**WEEK - 8 HANDSON SOLUTION**

**EXERCISE -1 :**

HP@HP MINGW64 ~

$ git --version

git version 2.49.0.windows.1

HP@HP MINGW64 ~

$ git config --global user.name "dhatchayini"

HP@HP MINGW64 ~

$ git config --global user.email "dhatchayinirajavelu845@gmail.com"

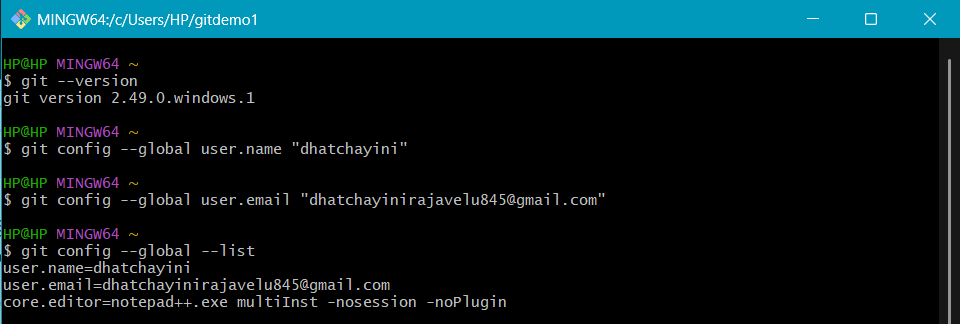
HP@HP MINGW64 ~

$ git config --global --list

user.name=dhatchayini

user.email=dhatchayinirajavelu845@gmail.com

core.editor=notepad++.exe multiInst -nosession -noPlugin



HP@HP MINGW64 ~

$ mkdir gitdemo1

HP@HP MINGW64 ~

$ cd gitdemo1

HP@HP MINGW64 ~/gitdemo1

$ git init

Initialized empty Git repository in C:/Users/HP/gitdemo1/.git/

HP@HP MINGW64 ~/gitdemo1 (master)

$ echo "Hello from gitdemo" >welcome.txt

HP@HP MINGW64 ~/gitdemo1 (master)

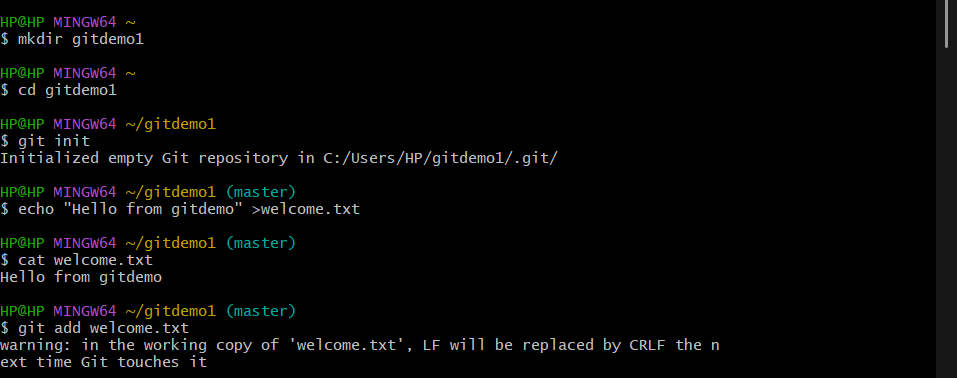
$ cat welcome.txt

Hello from gitdemo

HP@HP MINGW64 ~/gitdemo1 (master)

$ git add welcome.txt

warning: in the working copy of 'welcome.txt', LF will be replaced by CRLF the next time Git touches it



HP@HP MINGW64 ~/gitdemo1 (main)

$ git remote add origin <https://gitlab.com/dhatchayini/gitdemo1.git>

HP@HP MINGW64 ~/gitdemo1 (main)

$ git push -u origin main

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 4 threads

Compressing objects: 100% (2/2), done.

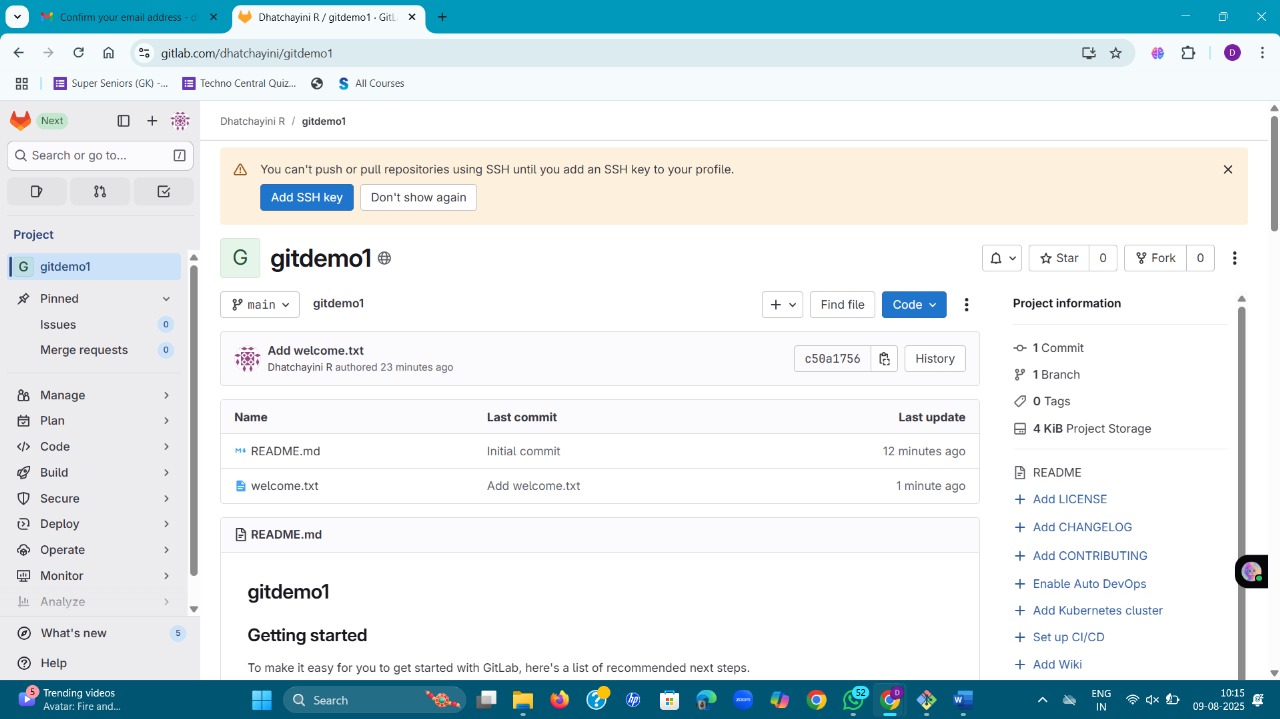
Writing objects: 100% (3/3), 311 bytes | 155.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

To https://gitlab.com/dhatchayini/gitdemo1.git

5a6a88a..c50a175 main -> main

branch 'main' set up to track 'origin/main'.



**EXERCISE -2 :**

**1.Explain git ignore:**

* .Keeps the repository clean.
* Prevents sensitive information (like passwords in .env files) from being uploaded.
* Saves space by ignoring large temporary files.
* Avoids conflicts from system-specific files (e.g., .DS\_Store on Mac, Thumbs.db on Windows).

**2.Explain how to ignore unwanted files using git ignore:**

**Step 1: Create .gitignore file**

In your project’s root directory:

touch .gitignore

Or create it manually using any text editor.

**Step 2: Add patterns to ignore**

Write the file/folder names or patterns you want to ignore.

Examples:

# Ignore log files

\*.log

# Ignore a folder named log

log/

# Ignore environment files

.env

# Ignore node\_modules folder

node\_modules/

# Ignore all .tmp files anywhere

\*.tmp

**Step 3: Save the file and check**

You can see which files are being ignored:

git status --ignored

Step 4: Important — Remove already tracked files

If a file is already tracked by Git, adding it to .gitignore won’t stop Git from tracking it until you remove it from tracking:

git rm --cached filename

Example:

git rm --cached .env

Then commit the changes:

git commit -m "Updated .gitignore"

HP@HP MINGW64 ~ (master)

$ cd /c/Users/HP/Documents/git-handson2

HP@HP MINGW64 ~/Documents/git-handson2 (master)

$ git clone https://gitlab.com/dhatchayini/git-handson2.git

Cloning into 'git-handson2'...

remote: Enumerating objects: 3, done.

remote: Counting objects: 100% (3/3), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

Receiving objects: 100% (3/3), done.

HP@HP MINGW64 ~/Documents/git-handson2 (master)

$ cd git-handson2

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ notepad++ .gitignore

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ echo "This is a log file"> example.log

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ mkdir log

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ echo "log folder file" > log/test.txt

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ git status --ignored

On branch main

Your branch is up to date with 'origin/main'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

.gitignore

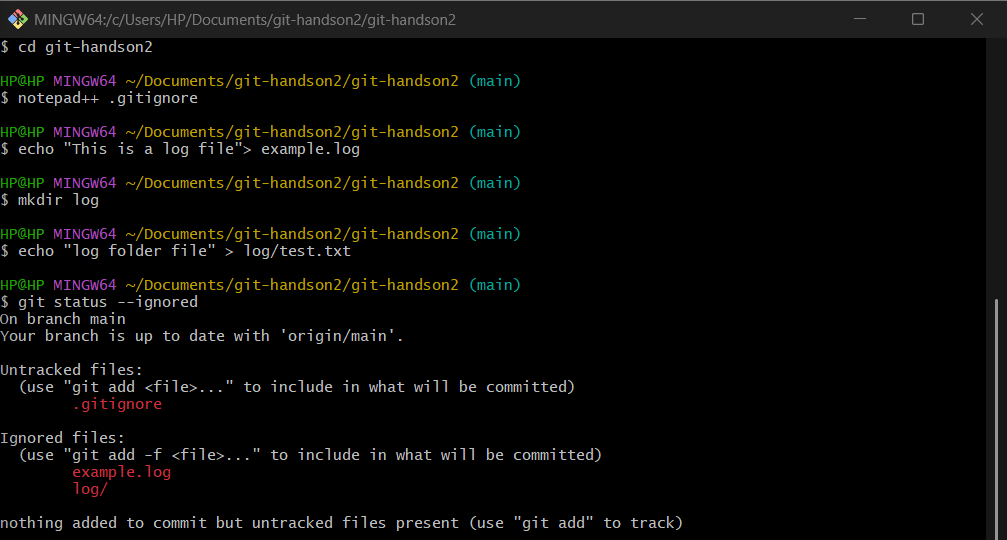
Ignored files:

(use "git add -f <file>..." to include in what will be committed)

example.log

log/

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



HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ git add .gitignore

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

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

[main 06a8342] Added .gitignore to ignore .log files and log folder

1 file changed, 2 insertions(+)

create mode 100644 .gitignore

HP@HP MINGW64 ~/Documents/git-handson2/git-handson2 (main)

$ git push origin main

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 4 threads

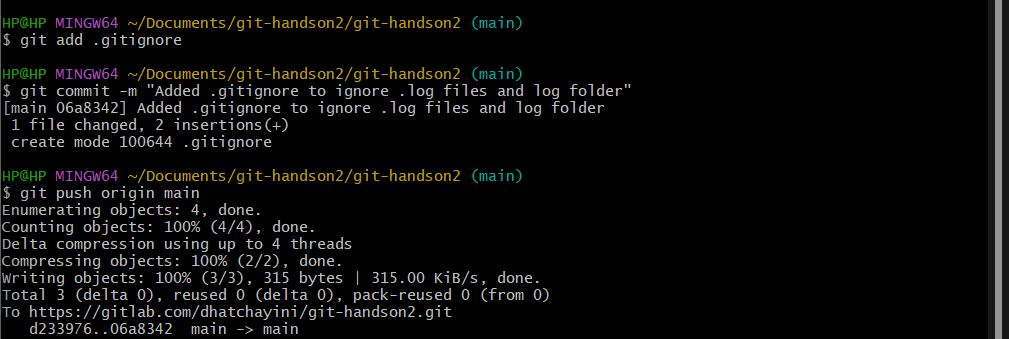
Compressing objects: 100% (2/2), done.

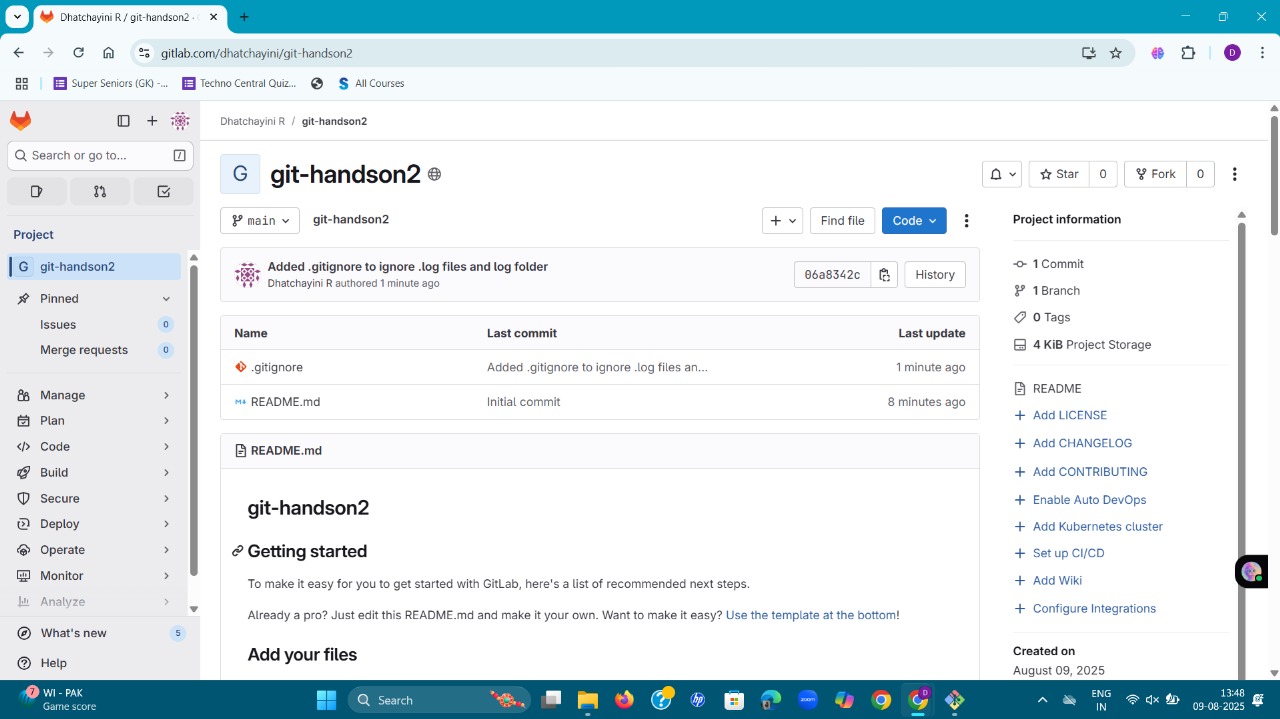
Writing objects: 100% (3/3), 315 bytes | 315.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

To https://gitlab.com/dhatchayini/git-handson2.git

d233976..06a8342 main -> main





**EXERCISE - 3 :**

HP@HP MINGW64 ~ (master)

$ git config --global diff.tool p4merge

HP@HP MINGW64 ~ (master)

$ git config --global mergetool.prompt false

HP@HP MINGW64 ~ (master)

$ git clone https://gitlab.com/dhatchayini/git-handson3.git

Cloning into 'git-handson3'...

remote: Enumerating objects: 3, done.

remote: Counting objects: 100% (3/3), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

Receiving objects: 100% (3/3), done.

HP@HP MINGW64 ~ (master)

$ cd git-handson3

HP@HP MINGW64 ~/git-handson3 (main)

$ git checkout -b GitNewBranch

Switched to a new branch 'GitNewBranch'

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ git branch

\* GitNewBranch

main

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ git branch -a

\* GitNewBranch

main

remotes/origin/HEAD -> origin/main

remotes/origin/main

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ echo "This is a test file for the branch" > file1.txt

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ git add file1.txt

warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ git commit -m "Add file1.txt on GitNewBranch"

[GitNewBranch cdd4d16] Add file1.txt on GitNewBranch

1 file changed, 1 insertion(+)

create mode 100644 file1.txt

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ git branch

\* GitNewBranch

main

HP@HP MINGW64 ~/git-handson3 (GitNewBranch)

$ git branch -a

\* GitNewBranch

main

remotes/origin/GitNewBranch

remotes/origin/HEAD -> origin/main

remotes/origin/main

t push origin main

Enumerating objects: 6, done.

Counting objects: 100% (6/6), done.

Delta compression using up to 4 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (4/4), 499 bytes | 99.00 KiB/s, done.

Total 4 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)

To https://gitlab.com/dhatchayini/git-handson3.git

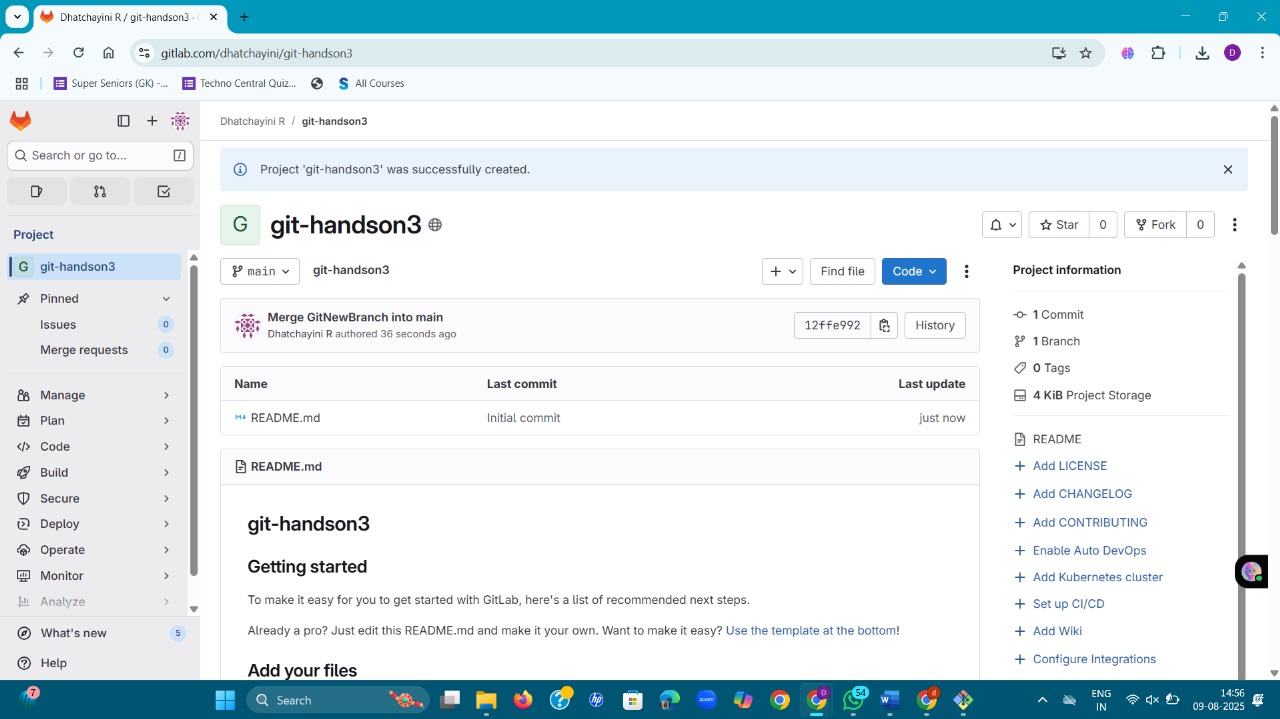
be0c12c..12ffe99 main -> main

HP@HP MINGW64 ~/git-handson3 (main)

$ git push origin --delete GitNewBranch

To https://gitlab.com/dhatchayini/git-handson3.git

- [deleted] GitNewBranch



**EXERCISE - 4 :**

**1.Explain how to resolve the conflict during merge.**

**1. Identify the conflict**

After running:

git merge <branch>

You might see:

CONFLICT (content): Merge conflict in hello.xml

Automatic merge failed; fix conflicts and then commit the result.

Check the status:

git status

**2. Open the conflicted file**

**Inside, you’ll see conflict markers inserted by Git:**

<<<<<<< HEAD

<greeting>Master version</greeting>

=======

<greeting>Branch updated</greeting>

>>>>>>> GitWork

**3. Decide the final content**

You have three options:

1. Keep your version (remove the other branch’s changes).
2. Keep their version (replace yours with theirs).
3. Combine both changes in a way that makes sense.

Example combined resolution:

<greeting>Master and Branch merged together</greeting>

**4. Remove conflict markers**

Delete all lines with:

<<<<<<<

=======

>>>>>>>

so that the file contains only valid content.

**5. Mark the conflict as resolved**

git add hello.xml

**6. Commit the merge**

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

HP@HP MINGW64 ~ (master)

$ mkdir Git-handson4

HP@HP MINGW64 ~ (master)

$ cd Git-handson4

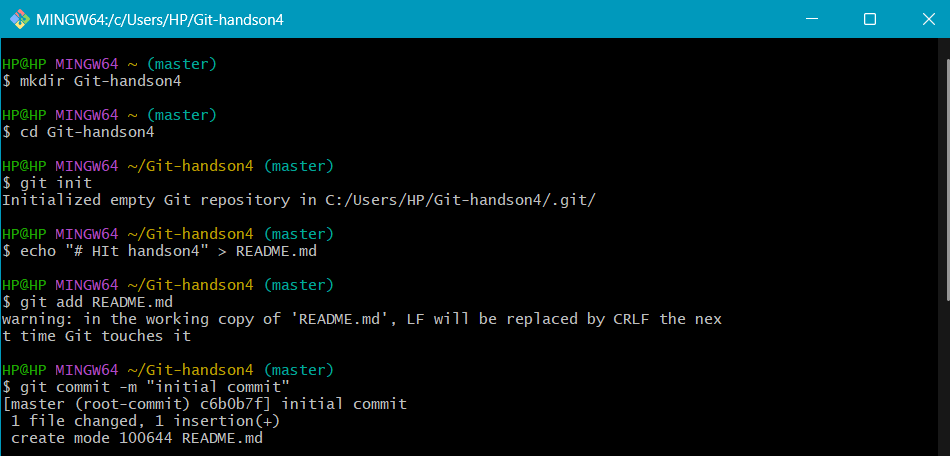
HP@HP MINGW64 ~/Git-handson4 (master)

$ git init

Initialized empty Git repository in C:/Users/HP/Git-handson4/.git/

HP@HP MINGW64 ~/Git-handson4 (master)

$ echo "# HIt handson4" > README.md



HP@HP MINGW64 ~/Git-handson4 (master)

$ git add README.md

warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

HP@HP MINGW64 ~/Git-handson4 (master)

$ git commit -m "initial commit"

[master (root-commit) c6b0b7f] initial commit

1 file changed, 1 insertion(+)

create mode 100644 README.md

HP@HP MINGW64 ~/Git-handson4 (master)

$ git status

On branch master

nothing to commit, working tree clean

HP@HP MINGW64 ~/Git-handson4 (master)

$ git checkout -b GitWork

Switched to a new branch 'GitWork'

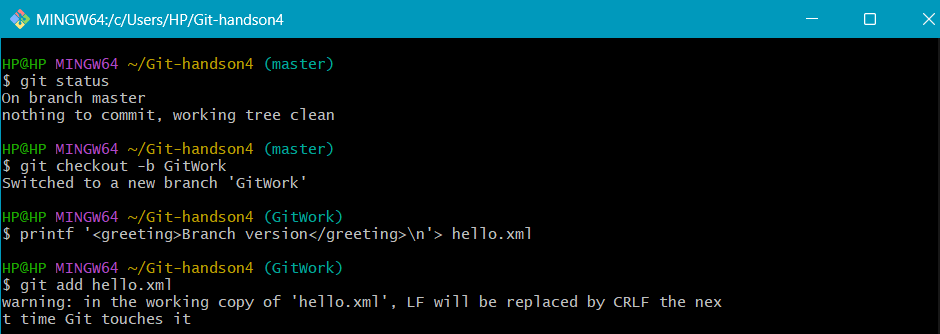
HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ printf '<greeting>Branch version</greeting>\n'> hello.xml

HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ git add hello.xml

warning: in the working copy of 'hello.xml', LF will be replaced by CRLF the next time Git touches it



HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ git commit -m "Add hello.xml on GitWork Branch"

[GitWork ae2bd50] Add hello.xml on GitWork Branch

1 file changed, 1 insertion(+)

create mode 100644 hello.xml

HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ printf '<greeting>Branch updated</greeting>\n'>hello.xml

HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ git status

On branch GitWork

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: hello.xml

no changes added to commit (use "git add" and/or "git commit -a")

HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ git add hello.xml

warning: in the working copy of 'hello.xml', LF will be replaced by CRLF the next time Git touches it

HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ git commit -m "Update hello.xml on GitWork branch"

[GitWork f54644c] Update hello.xml on GitWork branch

1 file changed, 1 insertion(+), 1 deletion(-)

HP@HP MINGW64 ~/Git-handson4 (GitWork)

$ git checkout master

Switched to branch 'master'

HP@HP MINGW64 ~/Git-handson4 (master)

$ printf '<greeting>Master version</greeting>\n' > hello.xml

HP@HP MINGW64 ~/Git-handson4 (master)

$ git add hello.xml

warning: in the working copy of 'hello.xml', LF will be replaced by CRLF the next time Git touches it

HP@HP MINGW64 ~/Git-handson4 (master)

$ git commit -m "Add hello.xml on master with different content"

[master 5f57485] Add hello.xml on master with different content

1 file changed, 1 insertion(+)

create mode 100644 hello.xml

HP@HP MINGW64 ~/Git-handson4 (master)

$ git log --oneline --graph --decorate --all

\* 5f57485 (**HEAD** -> **master**) Add hello.xml on master with different content

| \* f54644c (**GitWork**) Update hello.xml on GitWork branch

| \* ae2bd50 Add hello.xml on GitWork Branch

|/

\* c6b0b7f initial commit

HP@HP MINGW64 ~/Git-handson4 (master)

$ git diff master..GitWork -- hello.xml

**diff --git a/hello.xml b/hello.xml**

**index 9d27880..73f4bd5 100644**

**--- a/hello.xml**

**+++ b/hello.xml**

@@ -1 +1 @@

-<greeting>Master version</greeting>

+<greeting>Branch updated</greeting>

HP@HP MINGW64 ~/Git-handson4 (master)

$ git config --global diff.tool p4merge

HP@HP MINGW64 ~/Git-handson4 (master)

$ git config --global difftool.p4merge.cmd "\"C:/Program Files/Perforce/p4merge.exe\" \"$LOCAL\" \"$REMOTE\""

HP@HP MINGW64 ~/Git-handson4 (master)

$ git config --global merge.tool p4merge

HP@HP MINGW64 ~/Git-handson4 (master)

$ git config --global mergetool.p4merge.cmd "\"C:/Program Files/Perforce/p4merge.exe\" \"$BASE\" \"$LOCAL\" \"$REMOTE\" \"$MERGED\""

HP@HP MINGW64 ~/Git-handson4 (master)

$ git config --global mergetool.prompt false

HP@HP MINGW64 ~/Git-handson4 (master)

$ git checkout master

Already on 'master'

HP@HP MINGW64 ~/Git-handson4 (master)

$ git merge GitWork

Auto-merging hello.xml

CONFLICT (add/add): Merge conflict in hello.xml

Automatic merge failed; fix conflicts and then commit the result.

HP@HP MINGW64 ~/Git-handson4 (master|MERGING)

$ ^C

HP@HP MINGW64 ~/Git-handson4 (master|MERGING)

$ git status

On branch master

You have unmerged paths.

(fix conflicts and run "git commit")

(use "git merge --abort" to abort the merge)

Unmerged paths:

(use "git add <file>..." to mark resolution)

both added: hello.xml

no changes added to commit (use "git add" and/or "git commit -a")

HP@HP MINGW64 ~/Git-handson4 (master|MERGING)

$ git add hello.xml

HP@HP MINGW64 ~/Git-handson4 (master|MERGING)

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

[master 9dff145] Resolved merge conflict in hello.xml

HP@HP MINGW64 ~/Git-handson4 (master)

$ echo "\*.orig" >> .gitignore

HP@HP MINGW64 ~/Git-handson4 (master)

$ git add .gitignore

warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

HP@HP MINGW64 ~/Git-handson4 (master)

$ git commit -m "Ignore merge backup files"

[master 43976d5] Ignore merge backup files

1 file changed, 1 insertion(+)

create mode 100644 .gitignore

HP@HP MINGW64 ~/Git-handson4 (master)

$ git branch

GitWork

\* master

HP@HP MINGW64 ~/Git-handson4 (master)

$ git branch -d GitWork

Deleted branch GitWork (was f54644c).

HP@HP MINGW64 ~/Git-handson4 (master)

$ git log --oneline --graph --decorate

\* 43976d5 (**HEAD** -> **master**) Ignore merge backup files

\* 9dff145 Resolved merge conflict in hello.xml

|\

| \* f54644c Update hello.xml on GitWork branch

| \* ae2bd50 Add hello.xml on GitWork Branch

\* | 5f57485 Add hello.xml on master with different content

|/

\* c6b0b7f initial commit

HP@HP MINGW64 ~/Git-handson4 (master)

$ git remote add origin https://gitlab.com/dhatchayini/Git-handson4.git

HP@HP MINGW64 ~/Git-handson4 (master)

$ git push -u origin master

Enumerating objects: 18, done.

Counting objects: 100% (18/18), done.

Delta compression using up to 4 threads

Compressing objects: 100% (12/12), done.

Writing objects: 100% (18/18), 1.70 KiB | 45.00 KiB/s, done.

Total 18 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)

remote:

remote: To create a merge request for master, visit:

remote: https://gitlab.com/dhatchayini/git-handson4/-/merge\_requests/new?merge\_request%5Bsource\_branch%5D=master

remote:

To https://gitlab.com/dhatchayini/Git-handson4.git

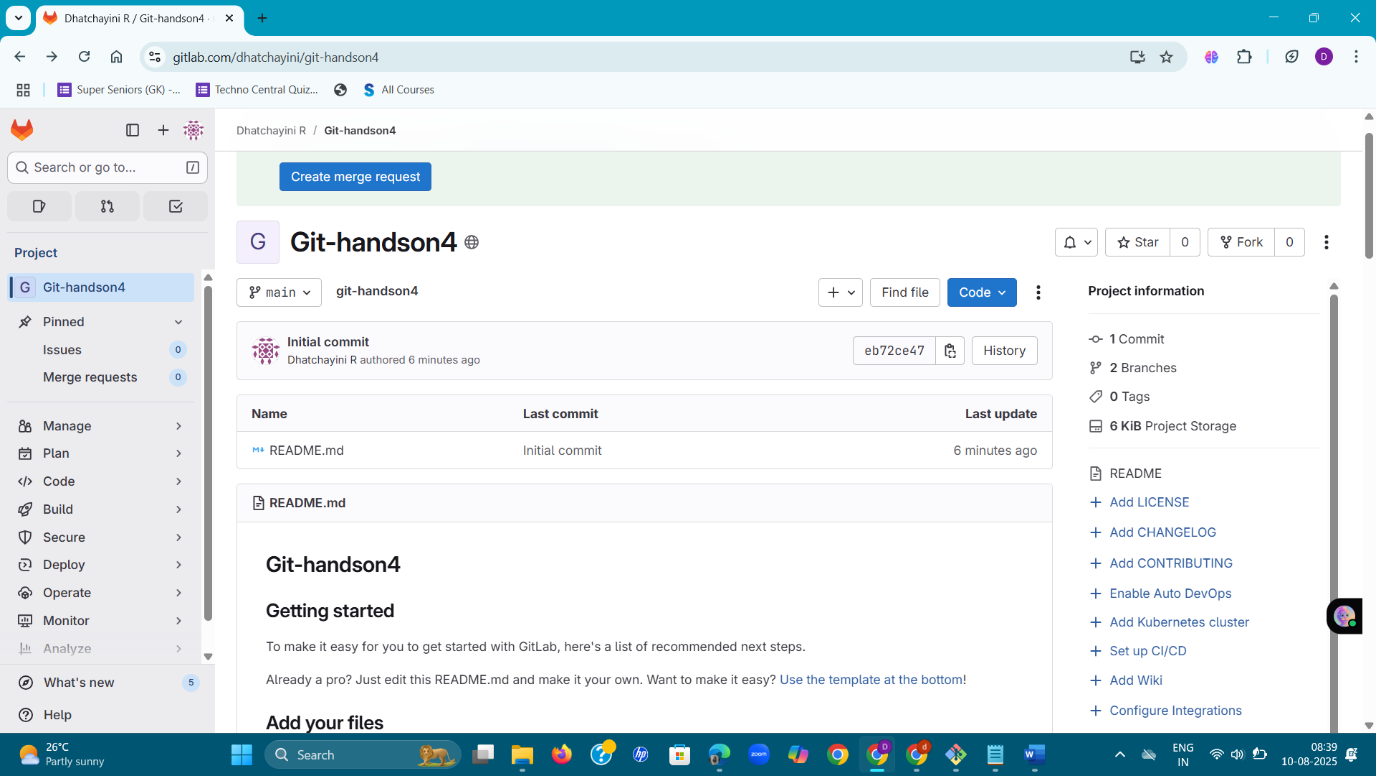
\* [new branch] master -> master

branch 'master' set up to track 'origin/master'.

HP@HP MINGW64 ~/Git-handson4 (master)

$ git push

Everything up-to-date



**EXERCISE - 5 :**

**Explain how to clean up and push back to remote Git:**

**1. Clean up your working directory**

Cleaning up means making sure your branch is in a *clean state* — no uncommitted changes, no untracked junk files.

Check status

git status

If you see nothing to commit, working tree clean → you’re good.  
If you see changes, you have two options:

Option A — Keep the changes

git add . git commit -m "Save work before push"

**2. Pull latest updates from remote**

Make sure your local branch is up to date before pushing.

git pull --rebase origin <branch-name>

Example:

git pull --rebase origin master

**3. Push your changes to remote**

git push origin <branch-name>

**Example:**

git push origin Git-T03-HOL\_002

If it’s the first push of a new branch:

git push -u origin Git-T03-HOL\_002

(-u sets the remote branch as the default for future pushes.)

**4. Verify on remote**

* Go to your repository in GitLab/GitHub.
* Check that your commit appears in the branch.

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git init

Initialized empty Git repository in C:/Users/HP/Documents/git-handson5/.git/

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git branch --show-current

master

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git status

On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git status --porcelain

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git add .

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git commit -m "Save work for Git-T03-HOL\_002"

On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git status --porcelain

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git add .

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git commit -m "Save work for Git-T03-HOL\_002"

On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git stash push -m "WIP before pull"

You do not have the initial commit yet

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git branch -a

HP@HP MINGW64 ~/Documents/git-handson5 (master)

$ git branch -r

HP@HP MINGW64 ~/Documents/git-handson5 (Git-T03-HOL\_002)

$ git branch

HP@HP MINGW64 ~/Documents/git-handson5 (Git-T03-HOL\_002)

$ git checkout -b Git-T03-HOL\_002

Switched to a new branch 'Git-T03-HOL\_002'

HP@HP MINGW64 ~/Documents/git-handson5 (Git-T03-HOL\_002)

$ echo "Hello Git HOL" > test.txt

HP@HP MINGW64 ~/Documents/git-handson5 (Git-T03-HOL\_002)

$ git add test.txt

warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next time Git touches it

HP@HP MINGW64 ~/Documents/git-handson5 (Git-T03-HOL\_002)

$ git commit -m "Initial commit for Git-T03-HOL\_002"

[Git-T03-HOL\_002 (root-commit) f0a41be] Initial commit for Git-T03-HOL\_002

1 file changed, 1 insertion(+)

create mode 100644 test.txt

HP@HP MINGW64 ~/Documents/git-handson5 (Git-T03-HOL\_002)

$ git push -u origin Git-T03-HOL\_002

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Writing objects: 100% (3/3), 249 bytes | 249.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

remote:

remote: To create a merge request for Git-T03-HOL\_002, visit:

remote: https://gitlab.com/dhatchayini/git-handson5/-/merge\_requests/new?merge\_request%5Bsource\_branch%5D=Git-T03-HOL\_002

remote:

To https://gitlab.com/dhatchayini/git-handson5.git

\* [new branch] Git-T03-HOL\_002 -> Git-T03-HOL\_002

branch 'Git-T03-HOL\_002' set up to track 'origin/Git-T03-HOL\_002'.

