Git & GitHub Sprints Notes

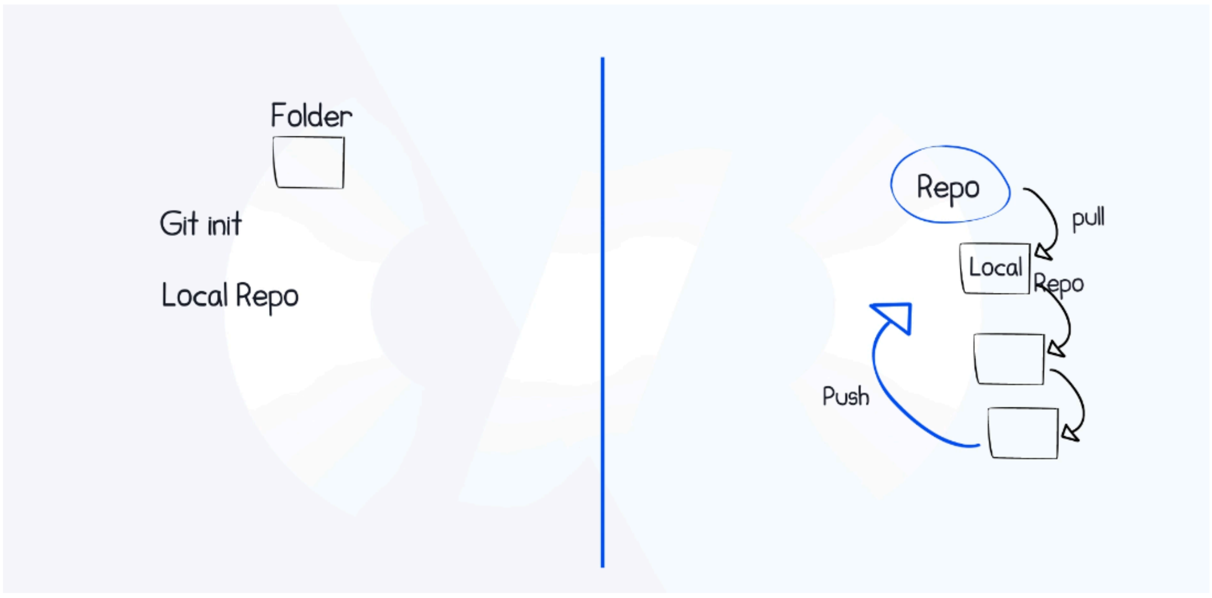
Intro to Git & GitHub

⇒ Git : is a distributed version-control system for tracking changes in source code during software development

Folder
Git init
Local Repo

⇒ for example if i have a local folder on my machine if you used git tool (`git init`) is a command in git if you use this command inside the folder the folder changed to local Repo

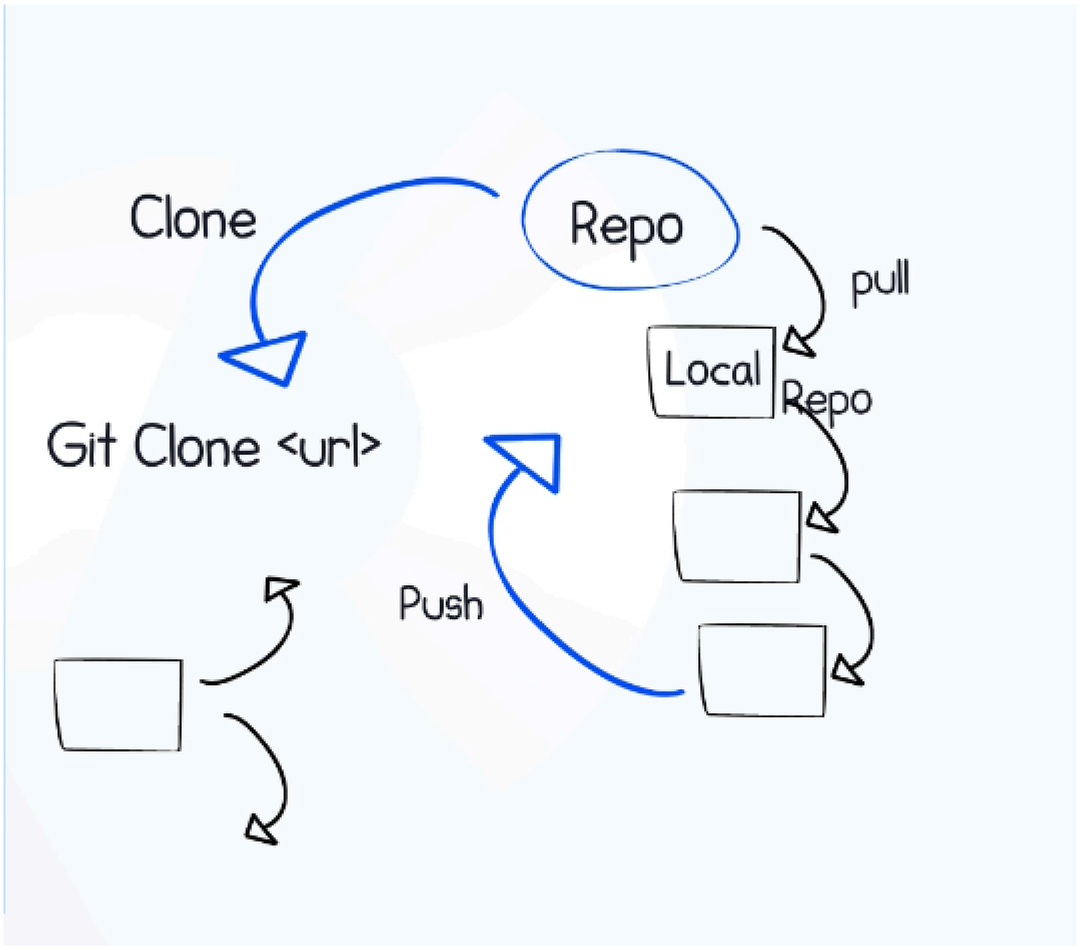
⇒ now this folder will be support git and can use it as a repo can i write the code in it and can tracking my changes in code and can i share this repo with another one to show my history of edits in code



⇒ it useful if you have a repo on cloud you can take a local copy (Pull) from it ad edit after finishing you can upload it in cloud

Notes :

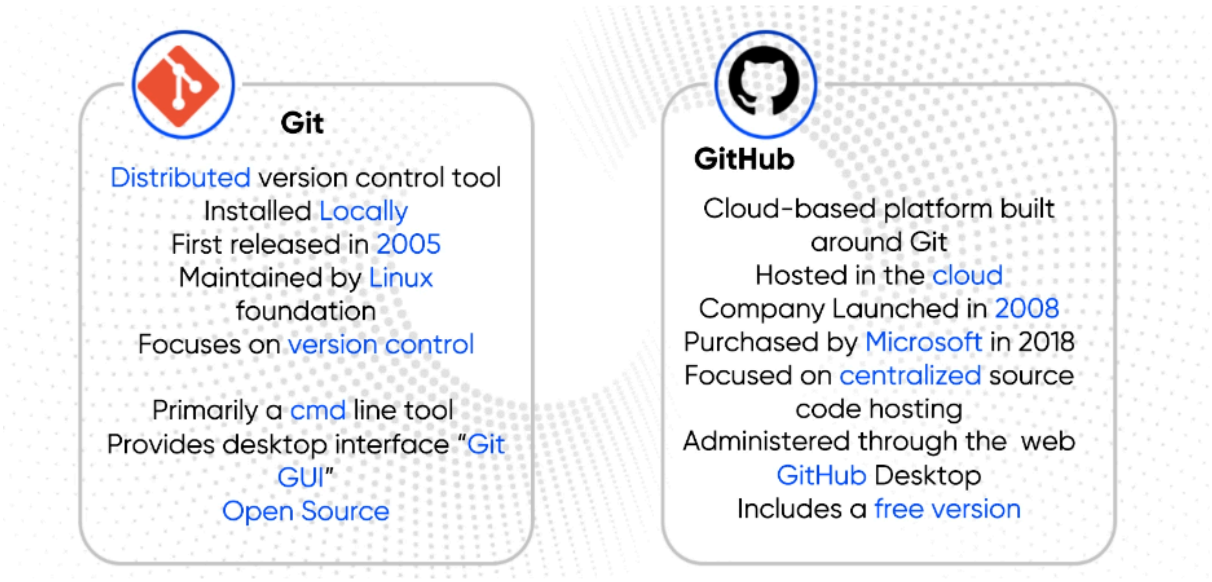
- pull ⇒ meaning to get a copy of repo local to make your edits
- Push ⇒ to upload the repo after editing ⇒



⇒ First time of Pull it called (cloning) to download the repo as a local in my machine by >> `git clone <URL>`

⇒ after finishing we push and merged the changes with the original Repo

⇒ GitHub : web-based platform used for version control and collaborative software development. It is built around Git, a distributed version control system created by Linus Torvalds.



Git

Distributed version control tool
Installed Locally
First released in 2005
Maintained by Linux foundation
Focuses on version control

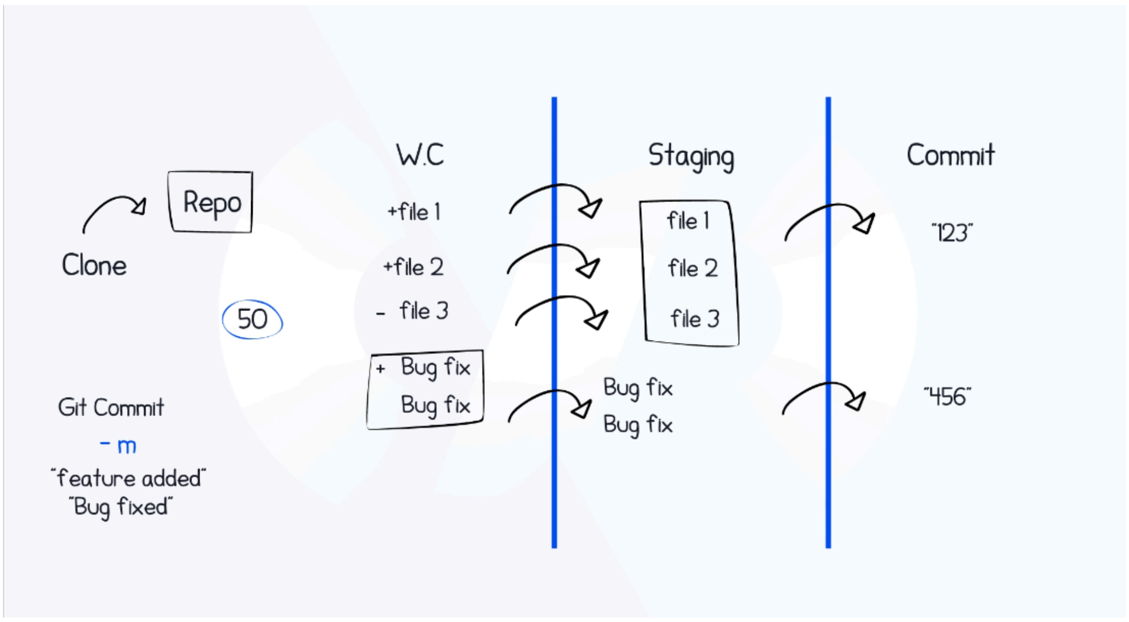
Primarily a cmd line tool
Provides desktop interface "Git GUI"
Open Source

GitHub

Cloud-based platform built around Git
Hosted in the cloud
Company Launched in 2008
Purchased by Microsoft in 2018
Focused on centralized source code hosting
Administered through the web
GitHub Desktop
Includes a free version

Working Copy, Staging & Commit

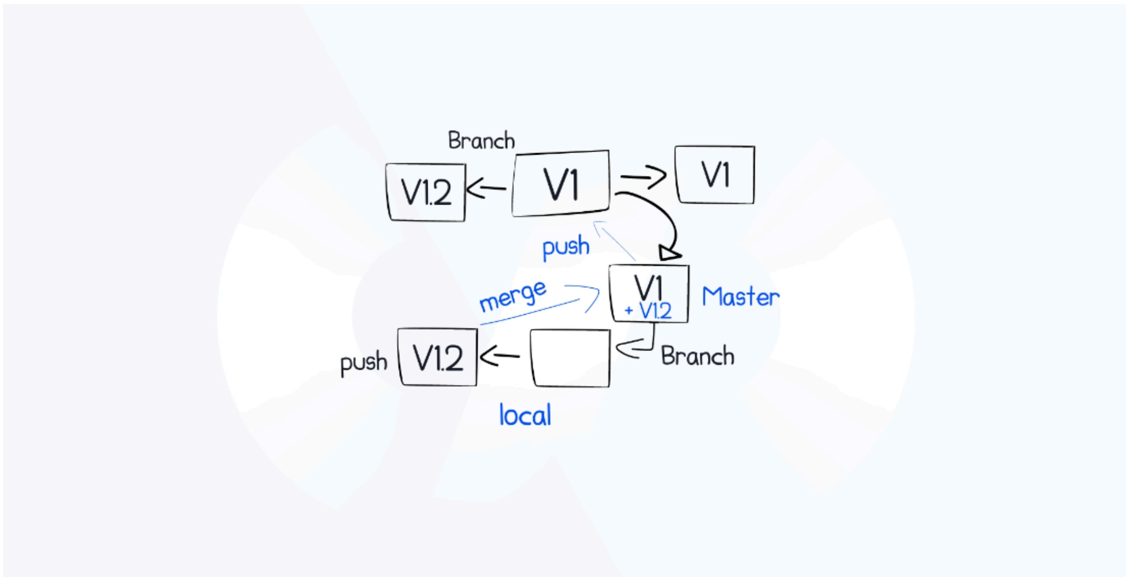
- ⇒ working copy : is the my edits in my local repo (my copy of repo)
- ⇒ staging : is the level before commit or before add edits in the original repo
- ⇒ commit : is the make a new version from the main repo (Feature Added) and it have commit number and it happened by using >> `git commit` also you can add a message by >> `git commit -m <your message>`
- Note : you Return to any level if you in working copy >> `git restore`



Branching and Merge

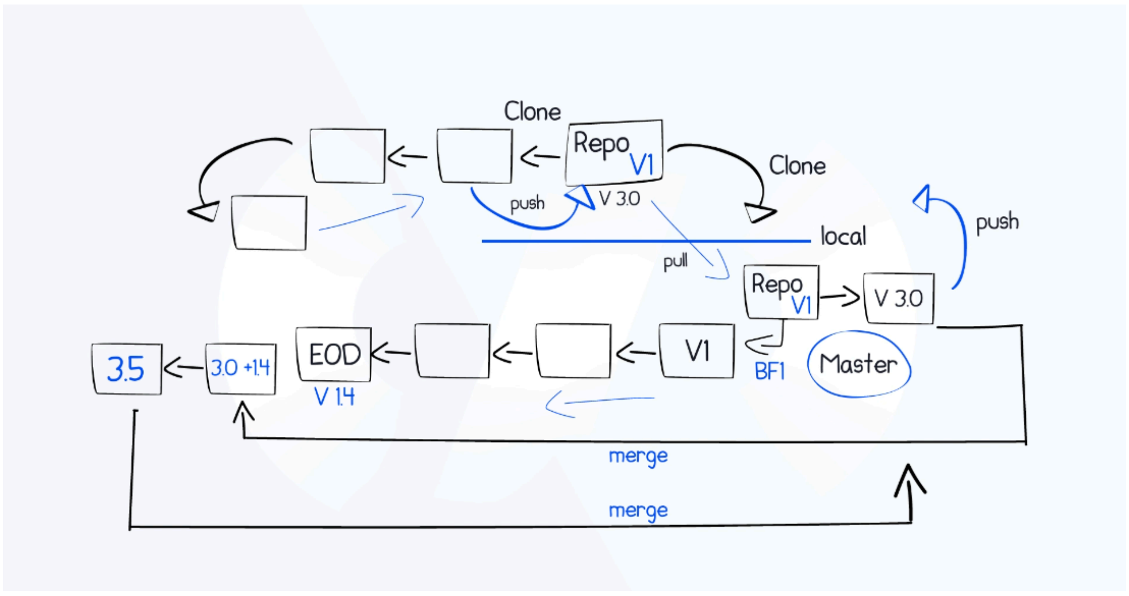
⇒ **branch** in Git is simply a pointer to a commit. Branches allow you to work on **features, fixes, or experiments** independently from the main codebase

⇒ Once you're done working on a branch, you typically **merge** it back into the main branch



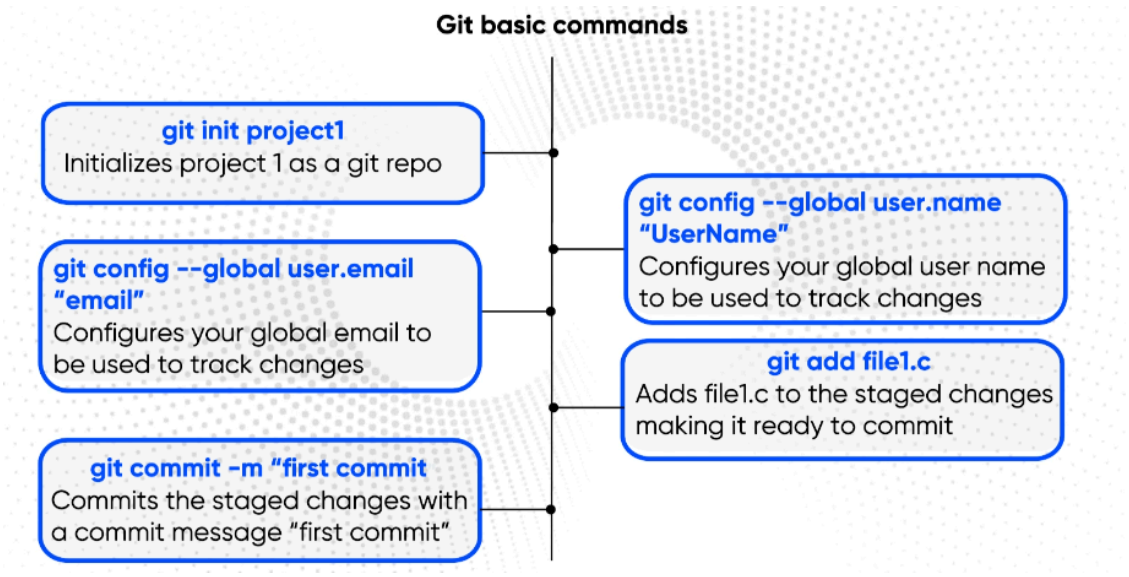
Git workflow

1. clone repo
2. branch for each feature
3. Development (EOD)
4. update local master
5. merge local master feature branch (Rebase)
6. Test
7. merge feature branch to local master
8. Push to cloud



Git Basic commands

https://git-scm.com/downloads ⇒ to download git



Git basic commands

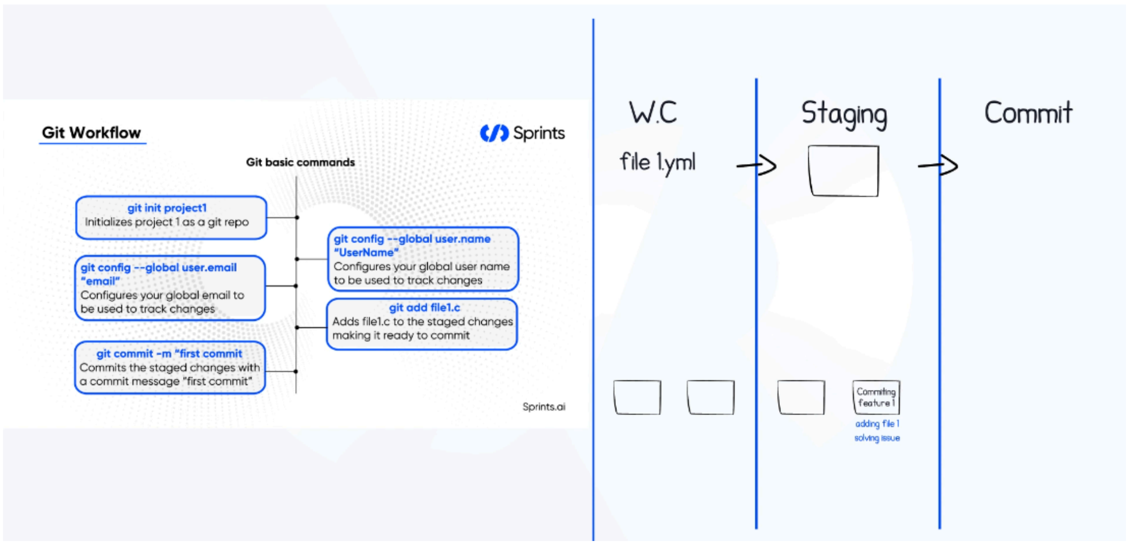
git init project1
Initializes project 1 as a git repo

git config --global user.email "email"
Configures your global email to be used to track changes

git commit -m "first commit
Commits the staged changes with a commit message "first commit"

git config --global user.name "UserName"
Configures your global user name to be used to track changes

git add file1.c
Adds file1.c to the staged changes making it ready to commit



Git Workflow

Sprints

Git basic commands

git init project1
Initializes project 1 as a git repo

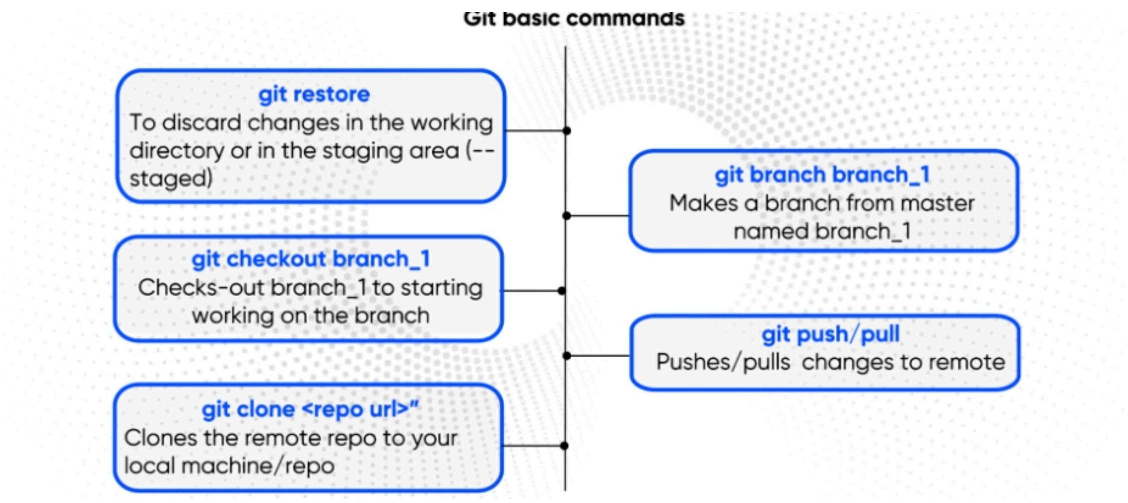
git config --global user.email "email"
Configures your global email to be used to track changes

git commit -m "first commit
Commits the staged changes with a commit message "first commit"

git config --global user.name "UserName"
Configures your global user name to be used to track changes

git add file1.c
Adds file1.c to the staged changes making it ready to commit

Sprints.ai



Git basic commands

git restore
To discard changes in the working directory or in the staging area (--staged)

git checkout branch_1
Checks-out branch_1 to starting working on the branch

git clone <repo url>"
Clones the remote repo to your local machine/repo

git branch branch_1
Makes a branch from master named branch_1

git push/pull
Pushes/pulls changes to remote

Git in action

⇒ make a folder to any place and going to the path of folder by `cd`

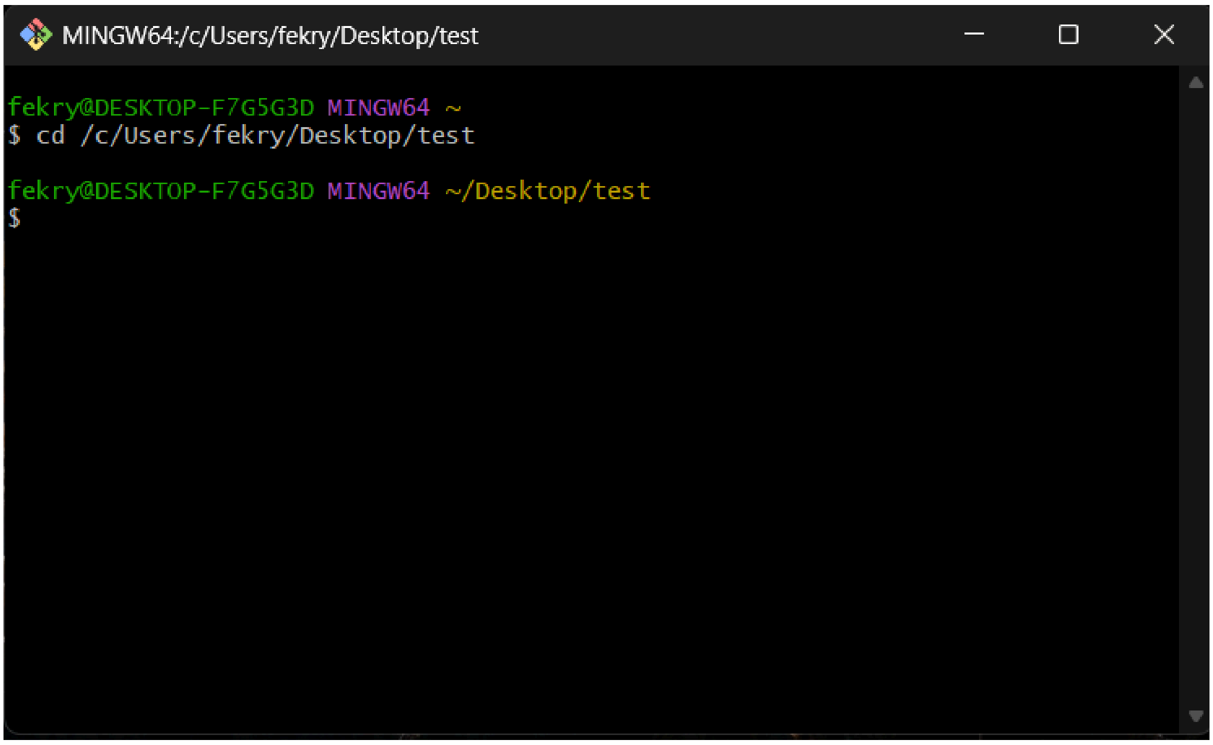
⇒ for example my folder called test and his path is `C:\Users\fekry\Desktop\test`

⇒ the terminal of git is built in Linux so we change the path to this format >> `/c/Users/fekry/Desktop/test`

```
cd /c/Users/fekry/Desktop/test
```

or

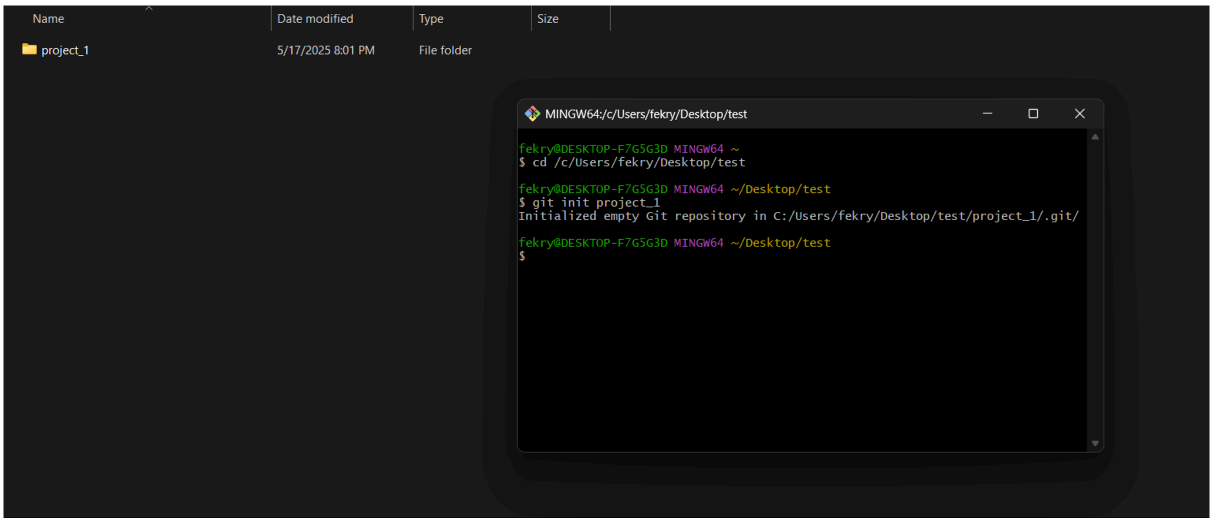
```
cd ~/Desktop/test
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~
$ cd /c/Users/fekry/Desktop/test

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test
$
```

⇒ create a local Repo

```
git init project_1
```



Name	Date modified	Type	Size
project_1	5/17/2025 8:01 PM	File folder	

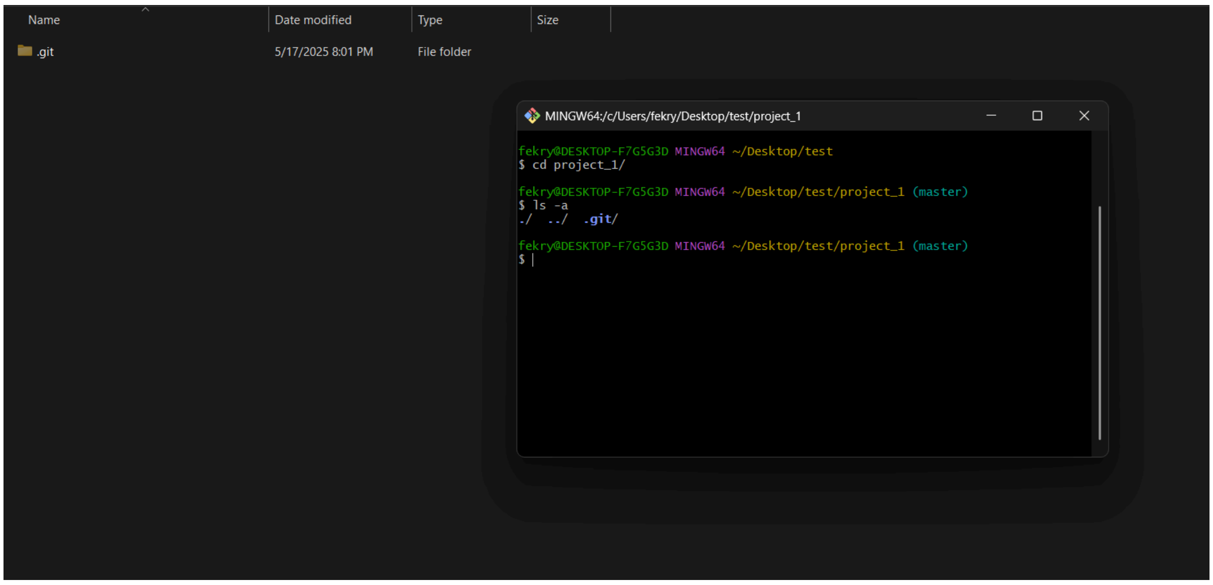
```
fekry@DESKTOP-F7G5G3D MINGW64 ~
$ cd /c/Users/fekry/Desktop/test

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test
$ git init project_1
Initialized empty Git repository in C:/Users/fekry/Desktop/test/project_1/.git/

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test
$
```

⇒ to show the all files or folders >> `ls` if you used Linux before it the same thing

```
ls
```

⇒ to go inside the project

```
cd project_1
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test
$ cd project_1/

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ ls -a
./  ../  .git/

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ |
```

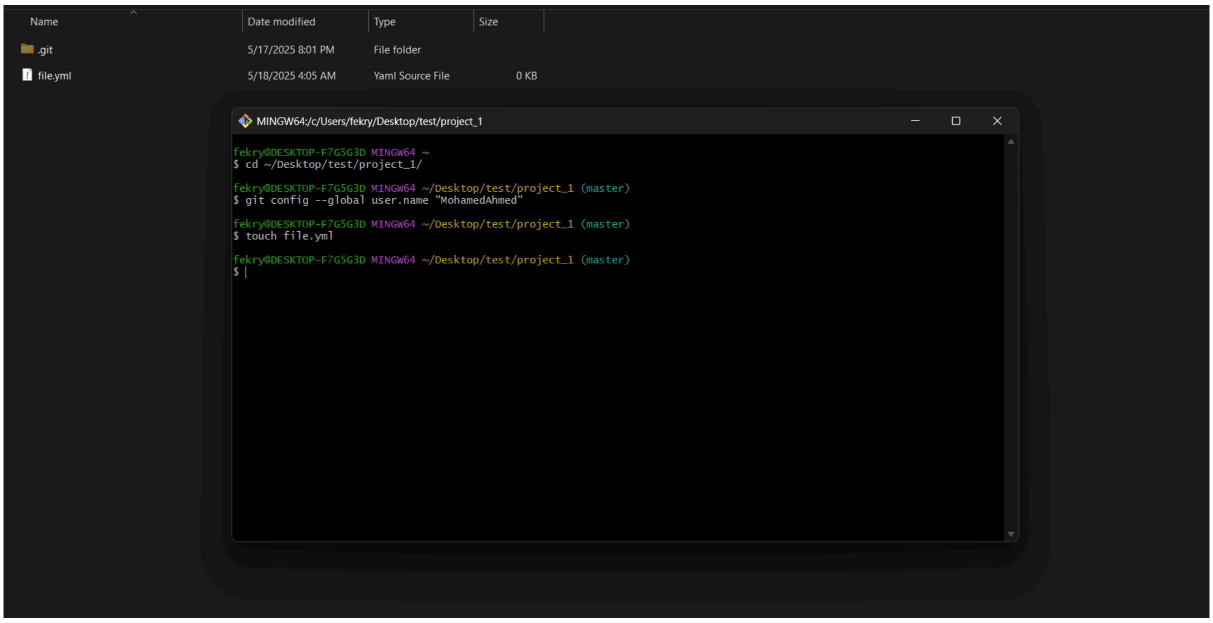
⇒ to add username to track the changes of this username and history of commits

```
git config --global user.name "MohamedAhmed"
```

⇒ now we will create file for example

```
touch file.yml
```

```
ls
```

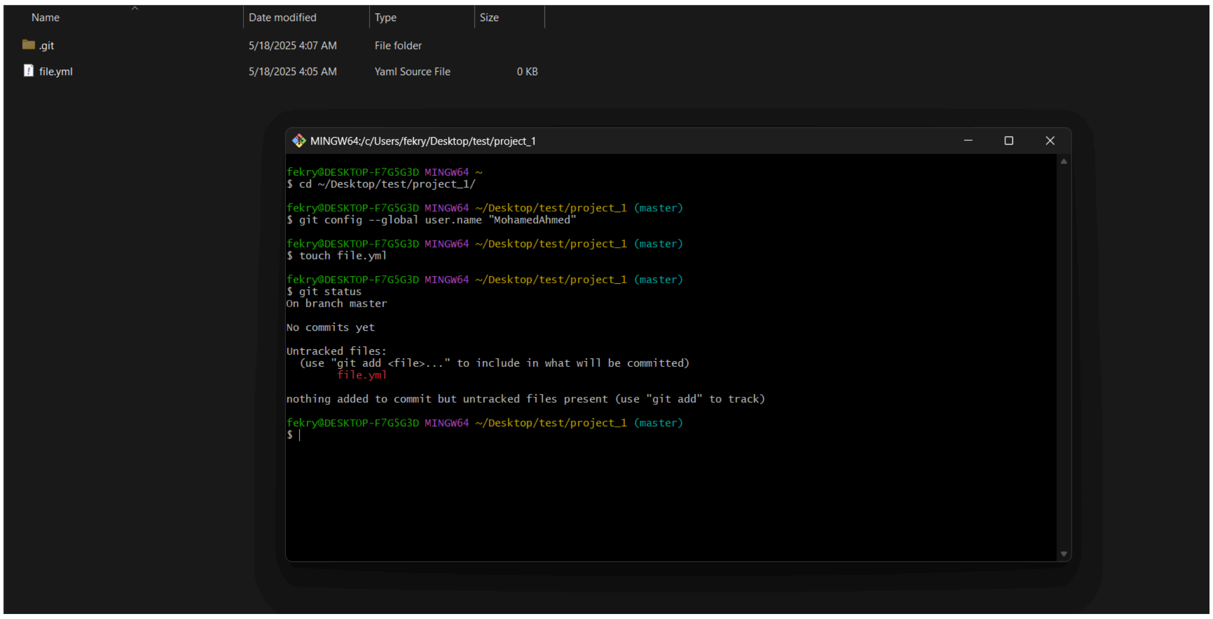


```
Name            Date modified         Type              Size
.git            5/17/2025 8:01 PM     File folder
file.yml        5/18/2025 4:05 AM     Yaml Source File  0 KB
```

```
MINGW64:/c/Users/fekry/Desktop/test/project_1

fekry@DESKTOP-F7G5G3D MINGW64 ~
$ cd ~/Desktop/test/project_1/

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git config --global user.name "MohamedAhmed"

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ touch file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ |
```

⇒ to show you the current state of your working directory and staging area

```
git status
```



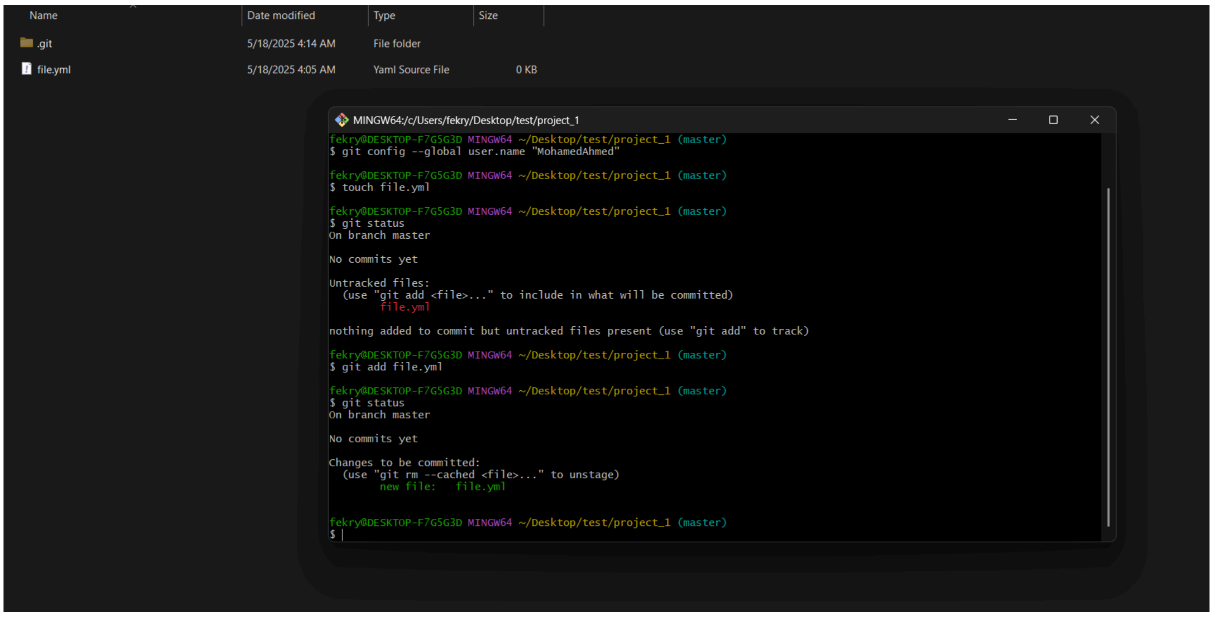
```
Name            Date modified         Type              Size
.git            5/18/2025 4:07 AM     File folder
file.yml        5/18/2025 4:05 AM     Yaml Source File  0 KB
```

```
MINGW64:/c/Users/fekry/Desktop/test/project_1

fekry@DESKTOP-F7G5G3D MINGW64 ~
$ cd ~/Desktop/test/project_1/

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git config --global user.name "MohamedAhmed"

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ touch file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master

No commits yet

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

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ |
```

⇒ to move the file from working copy to staging area >> `git add <name of file>`

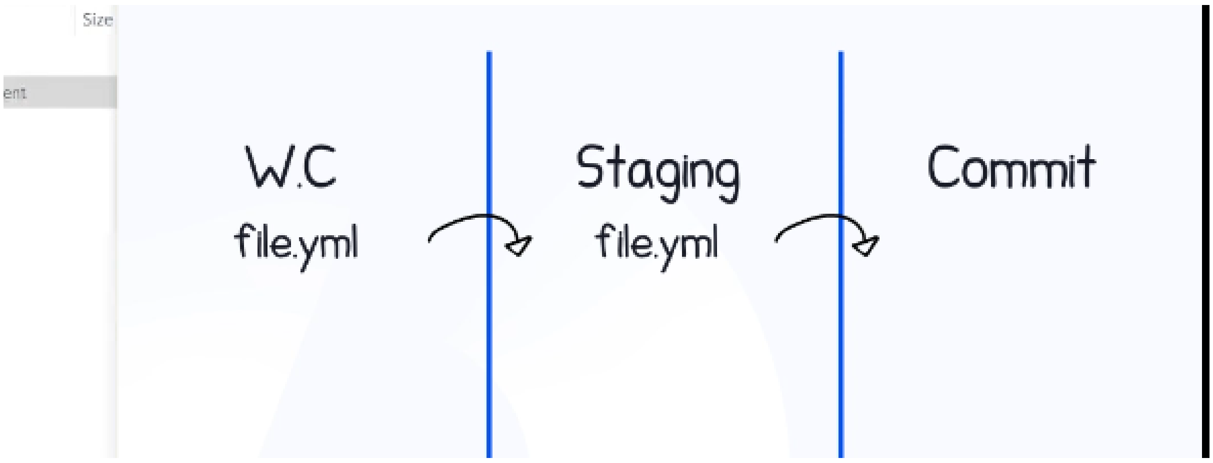
```
git add file.yml
```

⇒ now the file.yml is in staging area



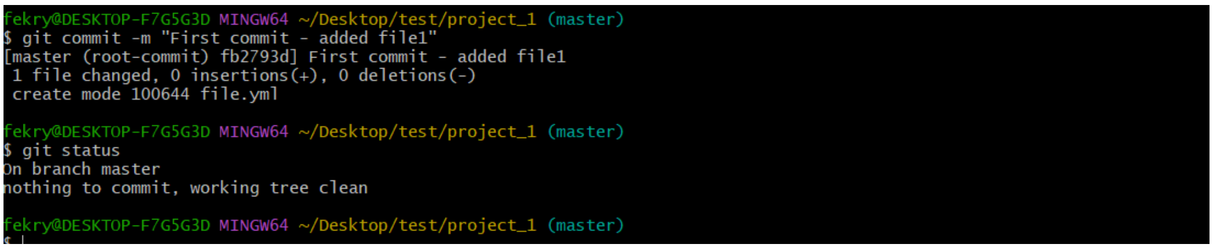
```
Name                Date modified        Type               Size
.git                5/18/2025 4:14 AM    File folder
file.yml            5/18/2025 4:05 AM    Yaml Source File   0 KB
```

```
MINGW64:/c/Users/fekry/Desktop/test/project_1
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git config --global user.name "MohamedAhmed"

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ touch file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master

No commits yet

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

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git add file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master

No commits yet

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


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ |
```



⇒ let's make a first commit >> `git commit -m <"your message">`

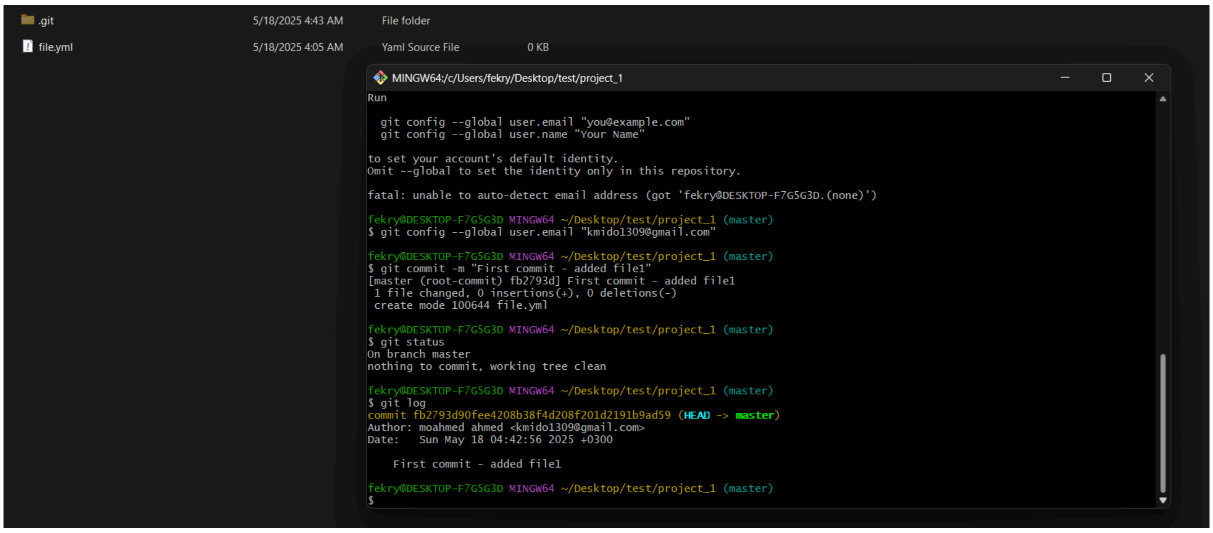
```
git commit -m "First commit - added file1"
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git commit -m "First commit - added file1"
[master (root-commit) fb2793d] First commit - added file1
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
```

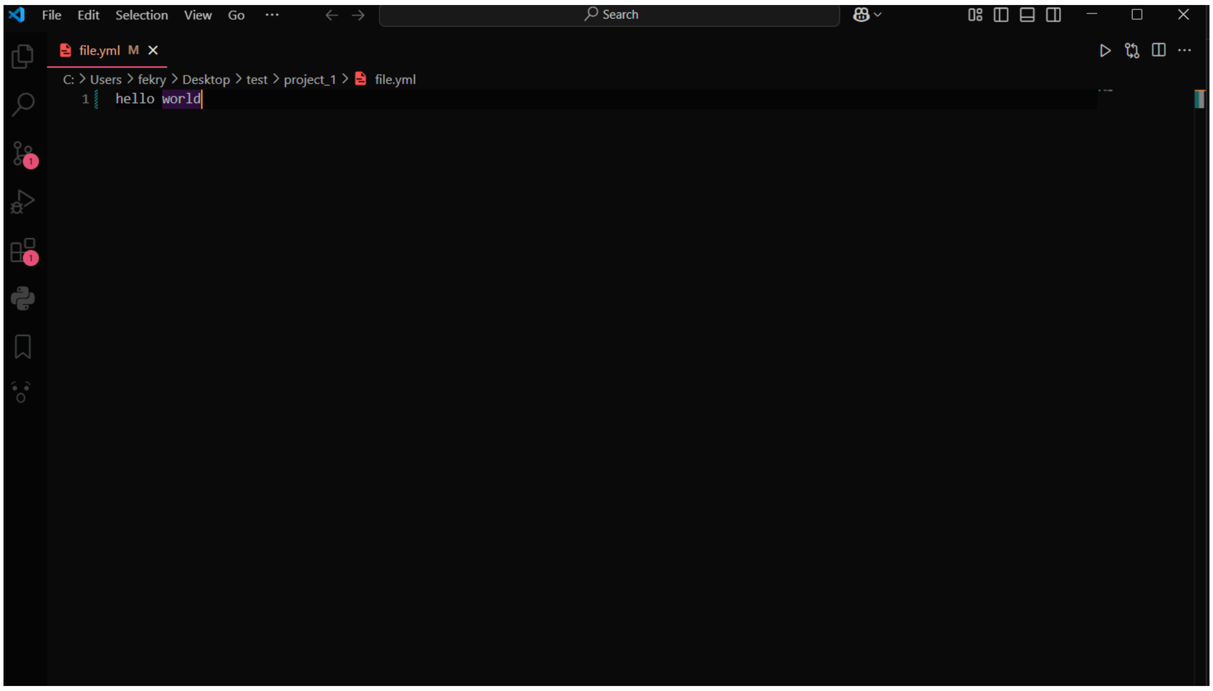
⇒ to show the history of commits

```
git log
```



```
Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'fekry@DESKTOP-F7G5G3D.(none)')

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git config --global user.email "kmido1309@gmail.com"

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git commit -m "First commit - added file1"
[master (root-commit) fb2793d] First commit - added file1
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git log
commit fb2793d90fee4208b38f4d208f201d2191b9ad59 (HEAD -> master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

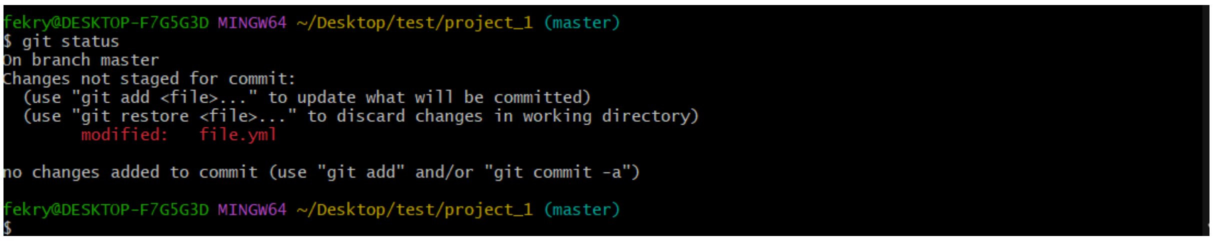
⇒ open the file and type any thing



```
hello world
```

⇒ now we will show the status

```
git status
```

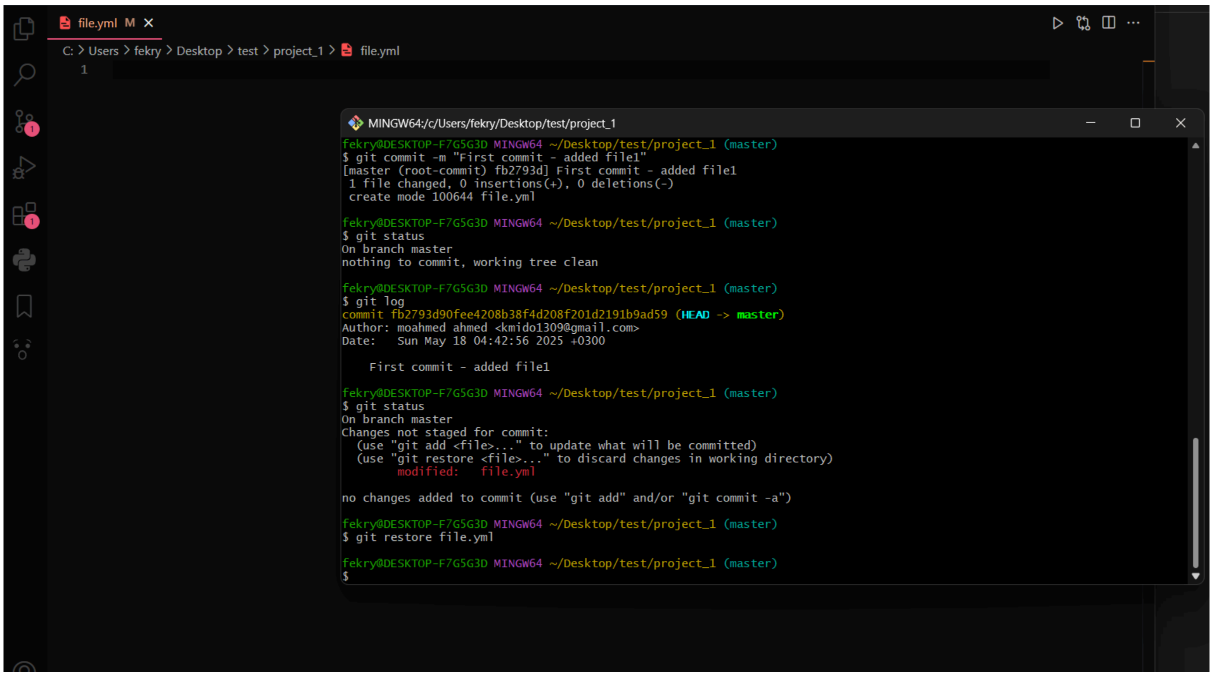


```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

⇒ if we need to discard changes

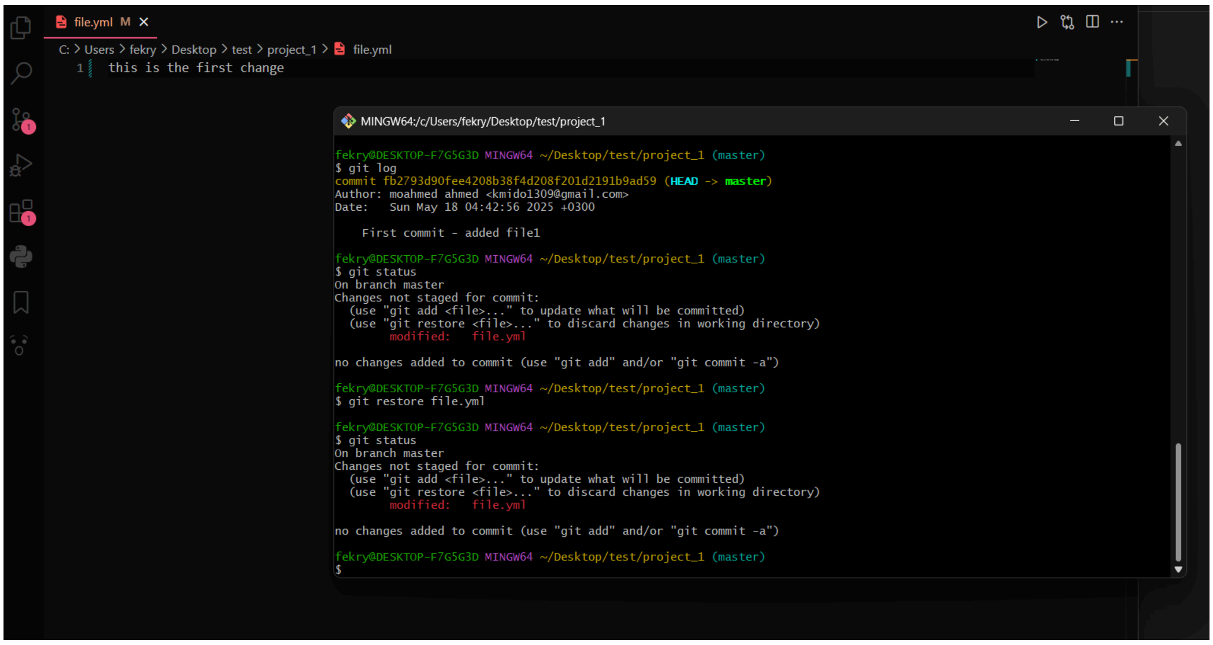
```
git restore file.yml
```

⇒ hello world will be deleted


file.yml M
C: > Users > fekry > Desktop > test > project_1 > file.yml
MINGW64:/c/Users/fekry/Desktop/test/project_1

```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git commit -m "First commit - added file1"
[master (root-commit) fb2793d] First commit - added file1
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git log
commit fb2793d90fee4208b38f4d208f201d2191b9ad59 (HEAD -> master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
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:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git restore file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

⇒ now we will make another change

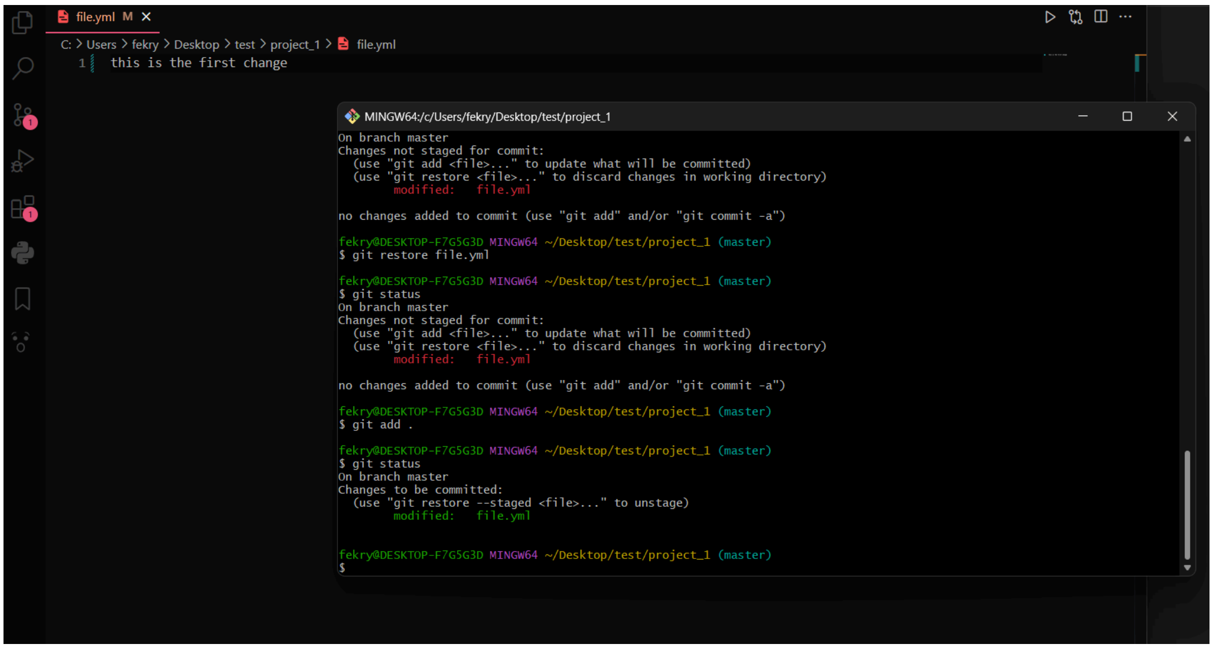

file.yml M
C: > Users > fekry > Desktop > test > project_1 > file.yml
MINGW64:/c/Users/fekry/Desktop/test/project_1

```
this is the first change
```

```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git log
commit fb2793d90fee4208b38f4d208f201d2191b9ad59 (HEAD -> master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
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:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git restore file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
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:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

⇒ now we need to add this changes to the staging area

```
git add .
```

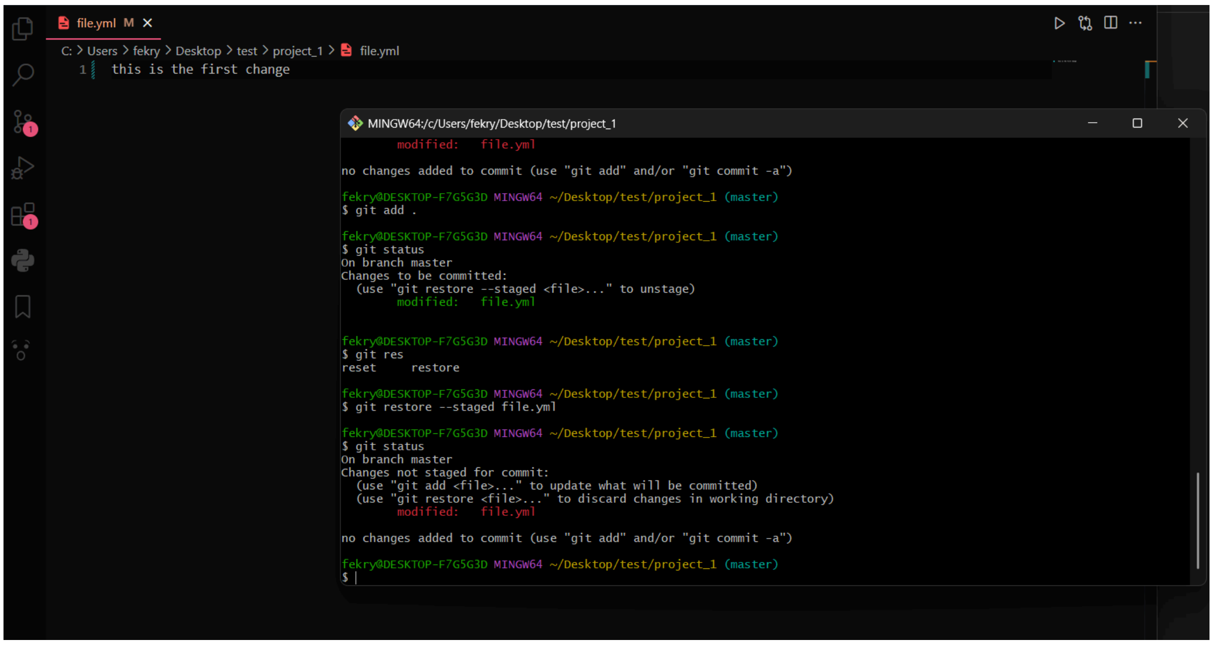

file.yml M
C: > Users > fekry > Desktop > test > project_1 > file.yml
MINGW64:/c/Users/fekry/Desktop/test/project_1

```
this is the first change
```

```
On branch master
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:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git restore file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
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:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file.yml


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

⇒ if we need to return it to the working copy

```
git restore --staged file.yml
```

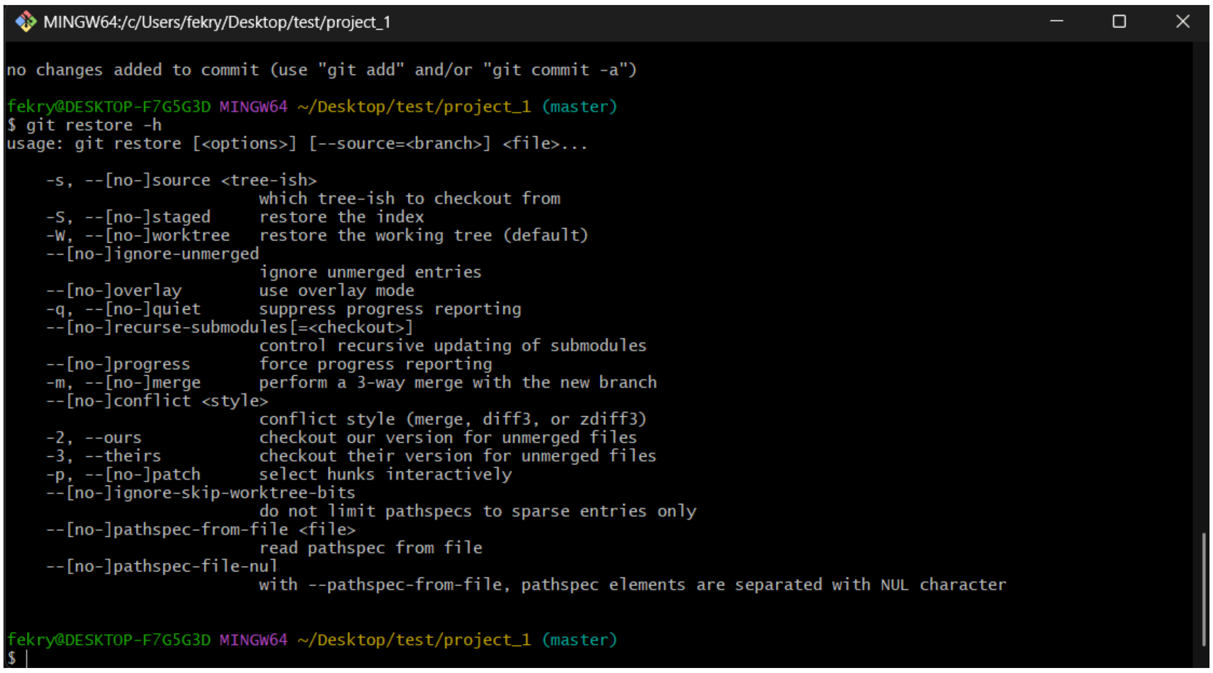

file.yml M
C: > Users > fekry > Desktop > test > project_1 > file.yml
MINGW64:/c/Users/fekry/Desktop/test/project_1

```
this is the first change
```

```
        modified:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file.yml


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git res
reset     restore

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git restore --staged file.yml

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
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:   file.yml

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ |
```

⇒ to show the all options of restore command

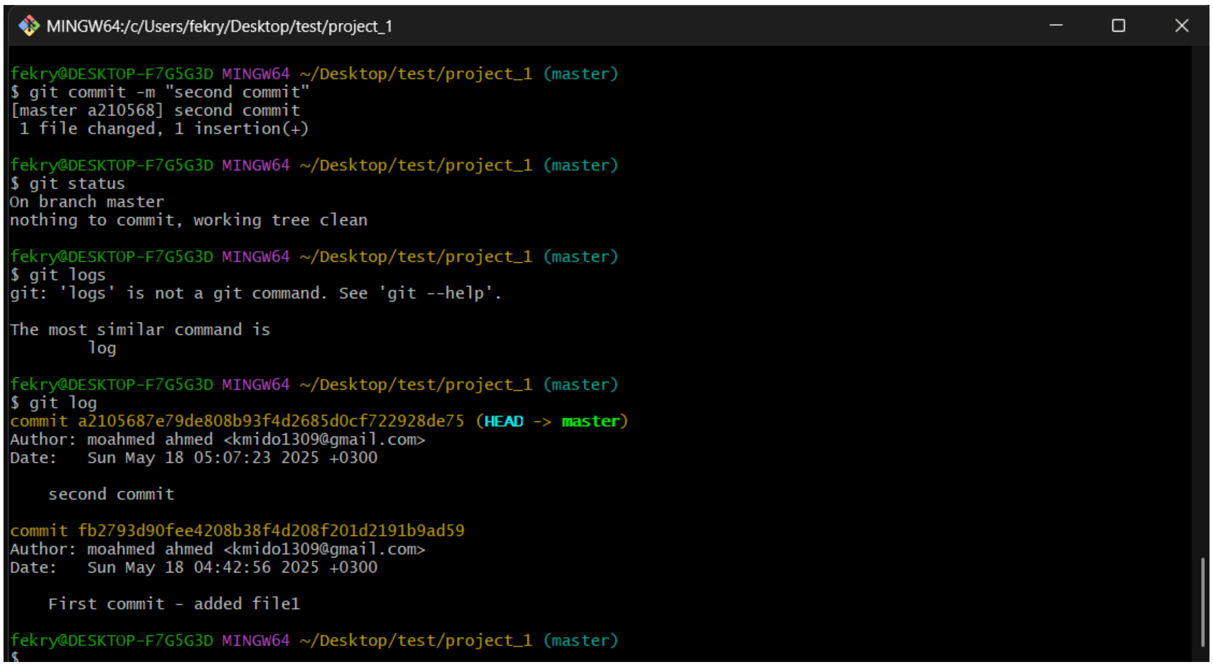
```
git restore -h
```



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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git restore -h
usage: git restore [<options>] [--source=<branch>] <file>...

    -s, --[no-]source <tree-ish>
                          which tree-ish to checkout from
    -S, --[no-]staged     restore the index
    -W, --[no-]worktree   restore the working tree (default)
    --[no-]ignore-unmerged
                          ignore unmerged entries
    --[no-]overlay        use overlay mode
    -q, --[no-]quiet      suppress progress reporting
    --[no-]recurse-submodules[=<checkout>]
                          control recursive updating of submodules
    --[no-]progress       force progress reporting
    -m, --[no-]merge      perform a 3-way merge with the new branch
    --[no-]conflict <style>
                          conflict style (merge, diff3, or zdiff3)
    -2, --ours            checkout our version for unmerged files
    -3, --theirs          checkout their version for unmerged files
    -p, --[no-]patch      select hunks interactively
    --[no-]ignore-skip-worktree-bits
                          do not limit pathspecs to sparse entries only
    --[no-]pathspec-from-file <file>
                          read pathspec from file
    --[no-]pathspec-file-nul
                          with --pathspec-from-file, pathspec elements are separated with NUL character

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

⇒ let's add our second commit

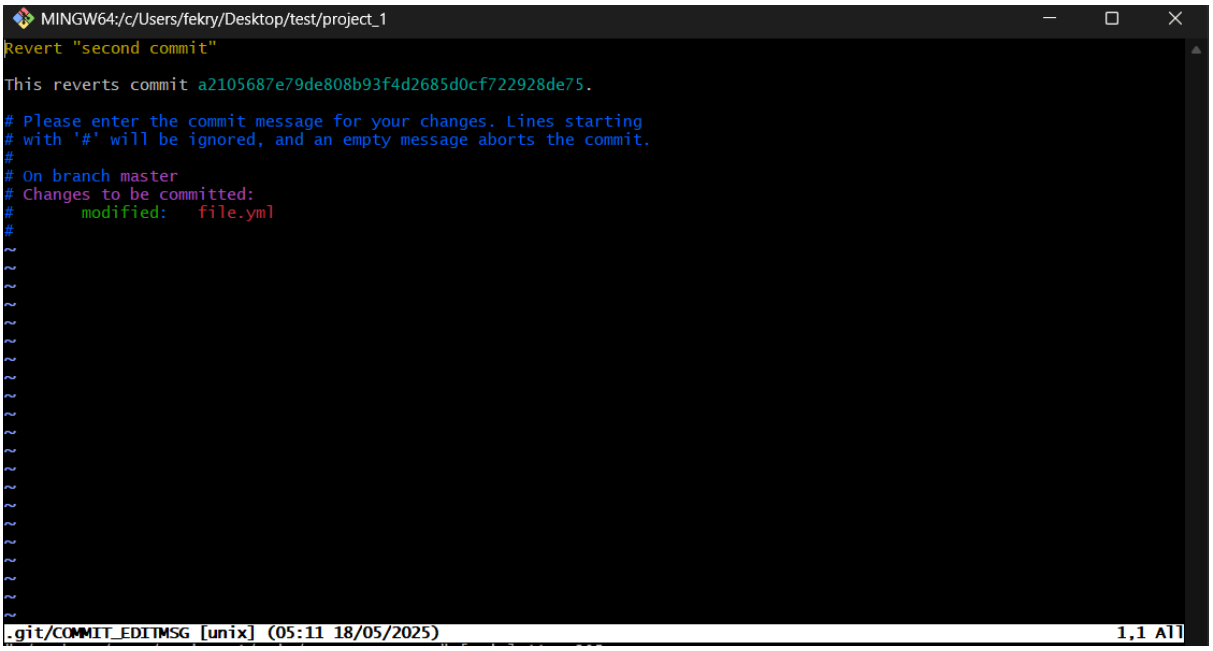
```
git commit -m "second commit"
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git commit -m "second commit"
[master a210568] second commit
 1 file changed, 1 insertion(+)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git logs
git: 'logs' is not a git command. See 'git --help'.

The most similar command is
        log

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git log
commit a2105687e79de808b93f4d2685d0cf722928de75 (HEAD -> master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:07:23 2025 +0300

    second commit

commit fb2793d90fee4208b38f4d208f201d2191b9ad59
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

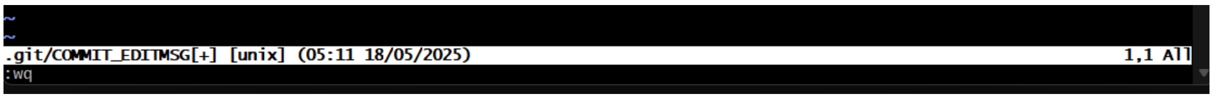
⇒ if we need to remove the commit >> `git revert <commit ID>`

```
git revert a2105687e79de808b93f4d2685d0cf722928de75
```



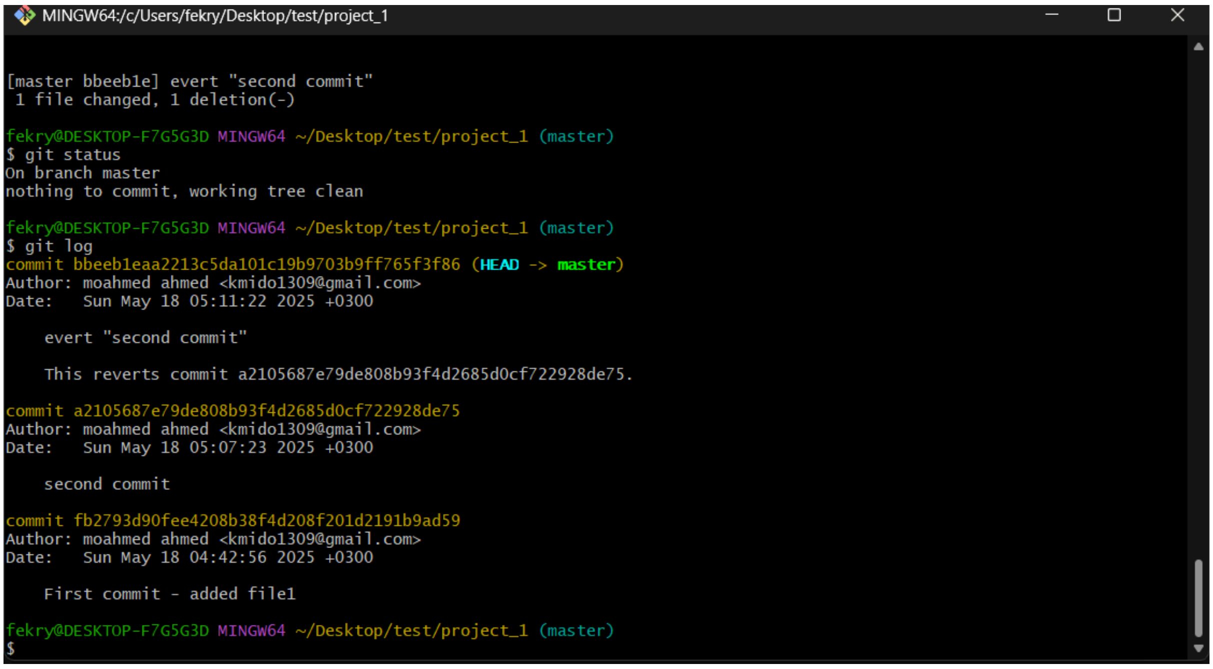
```
MINGW64:/c/Users/fekry/Desktop/test/project_1
Revert "second commit"

This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# Changes to be committed:
#       modified:   file.yml
#
~
.git/COMMIT_EDITMSG [unix] (05:11 18/05/2025)                    1,1 All
```

⇒ enter in the Esc button in your keyboard and write `:wq` and enter



```
~
~
.git/COMMIT_EDITMSG[+] [unix] (05:11 18/05/2025)                 1,1 All
:wq
```

```
git status
git log
```

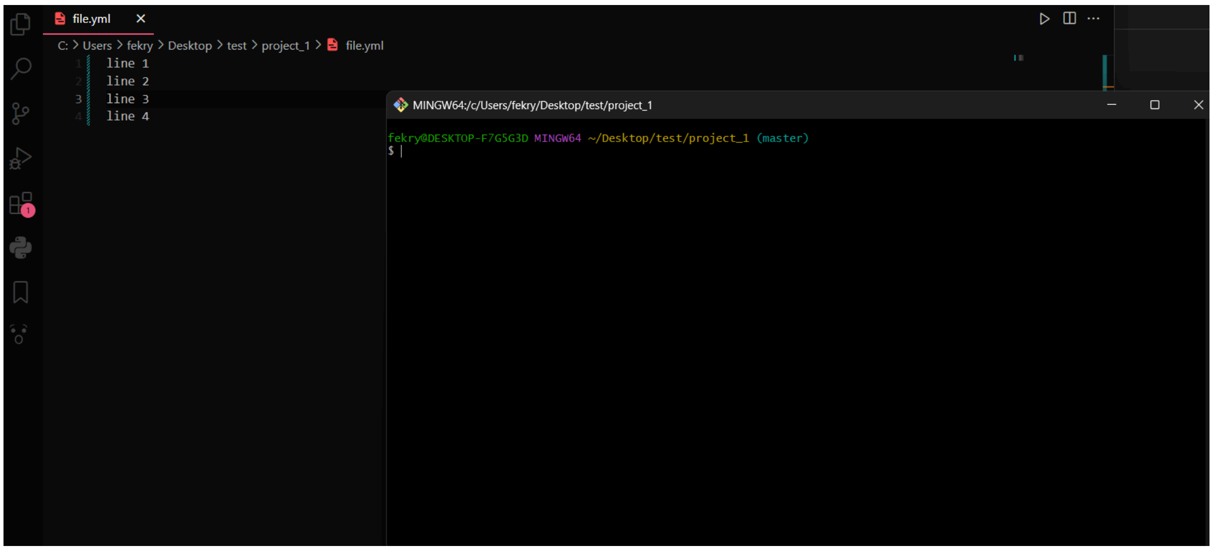


```
MINGW64:/c/Users/fekry/Desktop/test/project_1

[master bbeeb1e] evert "second commit"
 1 file changed, 1 deletion(-)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git status
On branch master
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git log
commit bbeeb1eaa2213c5da101c19b9703b9ff765f3f86 (HEAD -> master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:11:22 2025 +0300

    evert "second commit"

    This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.

commit a2105687e79de808b93f4d2685d0cf722928de75
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:07:23 2025 +0300

    second commit

commit fb2793d90fee4208b38f4d208f201d2191b9ad59
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

Branch & Merge in Action

⇒ we will write anything in master

⇒ and we will take a branch from master



```
file.yml
C: > Users > fekry > Desktop > test > project_1 > file.yml
line 1
line 2
line 3
line 4
```

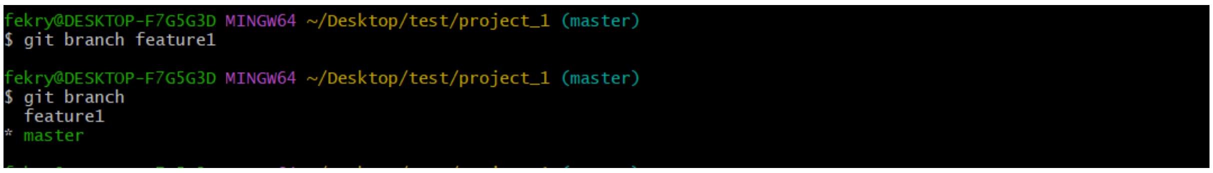
```
MINGW64:/c/Users/fekry/Desktop/test/project_1
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ |
```

⇒ to make a branch we use >> `git branch <branch name>`

```
git branch feature1
```

⇒ we need to know where i am in branches

```
git branch
```

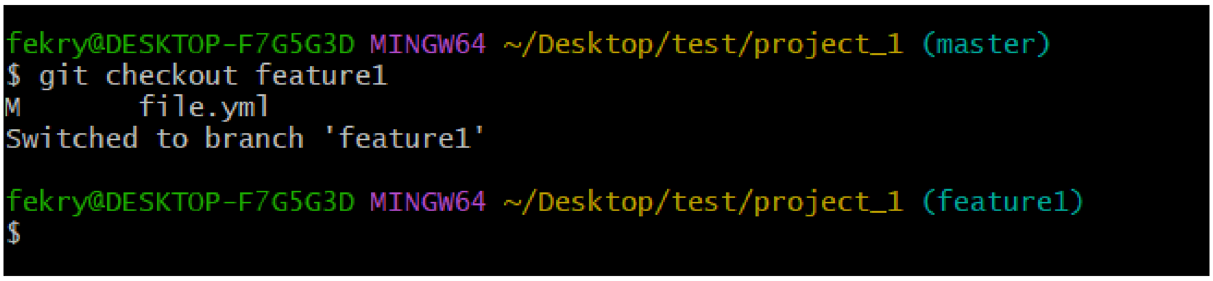


```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git branch feature1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git branch
  feature1
* master
```

⇒ ok we in the master

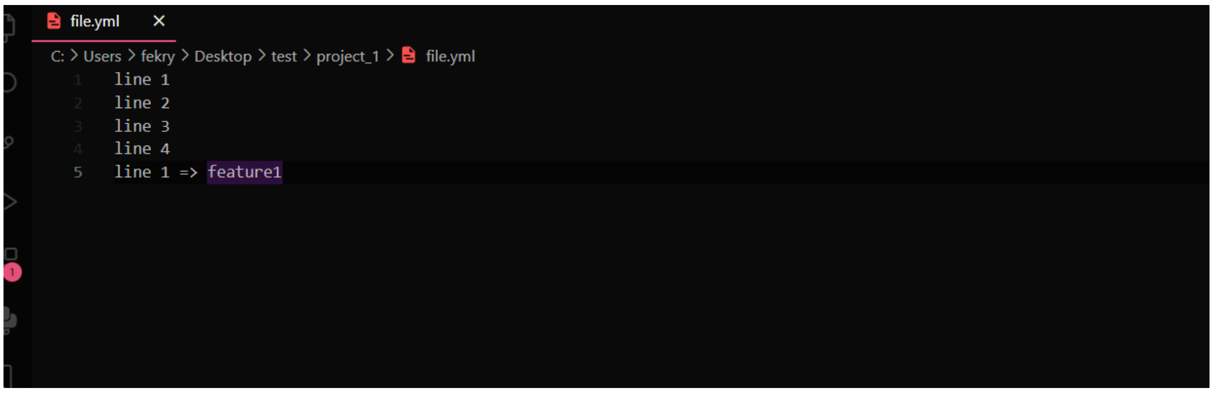
⇒ to move to the specific branch

```
git checkout feature1
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git checkout feature1
M       file.yml
Switched to branch 'feature1'

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$
```

⇒ now we will make a change and add a commit

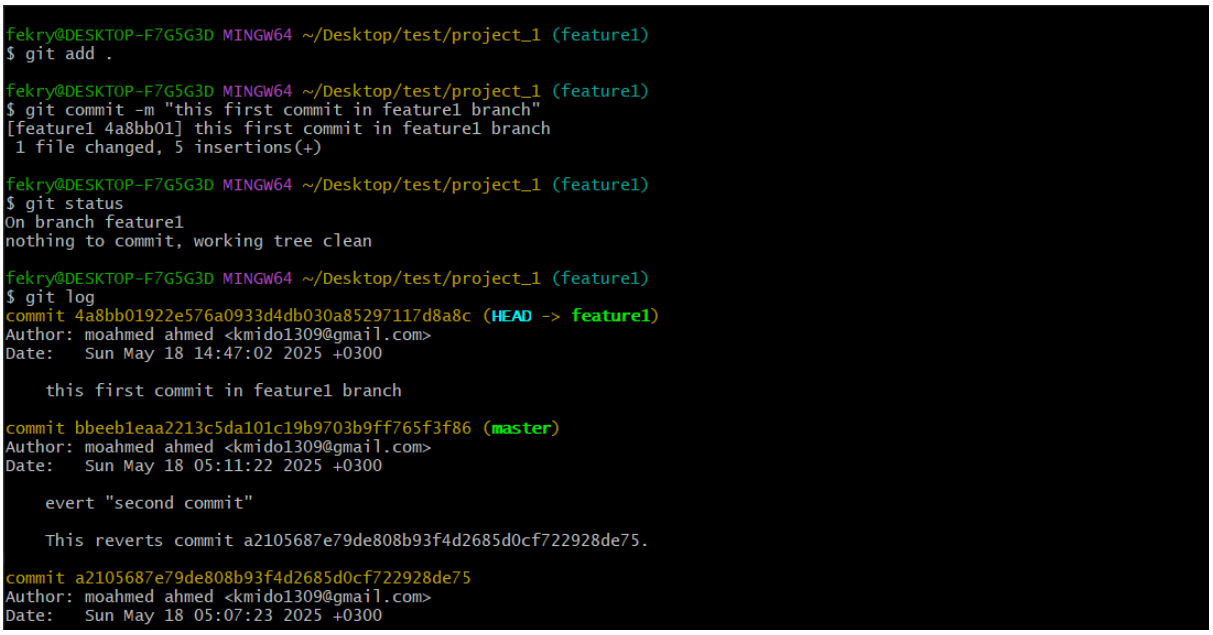


```
line 1
line 2
line 3
line 4
line 1 => feature1
```

```
git add .
```

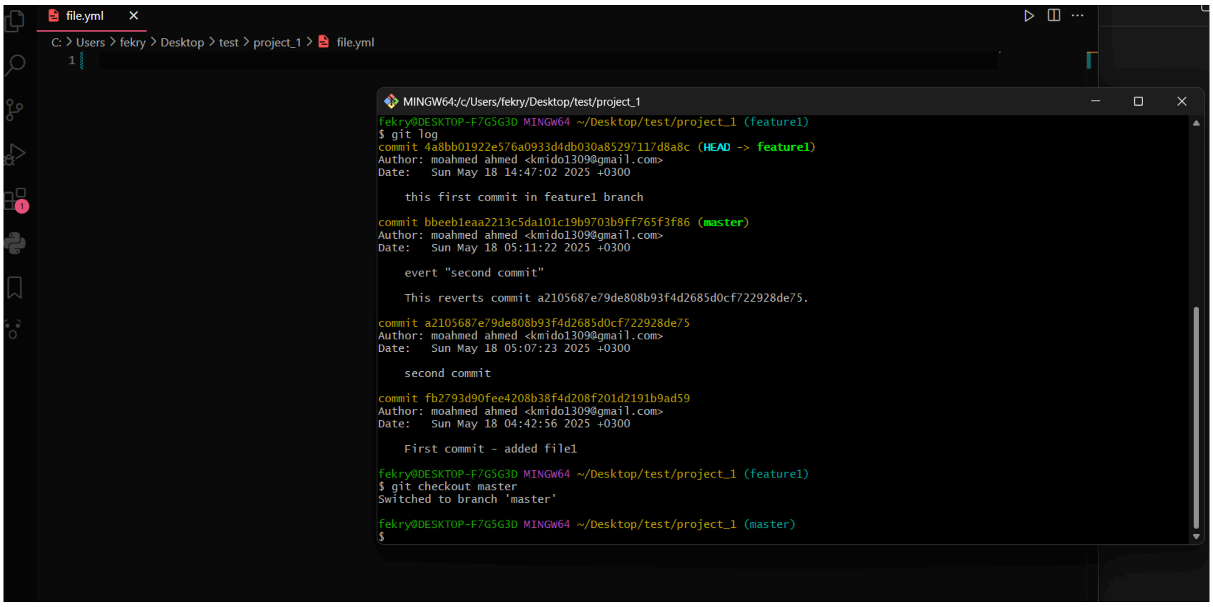
```
git commit -m "this first commit in feature1 branch"
git status
```

```
git log
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git commit -m "this first commit in feature1 branch"
[feature1 4a8bb01] this first commit in feature1 branch
 1 file changed, 5 insertions(+)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git status
On branch feature1
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git log
commit 4a8bb01922e576a0933d4db030a85297117d8a8c (HEAD -> feature1)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 14:47:02 2025 +0300

    this first commit in feature1 branch

commit bbeeb1eaa2213c5da101c19b9703b9ff765f3f86 (master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:11:22 2025 +0300

    evert "second commit"

    This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.

commit a2105687e79de808b93f4d2685d0cf722928de75
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:07:23 2025 +0300
```

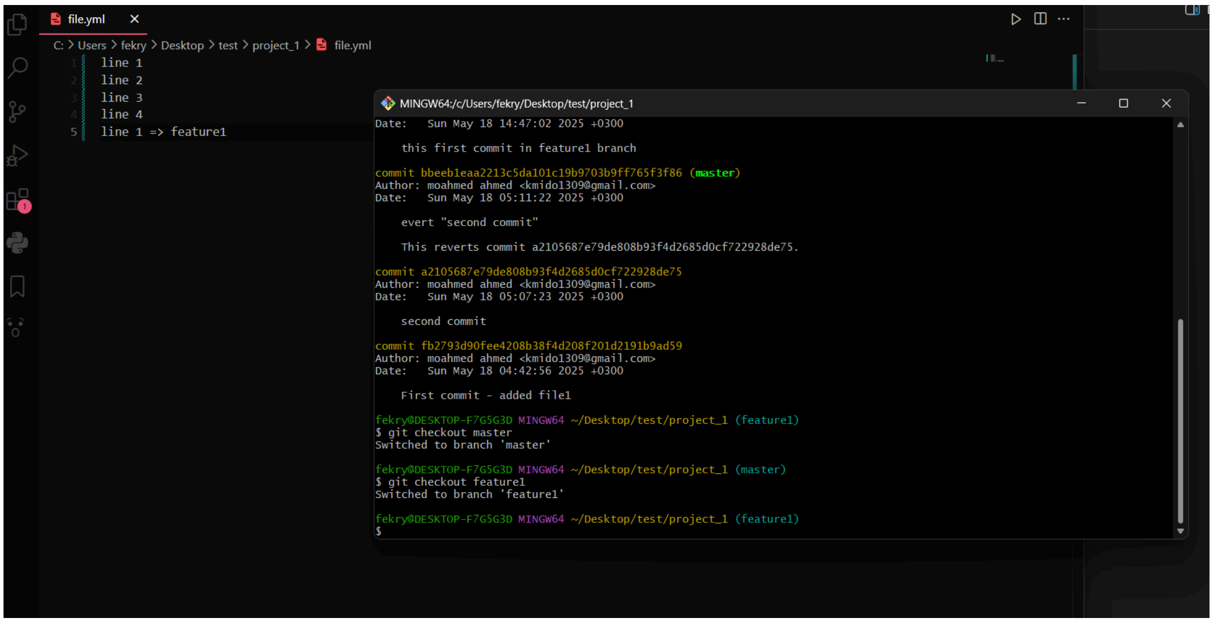
⇒ if we checkout into master the changes in branch didn't be in the master



file.yml
C: > Users > fekry > Desktop > test > project_1 > file.yml
MINGW64:/c/Users/fekry/Desktop/test/project_1

```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git log
commit 4a8bb01922e576a0933d4db030a85297117d8a8c (HEAD -> feature1)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 14:47:02 2025 +0300

    this first commit in feature1 branch

commit bbeeb1eaa2213c5da101c19b9703b9ff765f3f86 (master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:11:22 2025 +0300

    evert "second commit"

    This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.

commit a2105687e79de808b93f4d2685d0cf722928de75
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:07:23 2025 +0300

    second commit

commit fb2793d90fee4208b38f4d208f201d2191b9ad59
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git checkout master
Switched to branch 'master'

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```

⇒ if you checkout in the feature1 the changes will return



file.yml

C: > Users > fekry > Desktop > test > project_1 > file.yml

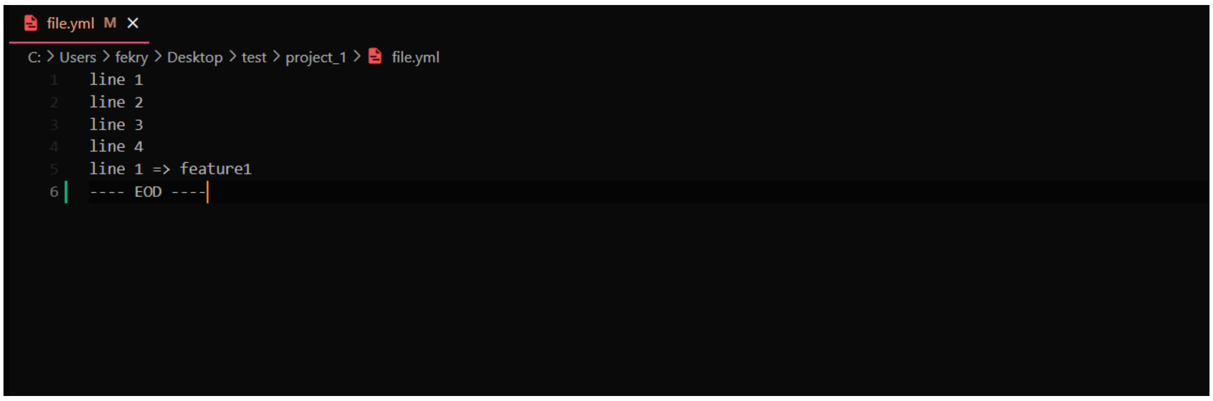
```
line 1
line 2
line 3
line 4
line 1 => feature1
```

MINGW64:/c/Users/fekry/Desktop/test/project_1

```
Date:   Sun May 18 14:47:02 2025 +0300

    this first commit in feature1 branch

commit bbeeb1eaa2213c5da101c19b9703b9ff765f3f86 (master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:11:22 2025 +0300

    evert "second commit"

    This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.

commit a2105687e79de808b93f4d2685d0cf722928de75
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:07:23 2025 +0300

    second commit

commit fb2793d90fee4208b38f4d208f201d2191b9ad59
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git checkout master
Switched to branch 'master'

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git checkout feature1
Switched to branch 'feature1'

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$
```

⇒ now we will make a merge

⇒ for example we will finish the development ad this to file and make a new commit



file.yml M

C: > Users > fekry > Desktop > test > project_1 > file.yml

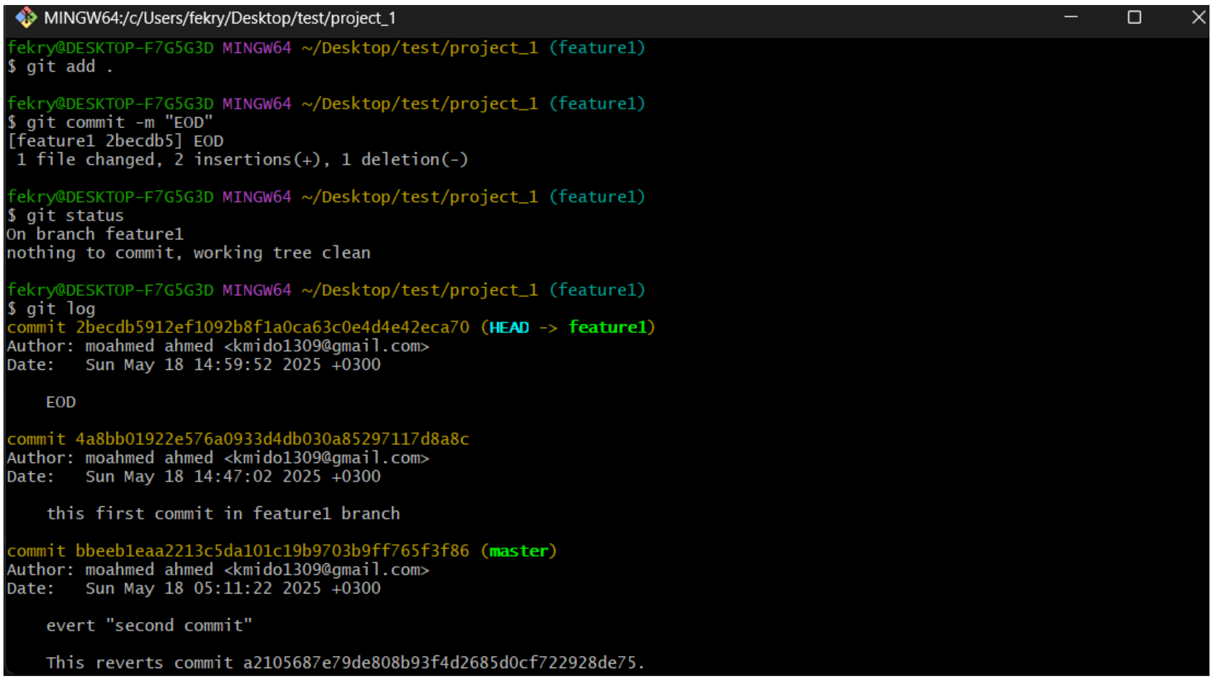
```
line 1
line 2
line 3
line 4
line 1 => feature1
---- EOD ----
```

```
git add .
```

```
git commit "EOD"
```

```
git status
```

```
git log
```



```
MINGW64:/c/Users/fekry/Desktop/test/project_1
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git commit -m "EOD"
[feature1 2becdb5] EOD
 1 file changed, 2 insertions(+), 1 deletion(-)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git status
On branch feature1
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git log
commit 2becdb5912ef1092b8f1a0ca63c0e4d4e42eca70 (HEAD -> feature1)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 14:59:52 2025 +0300

    EOD

commit 4a8bb01922e576a0933d4db030a85297117d8a8c
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 14:47:02 2025 +0300

    this first commit in feature1 branch

commit bbeeb1eaa2213c5da101c19b9703b9ff765f3f86 (master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:11:22 2025 +0300

    evert "second commit"

    This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.
```

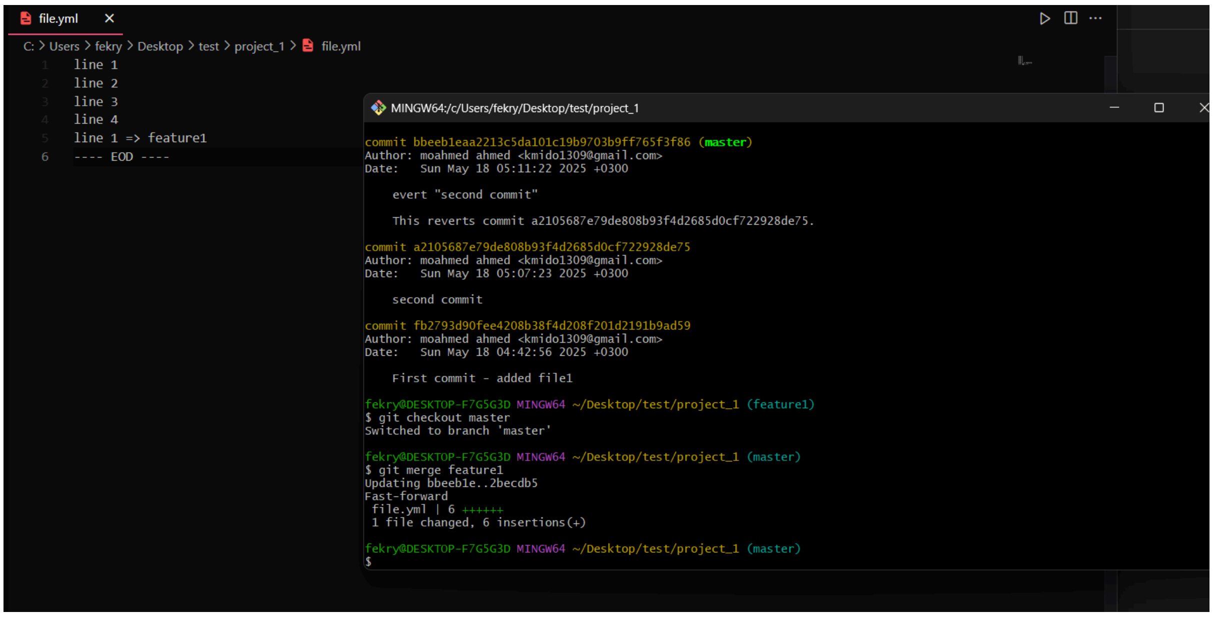
⇒ now we will merge feature1 in master to add the changes into master

⇒ first checkout into master

```
git checkout master
```

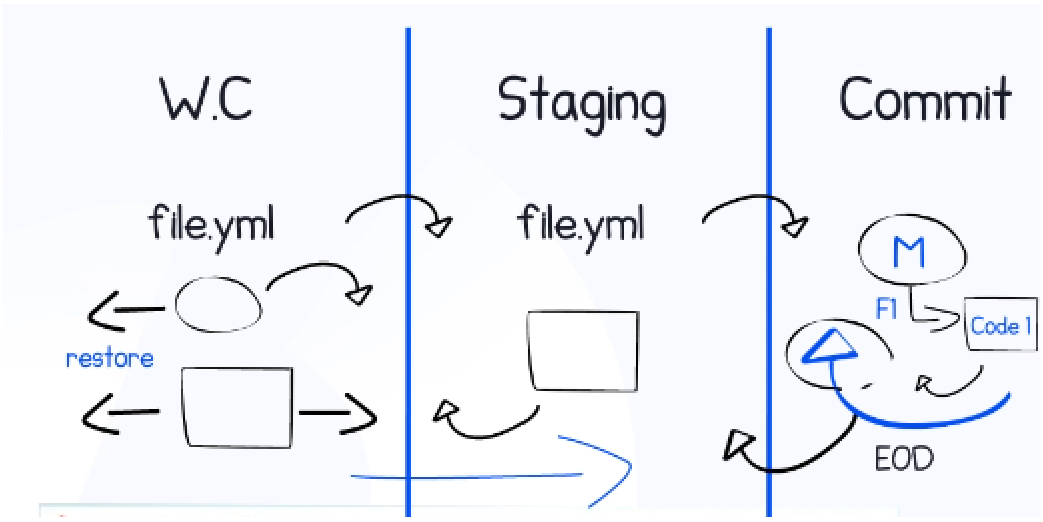
⇒ now merge feature1 with master

```
git merge feature1
```



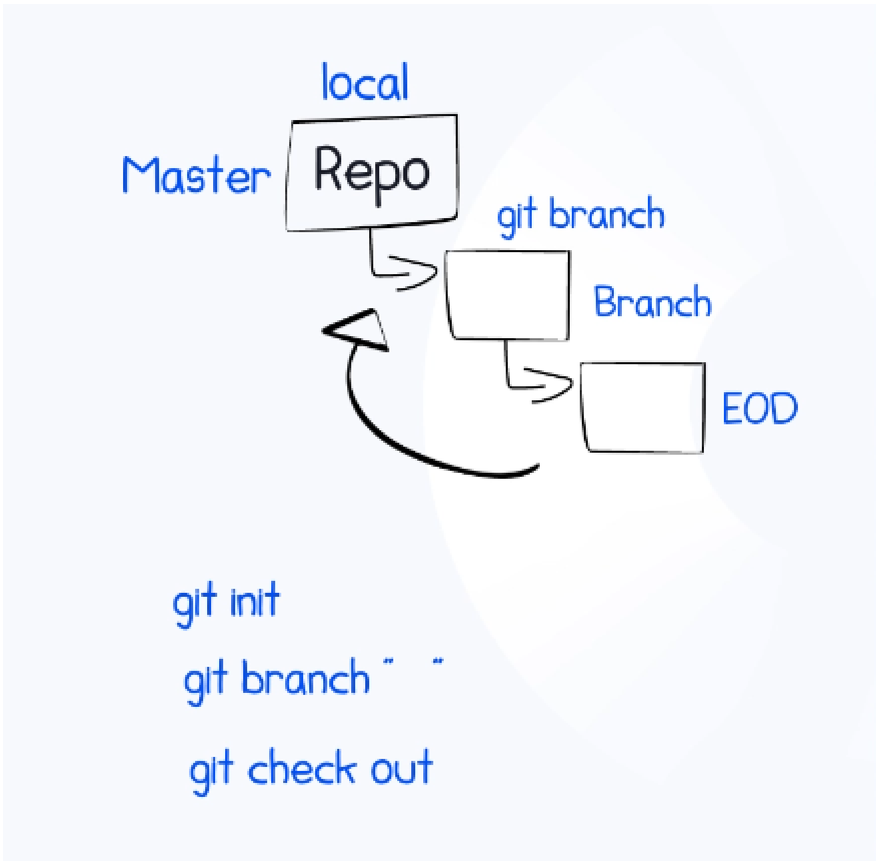
```
file.yml
C: > Users > fekry > Desktop > test > project_1 > file.yml
line 1
line 2
line 3
line 4
line 1 => feature1
---- EOD ----
```

```
MINGW64:/c/Users/fekry/Desktop/test/project_1
commit bbeeb1eaa2213c5da101c19b9703b9ff765f3f86 (master)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:11:22 2025 +0300

    evert "second commit"

    This reverts commit a2105687e79de808b93f4d2685d0cf722928de75.

commit a2105687e79de808b93f4d2685d0cf722928de75
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 05:07:23 2025 +0300

    second commit

commit fb2793d90fee4208b38f4d208f201d2191b9ad59
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 04:42:56 2025 +0300

    First commit - added file1

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (feature1)
$ git checkout master
Switched to branch 'master'

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$ git merge feature1
Updating bbeeb1e..2becdb5
Fast-forward
 file.yml | 6 ++++++
 1 file changed, 6 insertions(+)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/project_1 (master)
$
```



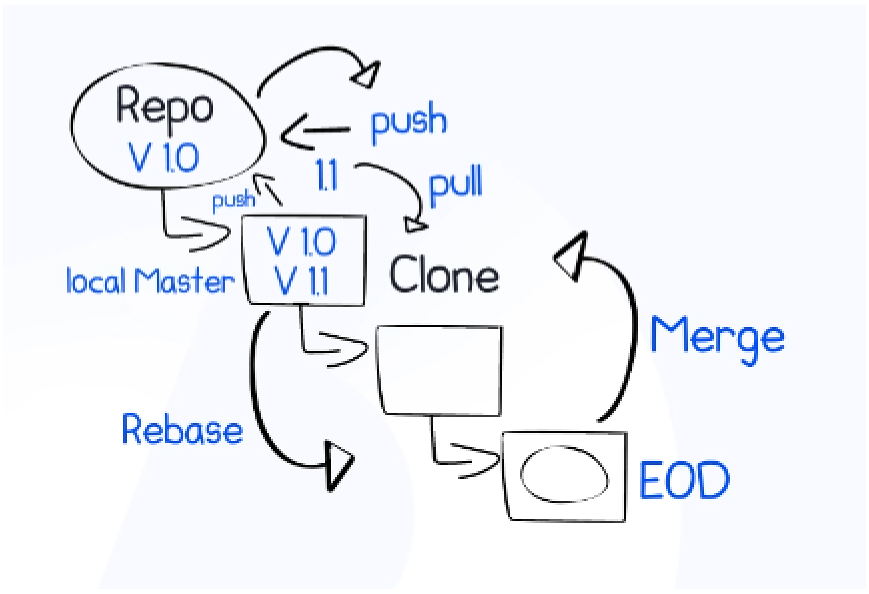
⇒ the all changes will be in master

Merge Conflicts

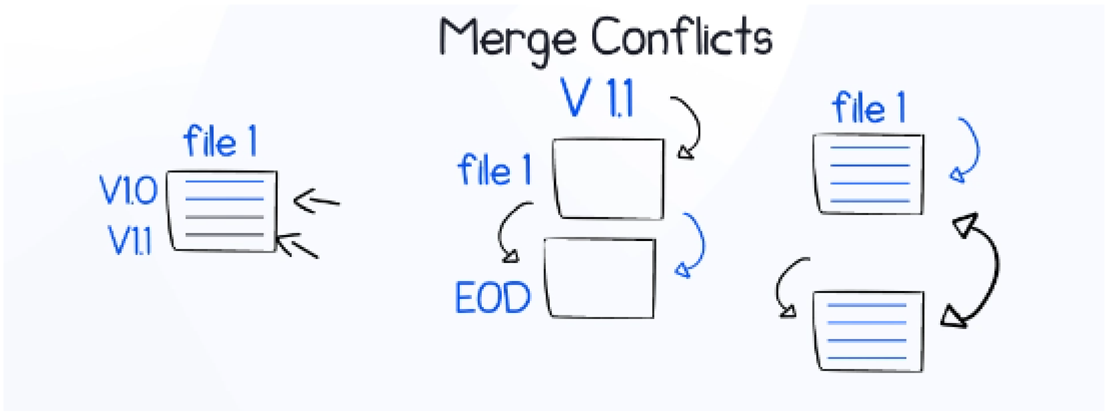


⇒ this is the steps that we made it in local

⇒ but in cloud it same steps but had some differents

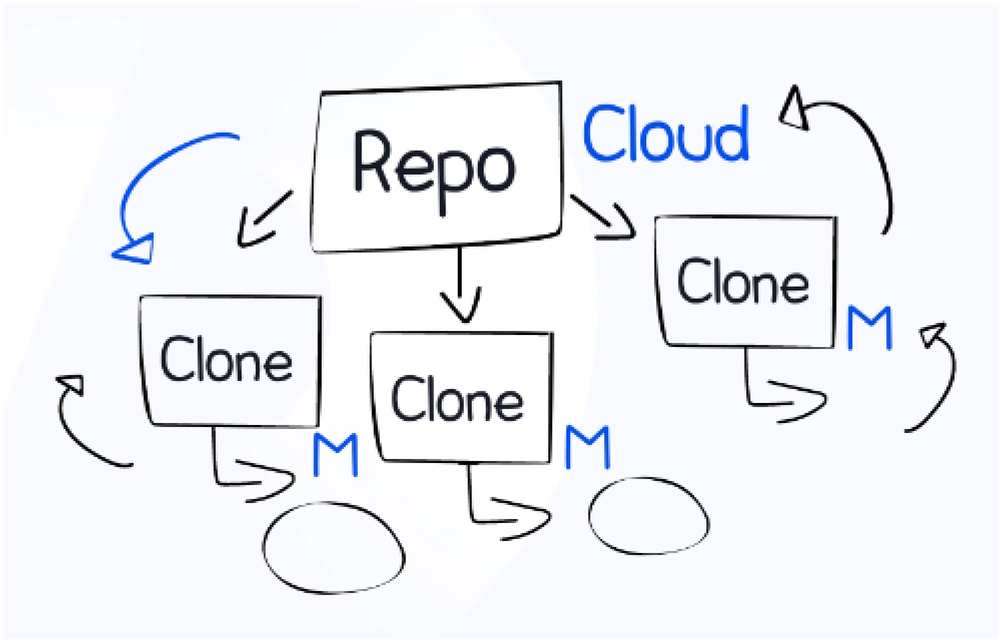


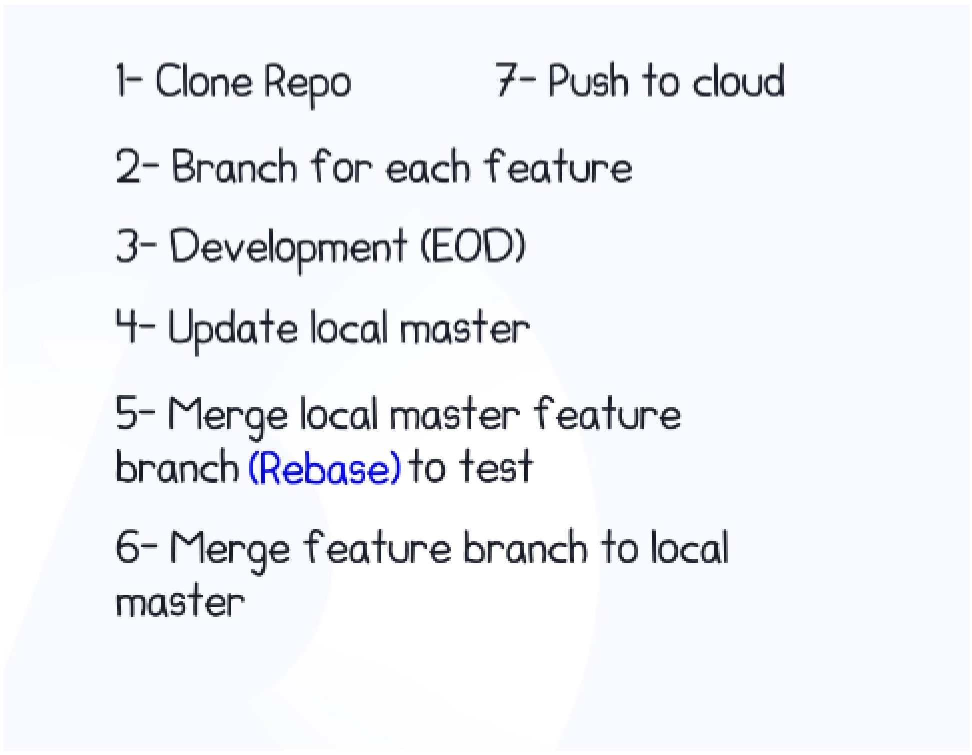
⇛ A
merge conflict : happens in Git when it can’t automatically combine changes from two branches usually because the same part of the same file was edited differently in both branches.



Github in Action

Git work flow





1- Clone Repo

2- Branch for each feature

3- Development (EOD)

4- Update local master

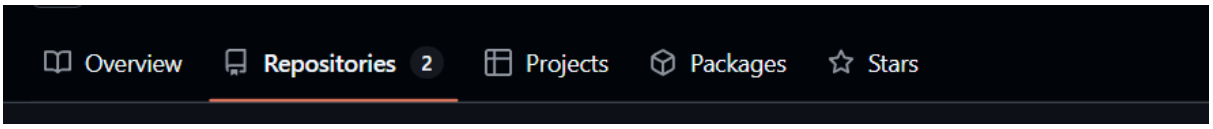
5- Merge local master feature branch (Rebase) to test

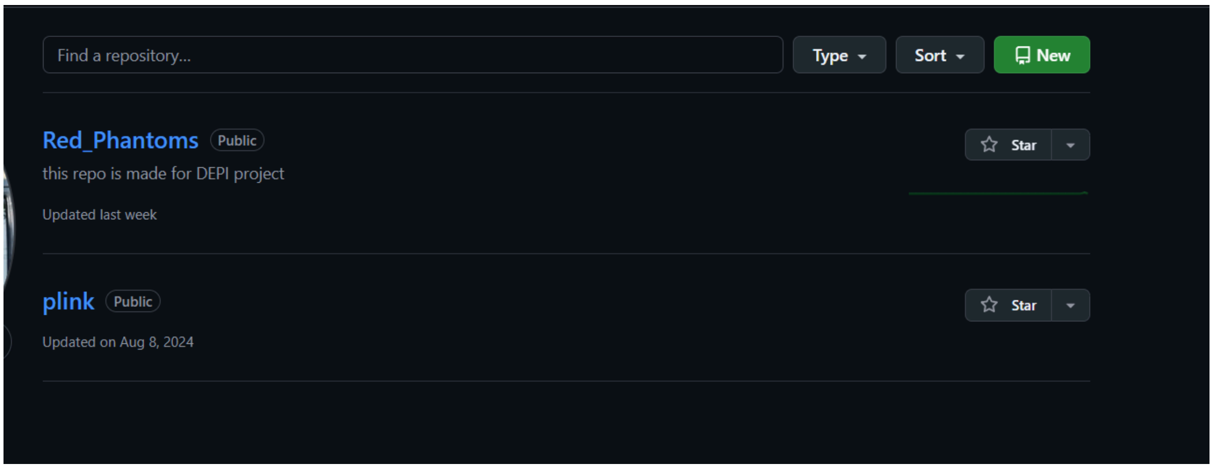
6- Merge feature branch to local master

7- Push to cloud

⇒ first make an account on GitHub >> https://github.com/

⇒ to make a a new repo go to Repository




Find a repository...

Red_Phantoms Public

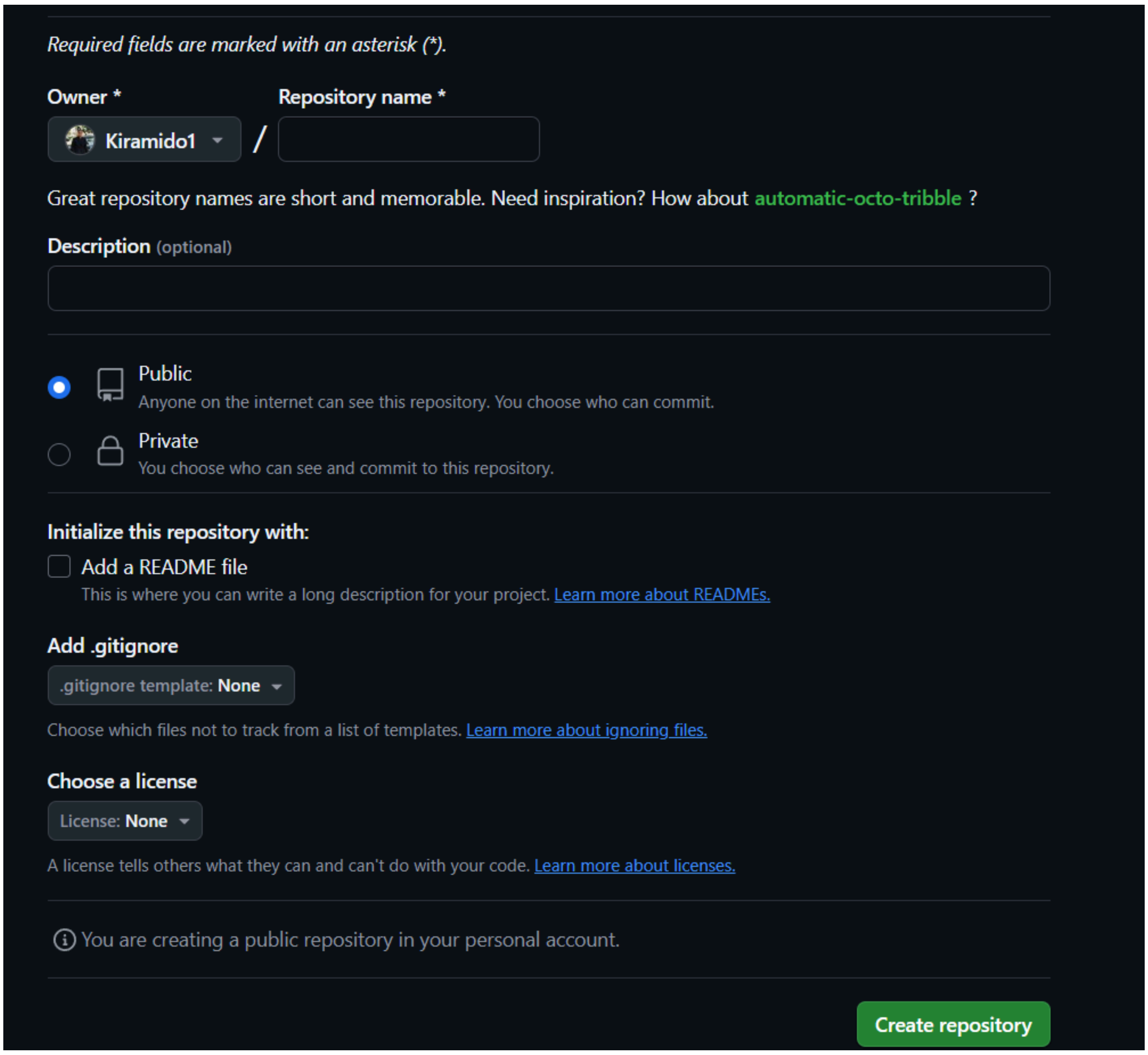
this repo is made for DEPI project

Updated last week

plink Public

Updated on Aug 8, 2024

⇒ chose new


Required fields are marked with an asterisk (*).

Owner * Kiramido1 / Repository name *

Great repository names are short and memorable. Need inspiration? How about automatic-octo-tribble ?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

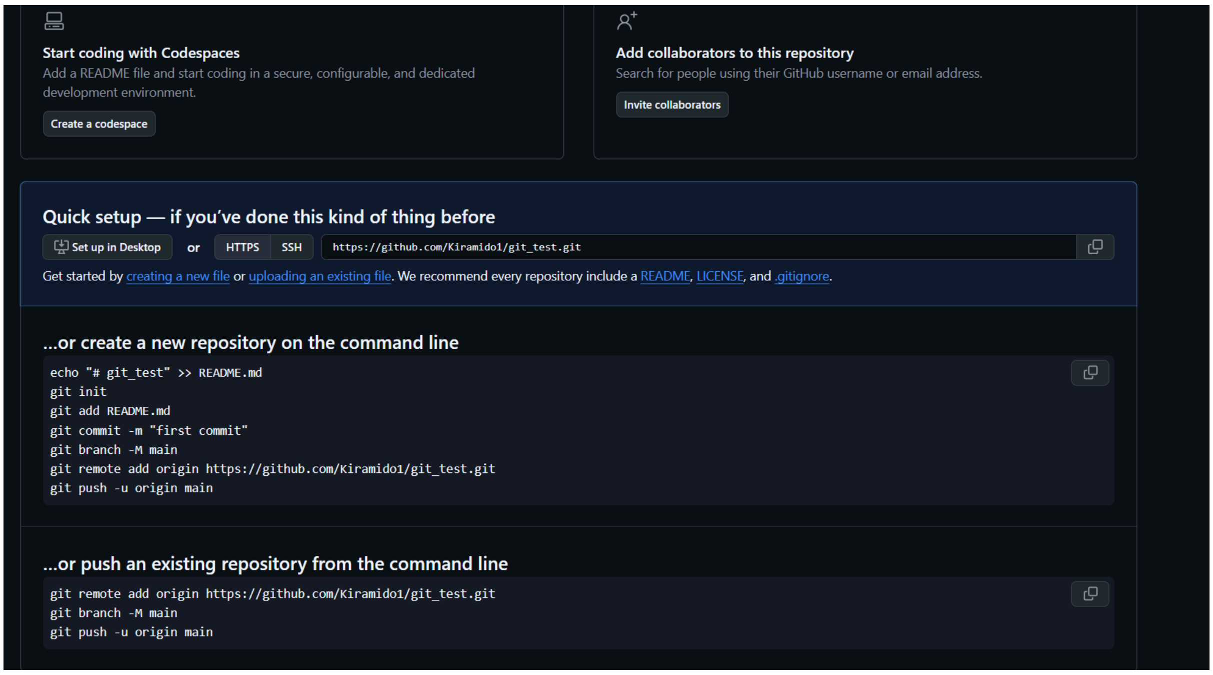
You are creating a public repository in your personal account.

⇒ add a name and description

⇒ and click on a create Repository

⇒ if you didn't understand the steps you can watch this >> https://www.youtube.com/watch?v=nyi8sTqF2m0

⇒ after making a new repo it give you some commands to help you



Start coding with Codespaces
Add a README file and start coding in a secure, configurable, and dedicated development environment.
Create a codespace

Add collaborators to this repository
Search for people using their GitHub username or email address.
Invite collaborators

Quick setup — if you've done this kind of thing before
Set up in Desktop or HTTPS SSH https://github.com/Kiramido1/git_test.git
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

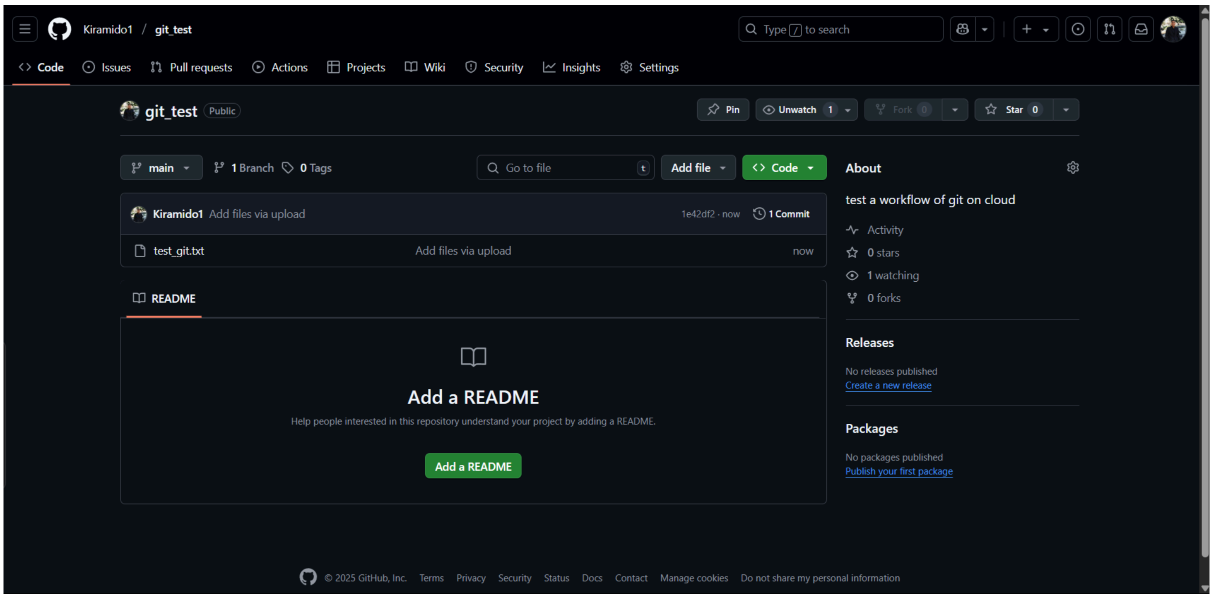
...or create a new repository on the command line

```
echo "# git_test" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Kiramido1/git_test.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/Kiramido1/git_test.git
git branch -M main
git push -u origin main
```

⇒ now upload a txt file for test

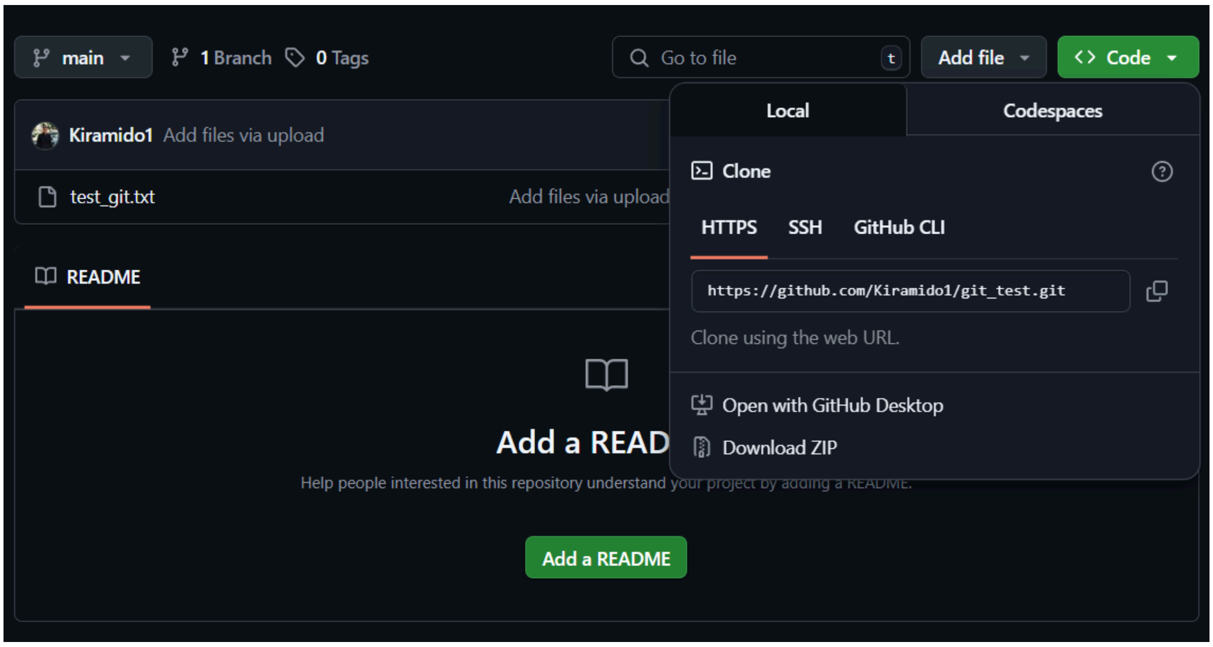


Kiramido1 / git_test
git_test Public
main · 1 Branch · 0 Tags
Kiramido1 Add files via upload · 1e42df2 · now · 1 Commit
test_git.txt · Add files via upload · now
README
Add a README
Help people interested in this repository understand your project by adding a README.
Add a README
About
test a workflow of git on cloud
Activity
0 stars
1 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package

⇒ now first step we need to clone repo

⇒ we will copy the link of repo

⇒ click on code and copy the link

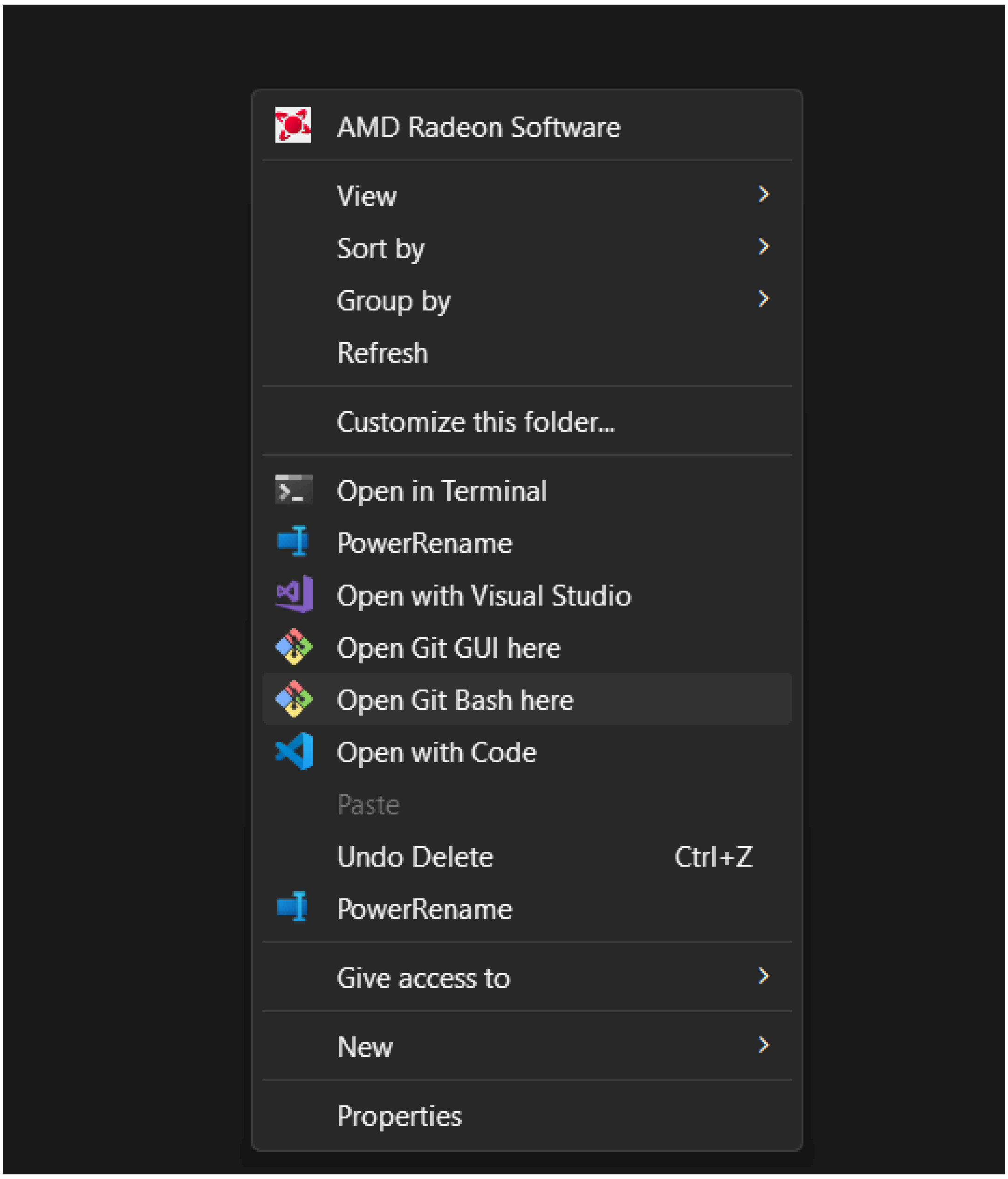


```
https://github.com/Kiramido1/git_test.git
```

Clone using the web URL.

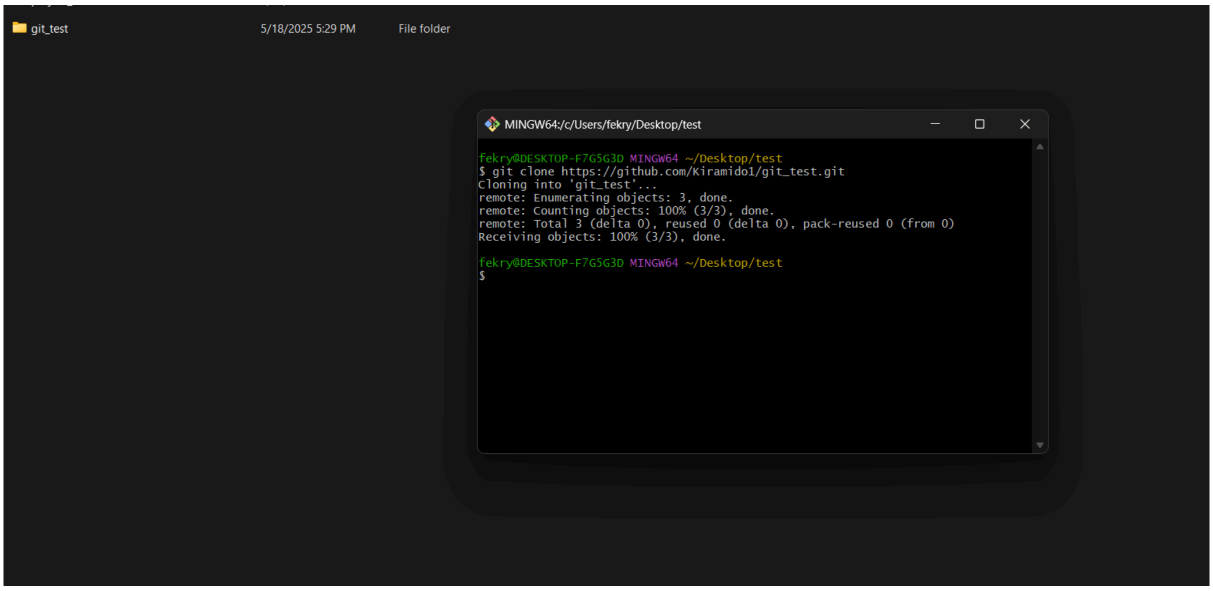
Help people interested in this repository understand your project by adding a README.

⇒ after coping the link going to folder and open the terminal of git (git bash)



⇒ now we will git clone the repo >> git clone <link of repo>

```
git clone https://github.com/Kiramido1/git_test.git
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test
$ git clone https://github.com/Kiramido1/git_test.git
Cloning into 'git_test'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test
$
```

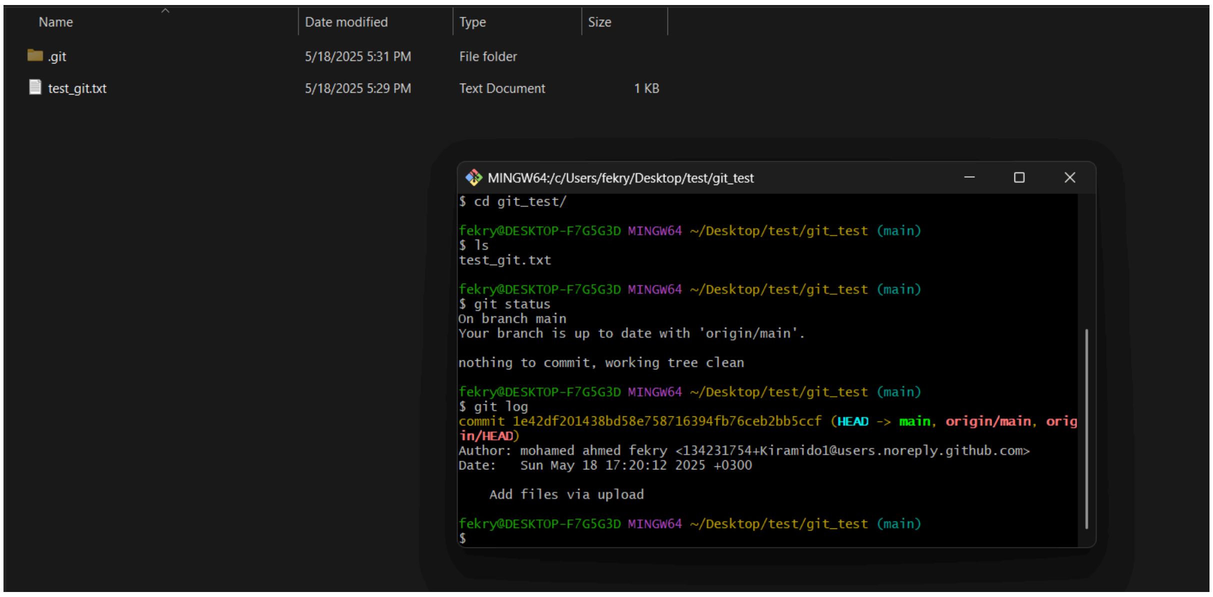
⇒ move inside the folder

```
ls
```

```
cd git_test/
```

```
git status
```

```
git log
```



Name	Date modified	Type	Size
.git	5/18/2025 5:31 PM	File folder	
test_git.txt	5/18/2025 5:29 PM	Text Document	1 KB

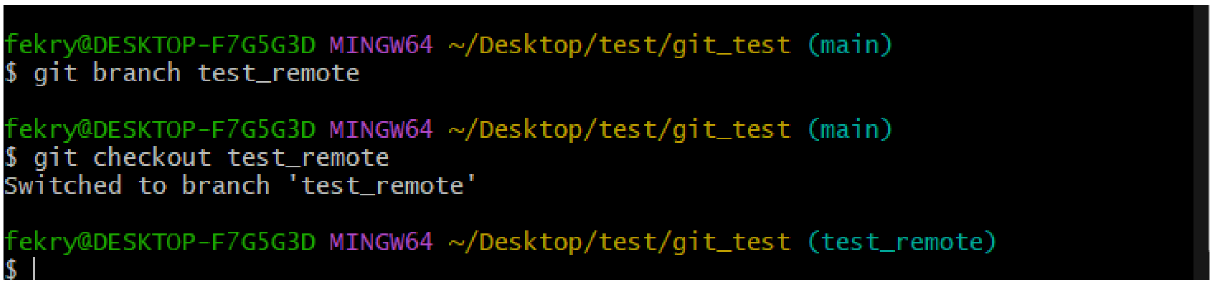
```
$ cd git_test/

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ ls
test_git.txt

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git log
commit 1e42df201438bd58e758716394fb76ceb2bb5ccf (HEAD -> main, origin/main, orig
in/HEAD)
Author: mohamed ahmed fekry <134231754+Kiramido1@users.noreply.github.com>
Date:   Sun May 18 17:20:12 2025 +0300

    Add files via upload

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$
```

⇒ now we will make a branch

```
git branch test_remote
```

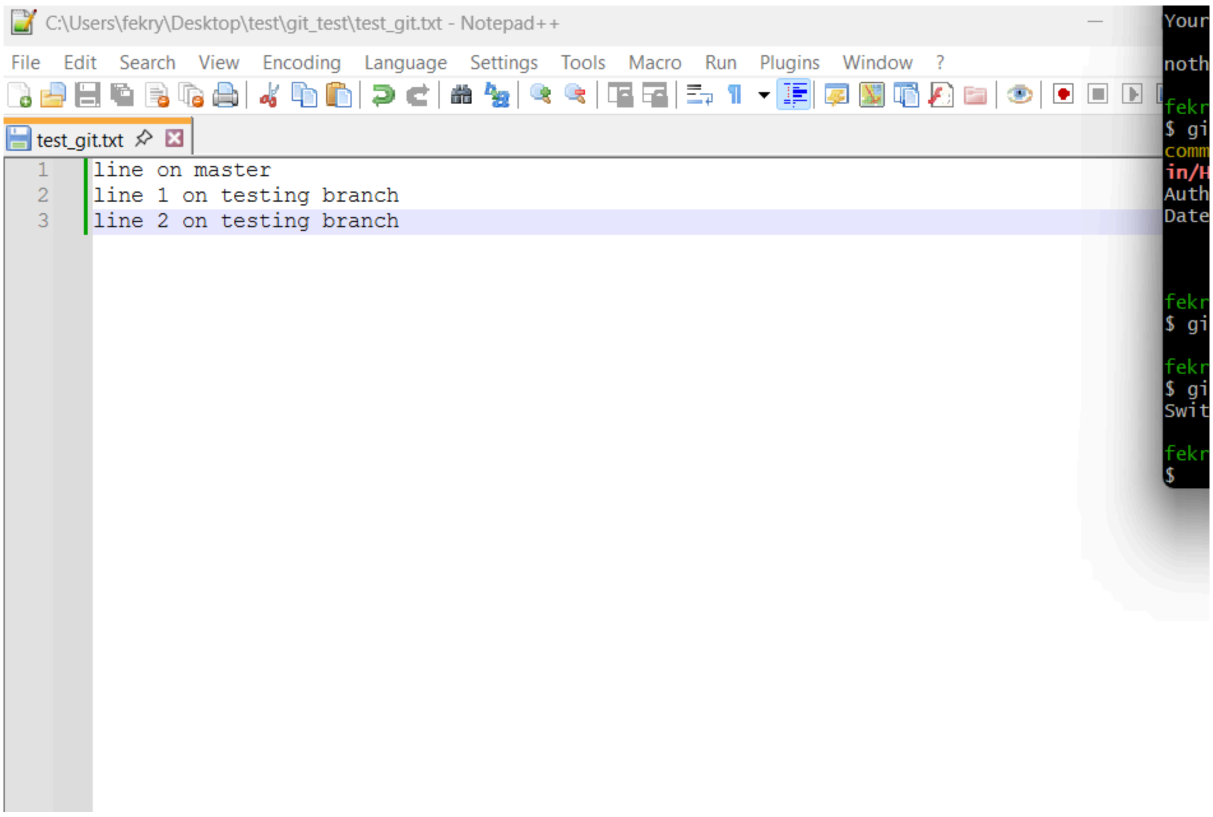
⇒ move to new branch

```
git checkout test_remote
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git branch test_remote

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git checkout test_remote
Switched to branch 'test_remote'

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ |
```

⇒ we will ad some changes



C:\Users\fekry\Desktop\test\git_test\test_git.txt - Notepad++

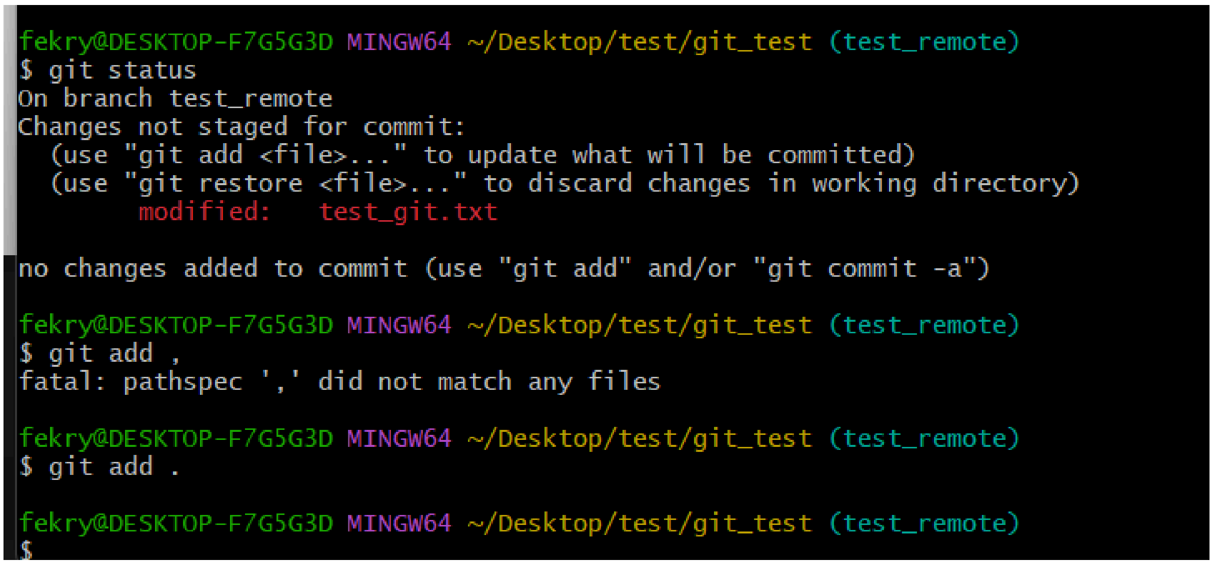
test_git.txt

```
line on master
line 1 on testing branch
line 2 on testing branch
```

```
git status
```

⇒ to add changes

```
git add .
```



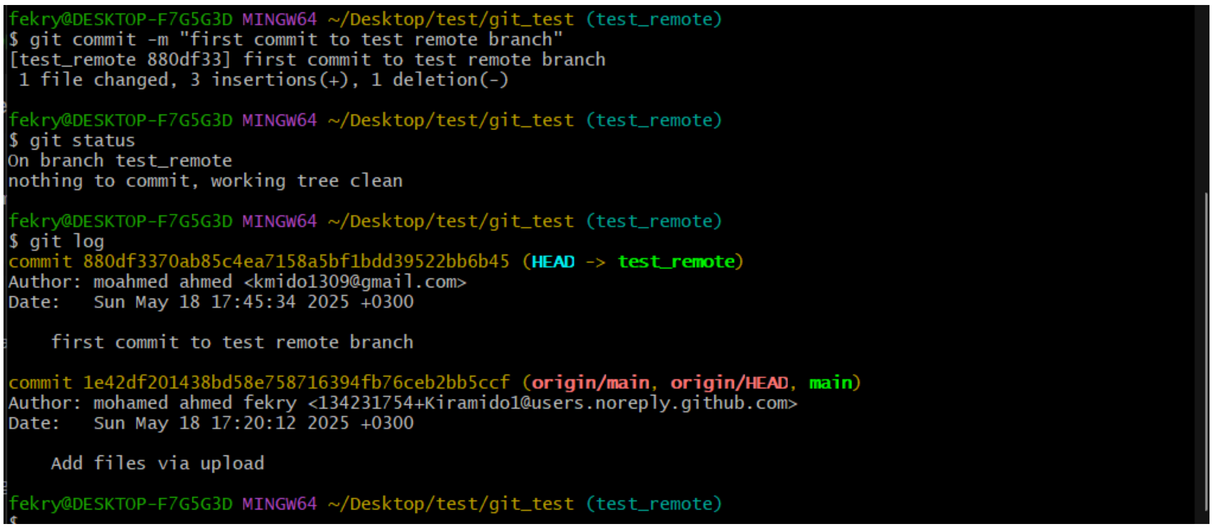
```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git status
On branch test_remote
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test_git.txt

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

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git add ,
fatal: pathspec ',' did not match any files

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$
```

⇒ now we will add a commit

```
git commit -m "first commit to test remote branch"
```

```
git status
```

```
git log
```

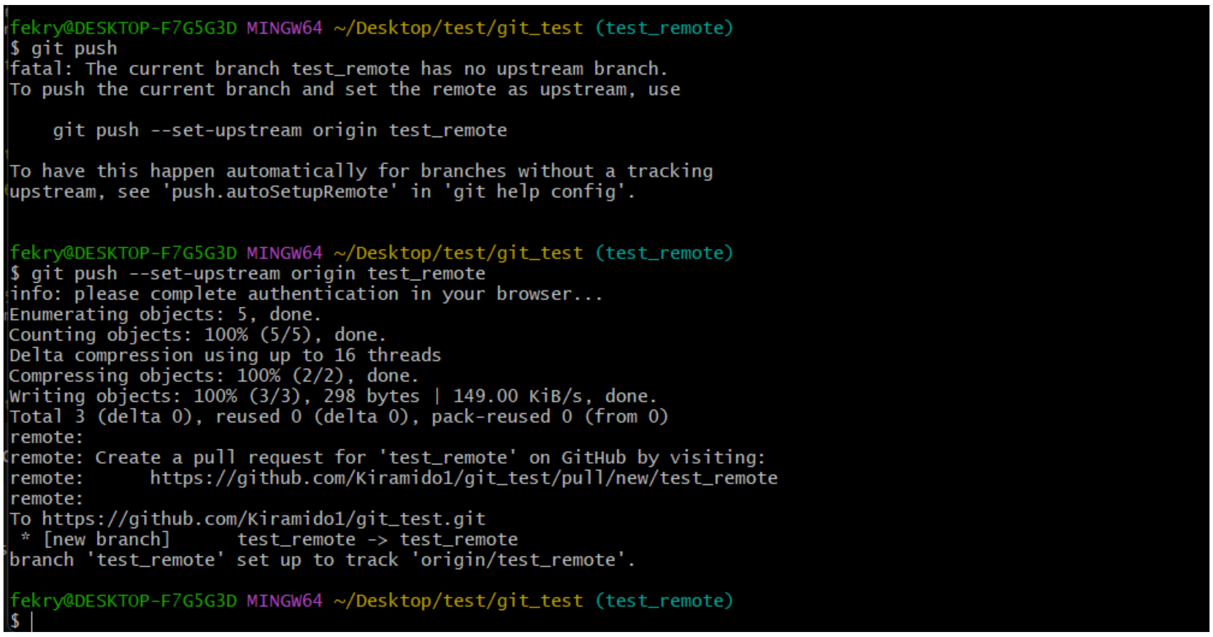


```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git commit -m "first commit to test remote branch"
[test_remote 880df33] first commit to test remote branch
 1 file changed, 3 insertions(+), 1 deletion(-)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git status
On branch test_remote
nothing to commit, working tree clean

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git log
commit 880df3370ab85c4ea7158a5bf1bdd39522bb6b45 (HEAD -> test_remote)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 17:45:34 2025 +0300

    first commit to test remote branch

commit 1e42df201438bd58e758716394fb76ceb2bb5ccf (origin/main, origin/HEAD, main)
Author: mohamed ahmed fekry <134231754+Kiramido1@users.noreply.github.com>
Date:   Sun May 18 17:20:12 2025 +0300

    Add files via upload

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$
```

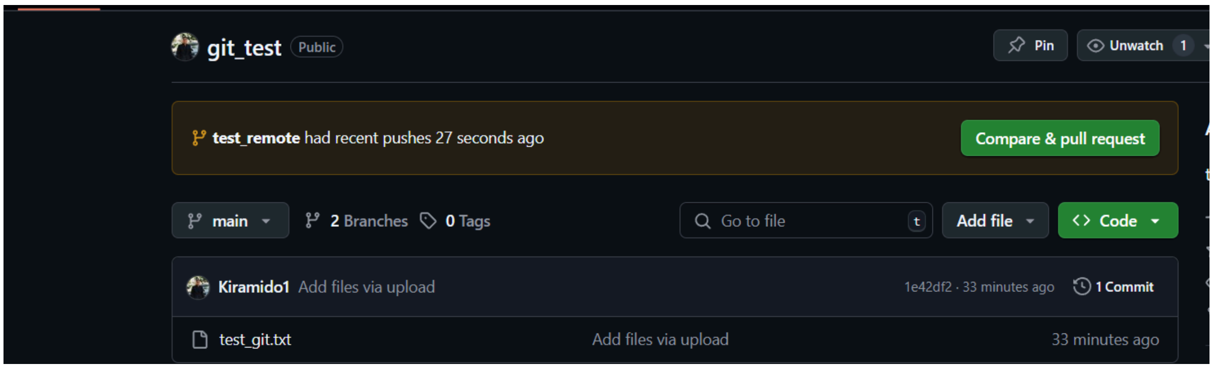
⇒ now we need to push our branch to cloud

```
git push --set-upstream origin test_remote
```

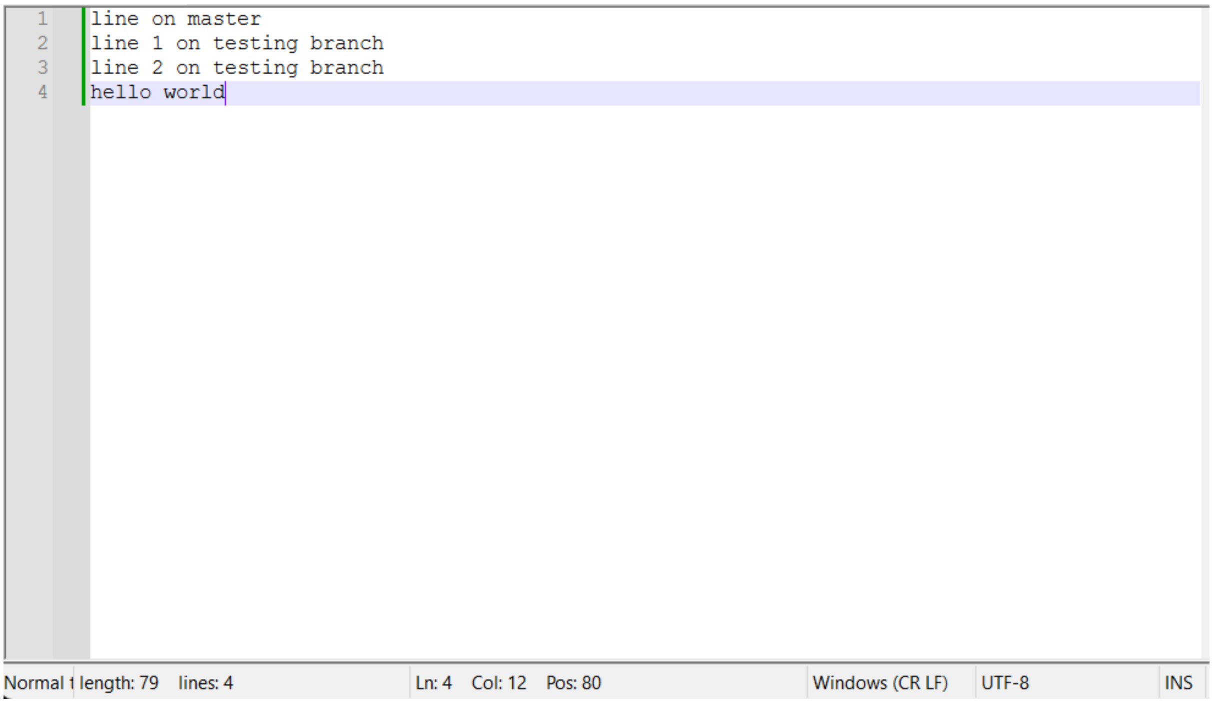
⇒ we write it once in the first push after it we write `git push`



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git push
fatal: The current branch test_remote has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin test_remote

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


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git push --set-upstream origin test_remote
info: please complete authentication in your browser...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 298 bytes | 149.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'test_remote' on GitHub by visiting:
remote:      https://github.com/Kiramido1/git_test/pull/new/test_remote
remote:
To https://github.com/Kiramido1/git_test.git
 * [new branch]      test_remote -> test_remote
branch 'test_remote' set up to track 'origin/test_remote'.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ 
```



⇒ lets make another change to make another commit & push



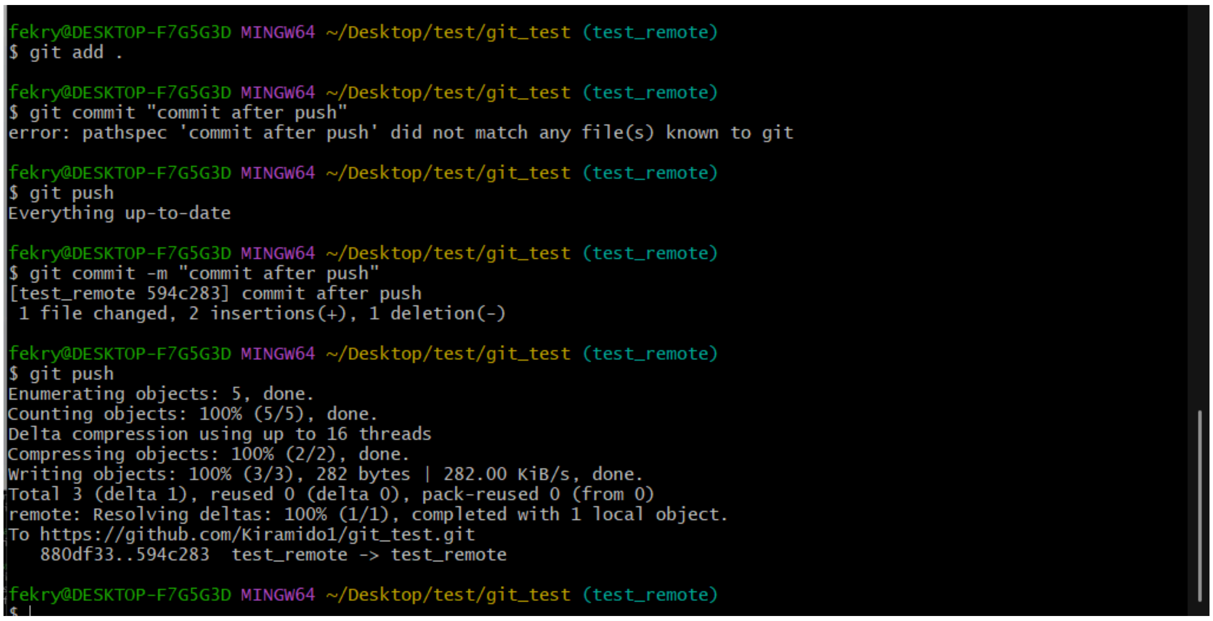
```
line on master
line 1 on testing branch
line 2 on testing branch
hello world
```

Normal t length: 79 lines: 4 Ln: 4 Col: 12 Pos: 80 Windows (CR LF) UTF-8 INS

```
git add .
```

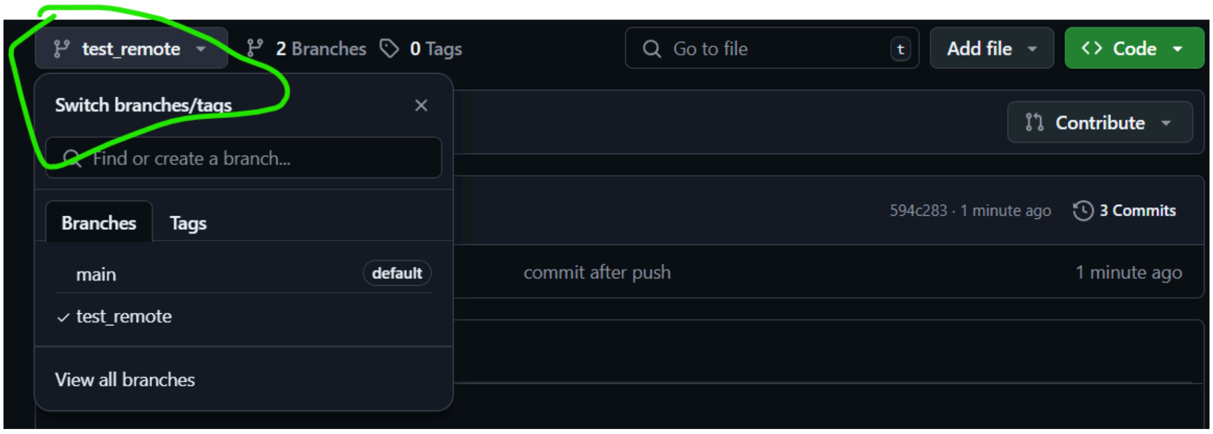
```
git commit "commit aftre push"
```

```
git push
```



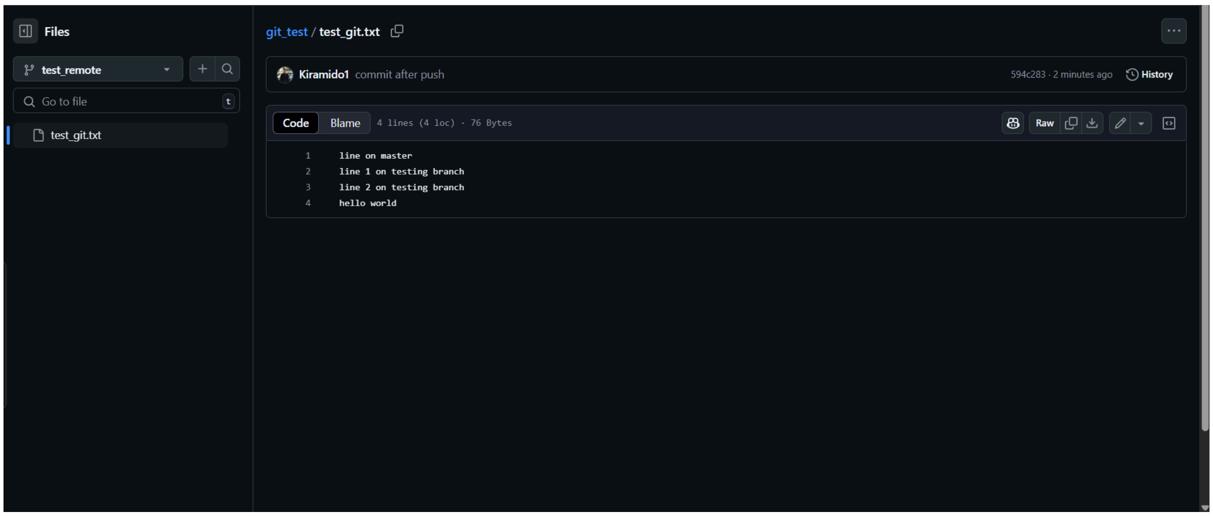
```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git commit "commit after push"
error: pathspec 'commit after push' did not match any file(s) known to git

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git push
Everything up-to-date

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git commit -m "commit after push"
[test_remote 594c283] commit after push
 1 file changed, 2 insertions(+), 1 deletion(-)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 282 bytes | 282.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Kiramido1/git_test.git
   880df33..594c283  test_remote -> test_remote

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ 
```

⇒ we want to see the changes on our branch on GitHub



⇒ after you choose your branch

⇒ open the file



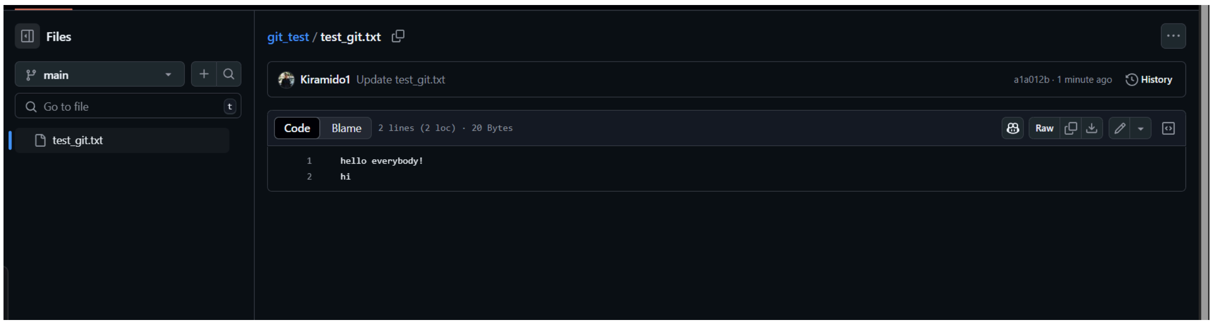
```
line on master
line 1 on testing branch
line 2 on testing branch
hello world
```

⇒ it's our changes

Remote Repo workflow

⇒ now we will update the main file

⇒ before it we will edit in the main repo



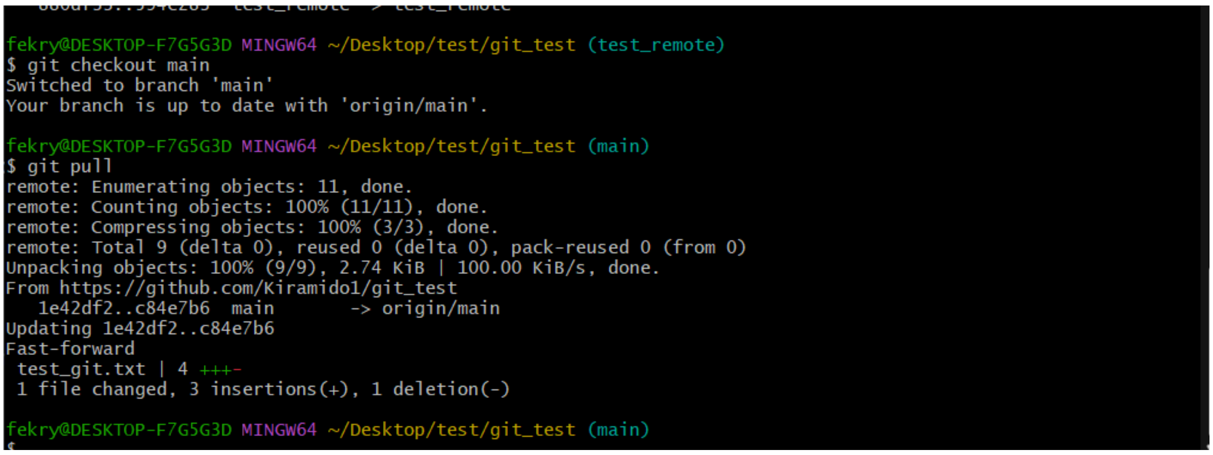
```
hello everybody!
hi
```

⇒ after changes we need to checkout main or master

```
git checkout main
```

⇒ to see the changes from local repo

```
git pull
```

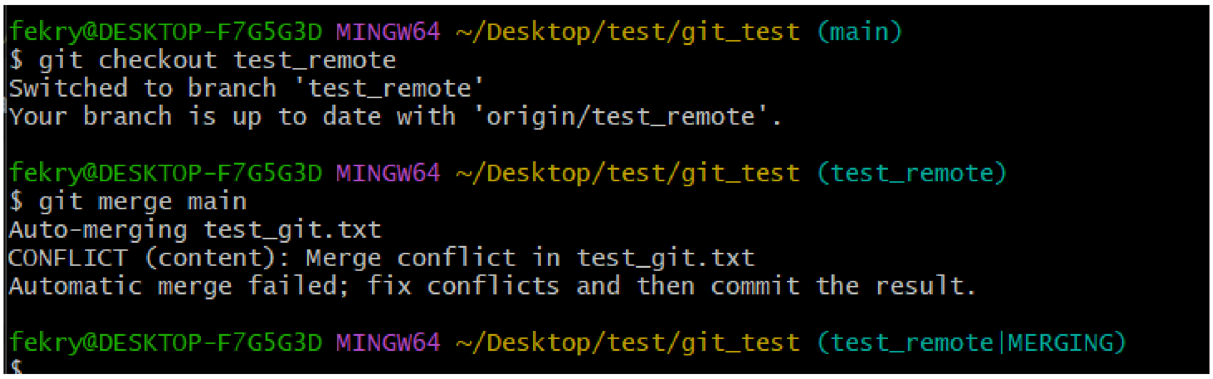


```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git pull
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 9 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (9/9), 2.74 KiB | 100.00 KiB/s, done.
From https://github.com/Kiramido1/git_test
   1e42df2..c84e7b6  main       -> origin/main
Updating 1e42df2..c84e7b6
Fast-forward
 test_git.txt | 4 +++-
 1 file changed, 3 insertions(+), 1 deletion(-)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
```

⇒ now if we merge this with our branch it will be conflict

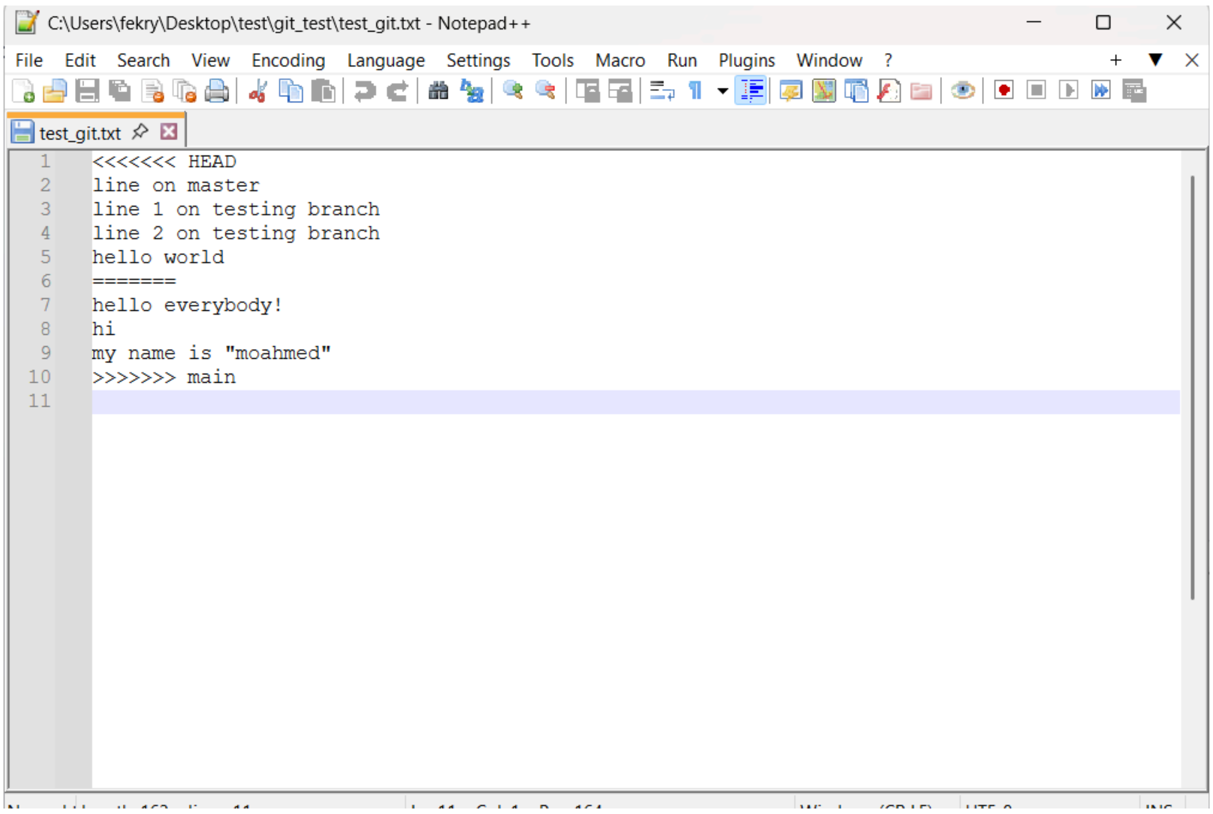
```
git checkout test_remote
```

```
git merge main
```



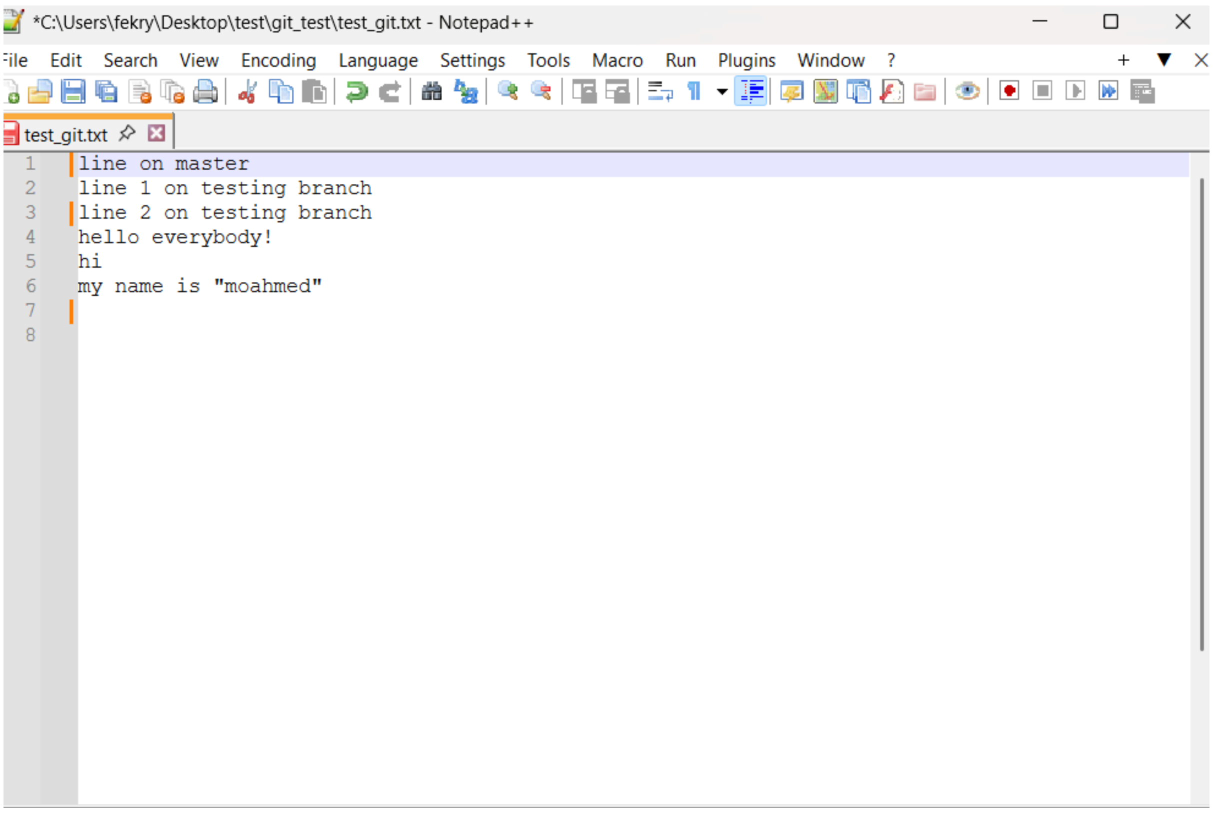
```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git checkout test_remote
Switched to branch 'test_remote'
Your branch is up to date with 'origin/test_remote'.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git merge main
Auto-merging test_git.txt
CONFLICT (content): Merge conflict in test_git.txt
Automatic merge failed; fix conflicts and then commit the result.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote|MERGING)
$
```

⇒ git working with confilict


C:\Users\fekry\Desktop\test\git_test\test_git.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
test_git.txt

```
<<<<<<< HEAD
line on master
line 1 on testing branch
line 2 on testing branch
hello world
=======
hello everybody!
hi
my name is "moahmed"
>>>>>>> main
```

⇒ it give you the changes of main & the main changes on your local repo and you choose


*C:\Users\fekry\Desktop\test\git_test\test_git.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
test_git.txt

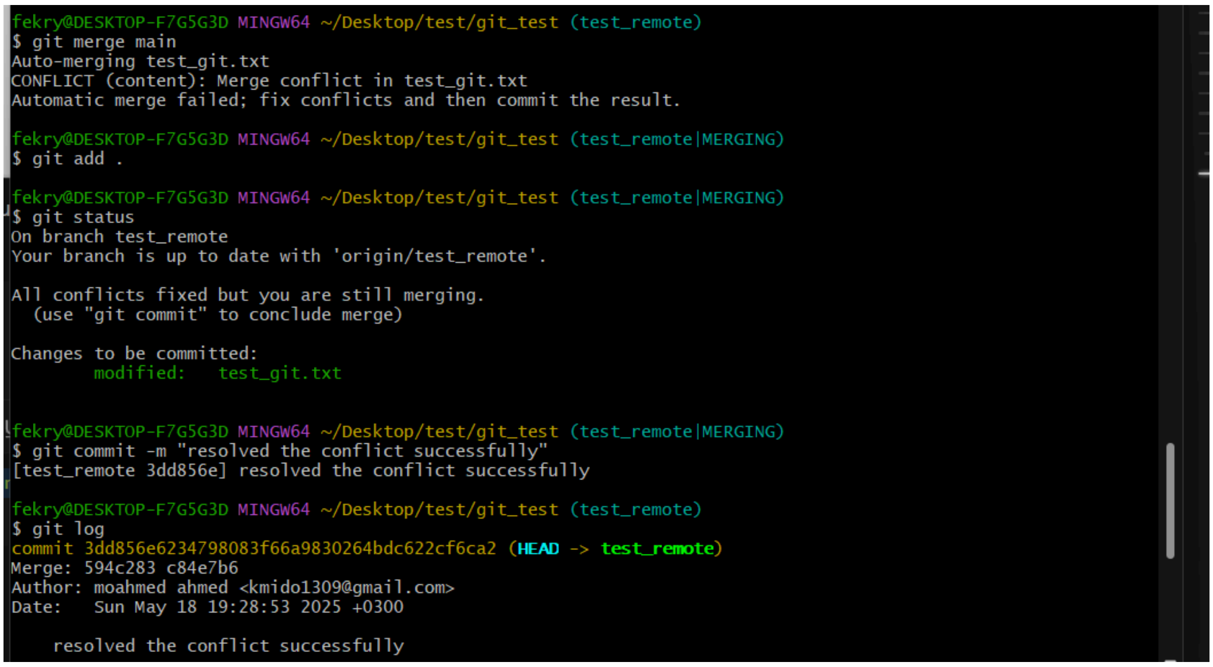
```
line on master
line 1 on testing branch
line 2 on testing branch
hello everybody!
hi
my name is "moahmed"
```

⇒ now we chose the changes manually and delete the rest

⇒ add our changes

```
git add .
```

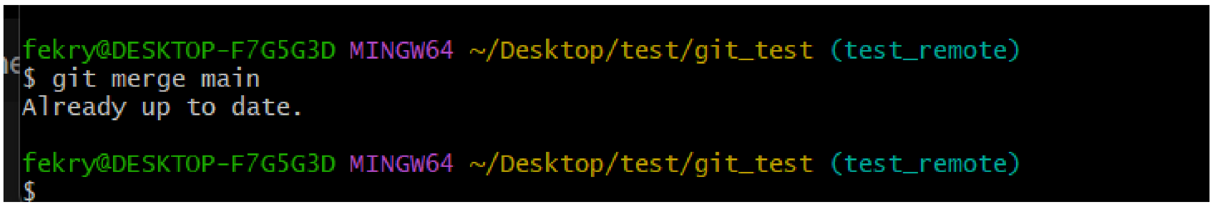
```
git commit -m "resolved the conflict successfully"
```



```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git merge main
Auto-merging test_git.txt
CONFLICT (content): Merge conflict in test_git.txt
Automatic merge failed; fix conflicts and then commit the result.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote|MERGING)
$ git add .

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote|MERGING)
$ git status
On branch test_remote
Your branch is up to date with 'origin/test_remote'.

All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
        modified:   test_git.txt


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote|MERGING)
$ git commit -m "resolved the conflict successfully"
[test_remote 3dd856e] resolved the conflict successfully

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git log
commit 3dd856e6234798083f66a9830264bdc622cf6ca2 (HEAD -> test_remote)
Merge: 594c283 c84e7b6
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 19:28:53 2025 +0300

    resolved the conflict successfully
```

⇒ now after Solvang conflicts we need to merge

```
git merge main
```



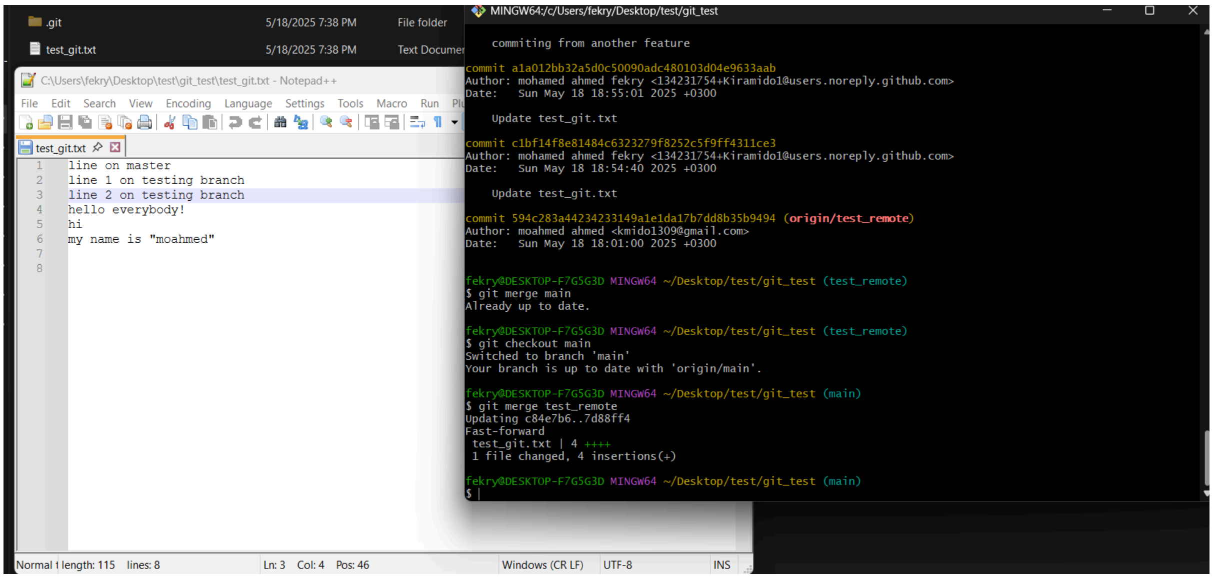
```
fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git merge main
Already up to date.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$
```

⇒ now we checkout to the main

```
git checkout main
```

⇒ we merge to test_remote with main

```
git merge test_remote
```



```
.git            5/18/2025 7:38 PM    File folder
test_git.txt    5/18/2025 7:38 PM    Text Documer
```

C:\Users\fekry\Desktop\test\git_test\test_git.txt - Notepad++

```
line on master
line 1 on testing branch
line 2 on testing branch
hello everybody!
hi
my name is "moahmed"
```

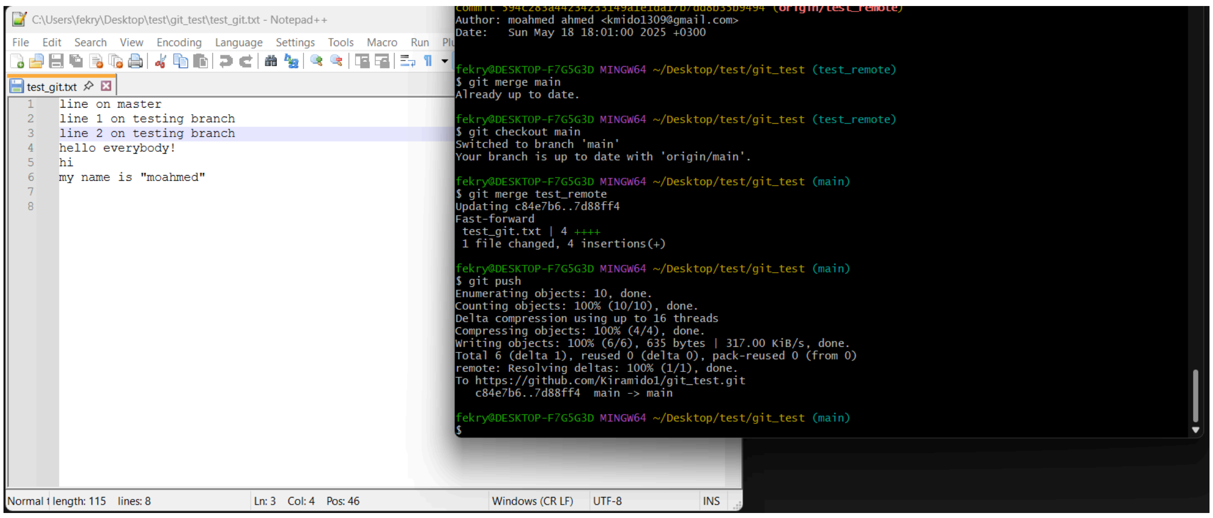
Normal t length: 115 lines: 8 Ln: 3 Col: 4 Pos: 46 Windows (CR LF) UTF-8 INS

MINGW64:/c/Users/fekry/Desktop/test/git_test

```
    commiting from another feature

commit a1a012bb32a5d0c50090adc480103d04e9633aab
Author: mohamed ahmed fekry <134231754+Kiramido1@users.noreply.github.com>
Date:   Sun May 18 18:55:01 2025 +0300

    Update test_git.txt

commit c1bf14f8e81484c6323279f8252c5f9ff4311ce3
Author: mohamed ahmed fekry <134231754+Kiramido1@users.noreply.github.com>
Date:   Sun May 18 18:54:40 2025 +0300

    Update test_git.txt

commit 594c283a44234233149a1e1da17b7dd8b35b9494 (origin/test_remote)
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 18:01:00 2025 +0300


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git merge main
Already up to date.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git merge test_remote
Updating c84e7b6..7d88ff4
Fast-forward
 test_git.txt | 4 ++++
 1 file changed, 4 insertions(+)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ |
```

⇒ now the final step is push

```
git push
```

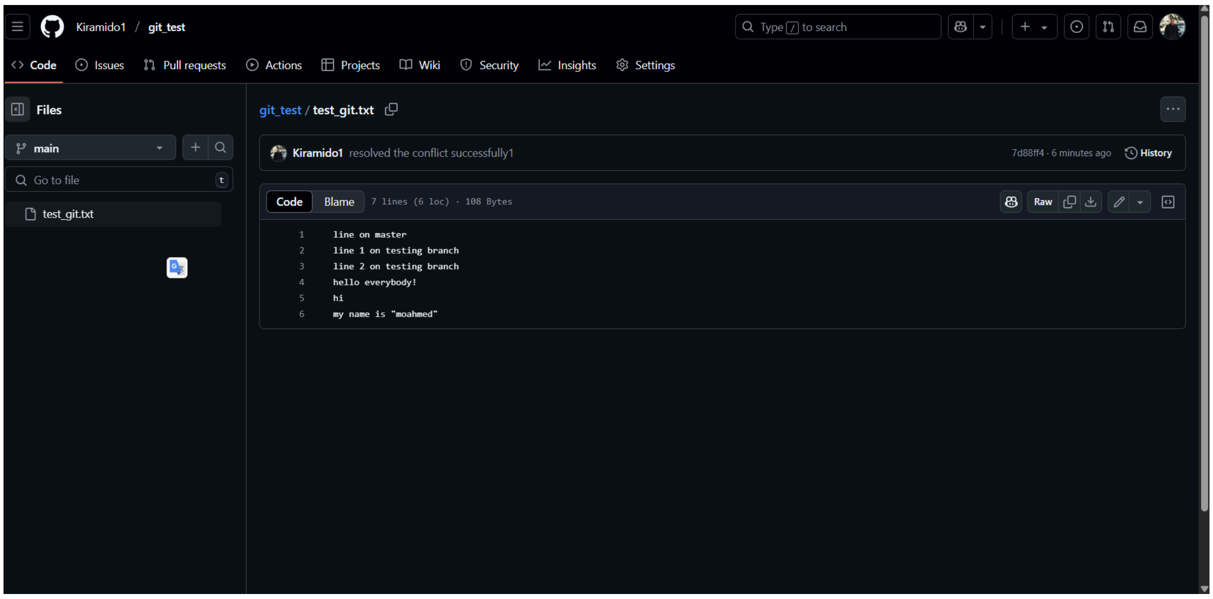


C:\Users\fekry\Desktop\test\git_test\test_git.txt - Notepad++

```
line on master
line 1 on testing branch
line 2 on testing branch
hello everybody!
hi
my name is "moahmed"
```

Normal t length: 115 lines: 8 Ln: 3 Col: 4 Pos: 46 Windows (CR LF) UTF-8 INS

```
Author: moahmed ahmed <kmido1309@gmail.com>
Date:   Sun May 18 18:01:00 2025 +0300


fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git merge main
Already up to date.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (test_remote)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git merge test_remote
Updating c84e7b6..7d88ff4
Fast-forward
 test_git.txt | 4 ++++
 1 file changed, 4 insertions(+)

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$ git push
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 635 bytes | 317.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Kiramido1/git_test.git
   c84e7b6..7d88ff4  main -> main

fekry@DESKTOP-F7G5G3D MINGW64 ~/Desktop/test/git_test (main)
$
```

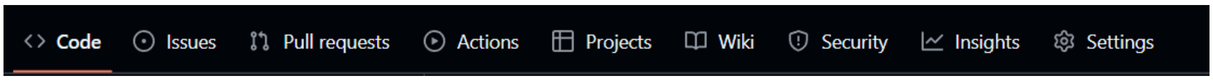


Kiramido1 resolved the conflict successfully1 7d88ff4 · 6 minutes ago

7 lines (6 loc) · 108 Bytes

```
line on master
line 1 on testing branch
line 2 on testing branch
hello everybody!
hi
my name is "moahmed"
```

More into Github



Tab	Description
Code	View and manage source code, branches, commits, and clone/download the repo.
Issues	Track bugs, feature requests, and tasks using discussion-based issue tracking.
Pull requests	Propose, review, and merge changes from branches or forks.
Actions	Set up CI/CD and automation workflows using GitHub Actions.
Projects	Manage tasks with Kanban-style project boards for planning and tracking.
Wiki	Create and share documentation or guides within the repository.
Security	Monitor vulnerabilities, configure Dependabot, and enable code scanning.
Insights	View repository analytics: commits, contributors, traffic, dependency graphs.

Settings	Configure repository options: branches, collaborators, visibility, webhooks.

UI Tools in Action

watch tis video ⇒ https://www.youtube.com/watch?v=hxbSj1cgQkY

summary

the most important commands

Basic Git Setup

Command	**Description**
`git config --global user.name "Your Name"`	Set your Git username globally
`git config --global user.email "you@example.com"`	Set your Git email globally
`git config --list`	View all current Git configuration settings

Repository Management

Command	**Description**
`git init`	Create a new Git repository in current folder
`git clone <url>`	Clone an existing repository from GitHub
`git status`	Show the status of changes in your working directory
`git add <file>`	Stage changes for commit
`git add .`	Stage **all** changes
`git commit -m "message"`	Commit staged changes with a message
`git log`	Show commit history
`git diff`	Show changes not yet staged or committed

Branching & Merging

Command	**Description**
`git branch`	List all branches

`git branch <branch-name>`	Create a new branch
`git checkout <branch-name>`	Switch to a different branch
`git checkout -b <branch-name>`	Create and switch to a new branch
`git merge <branch-name>`	Merge a branch into current branch
`git branch -d <branch-name>`	Delete a branch

Remote Repositories

Command	Description
`git remote -v`	View connected remotes
`git push`	Push local commits to the remote repository
`git push origin <branch>`	Push a specific branch
`git pull`	Fetch and merge changes from remote
`git fetch`	Fetch changes without merging

Undo & Recovery

Command	Description
`git reset <file>`	Unstage a file
`git checkout -- <file>`	Revert changes in a file
`git revert <commit>`	Revert a specific commit by creating a new one
`git reset --hard`	Remove all changes (use carefully!)