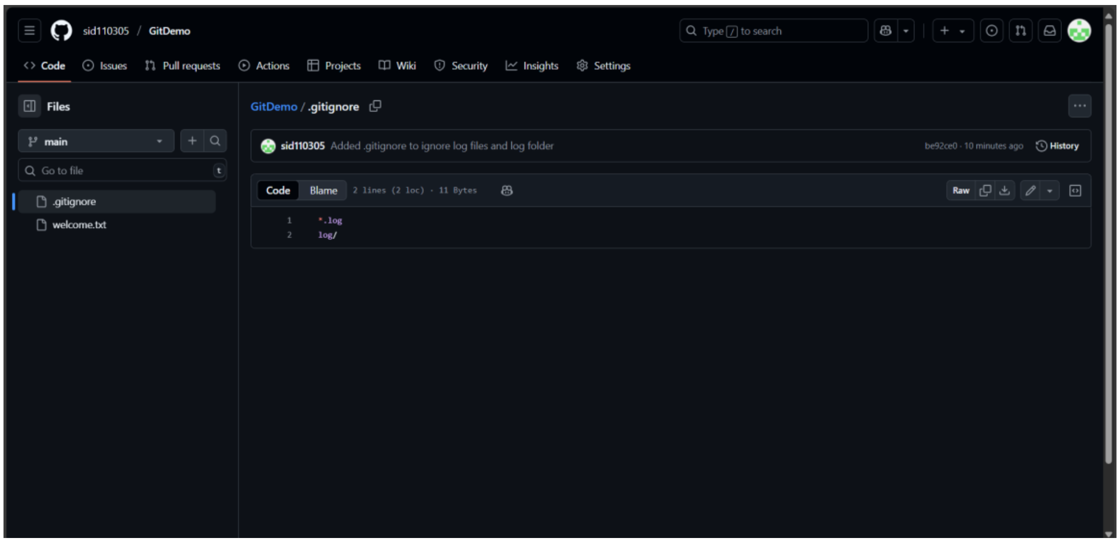
Step 6: Add Remote

```
git remote add origin https://github.com/YourUsername/GitDemo.git
```

Step 7: Push to GitHub

```
git pull origin master --allow-unrelated-histories
git push -u origin master
```

OUTPUT:



sid110305 Added .gitignore to ignore log files and log folder

be92ce0 · 10 minutes ago

2 lines (2 loc) · 11 Bytes

```
*.log
log/
```

3.Git-HOL

Branching (Git Bash)

1. **Create a New Branch**

```
git branch GitNewBranch
```

2. **List All Branches (Local & Remote)**

```
git branch -a
```

3. **Switch to the New Branch**

```
git checkout GitNewBranch
```

4. **Add a New File & Add Content**

```
echo "This is content for the new branch" > branchfile.txt
```

5. **Stage and Commit Changes**

```
git add branchfile.txt
git commit -m "Added branchfile.txt in GitNewBranch"
```

6. **Check Status**

```
git status
```

Merging (Git Bash)

1. **Switch Back to main Branch**

```
git checkout main
```

2. **View Differences Between main & Branch (CLI)**

```
git diff main GitNewBranch
```

3. **Merge Branch into main**

```
git merge GitNewBranch
```

4. **View Merge History**

```
git log --oneline --graph --decorate
```

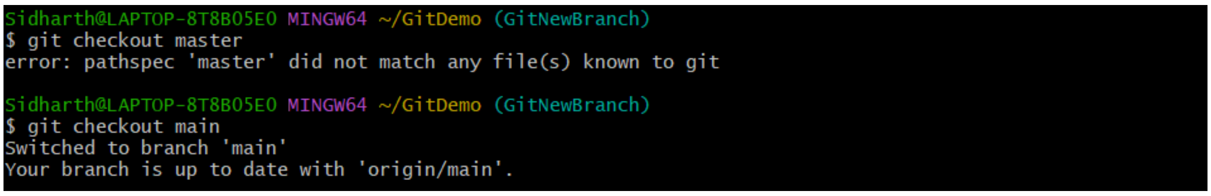
5. **Delete the Branch After Merging**

```
git branch -d GitNewBranch
```

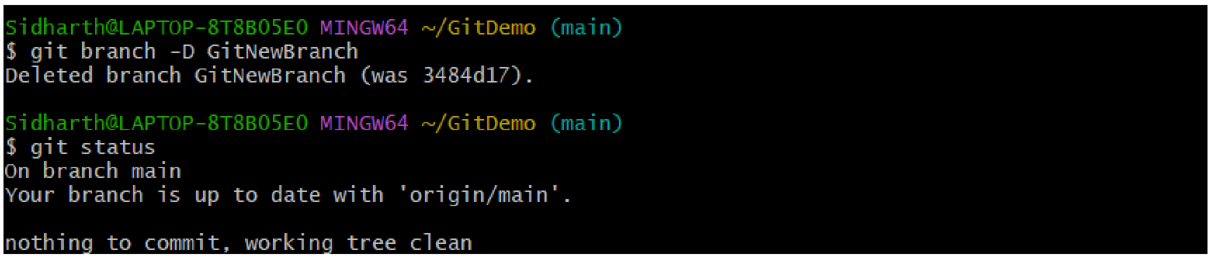
6. **Final Status Check**

git status

Output:



```
Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (GitNewBranch)
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (GitNewBranch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```



```
Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git branch -D GitNewBranch
Deleted branch GitNewBranch (was 3484d17).

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

4.GIT HOL

Steps 1. Verify if master/main is clean

```
git checkout main
git status
```

Step 2. Create a new branch GitWork and add a file

```
git branch GitWork
git checkout GitWork
echo "<message>Hello from GitWork branch</message>" > hello.xml
```

Step 3. Stage and commit changes in branch

```
git add hello.xml
git commit -m "Added hello.xml in GitWork branch"
```

Step 4. Switch back to master/main

```
git checkout main
```

Step 5. Add a different hello.xml in main

```
echo "<message>Hello from main branch</message>" > hello.xml
git add hello.xml
git commit -m "Added hello.xml in main branch"
```

Step 6. View commit history (all branches)

```
git log --oneline --graph --decorate --all
```

Step 7. Compare branches (CLI diff)

```
git diff main GitWork
```

Step 8. Compare branches (P4Merge – optional)

```
git difftool main GitWork
```

Step 9. Merge GitWork into main (expect conflict)

```
git merge GitWork
```

Step 10. Resolve conflict (3-way merge)

Open hello.xml → You'll see conflict markers:

<<<<<<< HEAD

<message>Hello from main branch</message> =======

<message>Hello from GitWork branch</message>

>>>>>>> GitWork

Manually edit to desired final content, e.g.:

<message>Hello from merged version</message>

Then:

```
git add hello.xml
git commit -m "Resolved merge conflict in hello.xml"
```

Step 11. Add .gitignore for backup files

```
echo "*.bak" >> .gitignore
git add .gitignore
git commit -m "Added .gitignore for backup files"
```

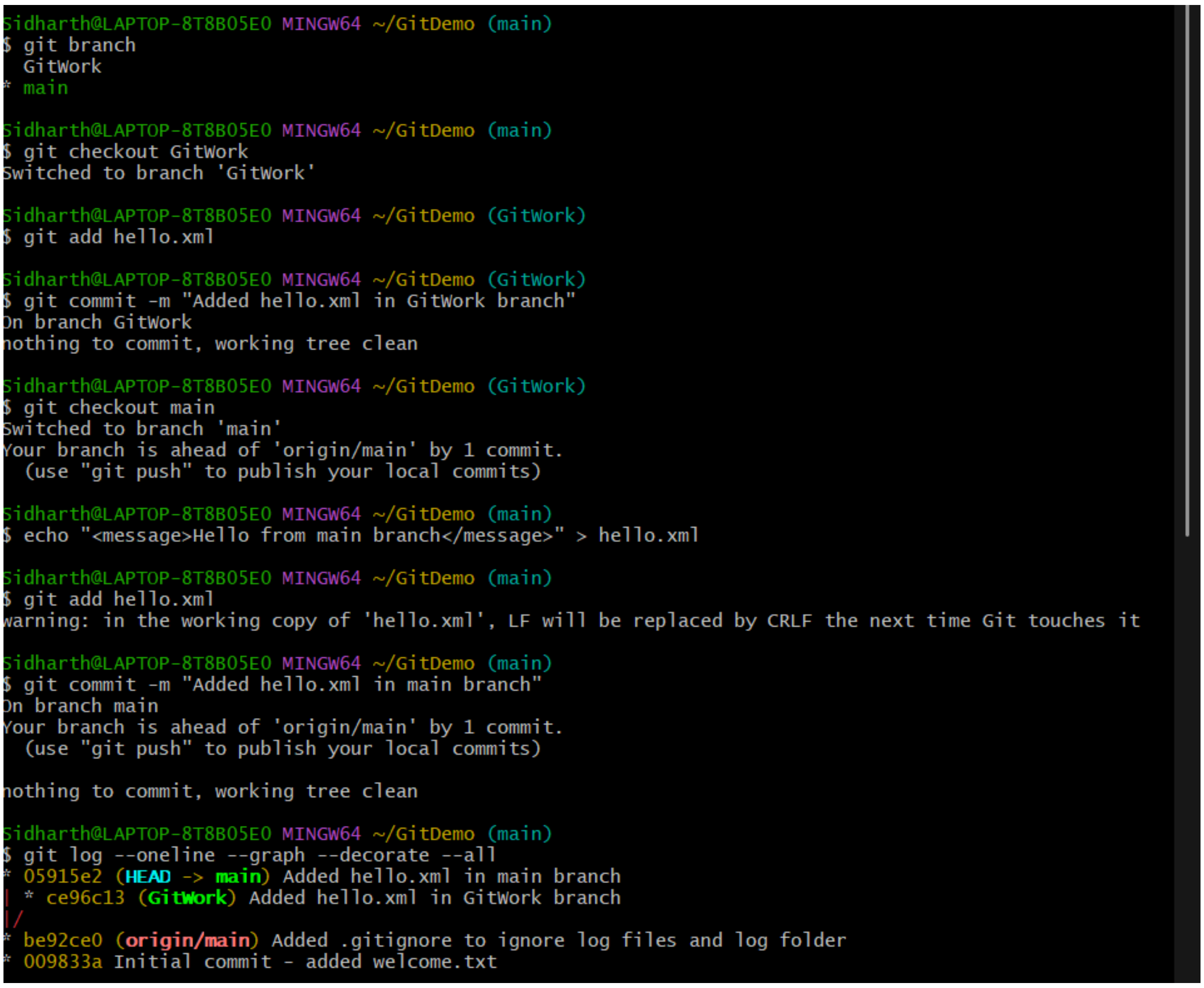
Step 12. Delete the merged branch

```
git branch -d GitWork
```

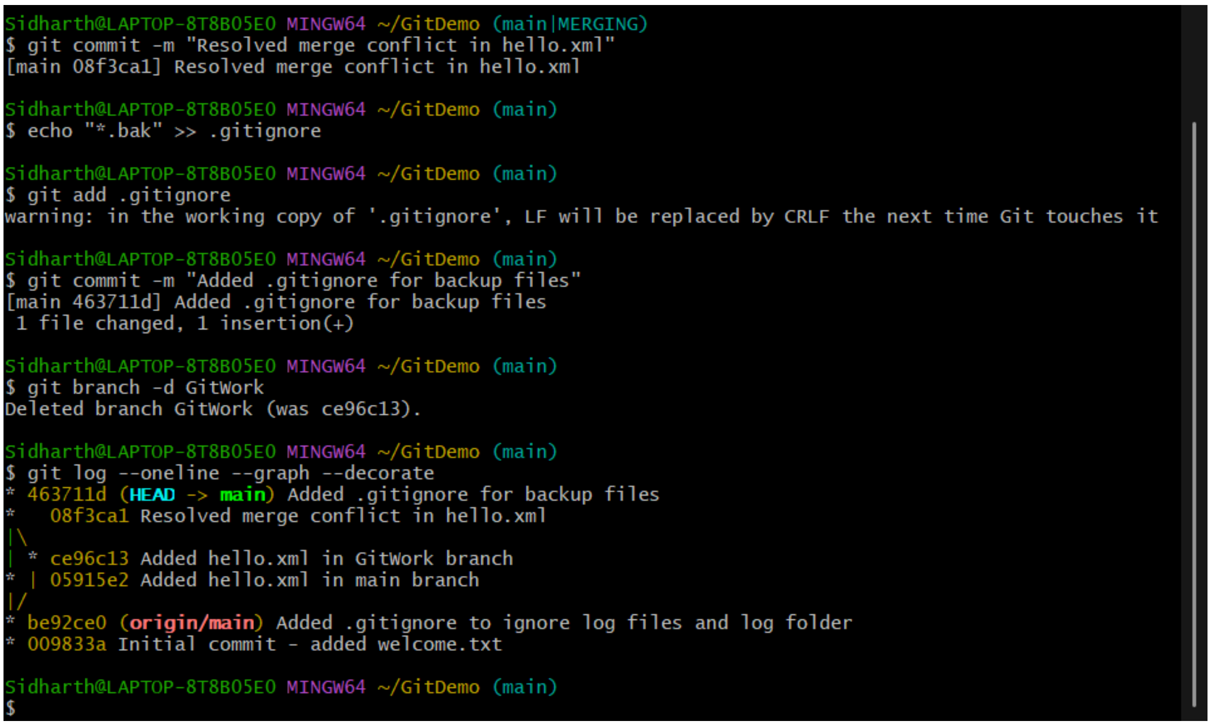
Step 13. View final commit history

```
git log --oneline --graph –decorate
```

Output:



```
Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git branch
  GitWork
* main

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git checkout GitWork
Switched to branch 'GitWork'

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (GitWork)
$ git add hello.xml

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (GitWork)
$ git commit -m "Added hello.xml in GitWork branch"
On branch GitWork
nothing to commit, working tree clean

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (GitWork)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ echo "<message>Hello from main branch</message>" > hello.xml

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git add hello.xml
warning: in the working copy of 'hello.xml', LF will be replaced by CRLF the next time Git touches it

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git commit -m "Added hello.xml in main branch"
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git log --oneline --graph --decorate --all
* 05915e2 (HEAD -> main) Added hello.xml in main branch
| * ce96c13 (GitWork) Added hello.xml in GitWork branch
|/
* be92ce0 (origin/main) Added .gitignore to ignore log files and log folder
* 009833a Initial commit - added welcome.txt
```



```
Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main|MERGING)
$ git commit -m "Resolved merge conflict in hello.xml"
[main 08f3ca1] Resolved merge conflict in hello.xml

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ echo "*.bak" >> .gitignore

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git add .gitignore
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git commit -m "Added .gitignore for backup files"
[main 463711d] Added .gitignore for backup files
 1 file changed, 1 insertion(+)

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git branch -d GitWork
Deleted branch GitWork (was ce96c13).

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git log --oneline --graph --decorate
* 463711d (HEAD -> main) Added .gitignore for backup files
*   08f3ca1 Resolved merge conflict in hello.xml
|\
| * ce96c13 Added hello.xml in GitWork branch
* | 05915e2 Added hello.xml in main branch
|/
* be92ce0 (origin/main) Added .gitignore to ignore log files and log folder
* 009833a Initial commit - added welcome.txt

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$
```

5.Git-HOL

Step 1. Verify if master is in clean state

```
git status
```

Step 2. List out all the available branches

```
git branch -a
```

Step 3. Pull the remote Git repository to the master branch

```
git checkout master
git pull origin master
```

Step 4. Push the pending changes from “Git-T03-HOL_002” to remote

```
git checkout Git-T03-HOL_002
git add .
git commit -m
git push origin Git-T03-HOL_002
```

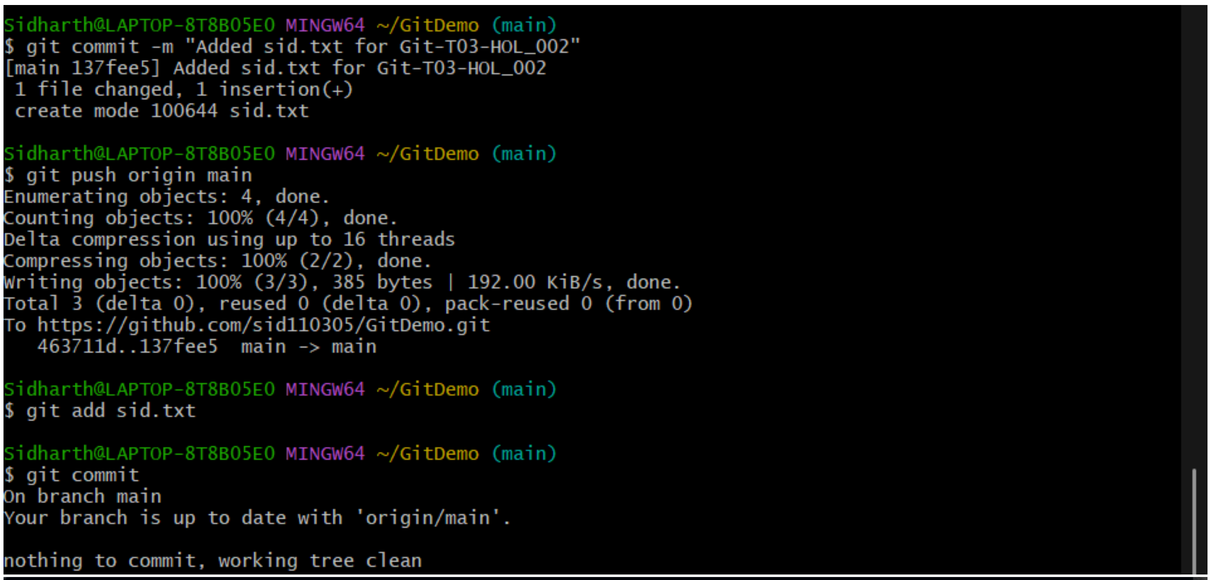
Step 5. Merge Git-T03-HOL_002 changes into master and push

```
git checkout master
git merge Git-T03-HOL_002
git push origin master
```

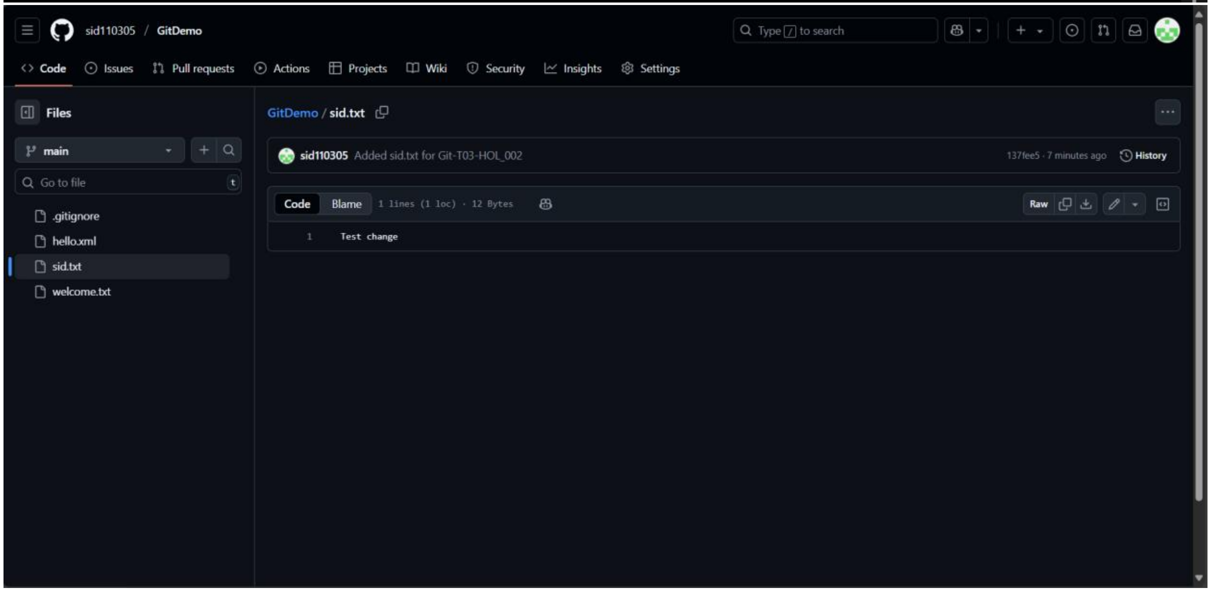
then:

```
git add sid.txt
git commit
git push origin master
```

Output:



```
Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git commit -m "Added sid.txt for Git-T03-HOL_002"
[main 137fee5] Added sid.txt for Git-T03-HOL_002
 1 file changed, 1 insertion(+)
 create mode 100644 sid.txt

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 385 bytes | 192.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/sid110305/GitDemo.git
   463711d..137fee5  main -> main

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git add sid.txt

Sidharth@LAPTOP-8T8B05E0 MINGW64 ~/GitDemo (main)
$ git commit
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```



sid110305 Added sid.txt for Git-T03-HOL_002 137fee5 · 7 minutes ago

1 lines (1 loc) · 12 Bytes

```
Test change
```