Step-by-step Guide Week 4 Daily Task

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops$ mkdir my-git-project
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops$ cd my-git-project
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git init
```

- You created a new directory called `my-git-project`.
- You moved into the newly created directory.
- You initialized a new Git repository in the directory, making it a version-controlled project.

```
Initialized empty Git repository in /mnt/d/Zionet Courses/Zionet Devops/my-git-project/.git/
```

תיקיית קבצים	27/05/2025 10:56	.git

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch -M main
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ echo "Initial content" > README.md
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add README.md
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git commit -m "Initial commit"
[main (root-commit) 33f4ca1] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch
* main
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$
```

- This renames the current branch (default `master`) to `main`. It's a common convention to use `main` as the default branch.
- This creates a file named `README.md` with the text **"Initial content"** inside.
- This stages the `README.md` file, telling Git you want to include it in the next commit.
- This commits the staged changes to the repository with the message **"Initial commit"**.
- This shows the current branches in the repository. The `*` indicates the active branch, which is `main`.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch feature-a
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch feature-b
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch
  feature-a
  feature-b
* main
```

- This command creates a new branch named feature-a. The branch is created but not yet checked out (you're still on main).
- Similarly, this command creates another branch named `feature-b`.
- This command lists all available branches:
- `main` (with an asterisk `*` indicating it is the currently active branch).

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-a
Switched to branch 'feature-a'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch
* feature-a
  feature-b
  main
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-b
Switched to branch 'feature-b'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git branch
  feature-a
* feature-b
  main
```

- git switch feature-a:
  This command moves you to the `feature-a` branch. The message confirms:
  `Switched to branch 'feature-a'.`
- git branch:
  The `*` indicates the current branch. You are now on `feature-a`.
- git switch feature-b
  This moves you to the `feature-b` branch. The message confirms:
  `Switched to branch 'feature-b'.`
- git branch:
  Again, the `*` indicates you are now on `feature-b`.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-a
Switched to branch 'feature-a'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ echo "Hello from feature-a" > greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git commit -m "Add greetings.txt in feature-a"
[feature-a da1535a] Add greetings.txt in feature-a
 1 file changed, 1 insertion(+)
 create mode 100644 greetings.txt
```

- git switch feature-a:
  You moved to the `feature-a` branch to make changes there.
- echo "Hello from feature-a" > greetings.txt:
  This command creates a new file called `greetings.txt` and writes the text **"Hello from feature-a"** into it.
- git add greetings.txt:
  You staged the file for the next commit. Git now knows you want to include this file in your commit.
- git commit -m "Add greetings.txt in feature-a"
  This commits the change with a descriptive message. The file `greetings.txt` is now part of the `feature-a` branch.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-b
Switched to branch 'feature-b'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ echo "Hello from feature-b" > greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git commit -m "Add greetings.txt in feature-b"
[feature-b aa7b4bf] Add greetings.txt in feature-b
 1 file changed, 1 insertion(+)
 create mode 100644 greetings.txt
```

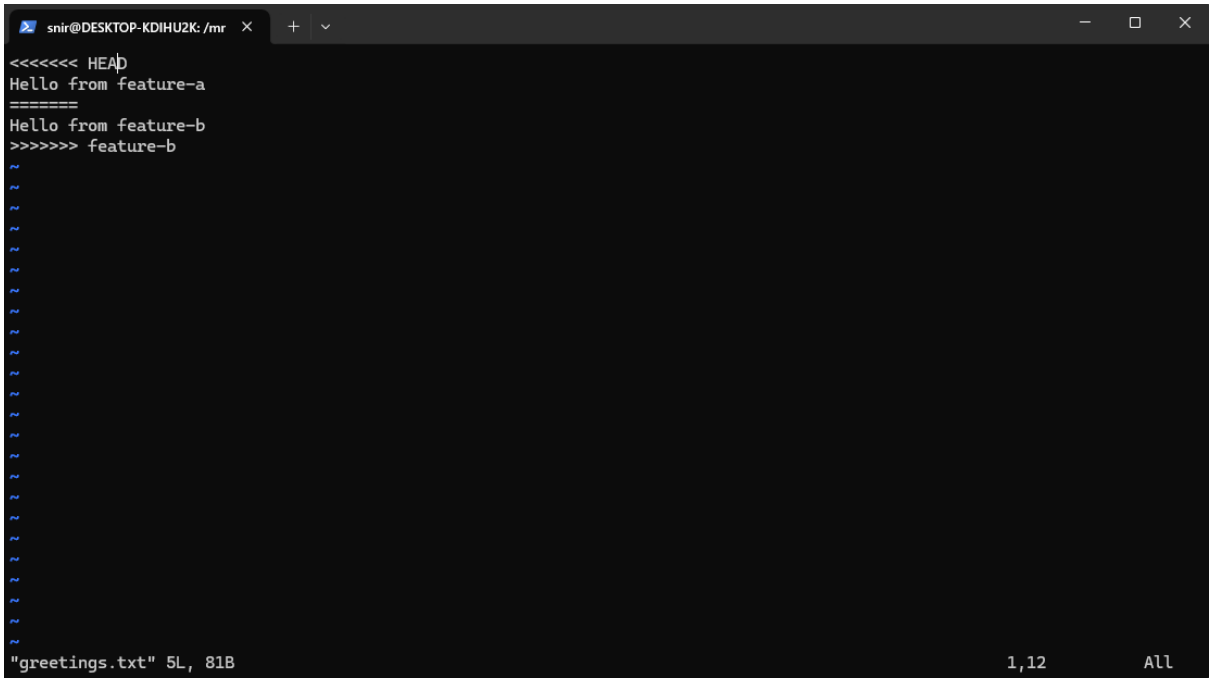
- git switch feature-a:
  You moved to the `feature-a` branch to make changes there.
- echo "Hello from feature-a" > greetings.txt:
  This command creates a new file called `greetings.txt` and writes the text **"Hello from feature-a"** into it.
- git add greetings.txt:
  You staged the file for the next commit. Git now knows you want to include this file in your commit.
- git commit -m "Add greetings.txt in feature-a"
  This commits the change with a descriptive message. The file `greetings.txt` is now part of the `feature-a` branch.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-a
Switched to branch 'feature-a'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git merge feature-b
Auto-merging greetings.txt
CONFLICT (add/add): Merge conflict in greetings.txt
Automatic merge failed; fix conflicts and then commit the result.
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ cat greetings.txt
<<<<<<< HEAD
Hello from feature-a
=======
Hello from feature-b
>>>>>>> feature-b
```

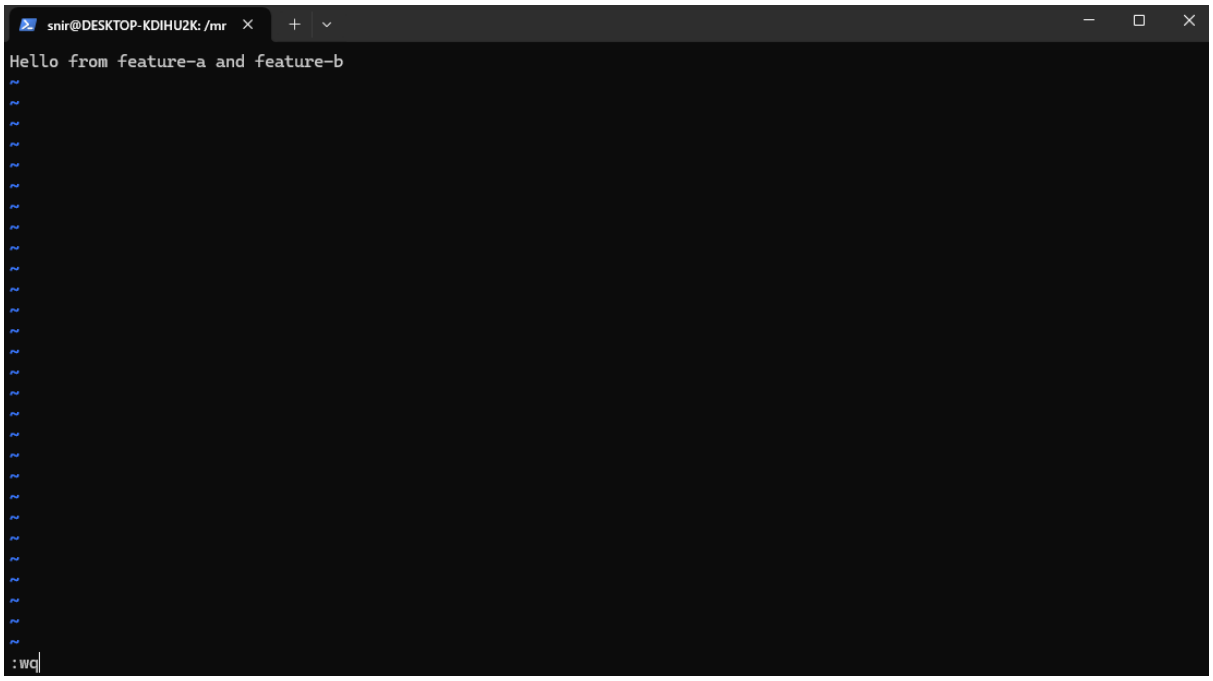
- git switch feature-a:
  You switched to `feature-a` to prepare for the merge.
- git merge feature-b:
  You tried to merge changes from `feature-b` into `feature-a`.
  Git tried to auto-merge the files, but **a conflict happened** because both branches modified the same file (`greetings.txt`).
- `<<<<<<< HEAD`
  shows the content from your current branch (`feature-a`).
- `=======`
  separates the two versions.
- `>>>>>>> feature-b` shows the content from `feature-b`.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ vim greetings.txt
```

- vim greetings.txt:
  You opened the file `greetings.txt` using the **Vim** text editor.
  Vim allows you to manually edit the file and resolve the merge conflict.



- **Remove** the conflict markers (<<<<<<<, =======, >>>>>>>).
- Decide what content you want to keep
- For example:



- Exit Insert mode:
  Press `Esc`.
- Save and Exit:
  Type :wq
- press `Enter`.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ cat greetings.txt
Hello from feature-a and feature-b
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git status
On branch feature-a
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:      greetings.txt

no changes added to commit (use "git add" and/or "git commit -a")
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git commit -m "Resolved merge conflicts and merged feature-b into feature-a"
[feature-a be6d59b] Resolved merge conflicts and merged feature-b into feature-a
```

- cat greetings.txt:
  This shows the merged content
- git status:
  Git reminds you that there are unresolved paths (`greetings.txt`) and you need to **add** the file to mark the conflict as resolved.
- git add greetings.txt:
  This marks the file as resolved and stages it for commit.
- git commit -m "Resolved merge conflicts and merged feature-b into feature-a"
  This creates a commit that finalizes the merge of `feature-b` into `feature-a` after resolving the conflict.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git log --oneline --graph --all
*   be6d59b (HEAD -> feature-a) Resolved merge conflicts and merged feature-b into feature-a
|\
| * aa7b4bf (feature-b) Add greetings.txt in feature-b
* | da1535a Add greetings.txt in feature-a
|/
* 33f4ca1 (main) Initial commit
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-a
Already on 'feature-a'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git rebase main
Auto-merging greetings.txt
CONFLICT (add/add): Merge conflict in greetings.txt
error: could not apply aa7b4bf... Add greetings.txt in feature-b
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
Could not apply aa7b4bf... Add greetings.txt in feature-b
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ cat greetings.txt
<<<<<<< HEAD
Hello from feature-a
=======
Hello from feature-b
>>>>>>> aa7b4bf (Add greetings.txt in feature-b)
```

- git log --oneline --graph –all:
  Shows a visual tree of commits and Helps you see how branches (`main`, `feature-a`, `feature-b`) relate to each other.
- git switch feature-a:
  You prepared to rebase `feature-a` onto `main`.
- git rebase main:
  You tried to replay the commits of `feature-a` on top of `main`.
  But... a **conflict happened** again in `greetings.txt` because both branches modified the same file.
- Resolved that with vm again

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add .
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git rebase --continue
```

- git add .
  After editing `greetings.txt` to resolve the conflict, you added the changes to the staging area.
- git rebase –continue:
  This command tells Git: "I’ve fixed the conflict, let’s proceed with the rebase."

```
GNU nano 6.2                    /mnt/d/Zionet Courses/Zionet Devops/my-git-project/.git/COMMIT_EDITMSG *
Add greetings.txt in feature-a and feature-b

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# interactive rebase in progress; onto 33f4ca1
# Last commands done (2 commands done):
#    pick da1535a Add greetings.txt in feature-a
#    pick aa7b4bf Add greetings.txt in feature-b
# No commands remaining.
# You are currently rebasing branch 'feature-a' on '33f4ca1'.
#
# Changes to be committed:
#       modified:   greetings.txt
#

^G Help    ^O Write Out  ^W Where Is   ^K Cut       ^T Execute   ^C Location  M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous  ^B Back      ^A Prev Word
^X Exit    ^R Read File  ^\ Replace    ^U Paste     ^J Justify   ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next      ^F Forward   ^→ Next Word
```

- You’re inside **GNU Nano** editor (or Vim-like editor) editing the **commit message** during the rebase process.
- Git paused the rebase to allow you to **write a commit message** for the rebased commit.
- You can **edit the commit message** as you like (or leave it as is).
- `Ctrl + O` → to save, `Enter` → to confirm the file name.
- `Ctrl + X` → to exit the editor

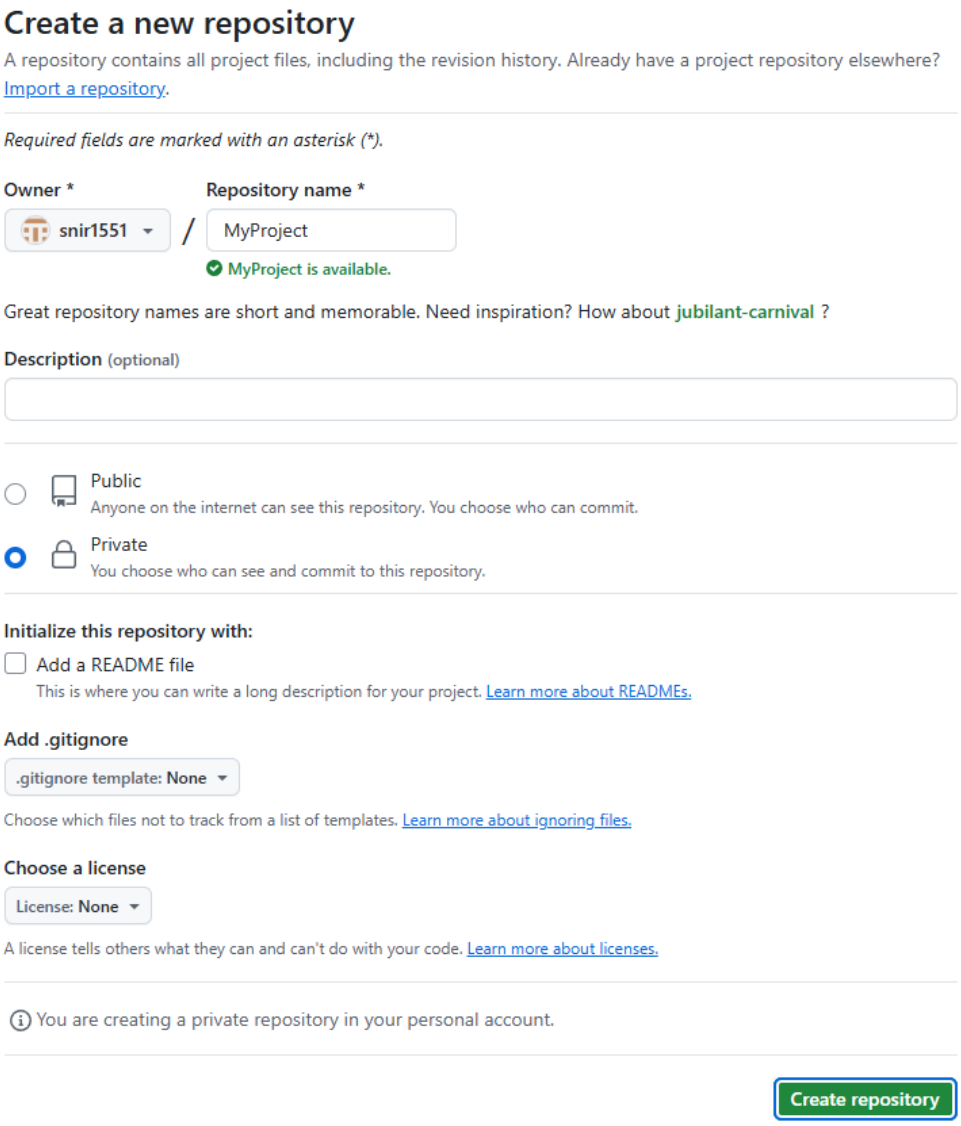
```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git log --oneline --graph --all
* 9c817f3 (HEAD -> feature-a) Add greetings.txt in feature-a and feature-b
* da1535a Add greetings.txt in feature-a
| * aa7b4bf (feature-b) Add greetings.txt in feature-b
|/
* 33f4ca1 (main) Initial commit
```

- `feature-a` has a **combined commit**:
  `9c817f3 Add greetings.txt in feature-a and feature-b`
  This is your **rebased commit** that includes the changes from both `feature-a` and `feature-b`.
- The old commits:
  da1535a (from feature-a) and aa7b4bf (from feature-b) were part of the history, but after the rebase, feature-a has the updated combined commit on top.
- The `main` branch still has the initial commit (`33f4fca`).

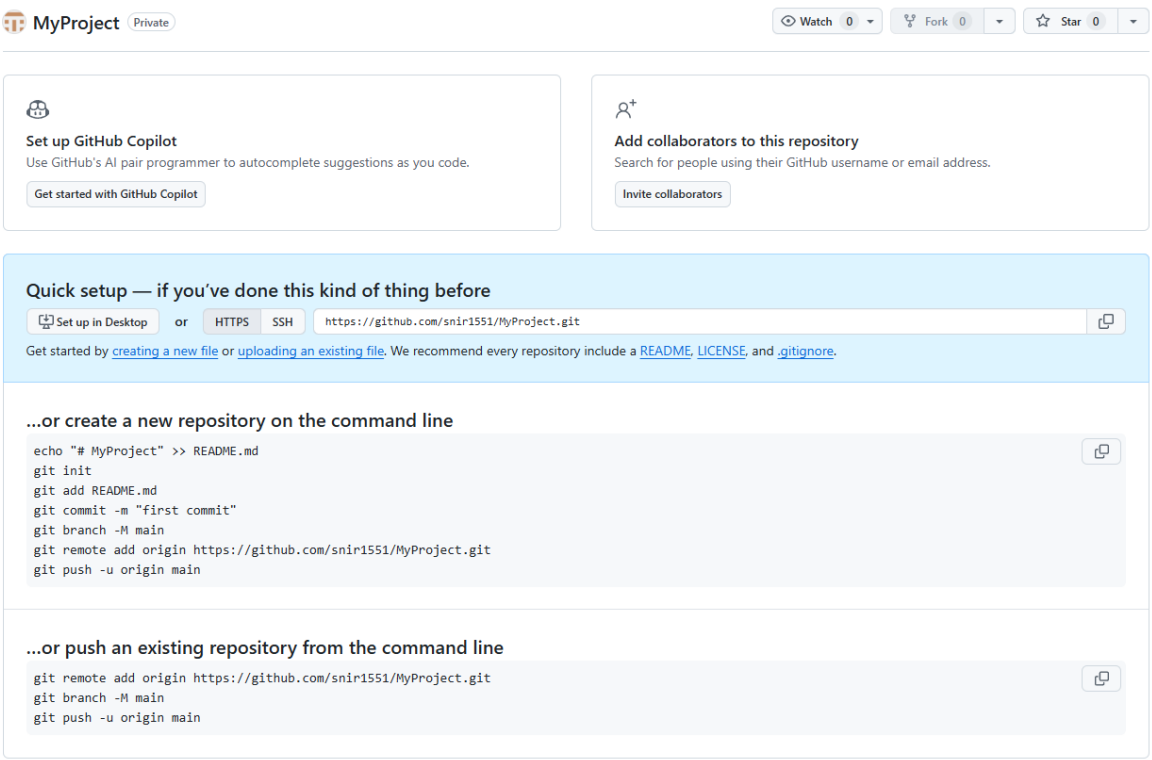
```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git log feature-b --oneline
aa7b4bf (feature-b) Add greetings.txt in feature-b
33f4ca1 (main) Initial commit
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch main
Switched to branch 'main'
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git cherry-pick aa7b4bf
[main 45ab58f] Add greetings.txt in feature-b
 Date: Tue May 27 11:08:53 2025 +0300
 1 file changed, 1 insertion(+)
 create mode 100644 greetings.txt
```

- git log feature-b –oneline:
  This shows the log of `feature-b`. The commit hash `aa7b4bf` represents the commit
- git switch main:
  You switched to the `main` branch to apply the commit there.
- git cherry-pick aa7b4bf:
  This copies the specific commit `aa7b4bf` from `feature-b` and applies it to the current branch (`main`).

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git log --oneline --graph --all
* 45ab58f (HEAD -> main) Add greetings.txt in feature-b
| * 9c817f3 (feature-a) Add greetings.txt in feature-a and feature-b
| * da1535a Add greetings.txt in feature-a
|/
| * aa7b4bf (feature-b) Add greetings.txt in feature-b
|/
* 33f4ca1 Initial commit
```



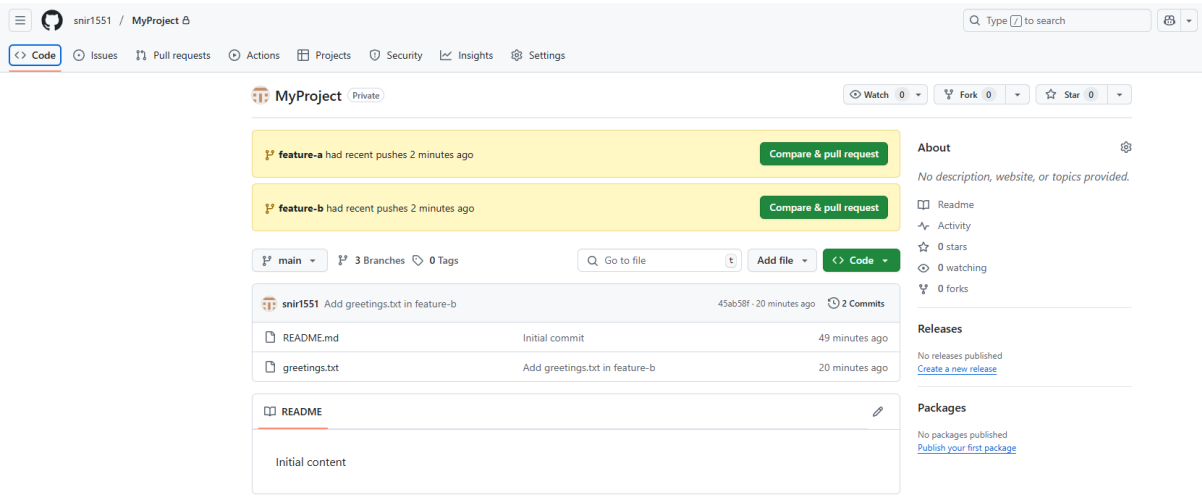
- **Owner**: The repository will be created under your GitHub account: snir1551
- **Repository Name**: You named your repository: `MyProject`.
- **Description (Optional)**: This is an optional field where you can write a short description of your project.
- **Visibility**: You selected Private:
- **Initialize the Repository (Optional)**:
- **Final Step**: Click the green **Create repository** button to create the repository on GitHub.



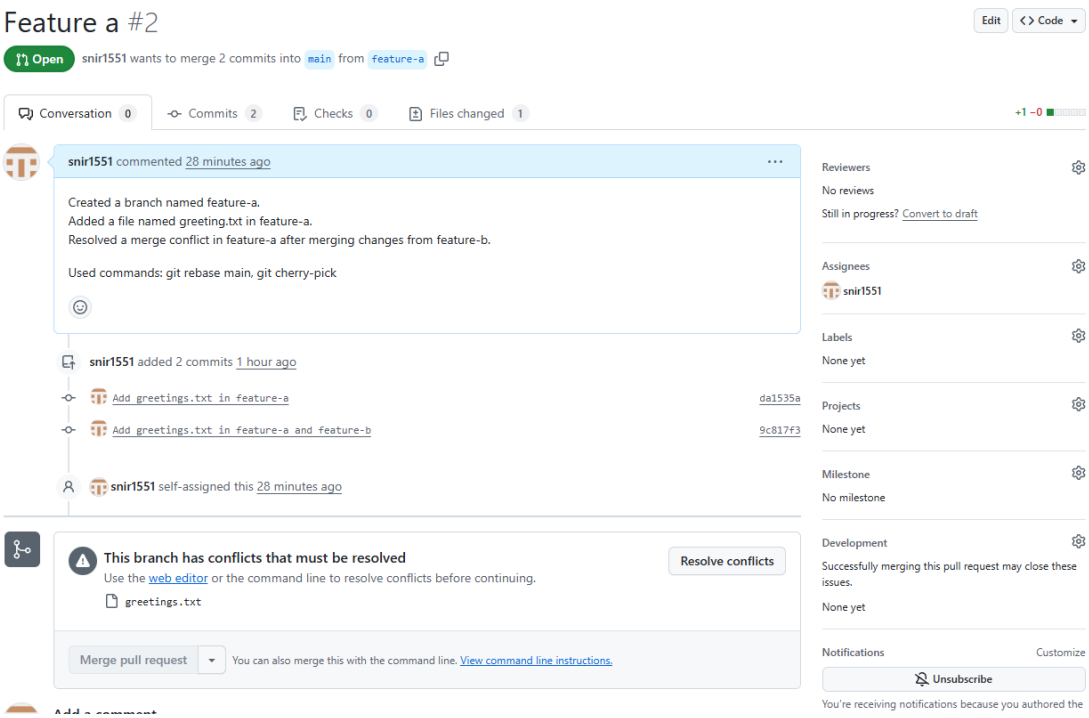
- After creating your new repository `MyProject` on GitHub, this is the **setup page** you see

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git remote set-url origin git@github.com:snir1551/MyProject.git
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git push -u origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 20 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 510 bytes | 25.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:snir1551/MyProject.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git push -u origin feature-a
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 20 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 607 bytes | 28.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature-a' on GitHub by visiting:
remote:      https://github.com/snir1551/MyProject/pull/new/feature-a
remote:
To github.com:snir1551/MyProject.git
 * [new branch]      feature-a -> feature-a
Branch 'feature-a' set up to track remote branch 'feature-a' from 'origin'.
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git push -u origin feature-b
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 309 bytes | 23.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature-b' on GitHub by visiting:
remote:      https://github.com/snir1551/MyProject/pull/new/feature-b
remote:
To github.com:snir1551/MyProject.git
 * [new branch]      feature-b -> feature-b
Branch 'feature-b' set up to track remote branch 'feature-b' from 'origin'.
```

- git remote set-url origin git@github.com:snir1551/MyProject.git
  You set the **SSH remote URL** for your GitHub repository.
- git push -u origin main
  The `main` branch was pushed and connected to the `origin` remote.
- git push -u origin feature-a
  The `feature-a` branch was pushed, and GitHub gave you a link to create a Pull Request:
- git push -u origin feature-b
- The `feature-b` branch was pushed too, with a similar Pull Request link:



- Click the **`Compare & pull request`** button for `feature-a` and `feature-b` to create PRs and start a code review/merge process.



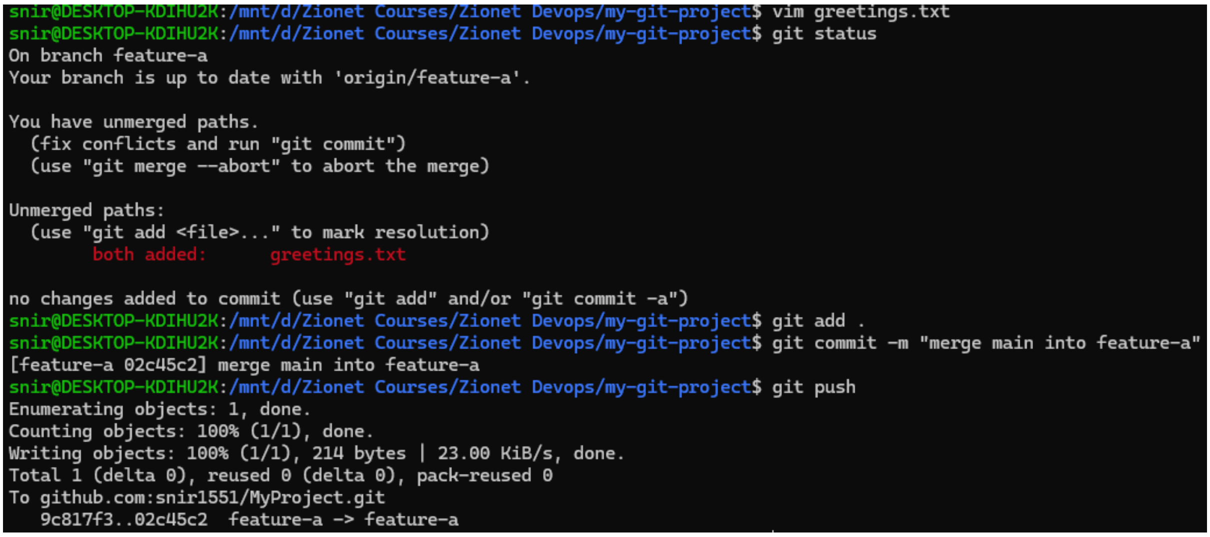
- You opened a **Pull Request** (PR) to merge the `feature-a` branch into `main`.
- Your comment in the PR clearly describes:
- Conflict Detected!
- You need to **resolve the conflict** before you can merge.
- **Use the CLI**:
  Pull the branch locally.
  Fix the conflict.
  Push the resolved file.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git merge main
Auto-merging greetings.txt
CONFLICT (add/add): Merge conflict in greetings.txt
Automatic merge failed; fix conflicts and then commit the result.
```

- git merge main:
  Git tried to **merge** the changes from `main` into your current branch, There's a **conflict** in the file `greetings.txt`.

```
Hello from feature-a and feature-b
~
~
~
:wq
```

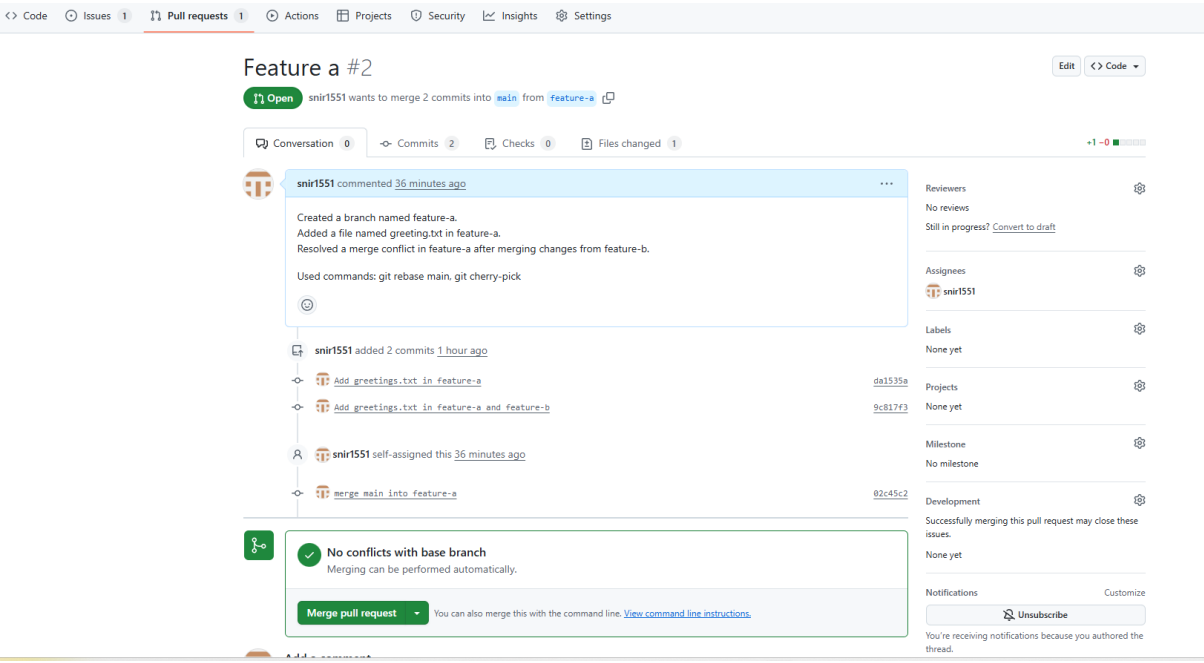
- Edit grettings.txt with vim like before



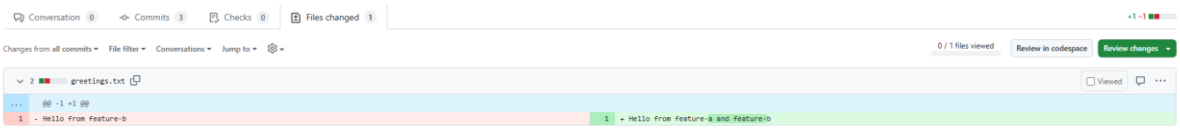
```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ vim greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git status
On branch feature-a
Your branch is up to date with 'origin/feature-a'.

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:      greetings.txt

no changes added to commit (use "git add" and/or "git commit -a")
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add .
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git commit -m "merge main into feature-a"
[feature-a 02c45c2] merge main into feature-a
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git push
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 214 bytes | 23.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:snir1551/MyProject.git
   9c817f3..02c45c2  feature-a -> feature-a
```

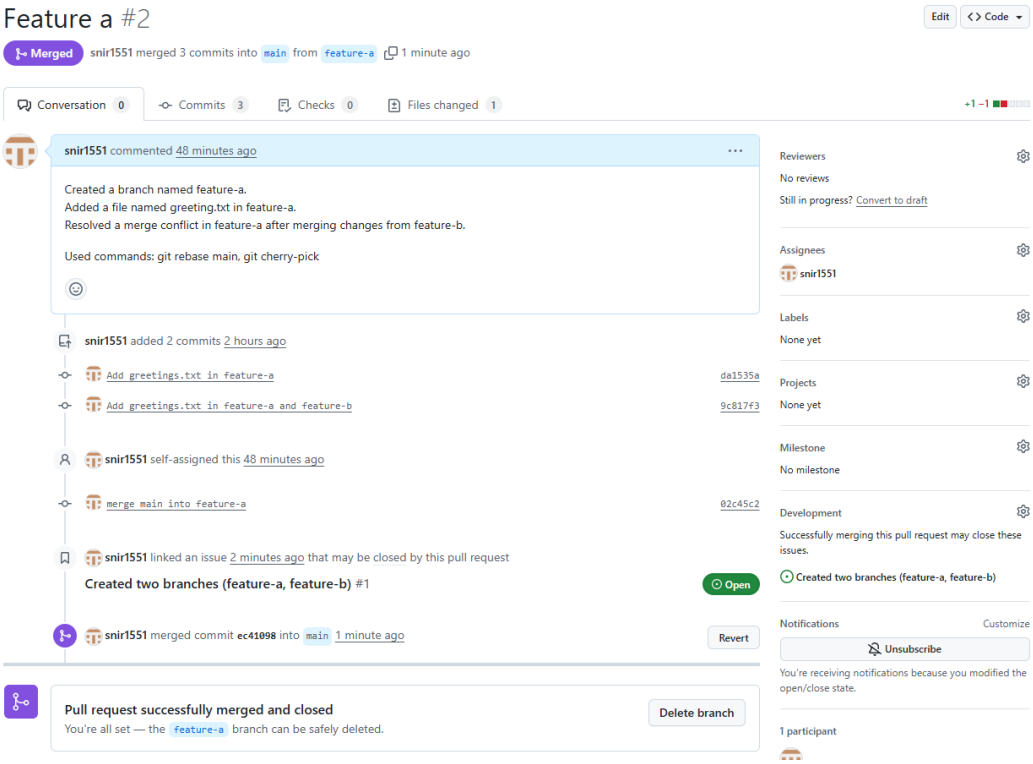
- vim greetings.txt:
  You opened greetings.txt in Vim and manually resolved the conflict by removing the conflict markers (e.g., <<<<<<<, =======, >>>>>>>).

- git status:
  Git confirmed that the conflict was resolved, and you were ready to stage and commit.
- git add .
  You staged the resolved file (`greetings.txt`).
- git commit -m "merge main into feature-a"
  You created a commit that finalizes the merge of `main` into `feature-a`.
- git push
  The `feature-a` branch is now **up-to-date on GitHub**, including the merge from `main`.



- **No conflicts** with the base branch (`main`).
- The **Merge pull request** button is now active and ready.





- Merged feature-a into main on GitHub.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git switch feature-a
Switched to branch 'feature-a'
Your branch is up to date with 'origin/feature-a'.
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ echo "Temporary changes" >> greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ cat greetings.txt
Hello from feature-a and feature-b
Temporary changes
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git status
On branch feature-a
Your branch is up to date with 'origin/feature-a'.

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:   greetings.txt

no changes added to commit (use "git add" and/or "git commit -a")
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git stash
Saved working directory and index state WIP on feature-a: 02c45c2 merge main into feature-a
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ cat greetings.txt
Hello from feature-a and feature-b
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git status
On branch feature-a
Your branch is up to date with 'origin/feature-a'.

nothing to commit, working tree clean
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$
```

- git switch feature-a:
  You switched to the `feature-a` branch.
- echo "Temporary changes" >> greetings.txt
  You appended "`Temporary changes`" to the file.
- cat greetings.txt:
  You saw what file contain
- git status:
  Git shows the file `greetings.txt` is modified but not yet staged for commit.
- git stash:
  This **temporarily saved** your local changes (the modification to `greetings.txt`) into the stash and restored the file to its previous state:
- cat greetings.txt:
  The file is back to the previous version (before the temporary changes):
- git status:
  Clean working tree! No changes.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git stash pop
On branch feature-a
Your branch is up to date with 'origin/feature-a'.

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:   greetings.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (5d0745d872f1365f0906f4eb3bfc83e3b239f33e)
```

- git stash pop:
  Applies the last stashed changes back into your working directory, Then **removes** (`drops`) that stash from the stash list.

```
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ echo "Fix bug and change commit" >> greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git add greetings.txt
snir@DESKTOP-KDIHU2K:/mnt/d/Zionet Courses/Zionet Devops/my-git-project$ git commit --amend
```

- echo "Fix bug and change commit" >> greetings.txt
  You appended the text "`Fix bug and change commit`" to the file `greetings.txt`.
- git add greetings.txt
  You added the modified file to the staging area, preparing it for commit.
- git commit –amend
  This opened the editor (likely Vim or Nano) to **edit the previous commit's message**. The `--amend` command lets you:
  - **Update the content** of the last commit,
  - **Change the commit message** if you want.