# Git-Hol 3

* Explain branching and merging

Branching

Branching allows you to create a new line of development within your Git repository. It allows you to:

Add a new feature or a bug fixwithout changing the base codebase(usually the'main ' or'master ' branch).

Each branch is akin to referring to a snapshot of your modifications.

Example:

You start at'main ' and create a new branch:

'bash '

git checkout -b feature/login-page

Now all your commits will be kept on'feature/login-page '.

Merging

Integration merges the modifications of one branch to another.

Typically, once you have completed building your feature, youmerge it backinto the master branch.

Git tries toauto-resolvethe changes; otherwise, you are presented with amerge conflictto manually resolve.

Example:

bash

/bin/bash

git checkout main

git merge feature/login-page

* Explain about creating a branch request in GitLab

In GitLab, "branch request" is not a formal term, but it is most commonly used to describe:

Checking out a new branchfor your feature/bugfix from the GitLab UI or terminal.

Via GitLab UI:

1. Go to your repository on GitLab.

2. Click onRepository > Branches .

3. PressNew branch .

4. Enter thebranch nameand name it following a branch like'main '.

5. ClickCreate branch .

Via Git (terminal) :

bash

git checkout -b feature/branch-name

git push origin feature/branch-name

* Explain about creating a merge request in GitLab

A Merge Request or an MR in GitLab is utilized to:

Propose to apply changes from one branch (e.g.,'feature/login-page ') to another (e.g.,'main ').

It's used forcode review ,discussions , andapprovalbefore merging.

Step-by-step process to form a Merge Request in GitLab:

1. Stage your version in your feature branch:

'bash '

git push origin feature/branch-name

2. Go to your project on GitLab.

3. GitLab will prompt regularly:"Create merge request"for fresh branches.

4. Click it or go to.

Merge Requests > New Merge Request5. Choose: Source branch:'feature/branch-name ' Target branch:'main ' 6. Add atitle, description , includereviewers , and optionally addlabels or milestones . 7. ClickSubmit merge request .Once you're approved, you or any user canmergeit into the target branch from the GitLab UI

MINGW64 ~/GitDemo (main)

$ git branch GitNewBranch

fatal: a branch named 'GitNewBranch' already exists

MINGW64 ~/GitDemo (main)

$ git branch -a

GitNewBranch

\* main

MINGW64 ~/GitDemo (main)

$ git checkout GitNewBranch

Switched to branch 'GitNewBranch'

MINGW64 ~/GitDemo (GitNewBranch)

$ echo "Changes in GitNewBranch" > branchfile.txt

git add branchfile.txt

git commit -m "Add branchfile.txt in GitNewBranch"

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

On branch GitNewBranch

nothing to commit, working tree clean

MINGW64 ~/GitDemo (GitNewBranch)

$ git status

On branch GitNewBranch

nothing to commit, working tree clean

MINGW64 ~/GitDemo (GitNewBranch)

$ git checkout master

error: pathspec 'master' did not match any file(s) known to git

MINGW64 ~/GitDemo (GitNewBranch)

$ git checkout main

Switched to branch 'main'

MINGW64 ~/GitDemo (main)

$

git diff main GitNewBranch

**diff --git a/branchfile.txt b/branchfile.txt**

**new file mode 100644**

**index 0000000..817435c**

**--- /dev/null**

**+++ b/branchfile.txt**

@@ -0,0 +1 @@

+Changes in GitNewBranch

MINGW64 ~/GitDemo (main)

$ git difftool main GitNewBranch

Viewing (1/1): 'branchfile.txt'

Launch 'p4merge' [Y/n]? y

The diff tool p4merge is not available as 'C:/Program Files/Perforce/p4merge.exe'

fatal: external diff died, stopping at branchfile.txt

MINGW64 ~/GitDemo (main)

$ git merge GitNewBranch

Updating 64a0f21..44e688b

Fast-forward

branchfile.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 branchfile.txt

MINGW64 ~/GitDemo (main)

$ git mergetool

No files need merging

MINGW64 ~/GitDemo (main)

$

MINGW64 ~/GitDemo (main)

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

\* 44e688b (**HEAD** -> **main**, **GitNewBranch**) Add branchfile.txt in GitNewBranch

\* 64a0f21 Add .gitignore to exclude log files and folders

\* 26000a4 hi hello

MINGW64 ~/GitDemo (main)

$ git branch -d GitNewBranch

Deleted branch GitNewBranch (was 44e688b).

MINGW64 ~/GitDemo (main)

$ git status

On branch main

nothing to commit, working tree clean

MINGW64 ~/GitDemo (main)