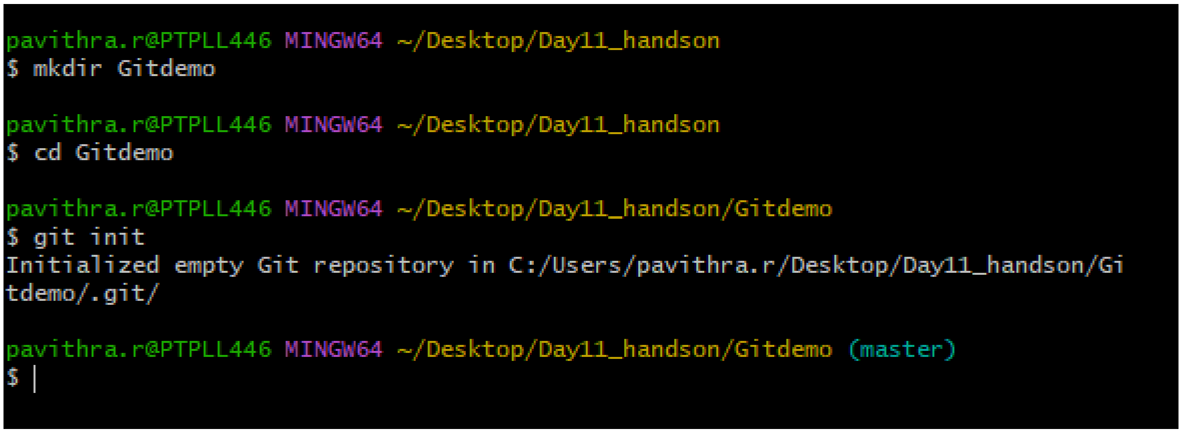
Name: Pavithra R
Date: 21/07/2025

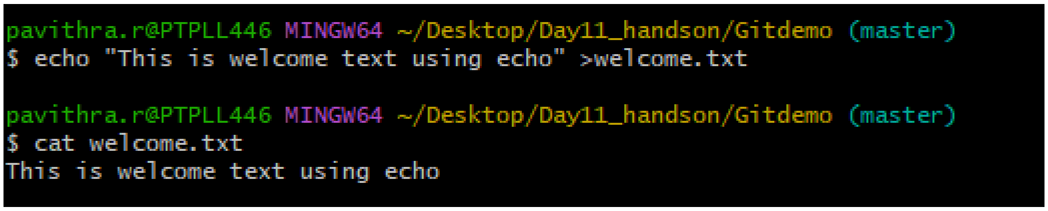
Git Commands

1. Open Git bash shell and create a new folder "**GitDemo**" and initialize as git repository
2. Git bash initializes the "**GitDemo**" repository.



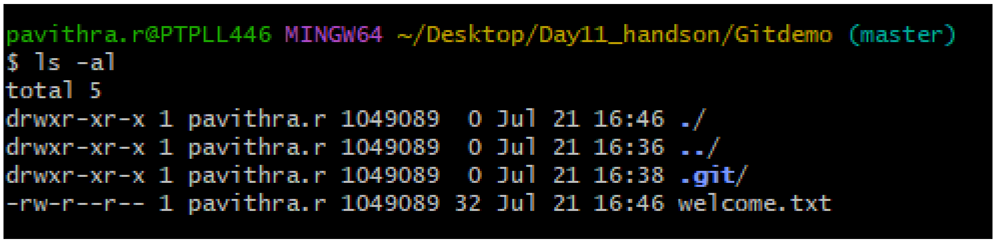
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson
$ mkdir Gitdemo

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson
$ cd Gitdemo

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo
$ git init
Initialized empty Git repository in C:/Users/pavithra.r/Desktop/Day11_handson/Gi
tdemo/.git/

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo (master)
$ |
```

3. Create a file **"welcome.txt"** and add content to the file (use echo or cat command)



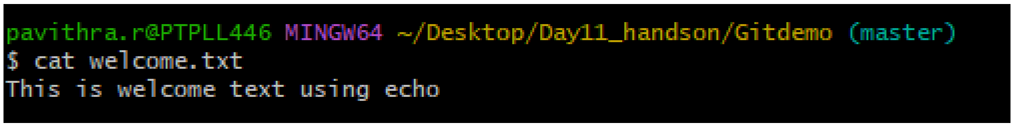
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo (master)
$ echo "This is welcome text using echo" >welcome.txt

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo (master)
$ cat welcome.txt
This is welcome text using echo
```

4.To verify if the file "welcome.txt" is created, execute ls-al



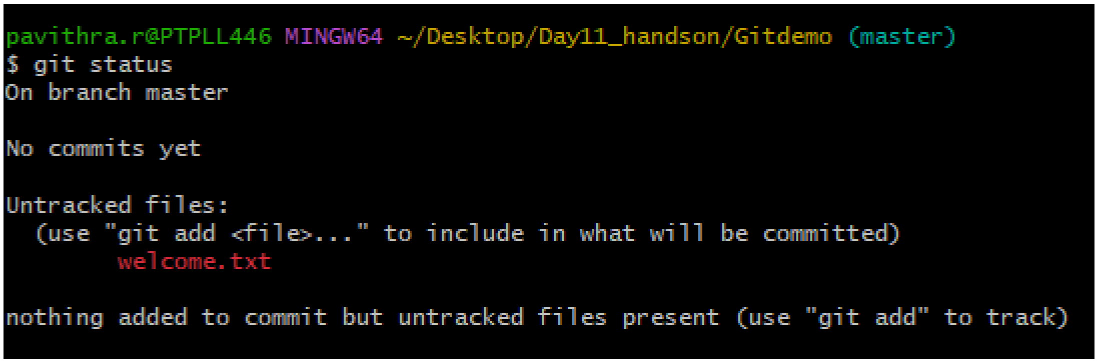
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo (master)
$ ls -al
total 5
drwxr-xr-x 1 pavithra.r 1049089  0 Jul 21 16:46 ./
drwxr-xr-x 1 pavithra.r 1049089  0 Jul 21 16:36 ../
drwxr-xr-x 1 pavithra.r 1049089  0 Jul 21 16:38 .git/
-rw-r--r-- 1 pavithra.r 1049089 32 Jul 21 16:46 welcome.txt
```

5. To verify the content, execute the command cat filename



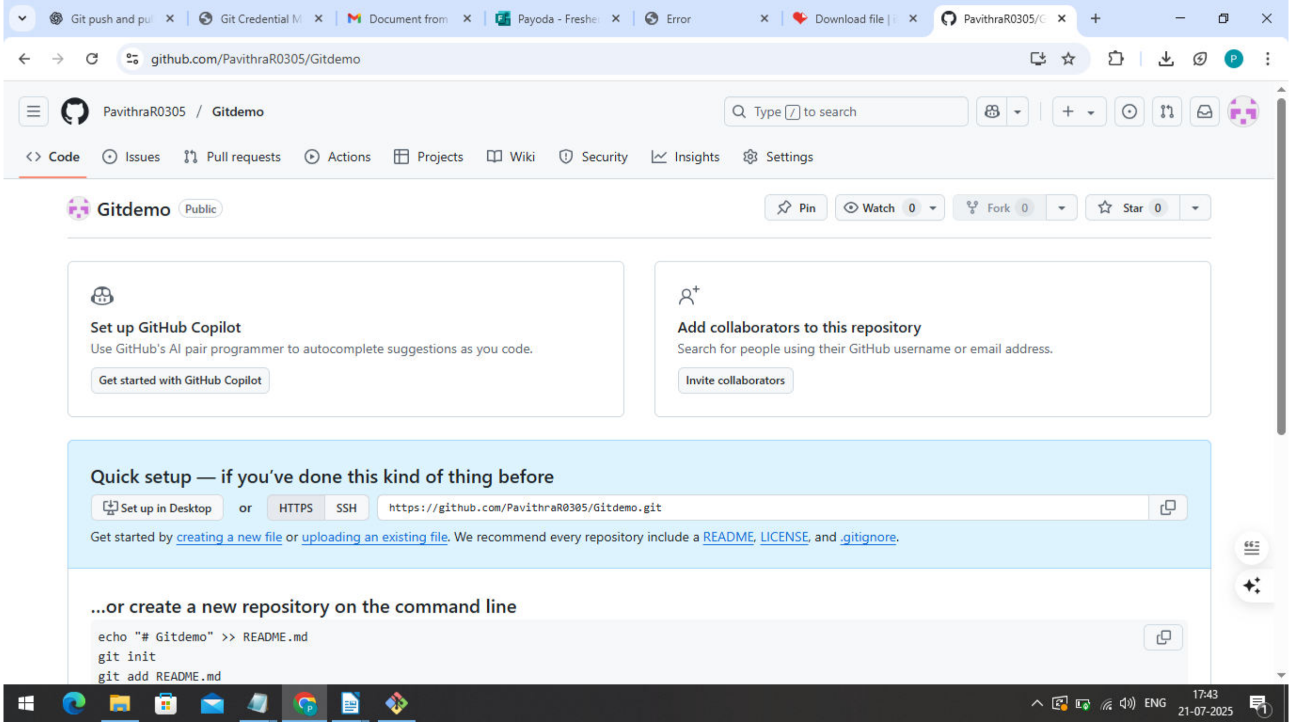
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo (master)
$ cat welcome.txt
This is welcome text using echo
```

6. Execute git status



```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handson/Gitdemo (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        welcome.txt

nothing added to commit but untracked files present (use "git add" to track)
```

7 Signup with GitHub and create a remote repository **“GitDemo”**



Set up GitHub Copilot

Use GitHub's AI pair programmer to autocomplete suggestions as you code.

Add collaborators to this repository

Search for people using their GitHub username or email address.

Quick setup — if you've done this kind of thing before

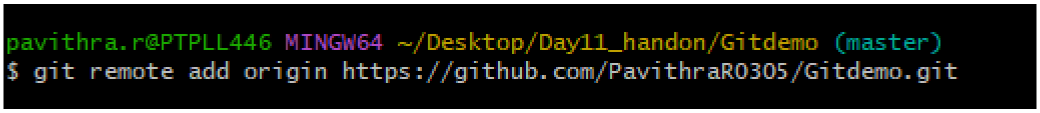
https://github.com/PavithraR0305/Gitdemo.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

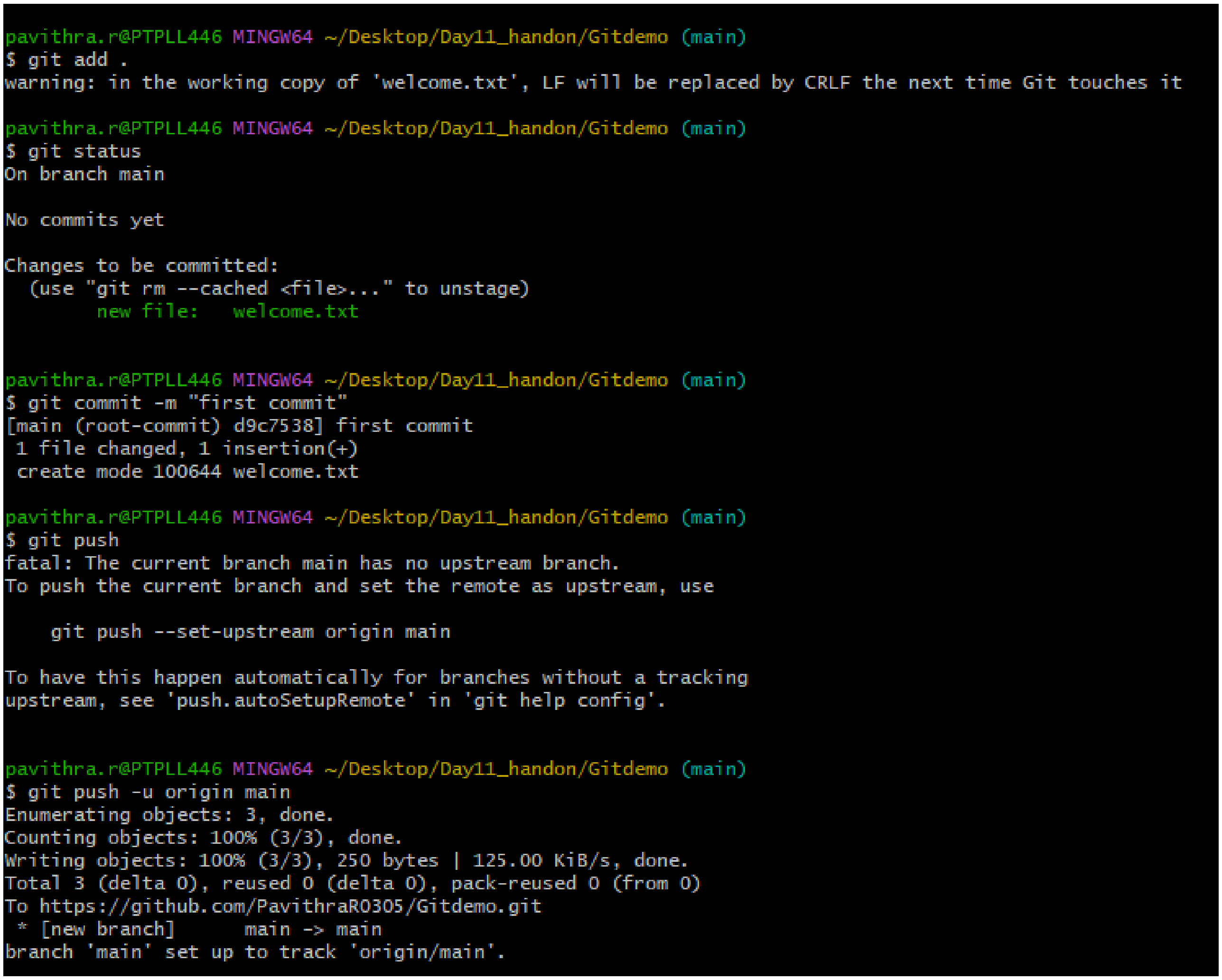
```
echo "# Gitdemo" >> README.md
git init
git add README.md
```

8. Add remote origin



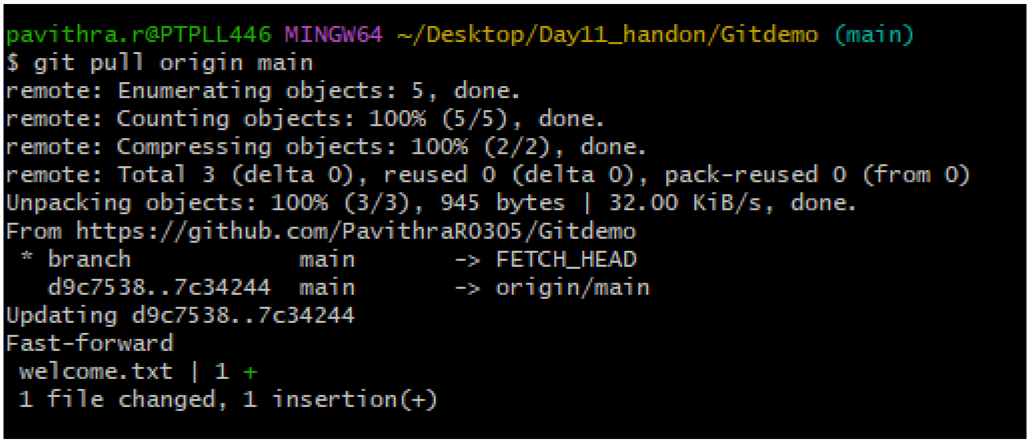
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (master)
$ git remote add origin https://github.com/PavithraR0305/Gitdemo.git
```

9.Create files and and push the local to remote repository.



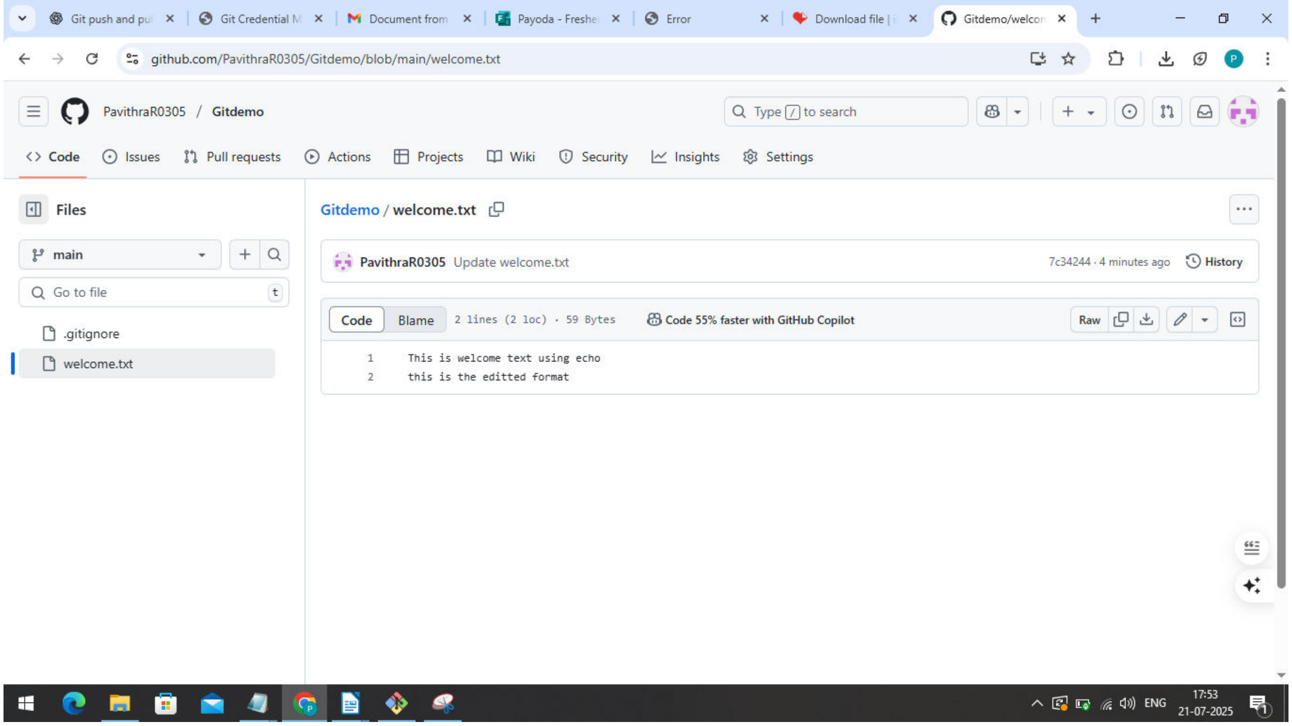
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git add .
warning: in the working copy of 'welcome.txt', LF will be replaced by CRLF the next time Git touches it

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   welcome.txt


pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git commit -m "first commit"
[main (root-commit) d9c7538] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 welcome.txt

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git push
fatal: The current branch main has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin main

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.


pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 250 bytes | 125.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/PavithraR0305/Gitdemo.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

10. Make changes in the remote and pull the changes to the local.



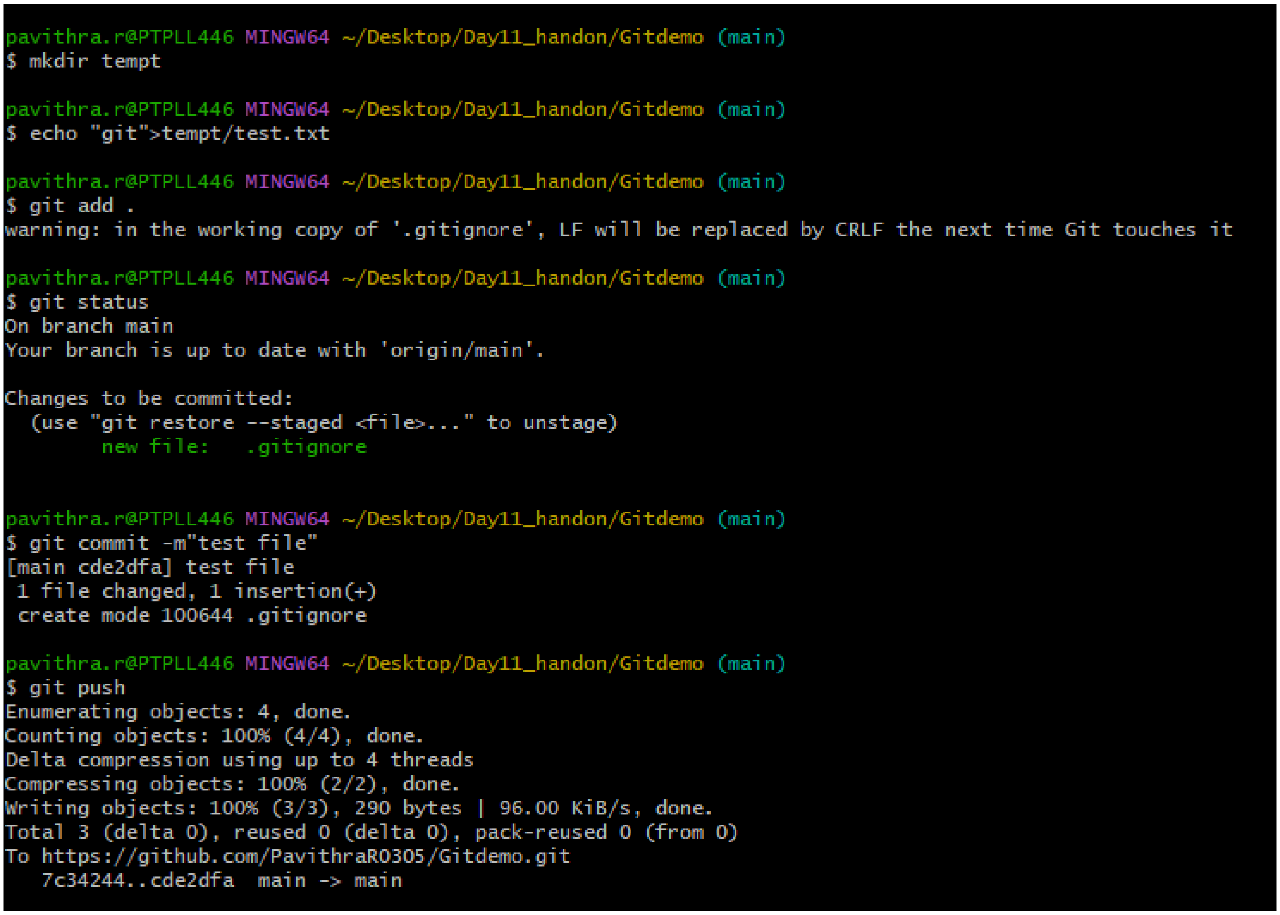
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git pull origin main
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 945 bytes | 32.00 KiB/s, done.
From https://github.com/PavithraR0305/Gitdemo
 * branch            main       -> FETCH_HEAD
   d9c7538..7c34244  main       -> origin/main
Updating d9c7538..7c34244
Fast-forward
 welcome.txt | 1 +
 1 file changed, 1 insertion(+)
```

11. Create a .gitignore file to ignore contents of temp folder



PavithraR0305 Update welcome.txt 7c34244 · 4 minutes ago

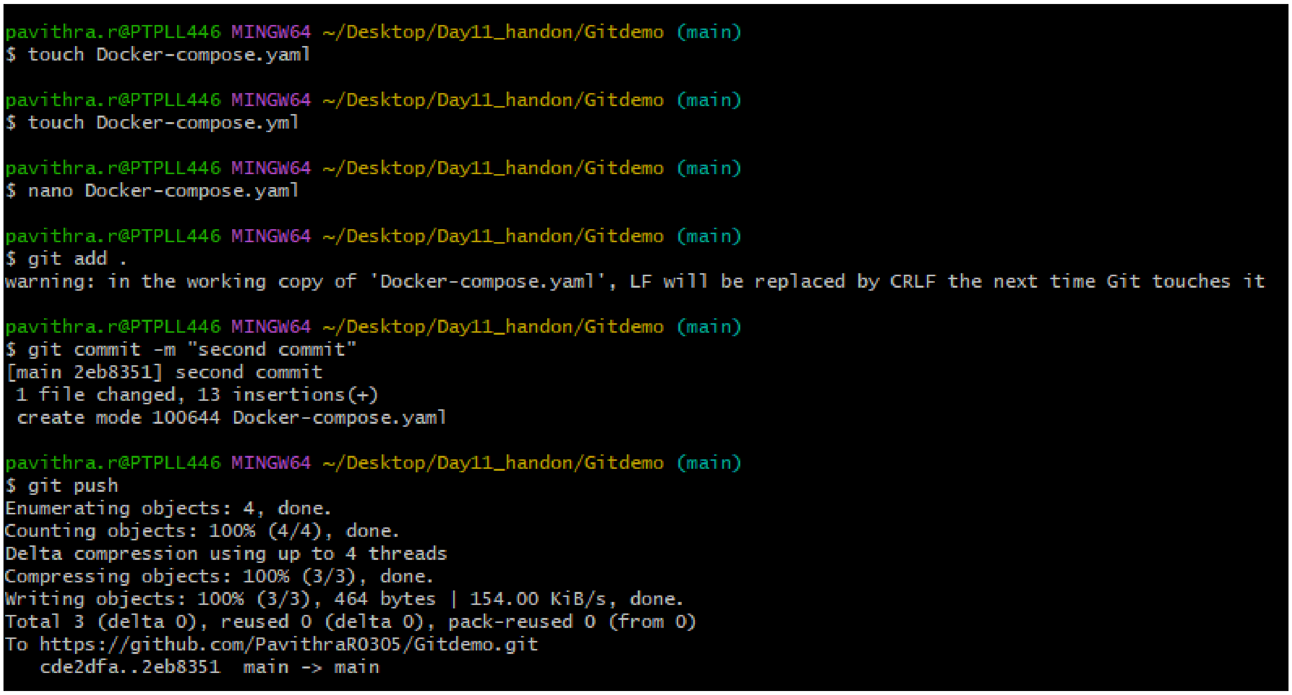
```
This is welcome text using echo
this is the editted format
```



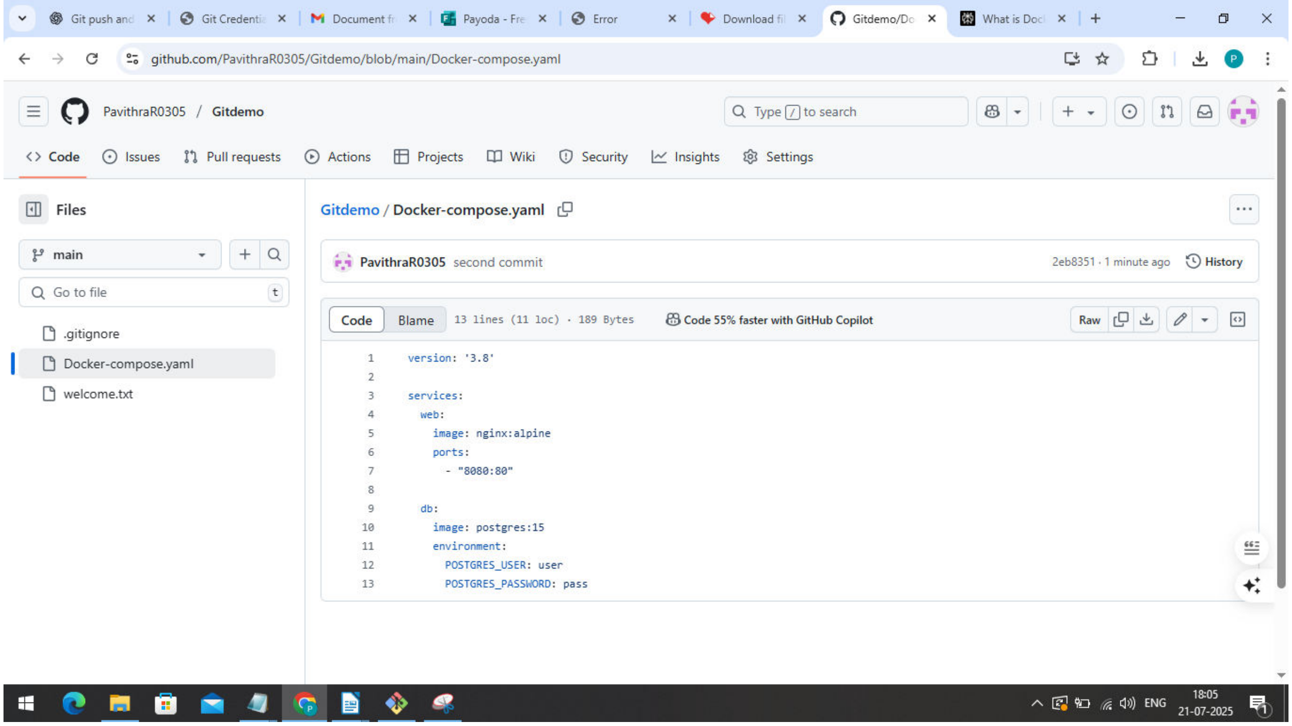
```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ mkdir tempt

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ echo "git">tempt/test.txt

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .gitignore


pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git commit -m"test file"
[main cde2dfa] test file
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 290 bytes | 96.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/PavithraR0305/Gitdemo.git
   7c34244..cde2dfa  main -> main
```

Docker

1. What is Docker-compose.yaml file? Give sample docker-compose file.



```
pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ touch Docker-compose.yaml

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ touch Docker-compose.yml

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ nano Docker-compose.yaml

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git add .
warning: in the working copy of 'Docker-compose.yaml', LF will be replaced by CRLF the next time Git touches it

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git commit -m "second commit"
[main 2eb8351] second commit
 1 file changed, 13 insertions(+)
 create mode 100644 Docker-compose.yaml

pavithra.r@PTPLL446 MINGW64 ~/Desktop/Day11_handon/Gitdemo (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 464 bytes | 154.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/PavithraR0305/Gitdemo.git
   cde2dfa..2eb8351  main -> main
```



github.com/PavithraR0305/Gitdemo/blob/main/Docker-compose.yaml

PavithraR0305 / Gitdemo

Gitdemo / Docker-compose.yaml

PavithraR0305 second commit 2eb8351 · 1 minute ago

13 lines (11 loc) · 189 Bytes

```
version: '3.8'

services:
  web:
    image: nginx:alpine
    ports:
      - "8080:80"

  db:
    image: postgres:15
    environment:
      POSTGRES_USER: user
      POSTGRES_PASSWORD: pass
```

2. What is DOCKER HUB

Docker Hub is a **cloud-based registry service** where you can S**tore**, **share**, and **distribute** Docker images, **Search** for public images (like Node.js, Python, Ubuntu, MySQL, etc.), **Push** your own custom images for use on any machine, **Collaborate** on containerized projects with teams

Uses:

- centralised distribution
- reliable base image
- efficiency
- collaboration
- extensibility

Docker Hub is the go-to-platform for most Docker users to share and consume container images, automate their pipeline, and manage both community and proprietary software deployments in a scalable, integrated environment