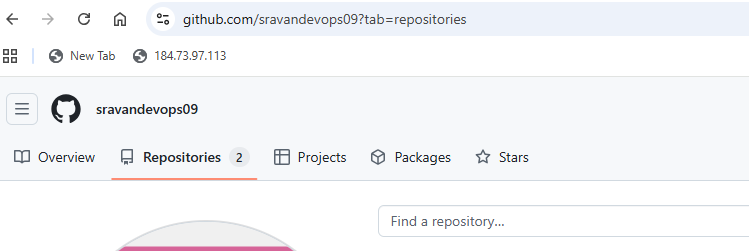
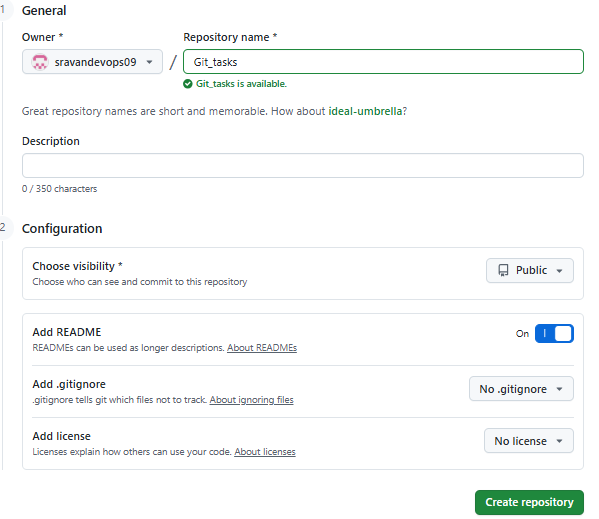
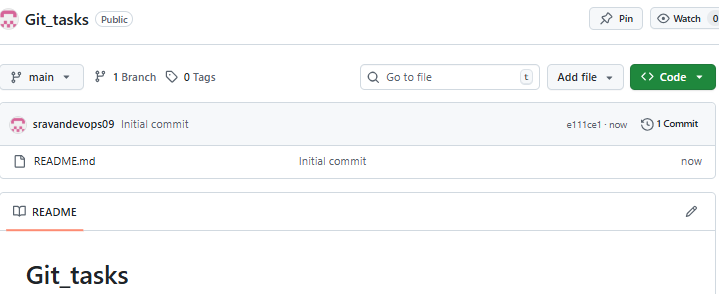
**GIT AND GITHUB TASKS**

**Install git.**

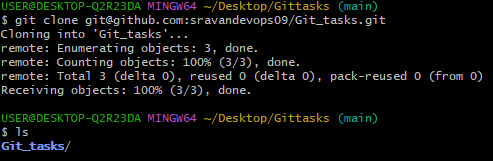


**Create a repo in github with README.md and .ignore file.**

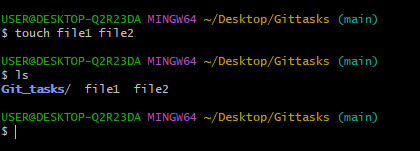




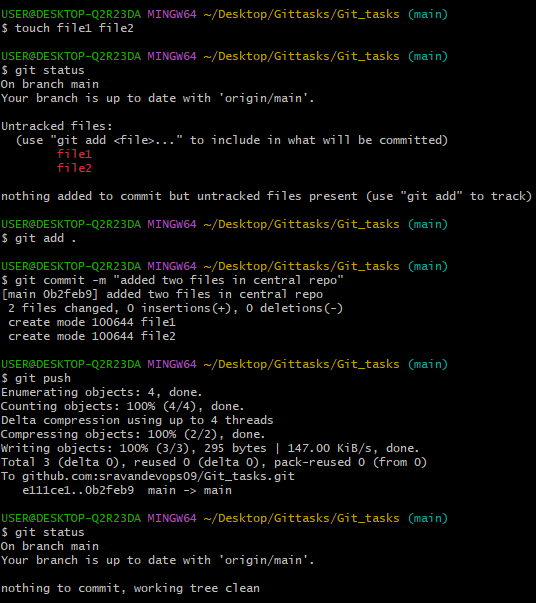
**Clone the created repo to local.**

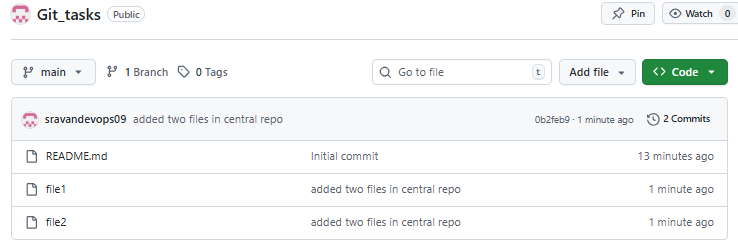


**Create two files in local repo.**

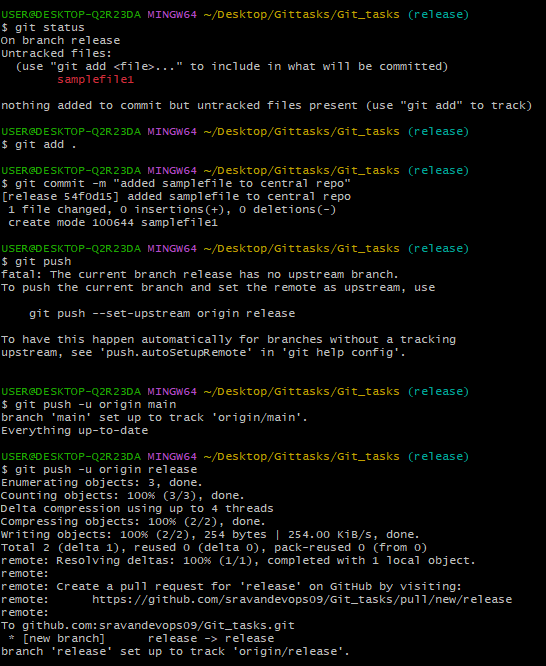


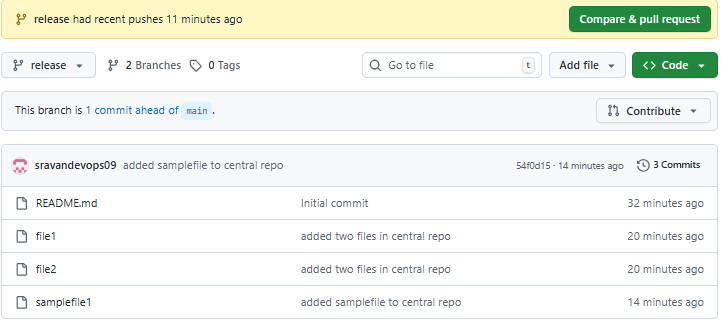
**Commit two files and push to central Repository.**



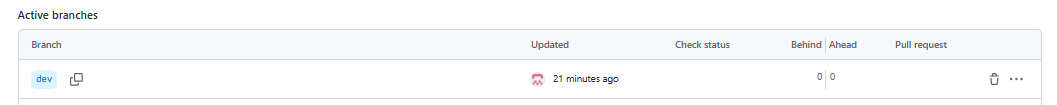


**Create a branch in local and create a sample file and push to central.**





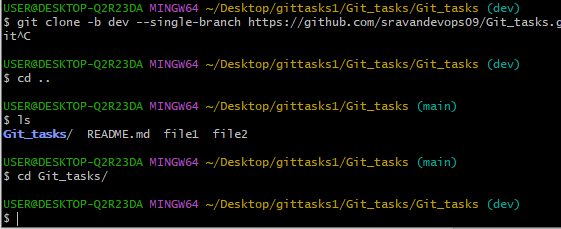
**Create a branch in github and clone that to local.**

**1 Created branch in github**

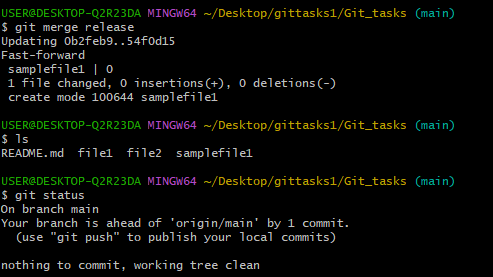
**Cloning in local**

**To clone particular branch:**

**git clone –b dev --single-branch https url**



**Merge the created branch with master in git local.**



**Merge the created branch with master in github by sending a pull request.**

Step 1: Push your branch to GitHub

git push origin release

Step 2: Go to GitHub Repository

 Click the **Branches** dropdown.

 Select your branch (release).

 Then click **New pull request**.

3: Create Pull Request

Base branch = master (or main, depending on your repo).

Compare branch = release

Add a title and description for the PR.

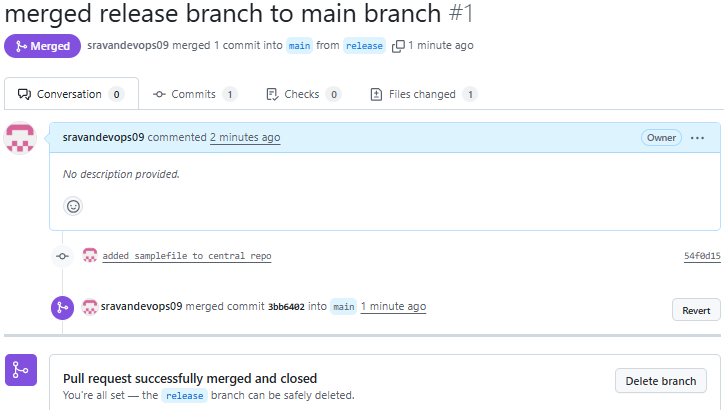
Click Create pull request.

Step 4: Merge the Pull Request

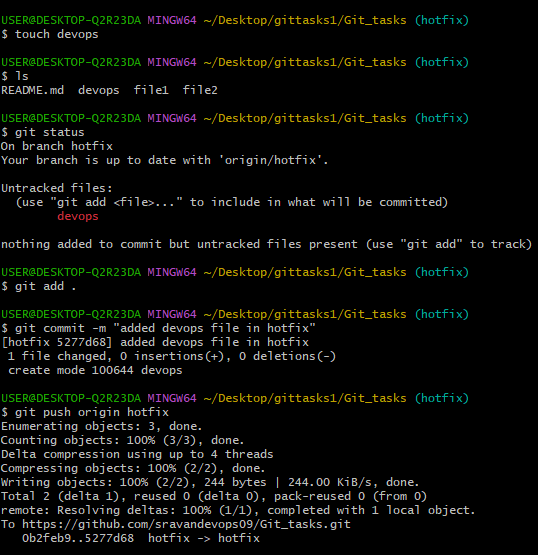
If you have permissions, you’ll see a **Merge pull request** button.

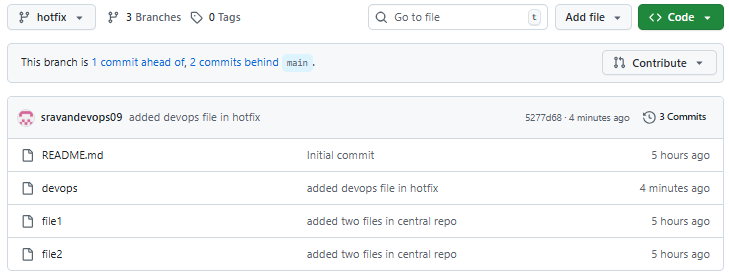
Click it, then confirm with **Confirm merge**.

Now release is merged into master in GitHub



**Create a file in local and send that to branch in github.**





**Clone only a branch from github to local.**

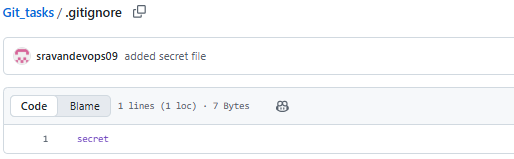
**Create a file with all passwords and make that untrackable with git.**

 Create file: secret

 Add to .gitignore

Then git add .

Commit

Git push  


**Make a commit and make that commit reset without savings changes.**

### Make a commit

Vi test

git add test

git commit -m "added test file"

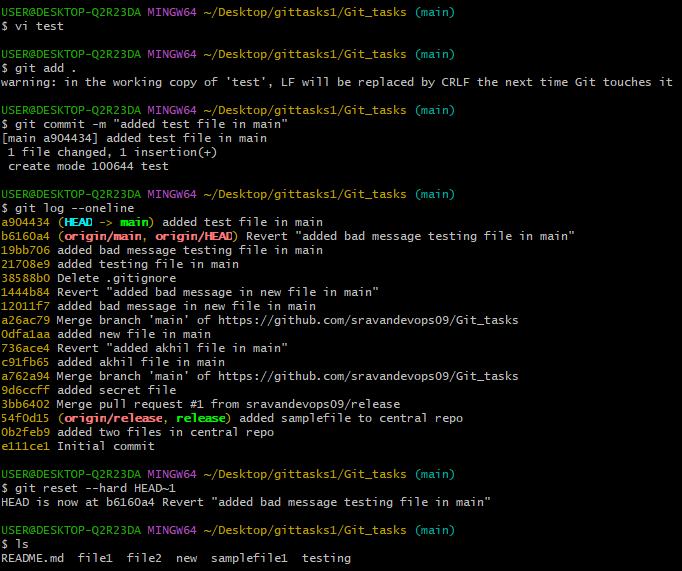
### Reset that commit without saving changes

git reset --hard HEAD~1

* HEAD~1 → move back one commit.
* --hard → also delete the changes from your working directory

## Result

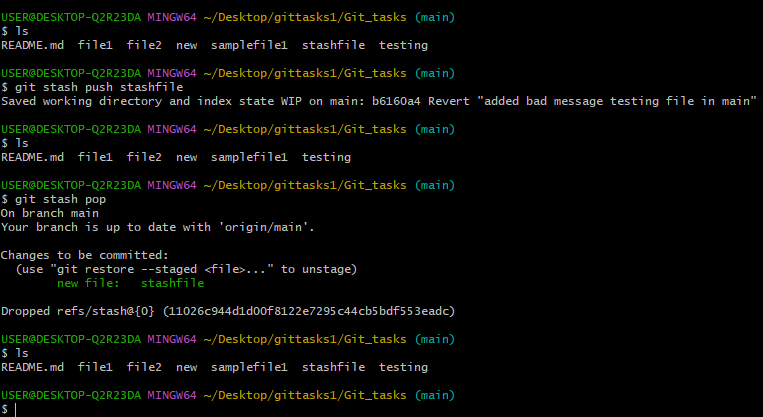
* The commit is gone.
* The file test.txt is gone too.
* Your repo looks exactly like before you created the commit.



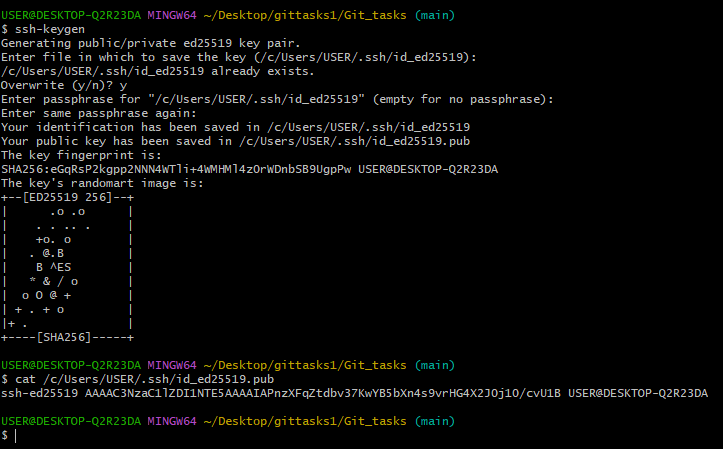
**Revert a commited commit to the older version.  
Undo the stash file and start working on that again.**

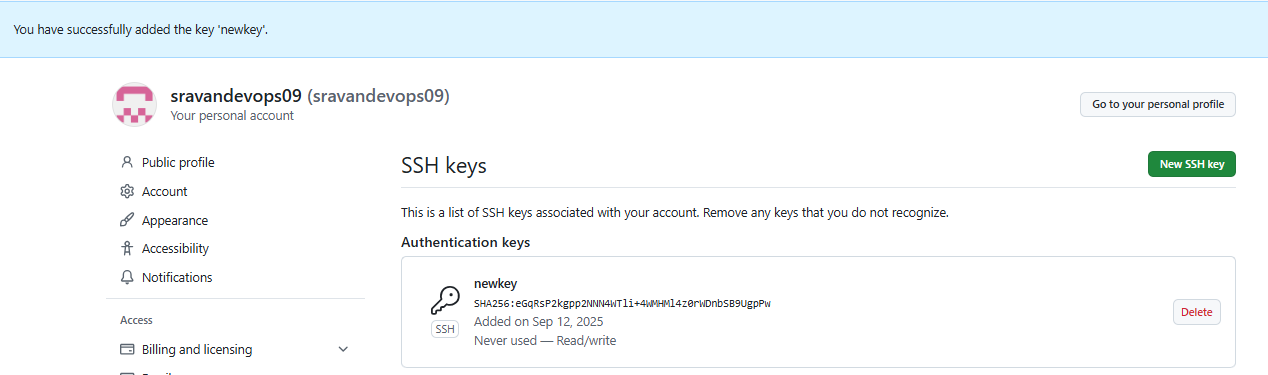
git stash push filename- move to stash

git stash pop - undo



**Generate a ssh-keygen and configure into github.**





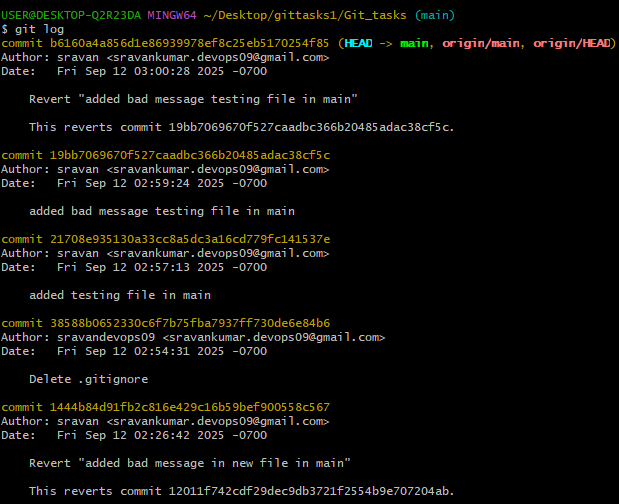
**Basic understanding of .git file.**

**In short:**

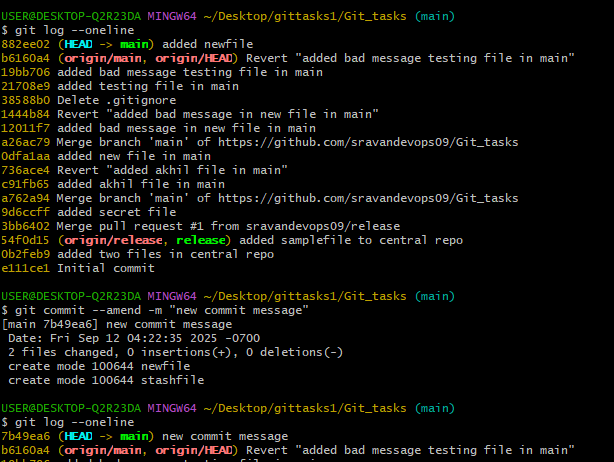
* **.git/objects** → all your data (commits, files, trees).
* **.git/refs** → labels pointing to commits (branches, tags).
* **HEAD** → tells which branch/commit you’re on.
* **index** → staging area.
* **logs** → record of moves you made.
* **config** → settings for this repo.

**Check all the logs of git**

**git log**



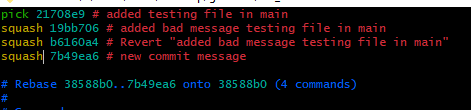
**Rename the commit message.**

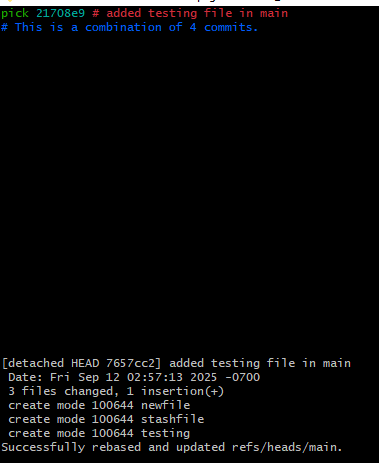


**Merge multiple commits into single commit.**

**git rebase -i HEAD~4**

**now change pick 🡪 squash**





**Configure webhooks to github.**

**Step 1: Connect to EC2**

ssh -i your-key.pem ec2-user@<EC2\_PUBLIC\_IP>

**Step 2: Install Python & Flask**

sudo dnf update -y

sudo dnf install python3-pip -y

pip3 install flask

**Step 3: Create Webhook Script in vi**

Create file:

vi webhook.py

**step 4 : Paste this code**:

from flask import Flask, request

app = Flask(\_\_name\_\_)

@app.route("/webhook", methods=["POST"])

def webhook():

data = request.json

print("✅ Webhook received:", data)

return "OK", 200

if \_\_name\_\_ == "\_\_main\_\_":

app.run(host="0.0.0.0", port=5000)

**step 5: save and quit**

**step 6 :Run Webhook Server**

python3 webhook.py

after running you will see the below o/p:

WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead. \* Running on all addresses (0.0.0.0) \* Running on http://127.0.0.1:5000 \* Running on <http://172.31.24.49:5000>

Step 7: Go to **AWS EC2 Console → Security Groups → Inbound Rules**

Add rule:

Type: **Custom TCP**

Port: 5000

Source: 0.0.0.0/0 (or just your IP for safety)

**Step 8 : Configure GitHub Webhook**

Repo → **Settings → Webhooks → Add webhook**

Payload URL:

http://<EC2\_PUBLIC\_IP>:5000/webhook

Content type: application/json

Choose event: **Push**

Save ✅

**Step 9: Create/Change a File**

Example: create a test file.

echo "Hello Webhook" > test.txt

**step 10 : Add & Commit**

git add test.txt

git commit -m "Test webhook trigger"

**step 11: Push to GitHub**

git push origin main

**step12 : Check EC2 Terminal**

On your EC2 where python3 webhook.py is running, you should see JSON payload from GitHub:

✅ Webhook received: {

"ref": "refs/heads/main",

"before": "...",

"after": "...",

"repository": { ... },

"pusher": { ... }

}

**Step 13: Verify in GitHub**

If nothing shows up, check:

* Repo → **Settings → Webhooks → Your webhook → Recent Deliveries**
* GitHub shows **Request → Response → Status (200 OK if success)**

