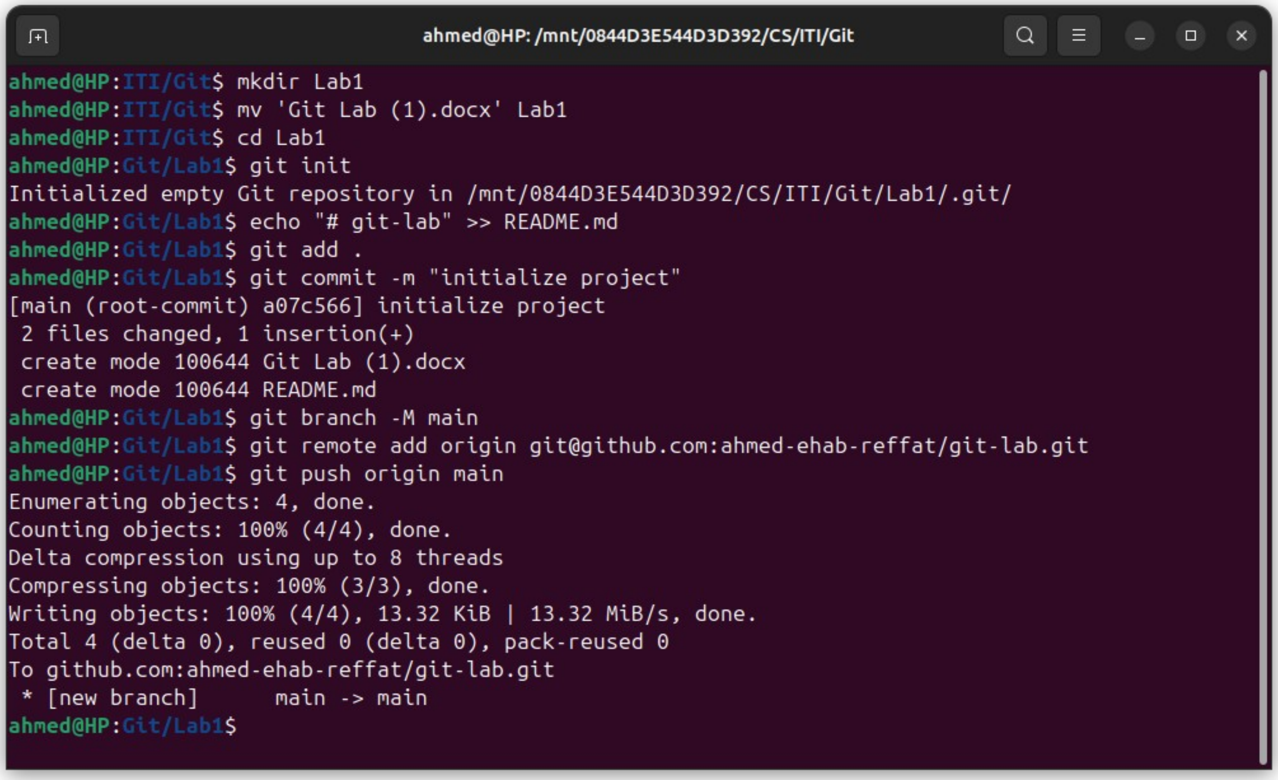
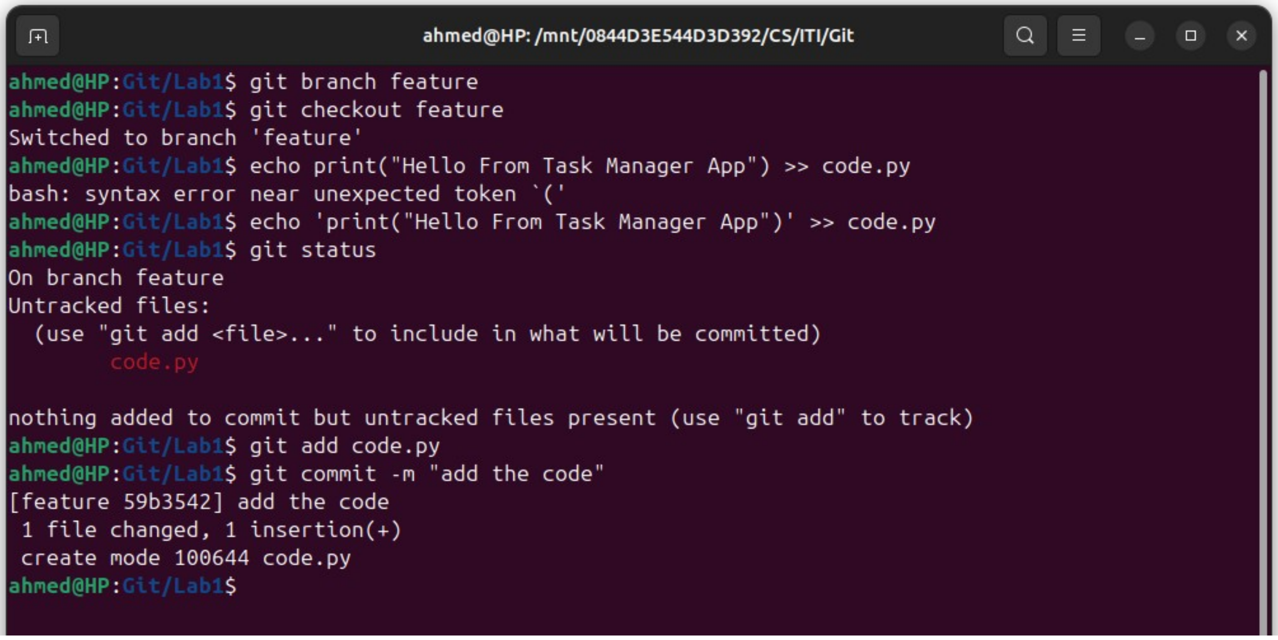
Ahmed Ehab Ahmed Reffat Open Source - Smart Village – 46

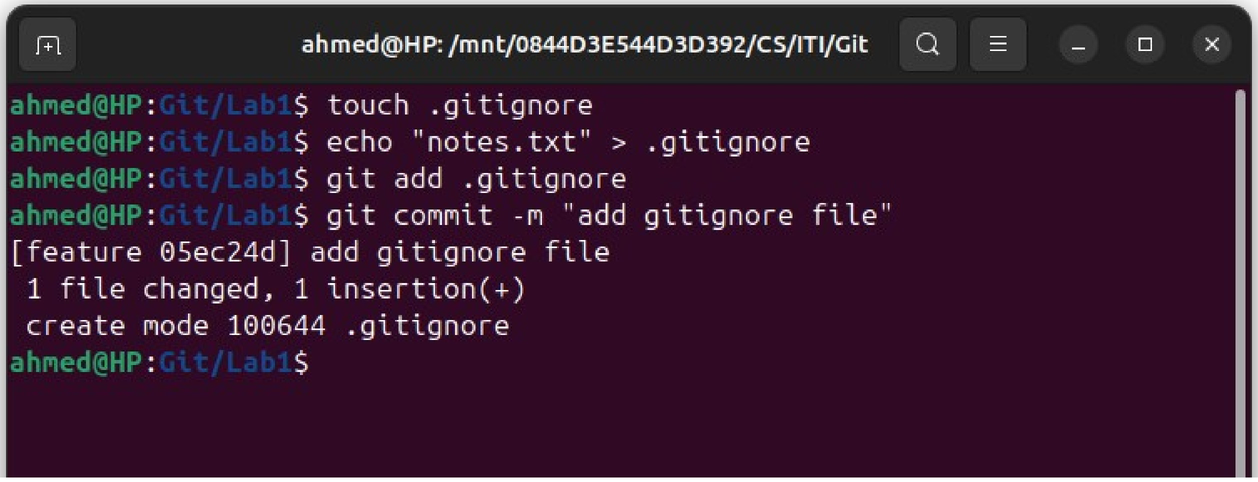
Git Lab-1



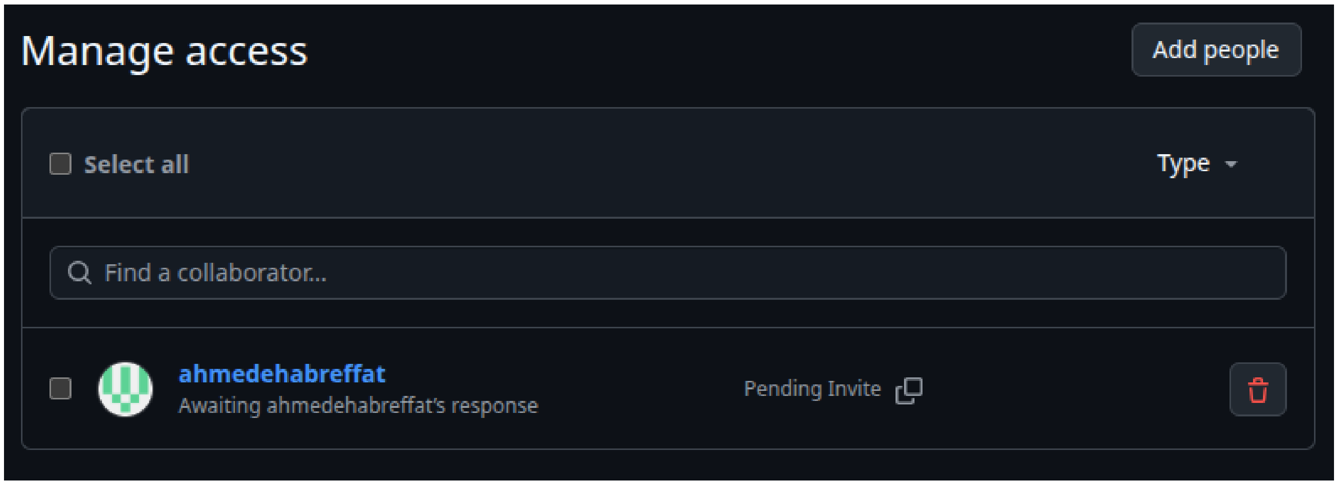
```
ahmed@HP:ITI/Git$ mkdir Lab1
ahmed@HP:ITI/Git$ mv 'Git Lab (1).docx' Lab1
ahmed@HP:ITI/Git$ cd Lab1
ahmed@HP:Git/Lab1$ git init
Initialized empty Git repository in /mnt/0844D3E544D3D392/CS/ITI/Git/Lab1/.git/
ahmed@HP:Git/Lab1$ echo "# git-lab" >> README.md
ahmed@HP:Git/Lab1$ git add .
ahmed@HP:Git/Lab1$ git commit -m "initialize project"
[main (root-commit) a07c566] initialize project
 2 files changed, 1 insertion(+)
 create mode 100644 Git Lab (1).docx
 create mode 100644 README.md
ahmed@HP:Git/Lab1$ git branch -M main
ahmed@HP:Git/Lab1$ git remote add origin git@github.com:ahmed-ehab-reffat/git-lab.git
ahmed@HP:Git/Lab1$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 13.32 KiB | 13.32 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:ahmed-ehab-reffat/git-lab.git
 * [new branch]      main -> main
ahmed@HP:Git/Lab1$
```



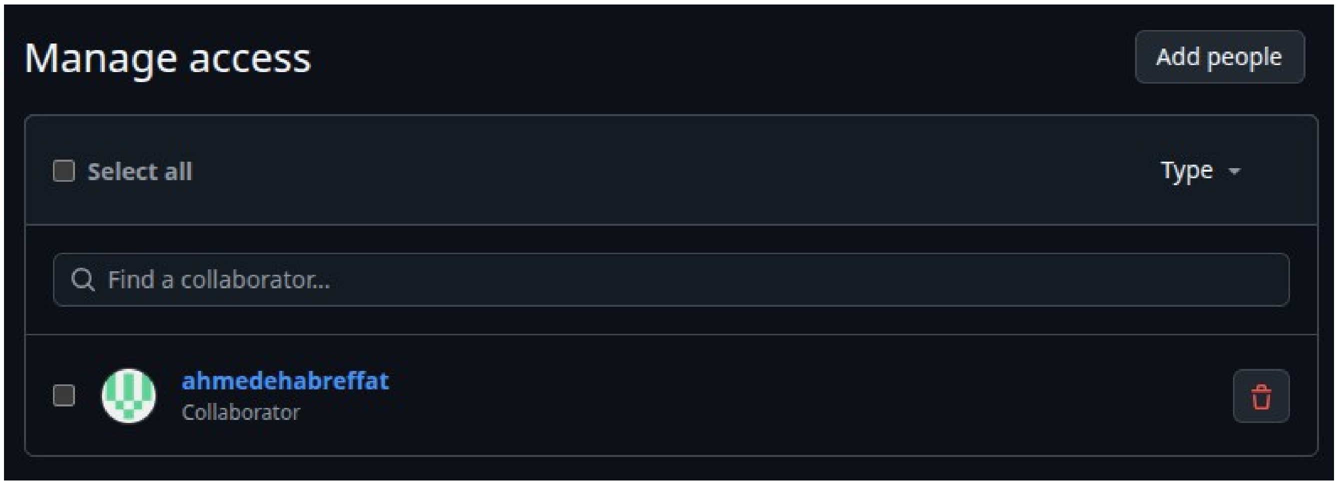
```
ahmed@HP:Git/Lab1$ git branch feature
ahmed@HP:Git/Lab1$ git checkout feature
Switched to branch 'feature'
ahmed@HP:Git/Lab1$ echo print("Hello From Task Manager App") >> code.py
bash: syntax error near unexpected token `('
ahmed@HP:Git/Lab1$ echo 'print("Hello From Task Manager App")' >> code.py
ahmed@HP:Git/Lab1$ git status
On branch feature
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        code.py

nothing added to commit but untracked files present (use "git add" to track)
ahmed@HP:Git/Lab1$ git add code.py
ahmed@HP:Git/Lab1$ git commit -m "add the code"
[feature 59b3542] add the code
 1 file changed, 1 insertion(+)
 create mode 100644 code.py
ahmed@HP:Git/Lab1$
```



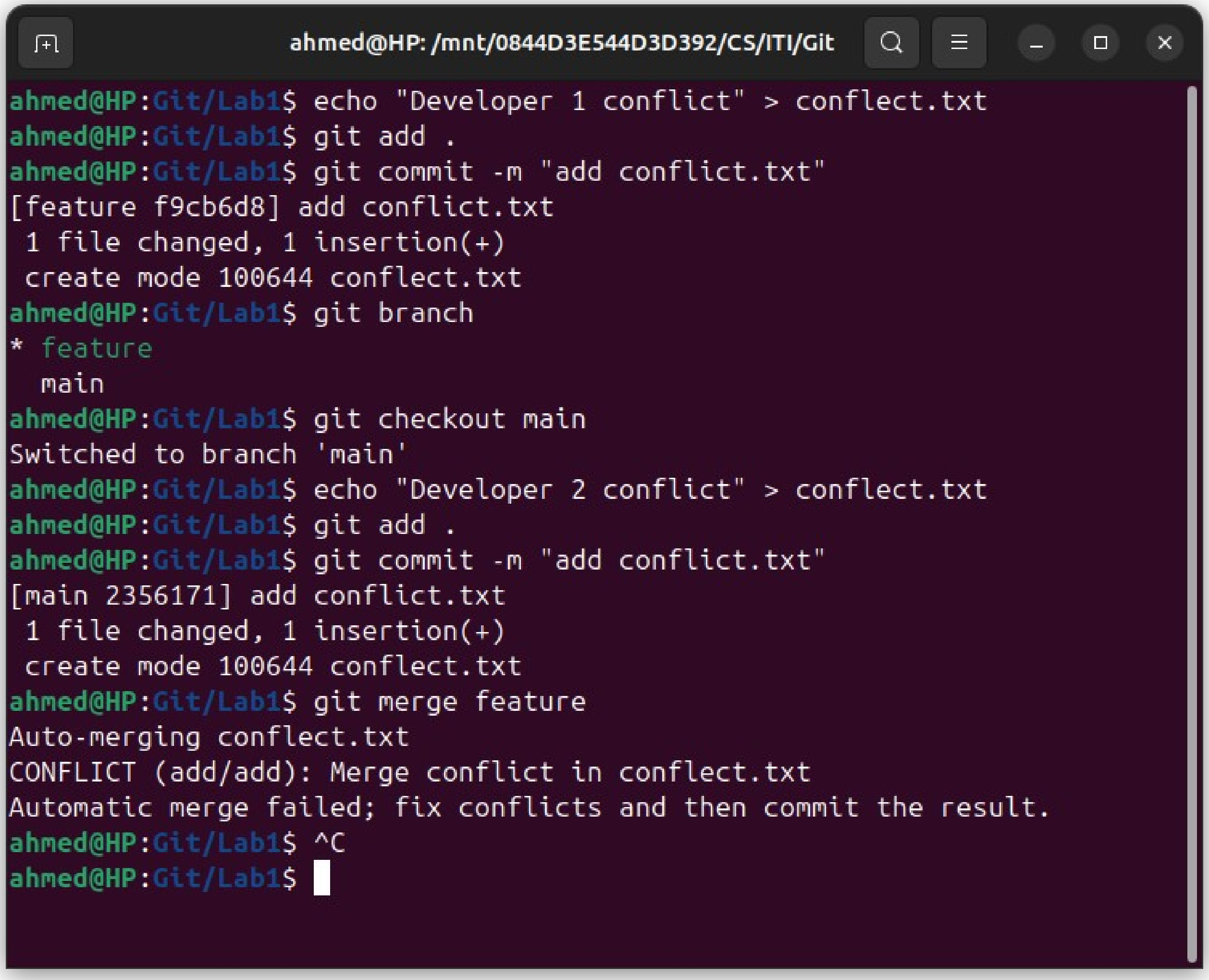
```
ahmed@HP:Git/Lab1$ touch .gitignore
ahmed@HP:Git/Lab1$ echo "notes.txt" > .gitignore
ahmed@HP:Git/Lab1$ git add .gitignore
ahmed@HP:Git/Lab1$ git commit -m "add gitignore file"
[feature 05ec24d] add gitignore file
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore
ahmed@HP:Git/Lab1$
```



ahmedehabreffat
Awaiting ahmedehabreffat's response
Pending Invite

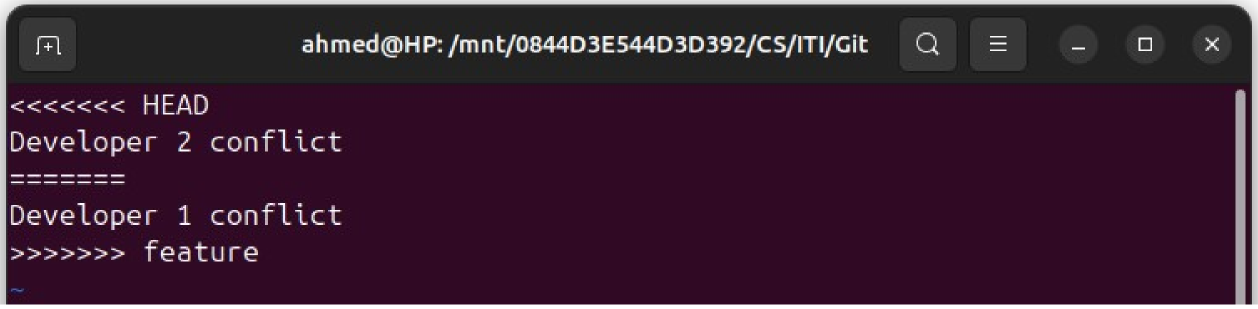


ahmedehabreffat
Collaborator



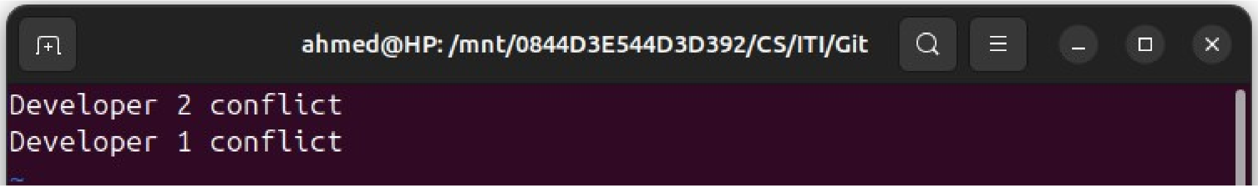
ahmed@HP: /mnt/0844D3E544D3D392/CS/ITI/Git

```
ahmed@HP:Git/Lab1$ echo "Developer 1 conflict" > conflect.txt
ahmed@HP:Git/Lab1$ git add .
ahmed@HP:Git/Lab1$ git commit -m "add conflict.txt"
[feature f9cb6d8] add conflict.txt
 1 file changed, 1 insertion(+)
 create mode 100644 conflect.txt
ahmed@HP:Git/Lab1$ git branch
* feature
  main
ahmed@HP:Git/Lab1$ git checkout main
Switched to branch 'main'
ahmed@HP:Git/Lab1$ echo "Developer 2 conflict" > conflect.txt
ahmed@HP:Git/Lab1$ git add .
ahmed@HP:Git/Lab1$ git commit -m "add conflict.txt"
[main 2356171] add conflict.txt
 1 file changed, 1 insertion(+)
 create mode 100644 conflect.txt
ahmed@HP:Git/Lab1$ git merge feature
Auto-merging conflect.txt
CONFLICT (add/add): Merge conflict in conflect.txt
Automatic merge failed; fix conflicts and then commit the result.
ahmed@HP:Git/Lab1$ ^C
ahmed@HP:Git/Lab1$ 
```



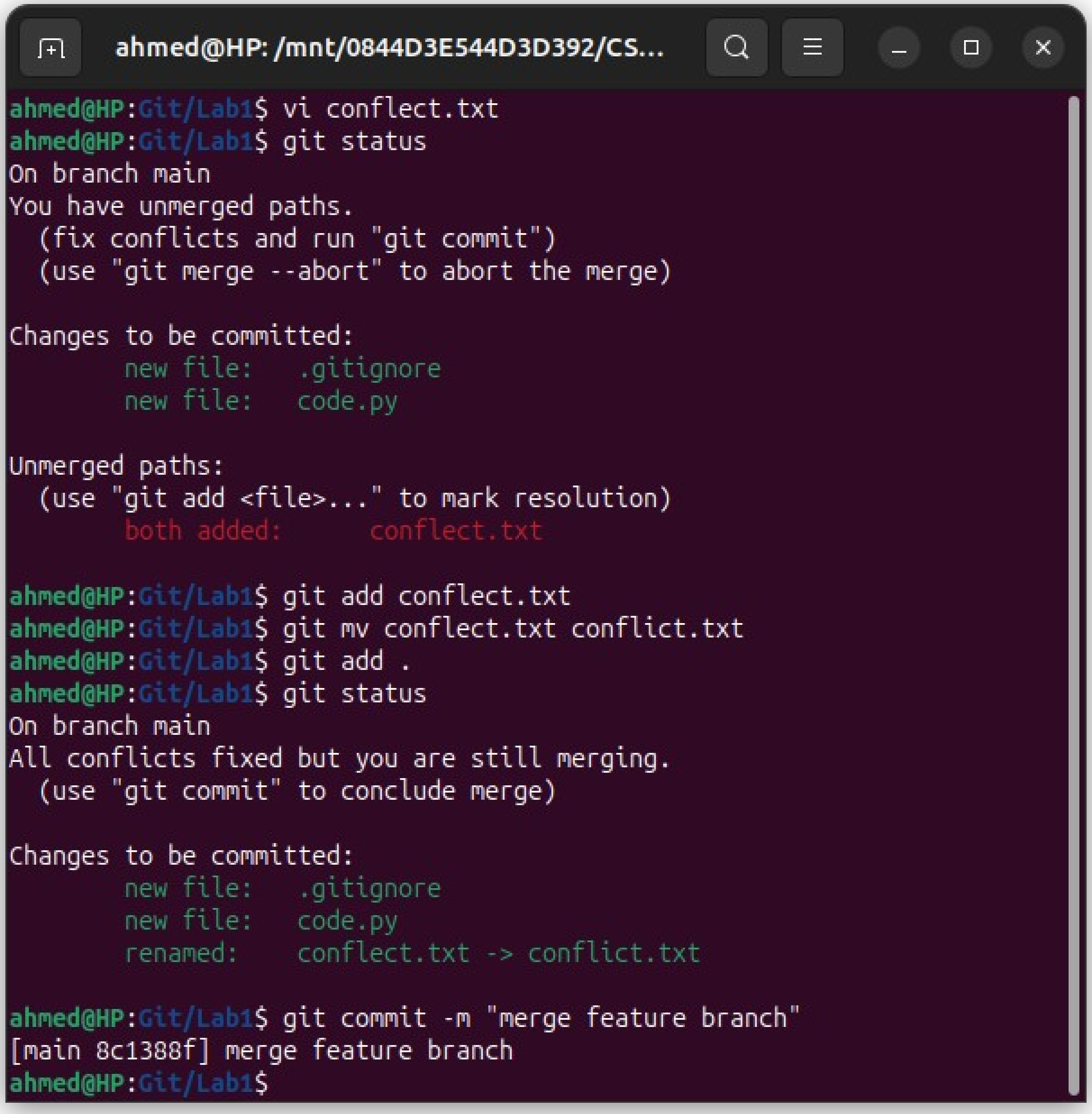
ahmed@HP: /mnt/0844D3E544D3D392/CS/ITI/Git

```
<<<<<<< HEAD
Developer 2 conflict
=======
Developer 1 conflict
>>>>>>> feature
~
```

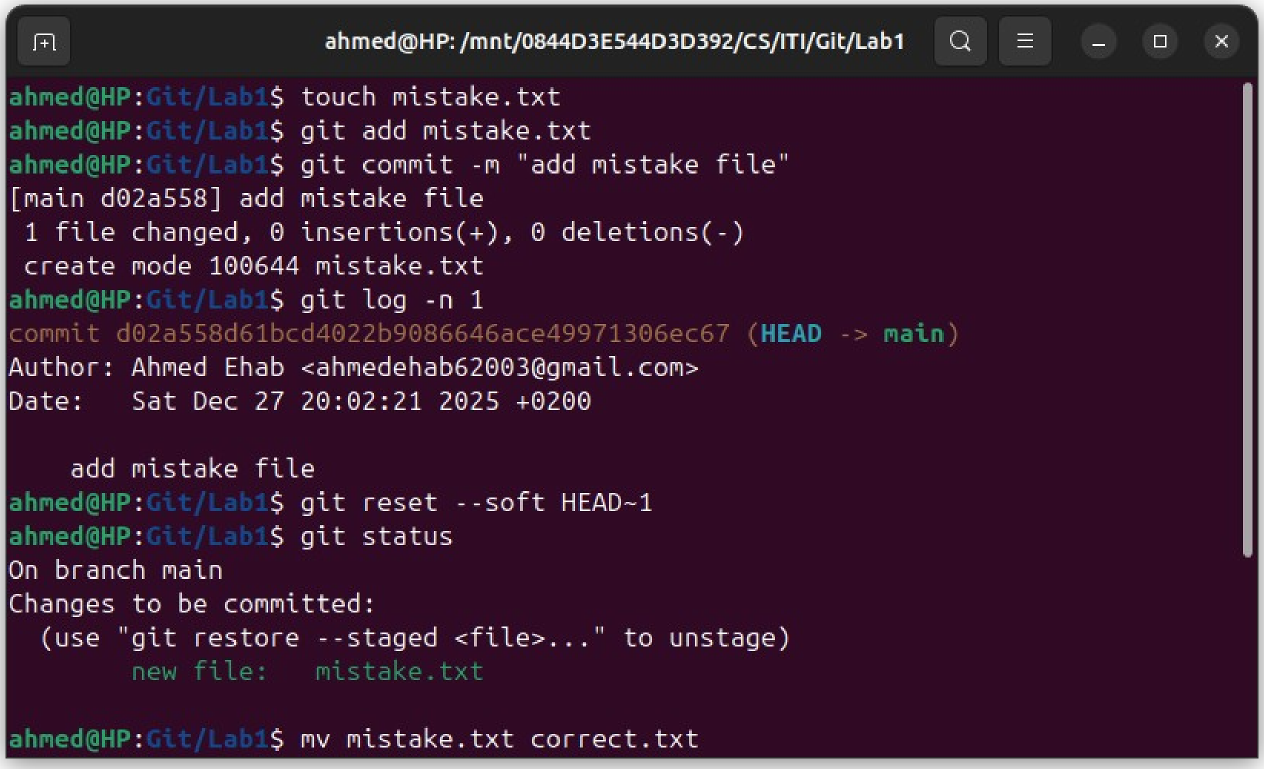


ahmed@HP: /mnt/0844D3E544D3D392/CS/ITI/Git

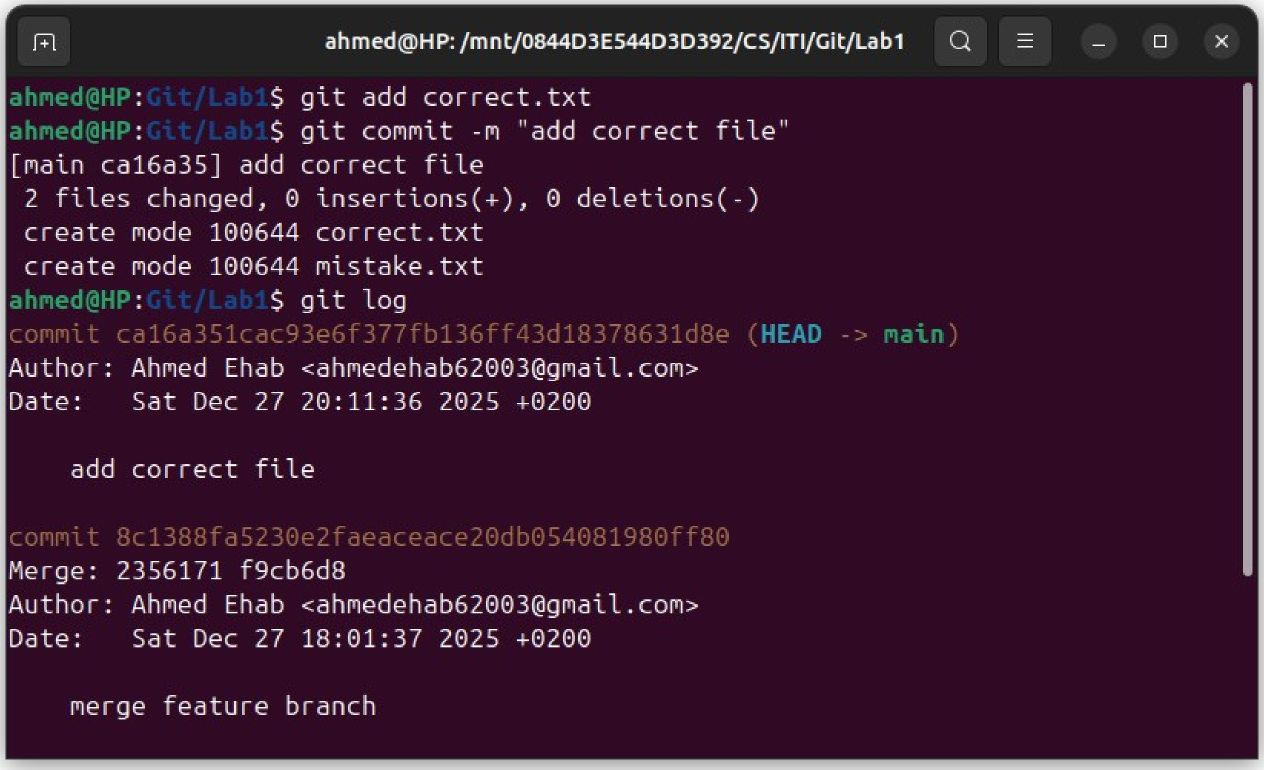
```
Developer 2 conflict
Developer 1 conflict
~
```


ahmed@HP: /mnt/0844D3E544D3D392/CS...

```
ahmed@HP:Git/Lab1$ vi conflect.txt
ahmed@HP:Git/Lab1$ git status
On branch main
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Changes to be committed:
        new file:   .gitignore
        new file:   code.py

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both added:      conflect.txt

ahmed@HP:Git/Lab1$ git add conflect.txt
ahmed@HP:Git/Lab1$ git mv conflect.txt conflict.txt
ahmed@HP:Git/Lab1$ git add .
ahmed@HP:Git/Lab1$ git status
On branch main
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
        new file:   .gitignore
        new file:   code.py
        renamed:    conflect.txt -> conflict.txt

ahmed@HP:Git/Lab1$ git commit -m "merge feature branch"
[main 8c1388f] merge feature branch
ahmed@HP:Git/Lab1$
```



```
ahmed@HP:Git/Lab1$ touch mistake.txt
ahmed@HP:Git/Lab1$ git add mistake.txt
ahmed@HP:Git/Lab1$ git commit -m "add mistake file"
[main d02a558] add mistake file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 mistake.txt
ahmed@HP:Git/Lab1$ git log -n 1
commit d02a558d61bcd4022b9086646ace49971306ec67 (HEAD -> main)
Author: Ahmed Ehab <ahmedehab62003@gmail.com>
Date:   Sat Dec 27 20:02:21 2025 +0200

    add mistake file
ahmed@HP:Git/Lab1$ git reset --soft HEAD~1
ahmed@HP:Git/Lab1$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   mistake.txt

ahmed@HP:Git/Lab1$ mv mistake.txt correct.txt
```



```
ahmed@HP:Git/Lab1$ git add correct.txt
ahmed@HP:Git/Lab1$ git commit -m "add correct file"
[main ca16a35] add correct file
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 correct.txt
 create mode 100644 mistake.txt
ahmed@HP:Git/Lab1$ git log
commit ca16a351cac93e6f377fb136ff43d18378631d8e (HEAD -> main)
Author: Ahmed Ehab <ahmedehab62003@gmail.com>
Date:   Sat Dec 27 20:11:36 2025 +0200

    add correct file

commit 8c1388fa5230e2faeaceace20db054081980ff80
Merge: 2356171 f9cb6d8
Author: Ahmed Ehab <ahmedehab62003@gmail.com>
Date:   Sat Dec 27 18:01:37 2025 +0200

    merge feature branch
```

reset --soft
Moves branch only, keeps staging and working directory

reset --mixed
(default) Moves branch, clears staging, keeps working directory

reset --hard
Moves branch, clears staging AND working directory == data loss possible!

revert
Creates a new commit that undoes changes == nothing is deleted == Safe - History Preserved