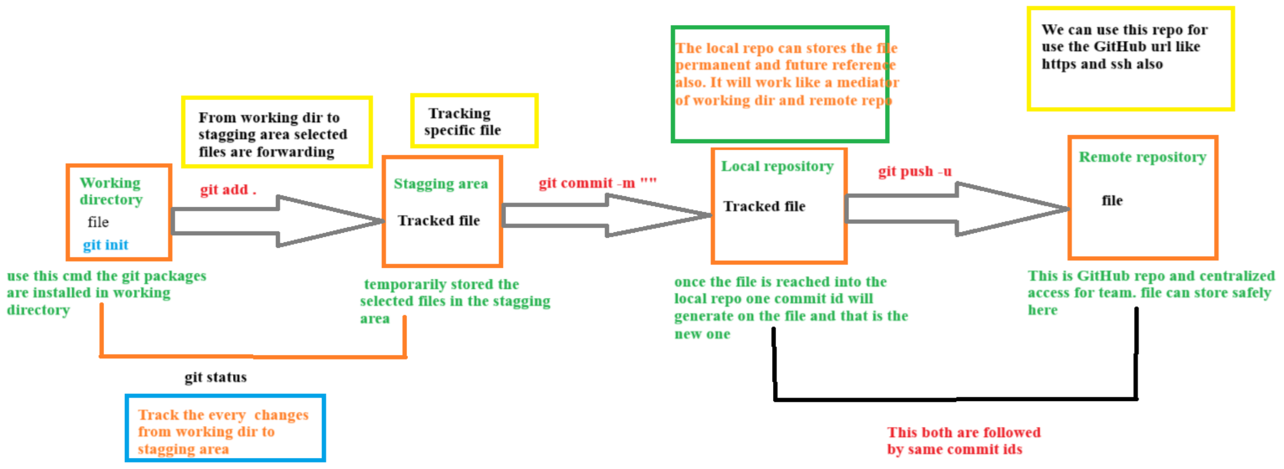
# TASK 4: Build a Version-Controlled DevOps Project with Git

**Git:** Git is a distributed version control system used to track changes in source code during software development. It helps teams collaborate, manage code versions, and keep a history of every change made to a project.

- Git can tracks the every changes from work directory to remote repository.

1. Working directory
2. Stagging area
3. Local repository
4. Remote repository

## Git Flow

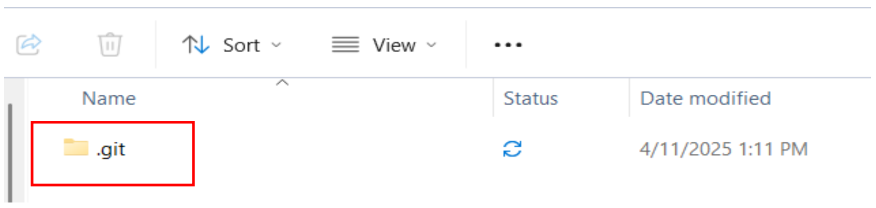

From working dir to stagging area selected files are forwarding
Tracking specific file
The local repo can stores the file permanent and future reference also. It will work like a mediator of working dir and remote repo
We can use this repo for use the GitHub url like https and ssh also
Working directory
file
git init
git add .
Stagging area
Tracked file
git commit -m ""
Local repository
Tracked file
git push -u
Remote repository
file
use this cmd the git packages are installed in working directory
temporarily stored the selected files in the stagging area
once the file is reached into the local repo one commit id will generate on the file and that is the new one
This is GitHub repo and centralized access for team. file can store safely here
git status
Track the every changes from working dir to stagging area
This both are followed by same commit ids

## Git connection:

- SSH keypair
- HTTPS
- PAT

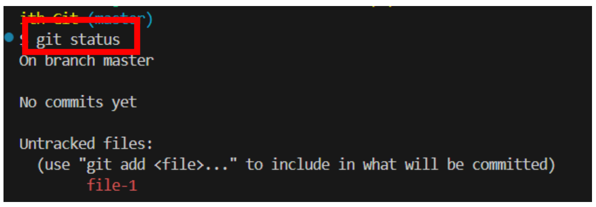
### GIT WORKFLOW COMMON CMDS

- Create a folder in local ------git init #.git is hidden file



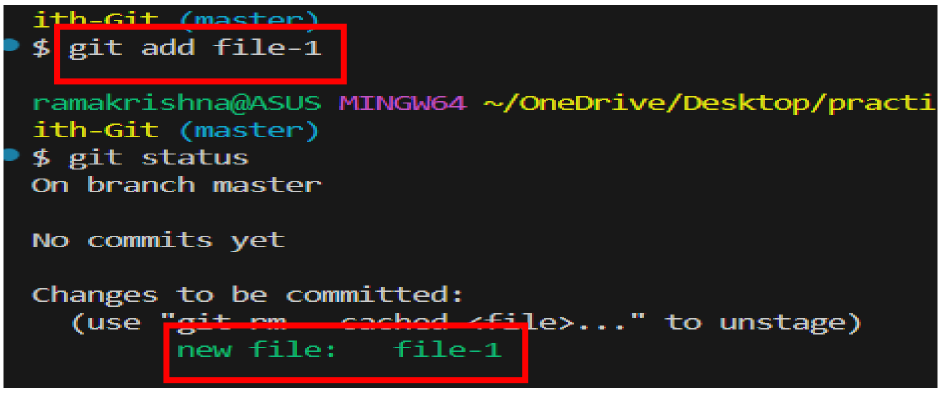
| Name | Status | Date modified |
|---|---|---|
| .git | | 4/11/2025 1:11 PM |

- git config --global user.name "username"
- git config --global user.email "email id"
- git clone -- This cmd use for a clone the repo from remote to local
- mkdir file-1
- git status



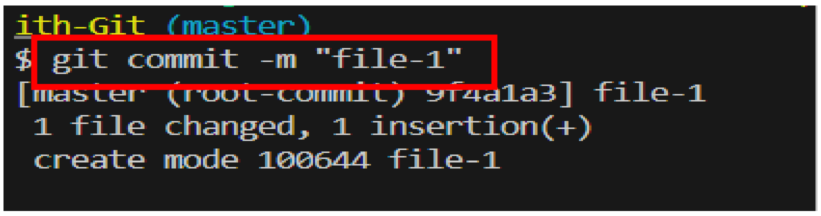
```
ith-Git (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file-1
```

- git add file-1



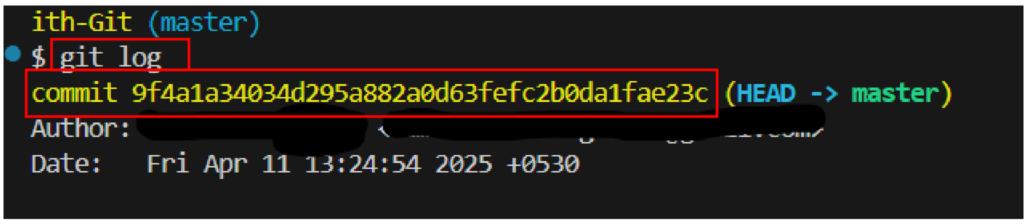
```
ith-Git (master)
$ git add file-1

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practi
ith-Git (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file-1
```

- git commit -m “message”



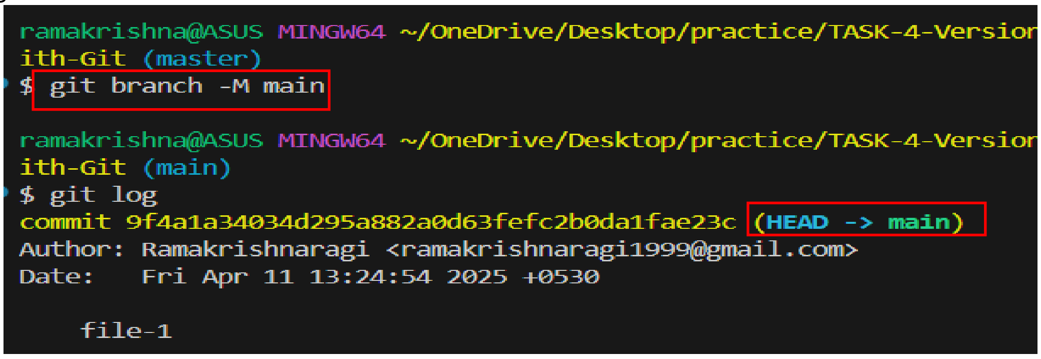
```
ith-Git (master)
$ git commit -m "file-1"
[master (root-commit) 9f4a1a3] file-1
 1 file changed, 1 insertion(+)
 create mode 100644 file-1
```

- git log



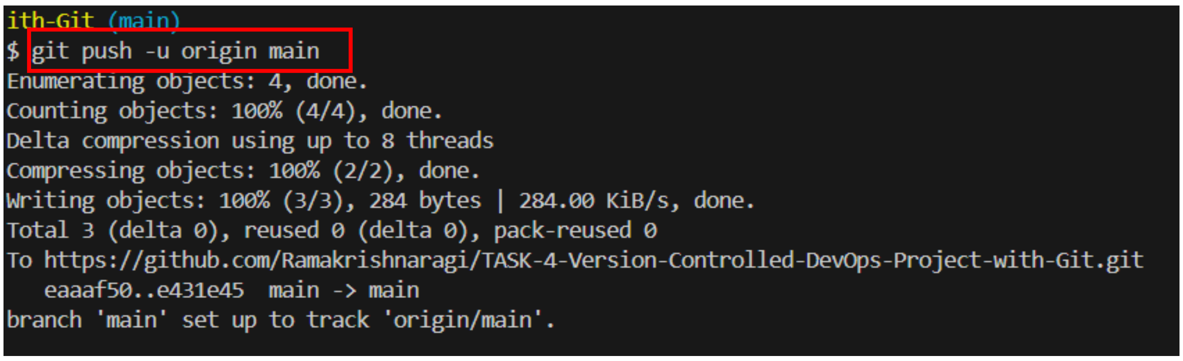
```
ith-Git (master)
$ git log
commit 9f4a1a34034d295a882a0d63fefc2b0da1fae23c (HEAD -> master)
Author: [redacted]
Date:   Fri Apr 11 13:24:54 2025 +0530
```

- git branch -M main



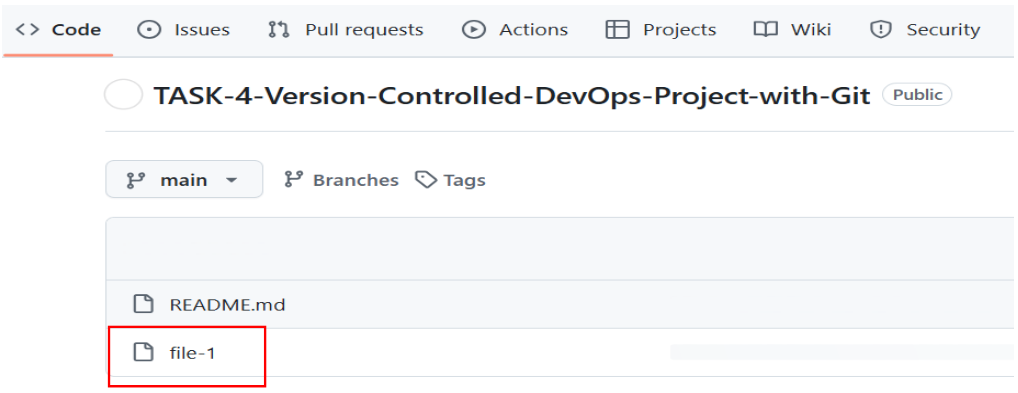
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version
ith-Git (master)
$ git branch -M main

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version
ith-Git (main)
$ git log
commit 9f4a1a34034d295a882a0d63fefc2b0da1fae23c (HEAD -> main)
Author: Ramakrishnaragi <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 13:24:54 2025 +0530

    file-1
```

- Git remote add origin <git repo url>
- Git push -u origin main



```
ith-Git (main)
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 284 bytes | 284.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git.git
   eaaaf50..e431e45  main -> main
branch 'main' set up to track 'origin/main'.
```

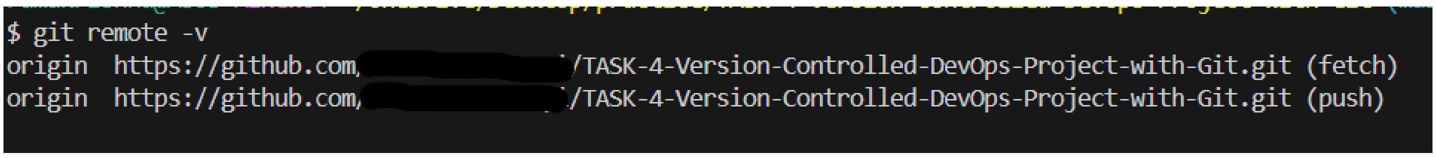
- Check the remote repo, the is available or not



TASK-4-Version-Controlled-DevOps-Project-with-Git Public

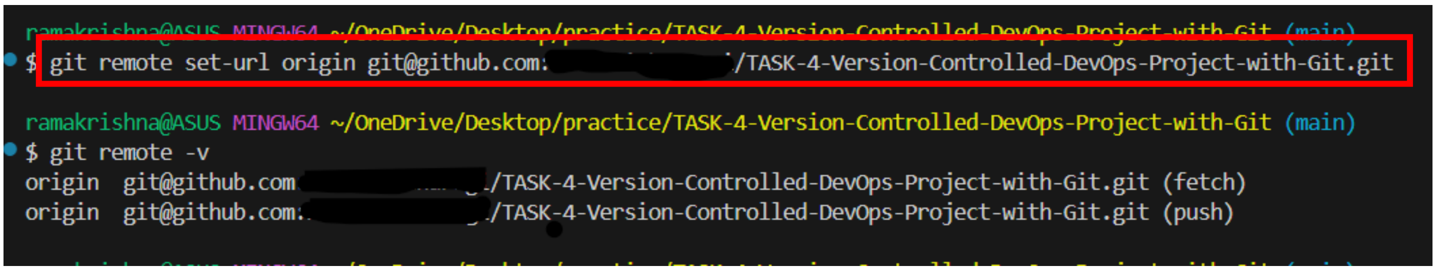
README.md
file-1

- Git remote -v # if we get an error: remote origin already exists



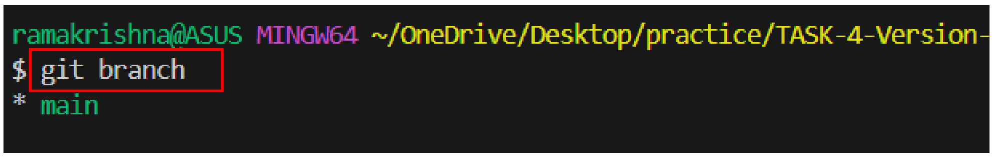
```
$ git remote -v
origin  https://github.com/[redacted]/TASK-4-Version-Controlled-DevOps-Project-with-Git.git (fetch)
origin  https://github.com/[redacted]/TASK-4-Version-Controlled-DevOps-Project-with-Git.git (push)
```

- Git remote set-url origin <git hub repo url> # remove existing origin map and add git remote
  git branch command ton check which branch you are in



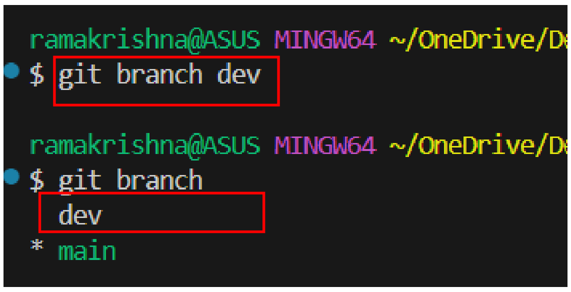
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (main)
$ git remote set-url origin git@github.com:[redacted]/TASK-4-Version-Controlled-DevOps-Project-with-Git.git

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (main)
$ git remote -v
origin  git@github.com[redacted]/TASK-4-Version-Controlled-DevOps-Project-with-Git.git (fetch)
origin  git@github.com:[redacted]/TASK-4-Version-Controlled-DevOps-Project-with-Git.git (push)
```

- Git branch # checking the branches are available



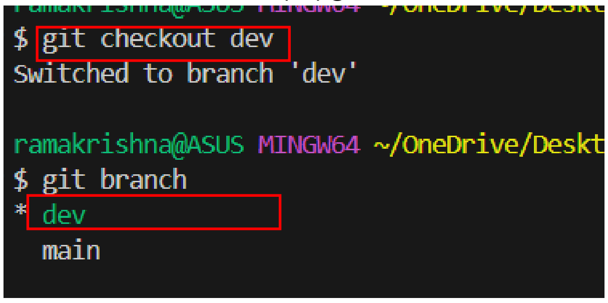
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-
$ git branch
* main
```

- Git branch dev # create a sub-branch comes under main branch



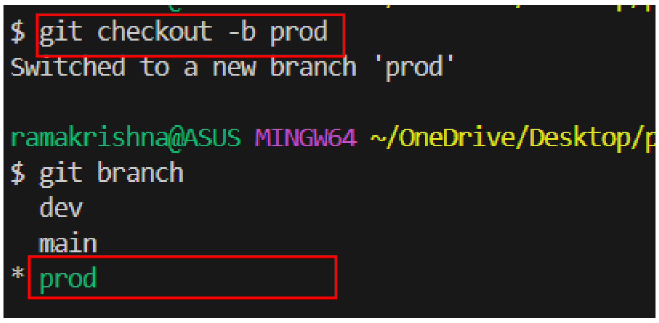
```
ramakrishna@ASUS MINGW64 ~/OneDrive/D
$ git branch dev

ramakrishna@ASUS MINGW64 ~/OneDrive/D
$ git branch
  dev
* main
```

- Git checkout <branch-name> (or) git switch <branch-name> #move from one branch to other branch



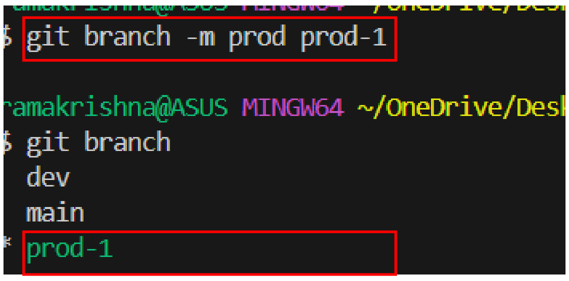
```
$ git checkout dev
Switched to branch 'dev'

ramakrishna@ASUS MINGW64 ~/OneDrive/Deskt
$ git branch
* dev
  main
```

- Git checkout -b <sub-branch-name> # this cmd use for create branch and switch to branch



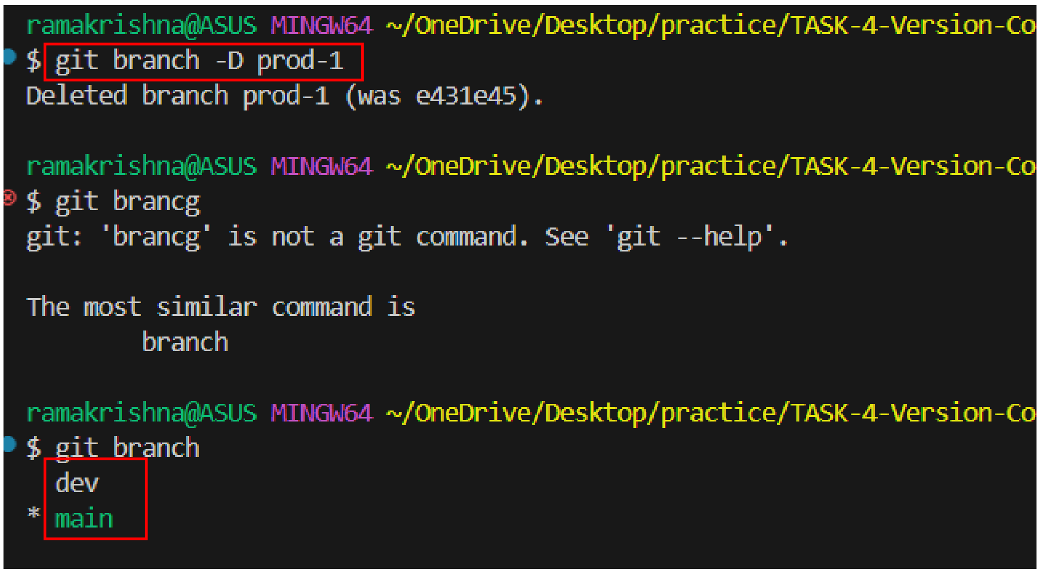
```
$ git checkout -b prod
Switched to a new branch 'prod'

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/p
$ git branch
  dev
  main
* prod
```

- git branch -m oldname new name -------> to rename a branch



```
$ git branch -m prod prod-1

amakrishna@ASUS MINGW64 ~/OneDrive/Desk
$ git branch
  dev
  main
* prod-1
```

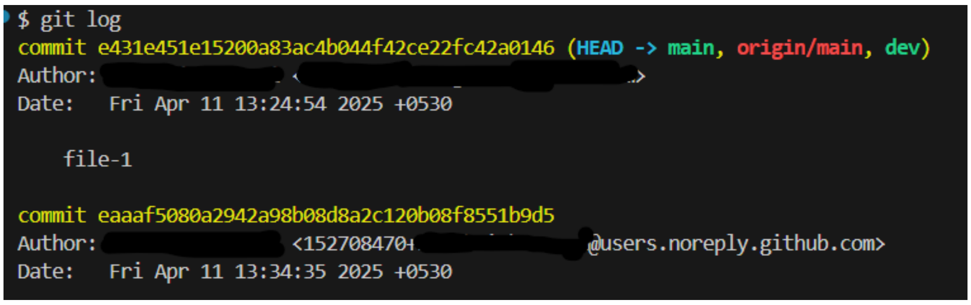
- Git branch -D <branch name> ----- to delete a branch



```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Co
$ git branch -D prod-1
Deleted branch prod-1 (was e431e45).

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Co
$ git brancg
git: 'brancg' is not a git command. See 'git --help'.

The most similar command is
        branch

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Co
$ git branch
  dev
* main
```

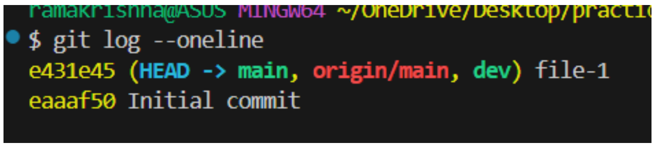
## Git logs cmds:

- Git log # check the logs



```
$ git log
commit e431e451e15200a83ac4b044f42ce22fc42a0146 (HEAD -> main, origin/main, dev)
Author: [redacted] <[redacted]>
Date:   Fri Apr 11 13:24:54 2025 +0530

    file-1

commit eaaaf5080a2942a98b08d8a2c120b08f8551b9d5
Author: [redacted] <152708470+[redacted]@users.noreply.github.com>
Date:   Fri Apr 11 13:34:35 2025 +0530
```

- Git log –oneline # it give in one line log



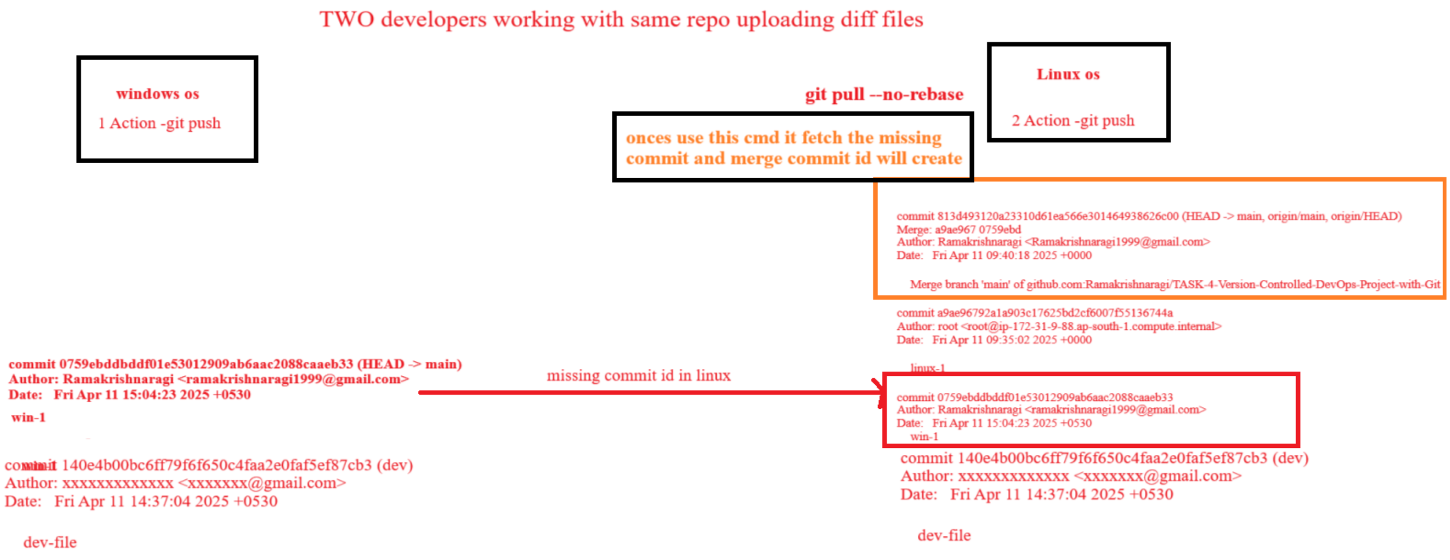
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practic
$ git log --oneline
e431e45 (HEAD -> main, origin/main, dev) file-1
eaaaf50 Initial commit
```

- Git pull #solve the merger conflicts and different commit ids

Team work with same repo and push and pull conflicts

# Here I am taking to team mates 1. Windows and 2. Linux

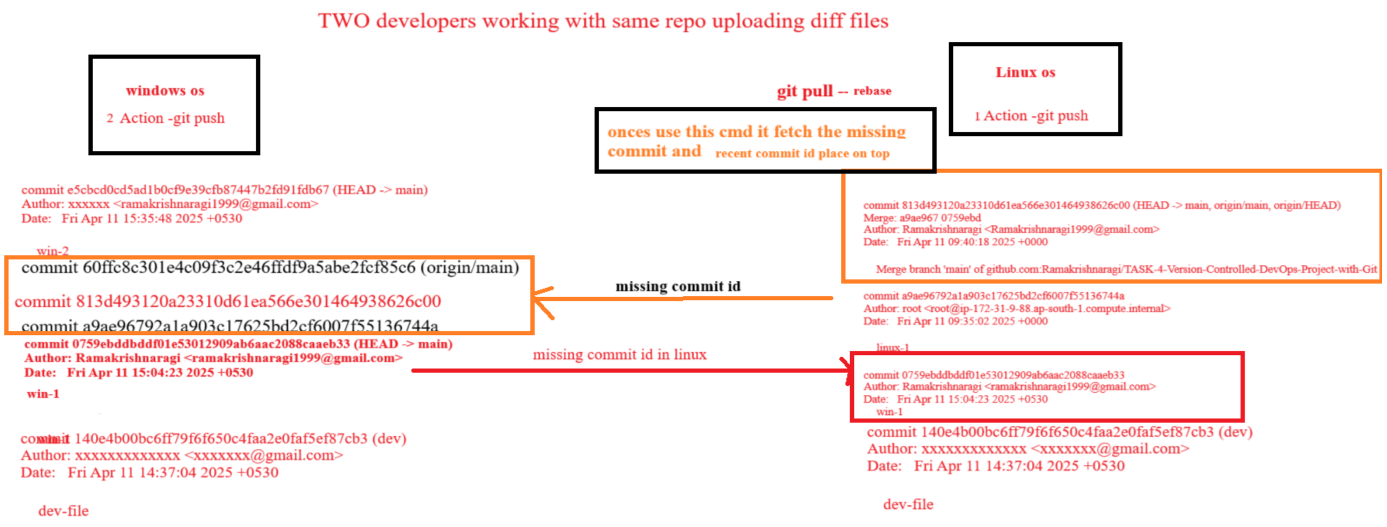
- git pull --no-rebase # it fetch the missing commit id and create a new merge commit id


TWO developers working with same repo uploading diff files
windows os
1 Action -git push
git pull --no-rebase
onces use this cmd it fetch the missing commit and merge commit id will create
Linux os
2 Action -git push
missing commit id in linux
win-1
linux-1
dev-file

```
commit 0759ebddbddf01e53012909ab6aac2088caaeb33 (HEAD -> main)
Author: Ramakrishnaragi <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 15:04:23 2025 +0530

 win-1

commit 140e4b00bc6ff79f6f650c4faa2e0faf5ef87cb3 (dev)
Author: xxxxxxxxxxxxx <xxxxxxx@gmail.com>
Date:   Fri Apr 11 14:37:04 2025 +0530

    dev-file
```

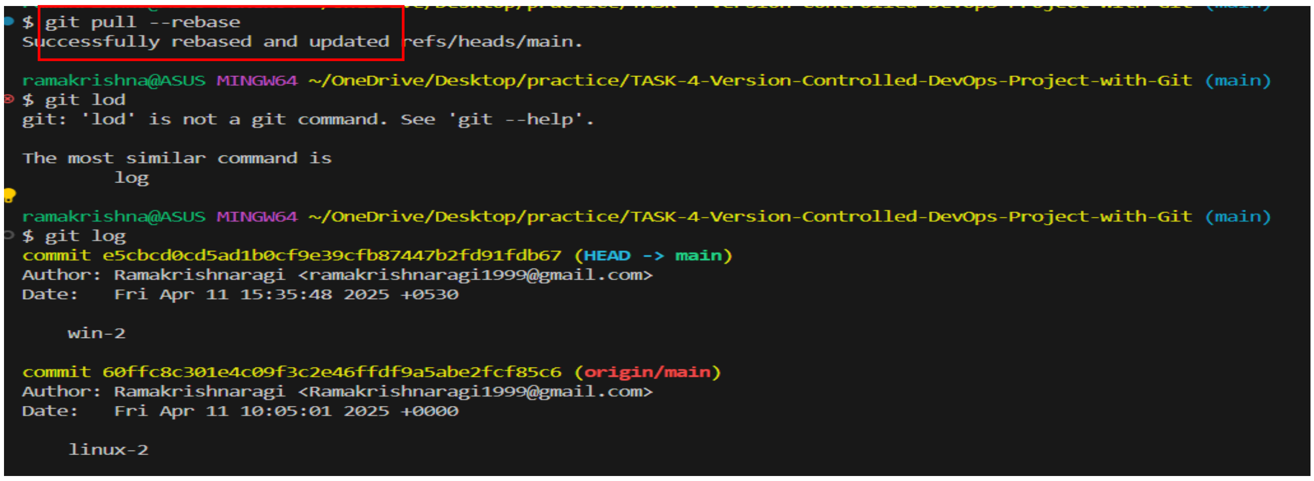
```
commit 813d493120a23310d61ea566e301464938626c00 (HEAD -> main, origin/main, origin/HEAD)
Merge: a9ae967 0759ebd
Author: Ramakrishnaragi <Ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 09:40:18 2025 +0000

    Merge branch 'main' of github.com:Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git

commit a9ae96792a1a903c17625bd2cf6007f55136744a
Author: root <root@ip-172-31-9-88.ap-south-1.compute.internal>
Date:   Fri Apr 11 09:35:02 2025 +0000

    linux-1

commit 0759ebddbddf01e53012909ab6aac2088caaeb33
Author: Ramakrishnaragi <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 15:04:23 2025 +0530
    win-1
commit 140e4b00bc6ff79f6f650c4faa2e0faf5ef87cb3 (dev)
Author: xxxxxxxxxxxxx <xxxxxxx@gmail.com>
Date:   Fri Apr 11 14:37:04 2025 +0530

    dev-file
```

- Git pull --rebase # it will fetch the missing the commit id and recent commit id will placed on the top


TWO developers working with same repo uploading diff files
windows os
2 Action -git push
git pull -- rebase
onces use this cmd it fetch the missing commit and recent commit id place on top
Linux os
1 Action -git push
missing commit id
missing commit id in linux
win-2
win-1
linux-1
dev-file

```
commit e5cbcd0cd5ad1b0cf9e39cfb87447b2fd91fdb67 (HEAD -> main)
Author: xxxxxx <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 15:35:48 2025 +0530

    win-2
commit 60ffc8c301e4c09f3c2e46ffdf9a5abe2fcf85c6 (origin/main)

commit 813d493120a23310d61ea566e301464938626c00
commit a9ae96792a1a903c17625bd2cf6007f55136744a
commit 0759ebddbddf01e53012909ab6aac2088caaeb33 (HEAD -> main)
Author: Ramakrishnaragi <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 15:04:23 2025 +0530

 win-1

commit 140e4b00bc6ff79f6f650c4faa2e0faf5ef87cb3 (dev)
Author: xxxxxxxxxxxxx <xxxxxxx@gmail.com>
Date:   Fri Apr 11 14:37:04 2025 +0530

    dev-file
```

```
commit 813d493120a23310d61ea566e301464938626c00 (HEAD -> main, origin/main, origin/HEAD)
Merge: a9ae967 0759ebd
Author: Ramakrishnaragi <Ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 09:40:18 2025 +0000

    Merge branch 'main' of github.com:Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git
commit a9ae96792a1a903c17625bd2cf6007f55136744a
Author: root <root@ip-172-31-9-88.ap-south-1.compute.internal>
Date:   Fri Apr 11 09:35:02 2025 +0000

    linux-1

commit 0759ebddbddf01e53012909ab6aac2088caaeb33
Author: Ramakrishnaragi <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 15:04:23 2025 +0530
    win-1
commit 140e4b00bc6ff79f6f650c4faa2e0faf5ef87cb3 (dev)
Author: xxxxxxxxxxxxx <xxxxxxx@gmail.com>
Date:   Fri Apr 11 14:37:04 2025 +0530

    dev-file
```

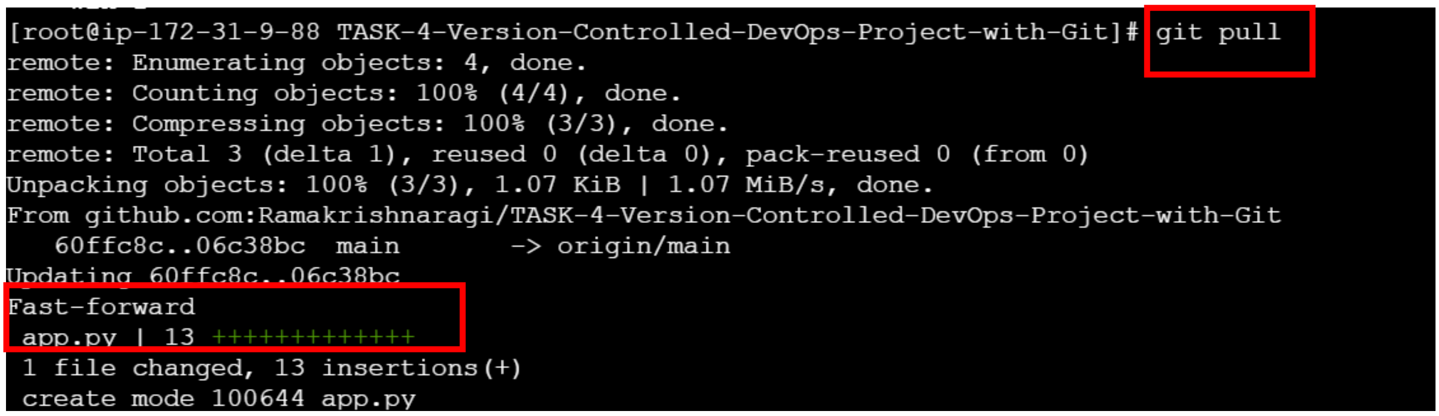


```
$ git pull --rebase
Successfully rebased and updated refs/heads/main.

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (main)
$ git lod
git: 'lod' is not a git command. See 'git --help'.

The most similar command is
        log

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (main)
$ git log
commit e5cbcd0cd5ad1b0cf9e39cfb87447b2fd91fdb67 (HEAD -> main)
Author: Ramakrishnaragi <ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 15:35:48 2025 +0530

    win-2

commit 60ffc8c301e4c09f3c2e46ffdf9a5abe2fcf85c6 (origin/main)
Author: Ramakrishnaragi <Ramakrishnaragi1999@gmail.com>
Date:   Fri Apr 11 10:05:01 2025 +0000

    linux-2
```

MERGER CONFLICTS ON SAME FILE OR DIFFERENT FILE

- Git pull –ff #minor changes only use this based on commit id is diff means it will not work

# I am add one file in remote repo and using git pull how it will let's see

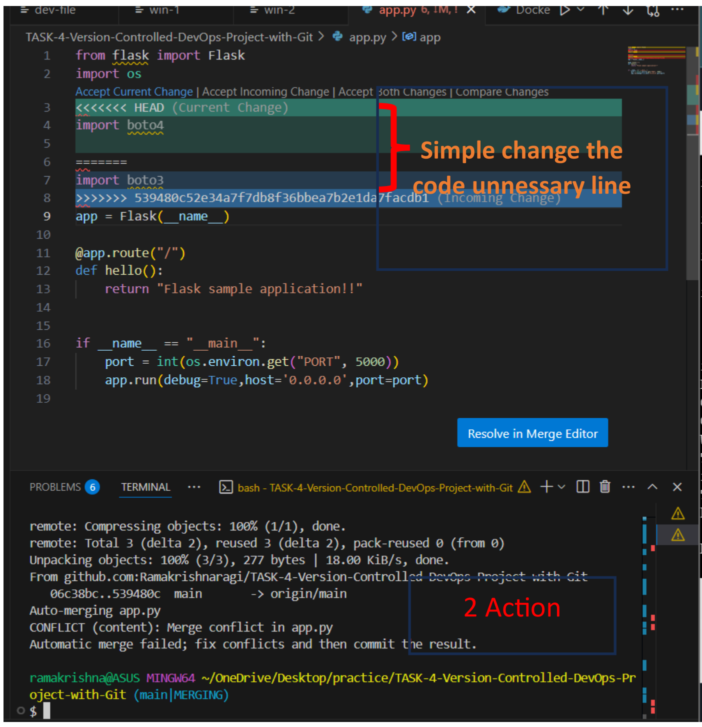


```
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]# git pull
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 1.07 KiB | 1.07 MiB/s, done.
From github.com:Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git
   60ffc8c..06c38bc  main       -> origin/main
Updating 60ffc8c..06c38bc
Fast-forward
 app.py | 13 +++++++++++++
 1 file changed, 13 insertions(+)
 create mode 100644 app.py
```

# then use the git pull for both os

# then change in code for one line and add, commit changes, push

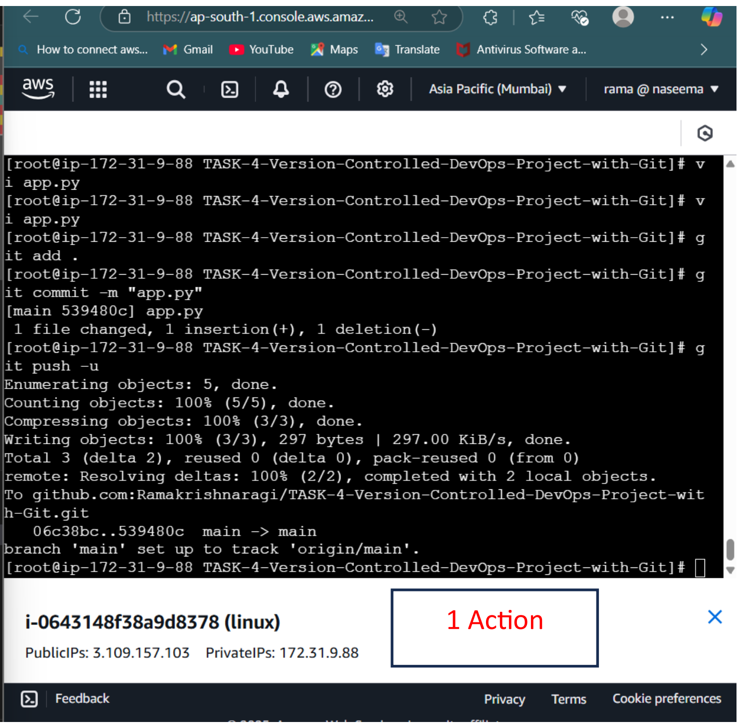
# Then execute the same change in same line code in windows os and check the errors



dev-file
win-1
win-2
app.py 6, 1M, !
Docke
TASK-4-Version-Controlled-DevOps-Project-with-Git > app.py > app
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
Simple change the code unnessary line
Resolve in Merge Editor
PROBLEMS 6
TERMINAL
bash - TASK-4-Version-Controlled-DevOps-Project-with-Git
2 Action

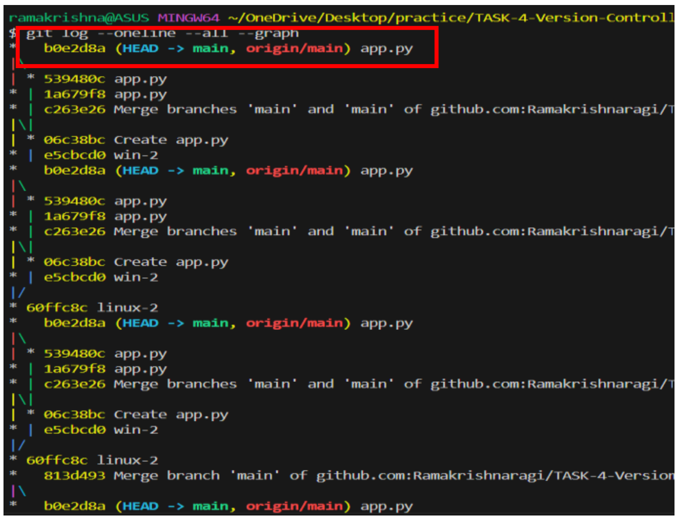
```
from flask import Flask
import os
<<<<<<< HEAD (Current Change)
import boto4

=======
import boto3
>>>>>>> 539480c52e34a7f7db8f36bbea7b2e1da7facdb1 (Incoming Change)
app = Flask(__name__)

@app.route("/")
def hello():
    return "Flask sample application!!"


if __name__ == "__main__":
    port = int(os.environ.get("PORT", 5000))
    app.run(debug=True,host='0.0.0.0',port=port)
```

```
remote: Compressing objects: 100% (1/1), done.
remote: Total 3 (delta 2), reused 3 (delta 2), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 277 bytes | 18.00 KiB/s, done.
From github.com:Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git
   06c38bc..539480c  main       -> origin/main
Auto-merging app.py
CONFLICT (content): Merge conflict in app.py
Automatic merge failed; fix conflicts and then commit the result.

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Pr
oject-with-Git (main|MERGING)
$
```



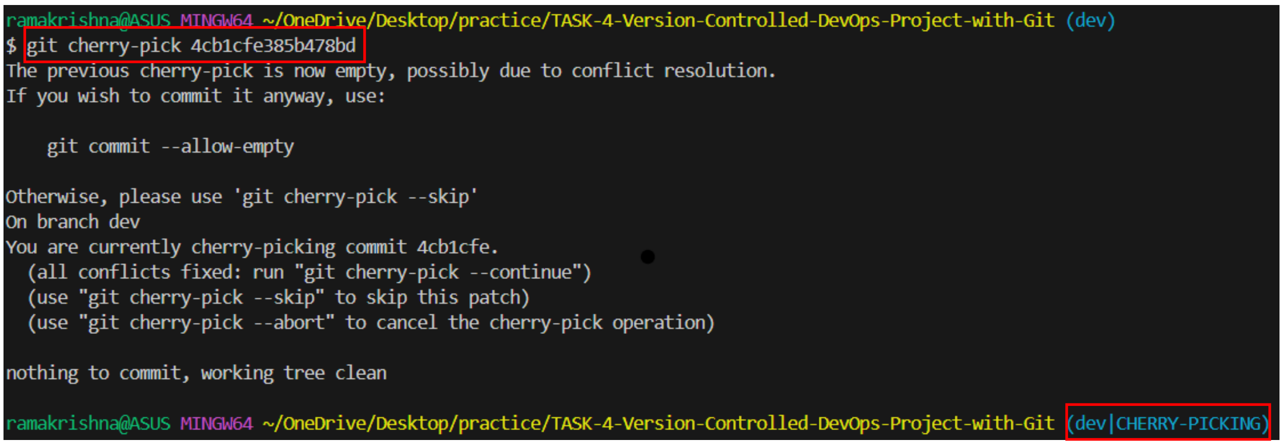
```
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]# v
i app.py
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]# v
i app.py
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]# g
it add .
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]# g
it commit -m "app.py"
[main 539480c] app.py
 1 file changed, 1 insertion(+), 1 deletion(-)
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]# g
it push -u
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 297 bytes | 297.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-wit
h-Git.git
   06c38bc..539480c  main -> main
branch 'main' set up to track 'origin/main'.
[root@ip-172-31-9-88 TASK-4-Version-Controlled-DevOps-Project-with-Git]#
```

- git log --oneline --all --graph # it shows the graphical representation

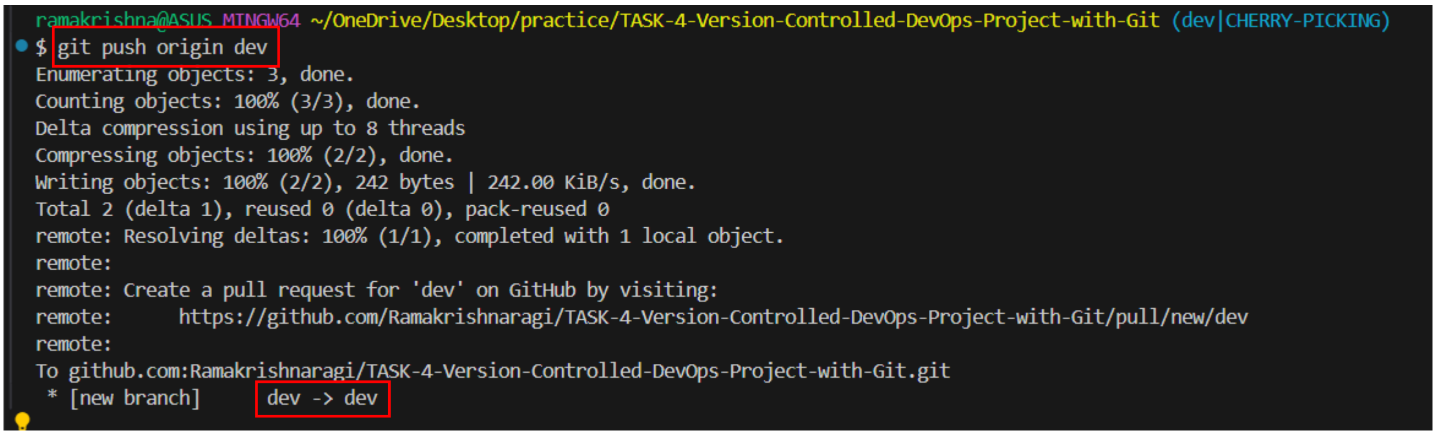


```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controll
$ git log --oneline --all --graph
*   b0e2d8a (HEAD -> main, origin/main) app.py
|\
| * 539480c app.py
* | 1a679f8 app.py
* | c263e26 Merge branches 'main' and 'main' of github.com:Ramakrishnaragi/T
|\|
| * 06c38bc Create app.py
* | e5cbcd0 win-2
*   b0e2d8a (HEAD -> main, origin/main) app.py
|\
| * 539480c app.py
* | 1a679f8 app.py
* | c263e26 Merge branches 'main' and 'main' of github.com:Ramakrishnaragi/T
|\|
| * 06c38bc Create app.py
* | e5cbcd0 win-2
|/
* 60ffc8c linux-2
*   b0e2d8a (HEAD -> main, origin/main) app.py
|\
| * 539480c app.py
* | 1a679f8 app.py
* | c263e26 Merge branches 'main' and 'main' of github.com:Ramakrishnaragi/T
|\|
| * 06c38bc Create app.py
* | e5cbcd0 win-2
|/
* 60ffc8c linux-2
*   813d493 Merge branch 'main' of github.com:Ramakrishnaragi/TASK-4-Version
|\
*   b0e2d8a (HEAD -> main, origin/main) app.py
```

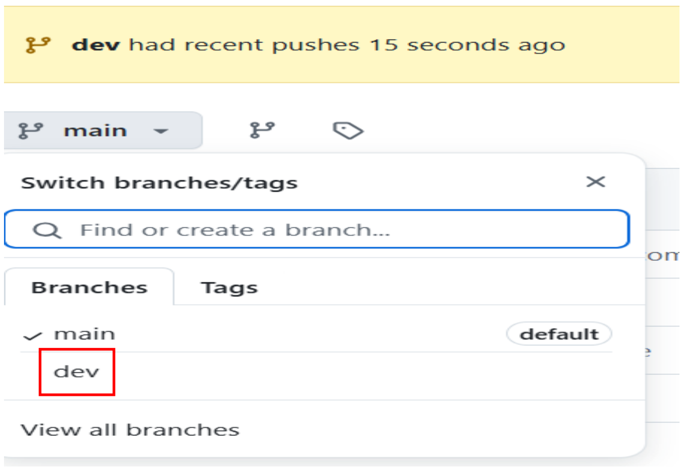
- Git cherry pick # if we want merge specific commit into upstream branch (like main) cherry pick will help us
    - git cherry-pic <committed>
    - git push origin<branch name>



```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (dev)
$ git cherry-pick 4cb1cfe385b478bd
The previous cherry-pick is now empty, possibly due to conflict resolution.
If you wish to commit it anyway, use:

    git commit --allow-empty

Otherwise, please use 'git cherry-pick --skip'
On branch dev
You are currently cherry-picking commit 4cb1cfe.
  (all conflicts fixed: run "git cherry-pick --continue")
  (use "git cherry-pick --skip" to skip this patch)
  (use "git cherry-pick --abort" to cancel the cherry-pick operation)

nothing to commit, working tree clean

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (dev|CHERRY-PICKING)
```

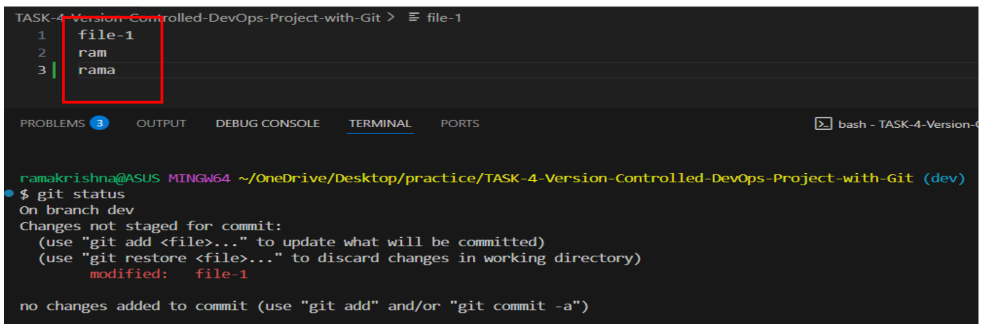


```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (dev|CHERRY-PICKING)
$ git push origin dev
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 242 bytes | 242.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'dev' on GitHub by visiting:
remote:      https://github.com/Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git/pull/new/dev
remote:
To github.com:Ramakrishnaragi/TASK-4-Version-Controlled-DevOps-Project-with-Git.git
 * [new branch]      dev -> dev
```



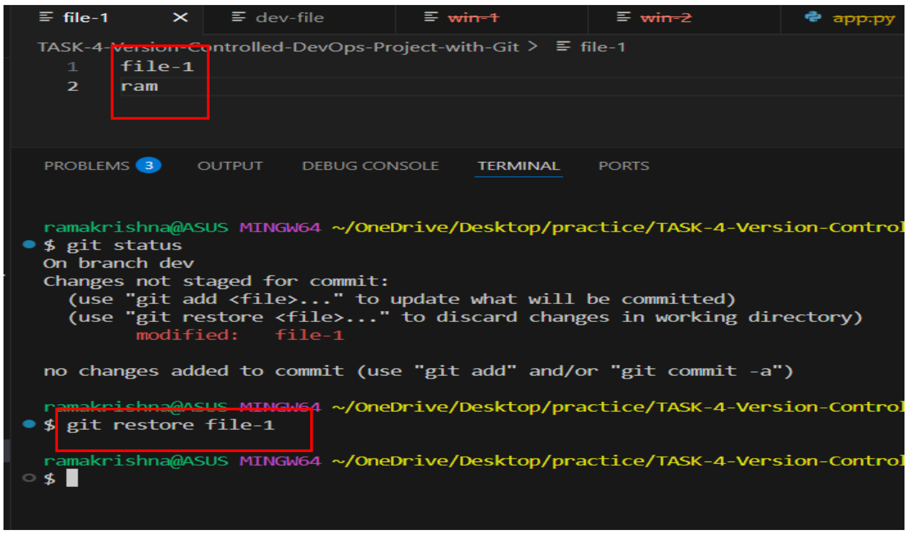
dev had recent pushes 15 seconds ago

- Git restore # This cmd can use for undo the changes in working dir only
    - Git restore <filename>



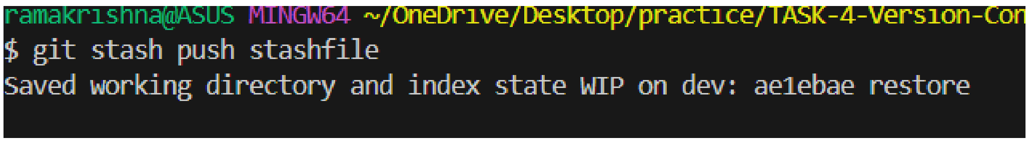
```
file-1
ram
rama
```

```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (dev)
$ git status
On branch dev
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-1

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



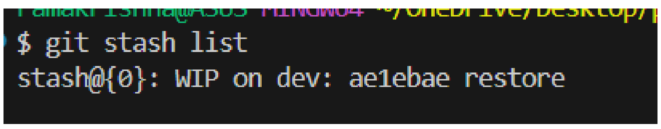
```
file-1    dev-file    win-1    win-2    app.py
TASK-4-Version-Controlled-DevOps-Project-with-Git > file-1
file-1
ram

PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Control
$ git status
On branch dev
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-1

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

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Control
$ git restore file-1

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Control
$ 
```

- Git stash
  - Git stash push <filename>



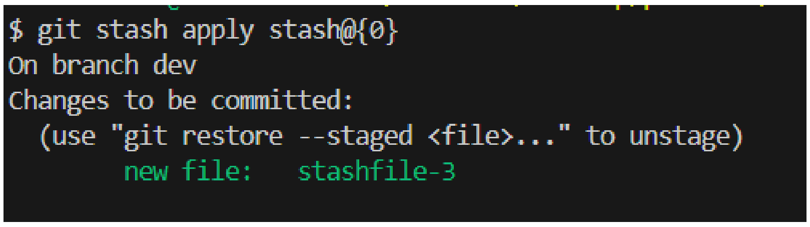
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Con
$ git stash push stashfile
Saved working directory and index state WIP on dev: ae1ebae restore
```

  - Git stash list



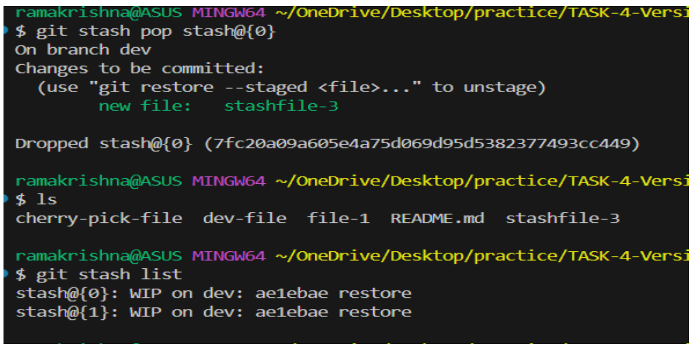
```
$ git stash list
stash@{0}: WIP on dev: ae1ebae restore
```

  - git stash apply stash@{0} # It act for a reback the file into the stagging area



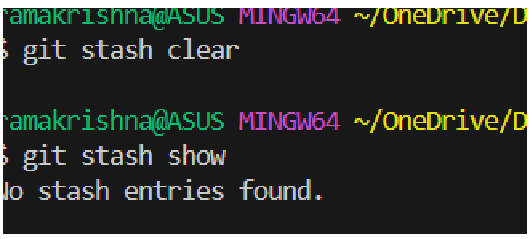
```
$ git stash apply stash@{0}
On branch dev
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   stashfile-3
```

  - git stash pop stash@{0} # it act for a Dropped stash



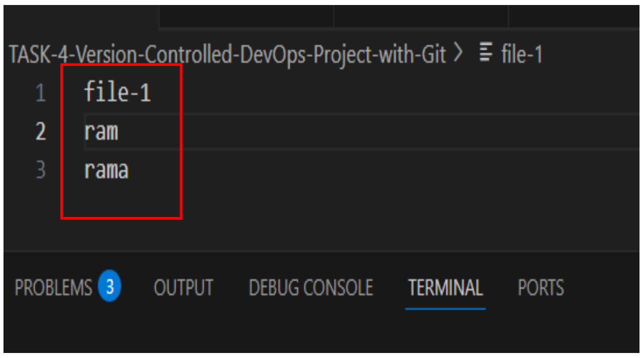
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Versi
$ git stash pop stash@{0}
On branch dev
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   stashfile-3

Dropped stash@{0} (7fc20a09a605e4a75d069d95d5382377493cc449)

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Versi
$ ls
cherry-pick-file  dev-file  file-1  README.md  stashfile-3

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Versi
$ git stash list
stash@{0}: WIP on dev: ae1ebae restore
stash@{1}: WIP on dev: ae1ebae restore
```

  - **git stash clear # clear the all stash files**

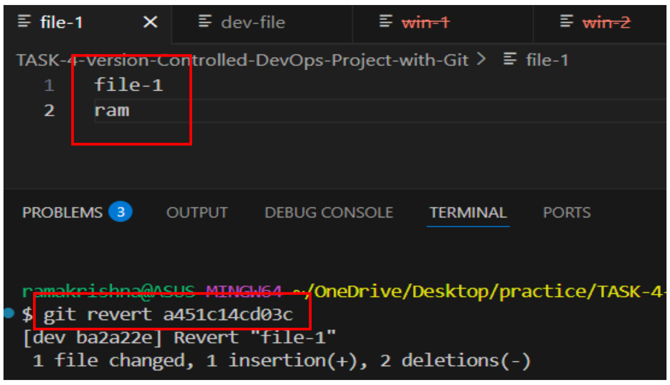


```
amakrishna@ASUS MINGW64 ~/OneDrive/D
 git stash clear

amakrishna@ASUS MINGW64 ~/OneDrive/D
 git stash show
o stash entries found.
```

➢ git revert <commit_id> # it can create a extra commit id and clear the data in the file. This is use for after the commit_id



```
TASK-4-Version-Controlled-DevOps-Project-with-Git > file-1
file-1
ram
rama
```

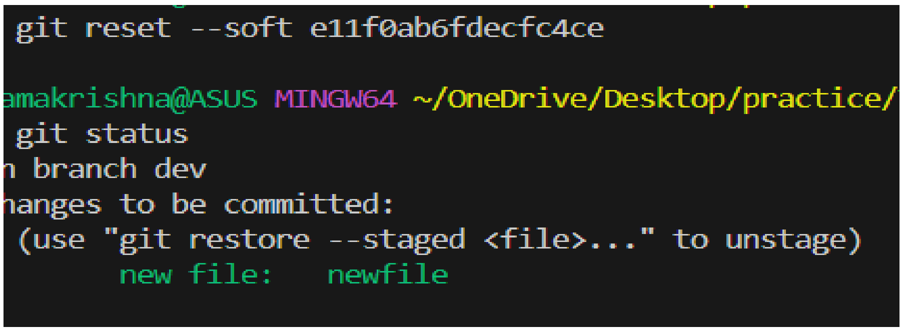


```
TASK-4-Version-Controlled-DevOps-Project-with-Git > file-1
file-1
ram
```

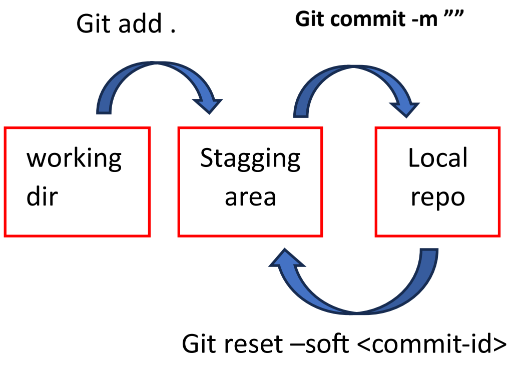
```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-
$ git revert a451c14cd03c
[dev ba2a22e] Revert "file-1"
 1 file changed, 1 insertion(+), 2 deletions(-)
```

➢ Git reset # git reset changes your branch's history or staging area, depending on the mode you use:

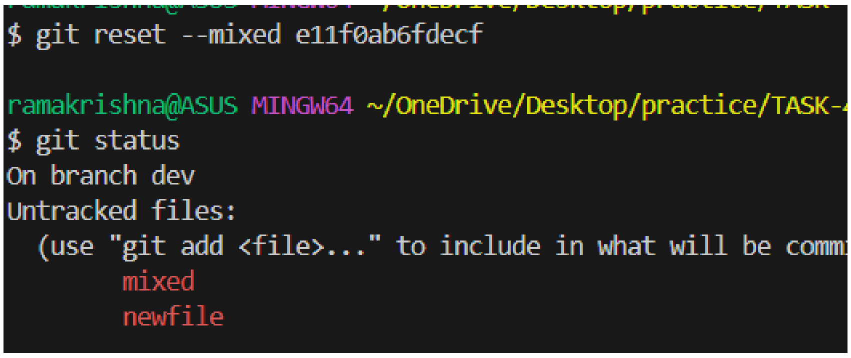
- Git reset --soft<commit id> # use after git commit



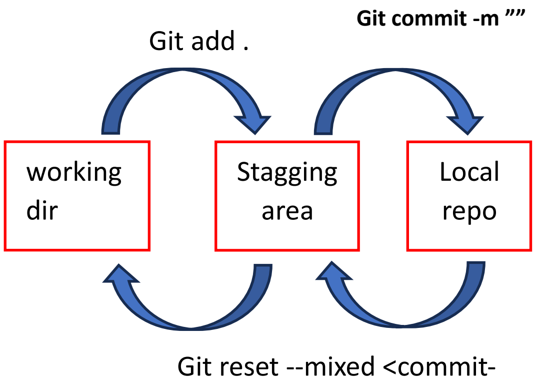
```
git reset --soft e11f0ab6fdecfc4ce

amakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/
 git status
n branch dev
hanges to be committed:
 (use "git restore --staged <file>..." to unstage)
        new file:   newfile
```



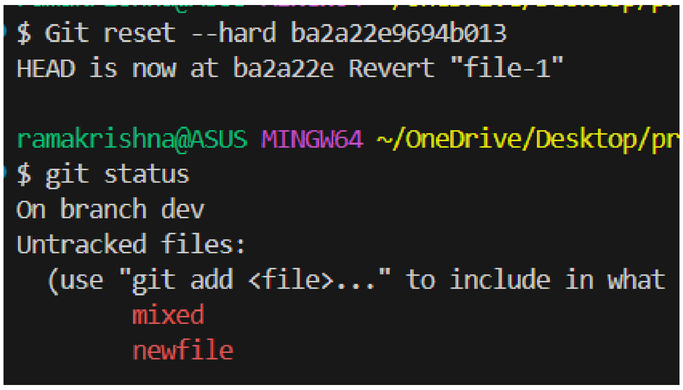
- Git reset --mixed<commit id>



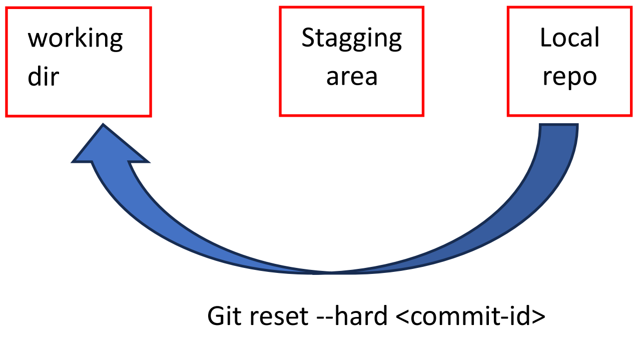
```
$ git reset --mixed e11f0ab6fdecf

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4
$ git status
On branch dev
Untracked files:
  (use "git add <file>..." to include in what will be comm
        mixed
        newfile
```



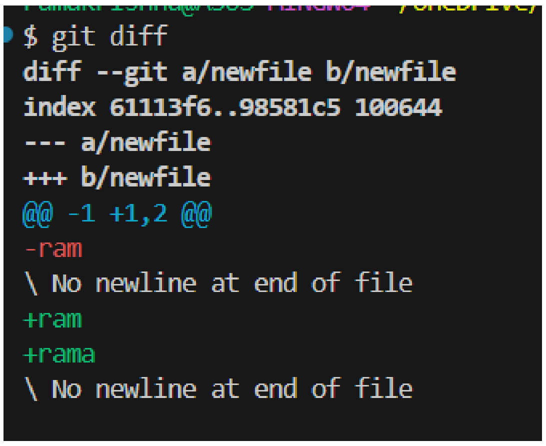
- Git reset --hard <commit -d>



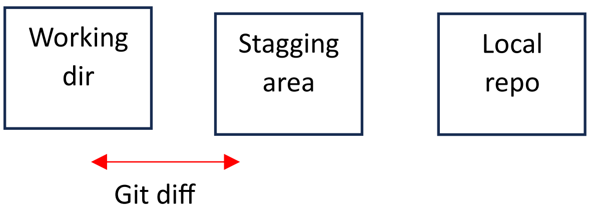
```
$ Git reset --hard ba2a22e9694b013
HEAD is now at ba2a22e Revert "file-1"

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/pr
$ git status
On branch dev
Untracked files:
  (use "git add <file>..." to include in what
        mixed
        newfile
```



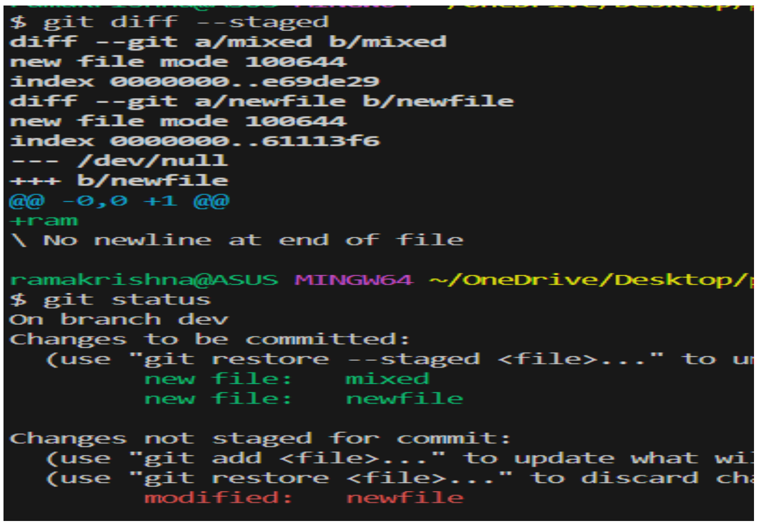
- Git diff # it show the changes between in the working and stagging area



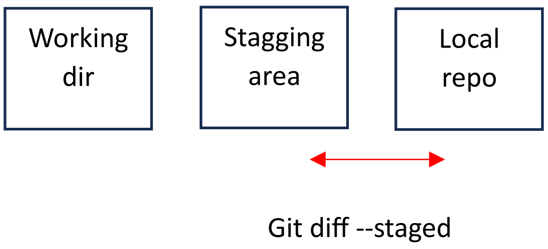
```
$ git diff
diff --git a/newfile b/newfile
index 61113f6..98581c5 100644
--- a/newfile
+++ b/newfile
@@ -1 +1,2 @@
-ram
\ No newline at end of file
+ram
+rama
\ No newline at end of file
```



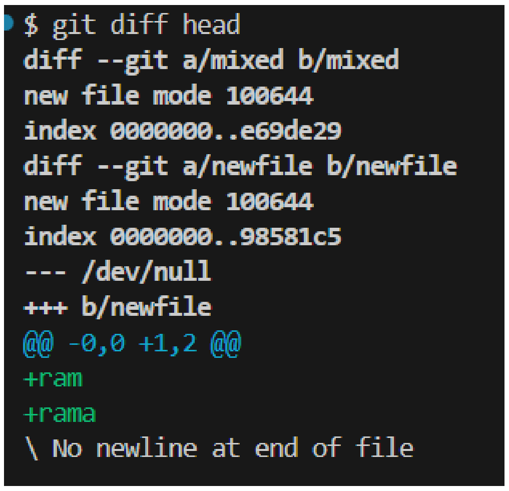
- Git diff --staged



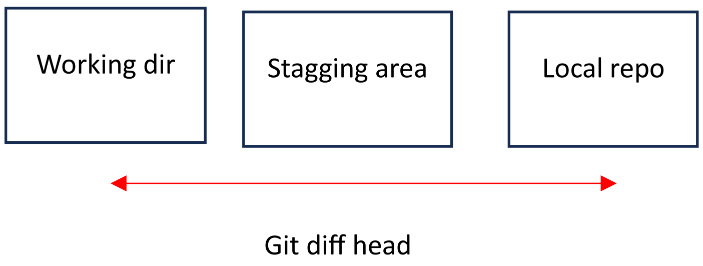
```
$ git diff --staged
diff --git a/mixed b/mixed
new file mode 100644
index 0000000..e69de29
diff --git a/newfile b/newfile
new file mode 100644
index 0000000..61113f6
--- /dev/null
+++ b/newfile
@@ -0,0 +1 @@
+ram
\ No newline at end of file

ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/
$ git status
On branch dev
Changes to be committed:
  (use "git restore --staged <file>..." to u
        new file:   mixed
        new file:   newfile

Changes not staged for commit:
  (use "git add <file>..." to update what wi
  (use "git restore <file>..." to discard ch
        modified:   newfile
```



- Git diff head #

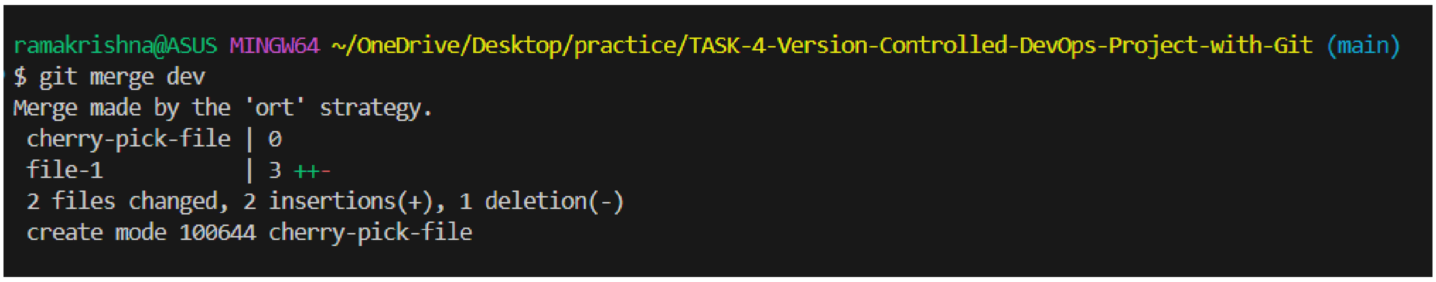


```
$ git diff head
diff --git a/mixed b/mixed
new file mode 100644
index 0000000..e69de29
diff --git a/newfile b/newfile
new file mode 100644
index 0000000..98581c5
--- /dev/null
+++ b/newfile
@@ -0,0 +1,2 @@
+ram
+rama
\ No newline at end of file
```



- Git pull = git fetch + git merge # pull remote repo
  - Git pull <remote url>

- Git merge <branchname> merging the dev in main branch



```
ramakrishna@ASUS MINGW64 ~/OneDrive/Desktop/practice/TASK-4-Version-Controlled-DevOps-Project-with-Git (main)
$ git merge dev
Merge made by the 'ort' strategy.
 cherry-pick-file | 0
 file-1           | 3 ++-
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 cherry-pick-file
```