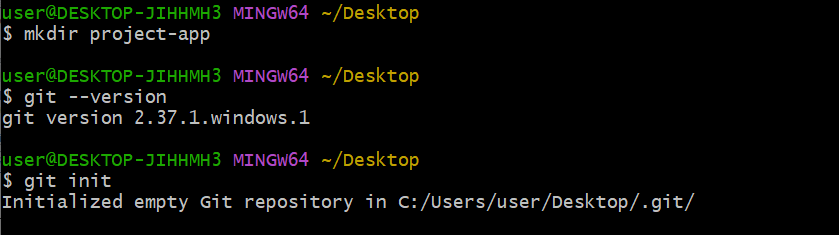
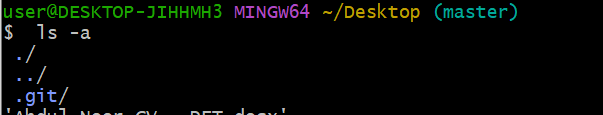
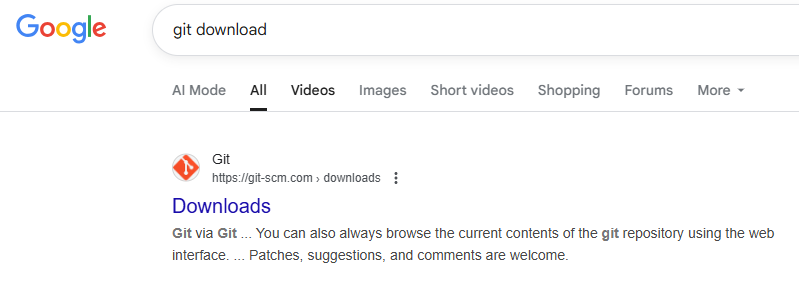
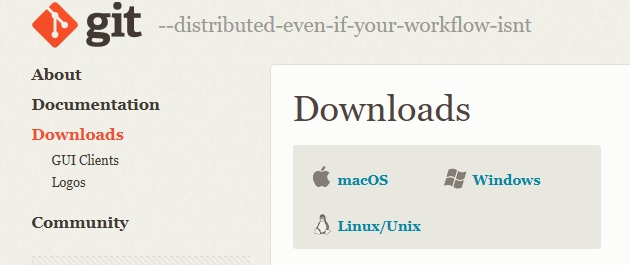
1)Install git.





If the version not displayed then need to ownload git from google > download git >



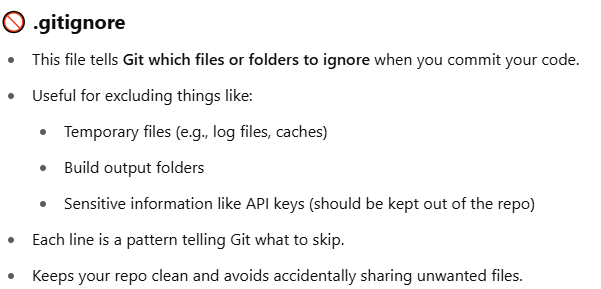


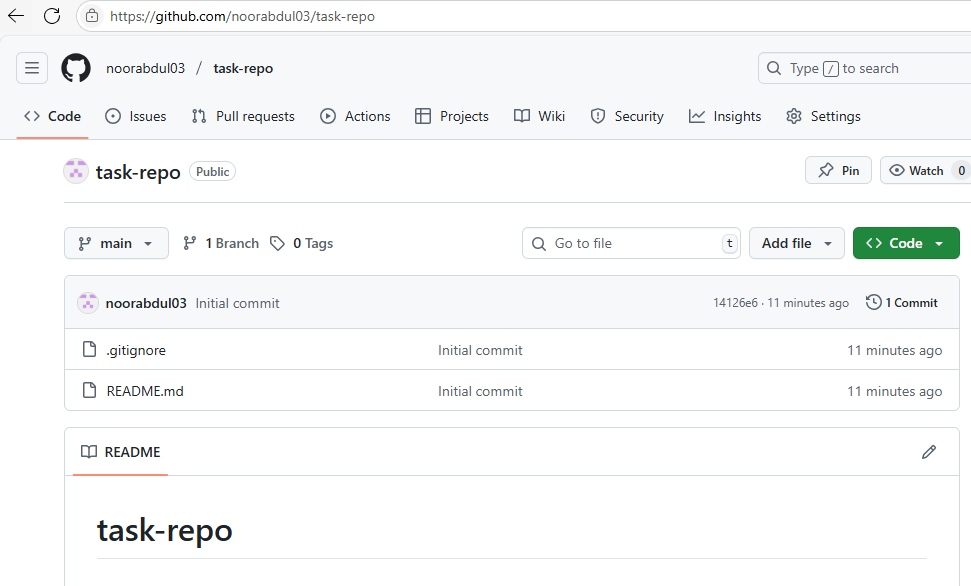
Click on windows if for windows

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

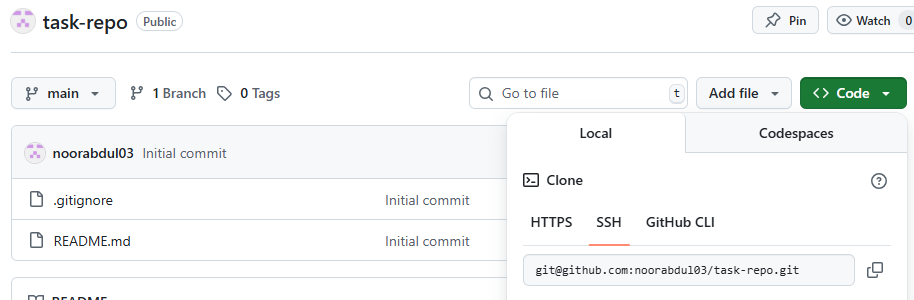
**📖 README.md**

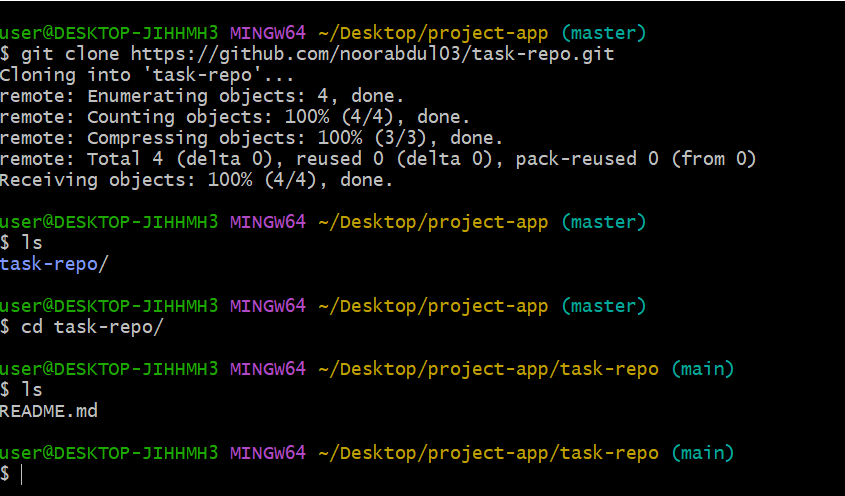
* It’s a **Markdown file** that usually explains your project.
* It often contains:
  + What the project is about
  + How to install and use it
  + Features, examples, and more
* When someone visits your GitHub repo, this file is displayed right on the main page.
* Helps others (and your future self) understand what the project does.





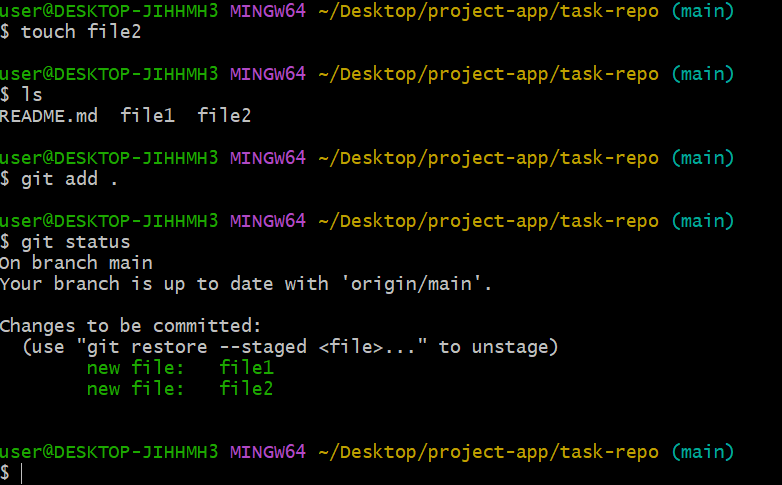
3) Clone the created repo to local



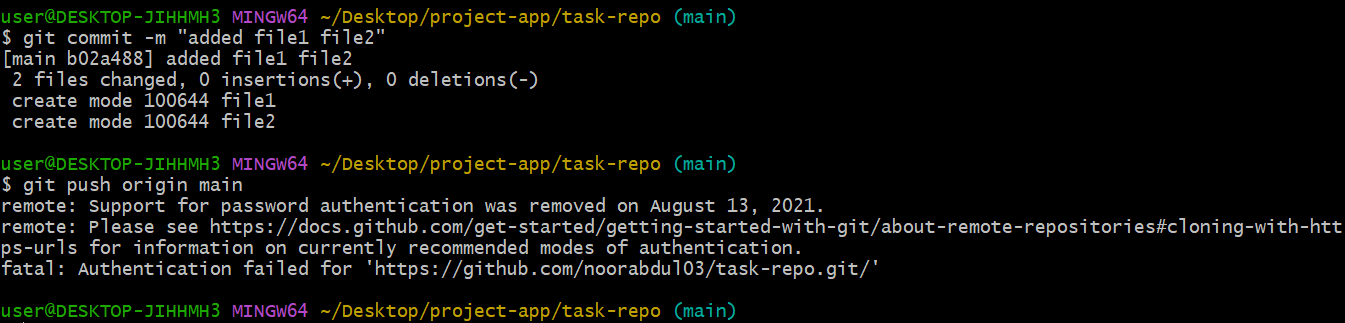


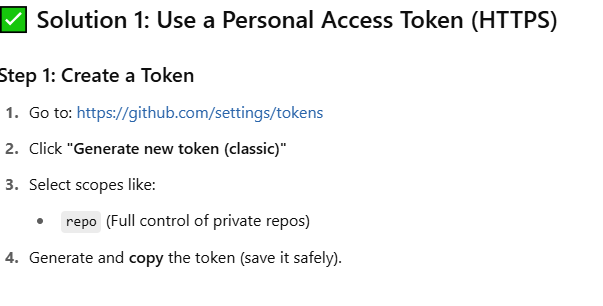
Analyse about directory , cd task repo, .

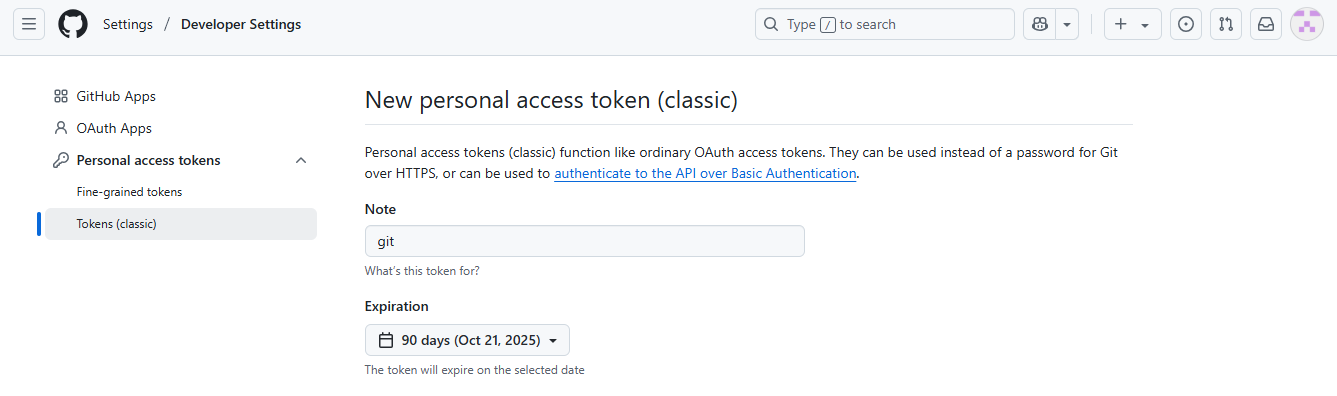
4) Create two files in local repo.

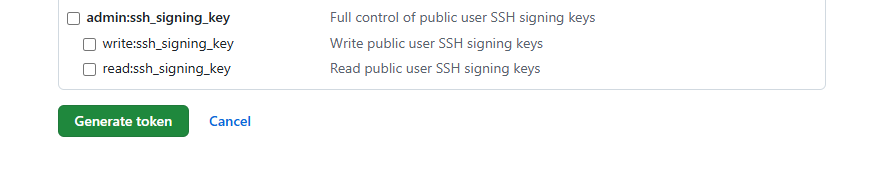


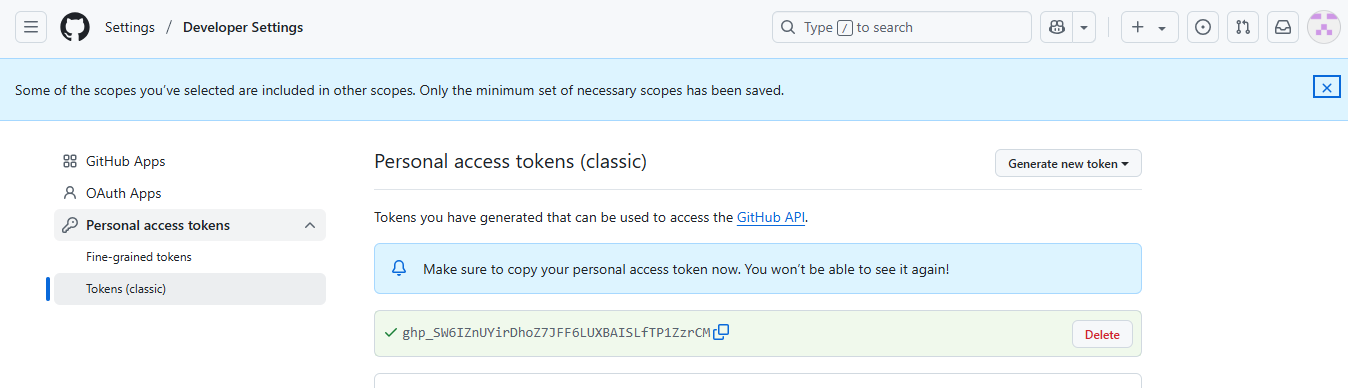
5) Commit two files and push to central Repository.



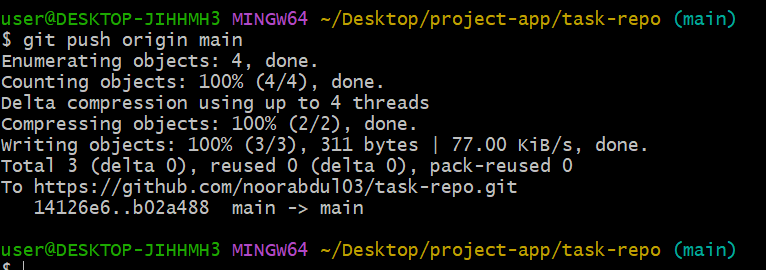






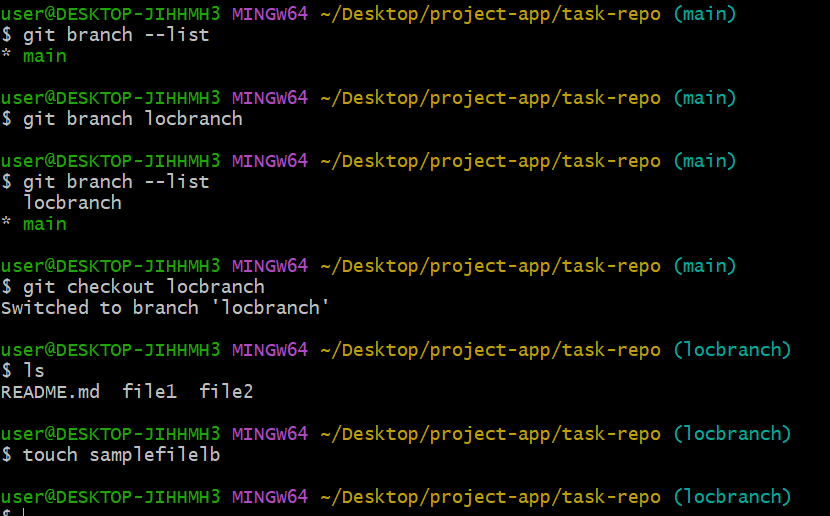


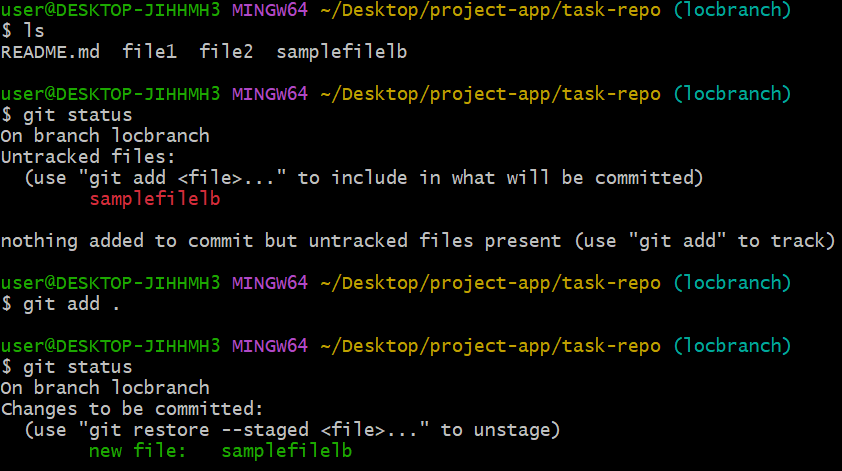
**ghp\_SW6IZnUYirDhoZ7JFF6LUXBAISLfTP1ZzrCM**

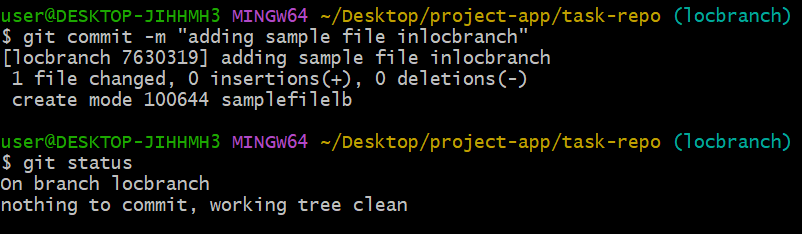


6)Create a branch in local and create a sample file and push to central.

ghp\_DgGEwqzrA06v3bm38UF8gXMiW5lNE31uWjUY 





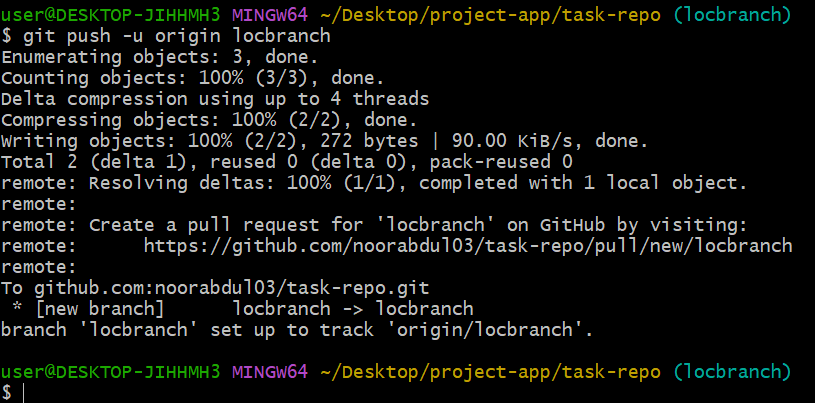


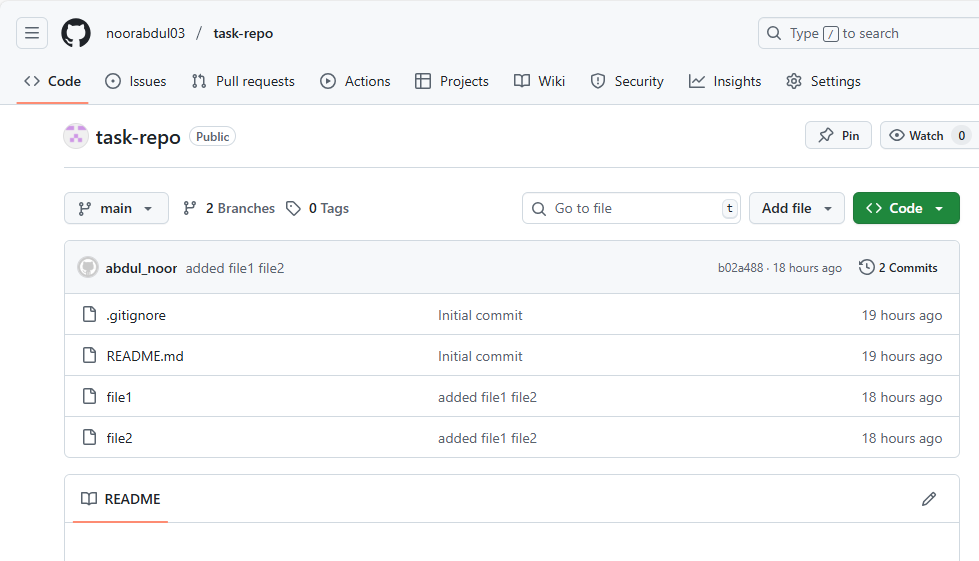
It was not pushing throughing the error or that I have changed the HTTPS to SSH to avoid every time giving username and password. To change this

git remote add origin url (SSH)

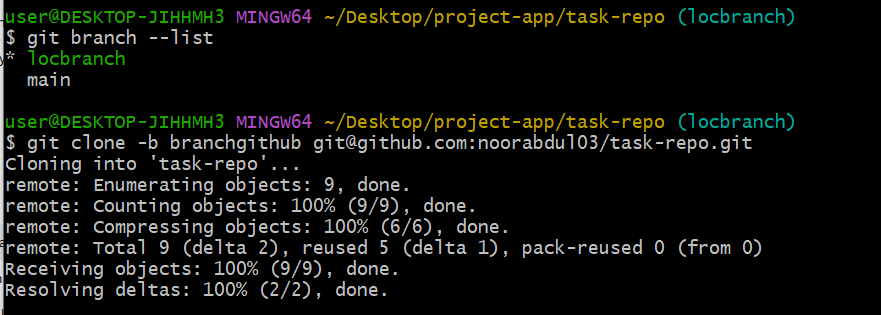


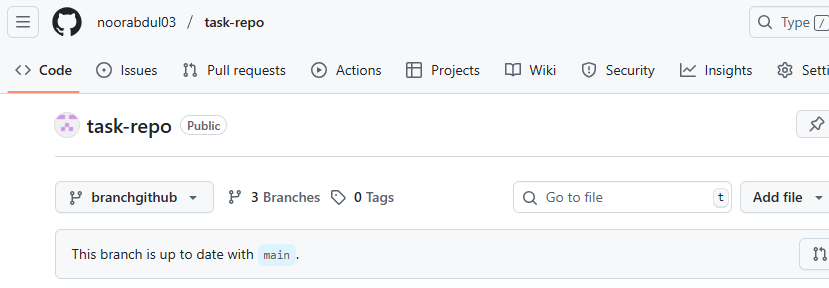
Now it did push and its not asking the credentials.



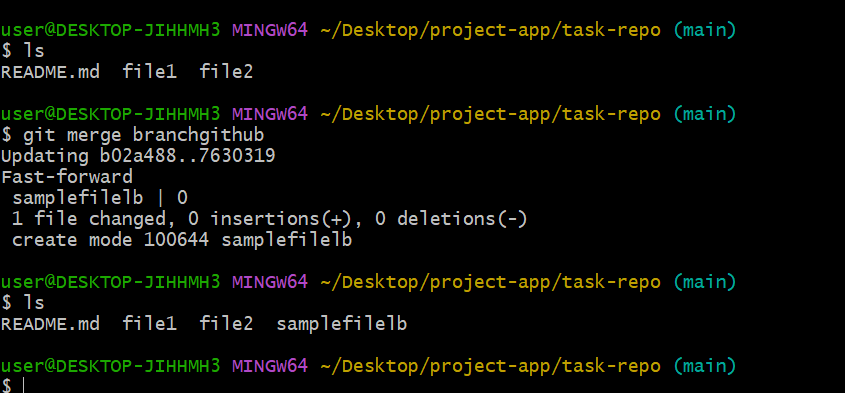


7) Create a branch in github and clone that to local.

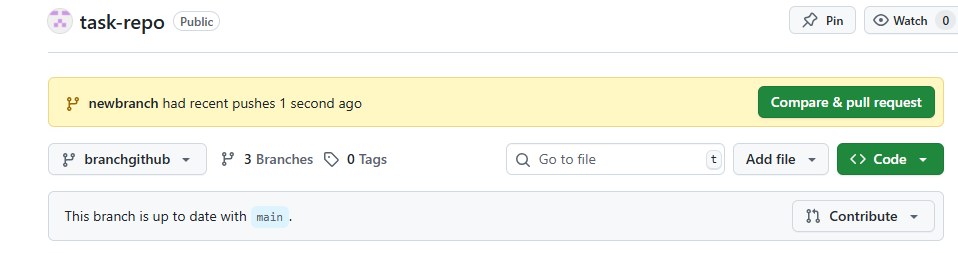


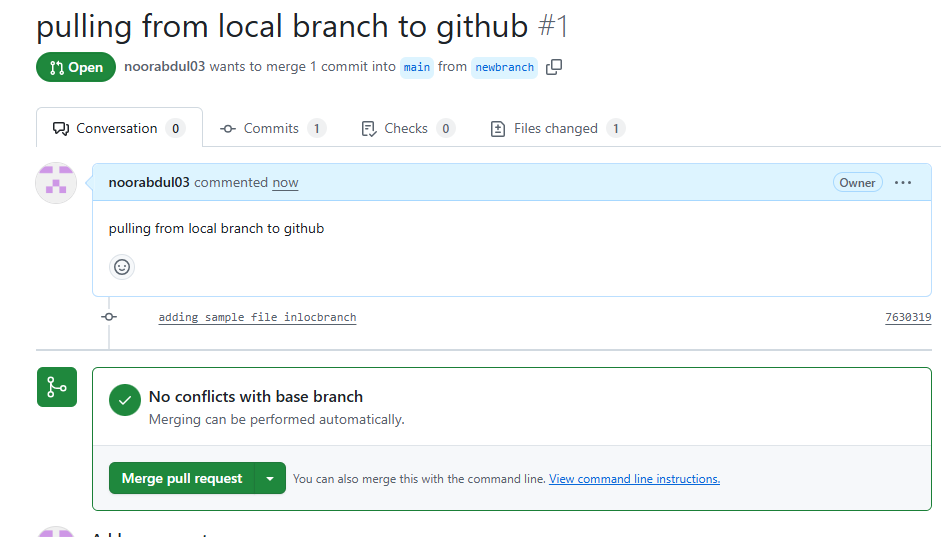


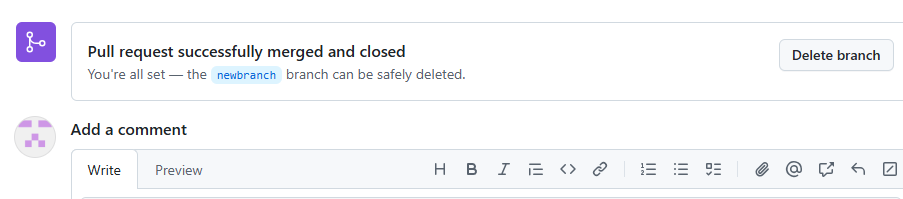
8)Merge the created branch with master in git local.



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



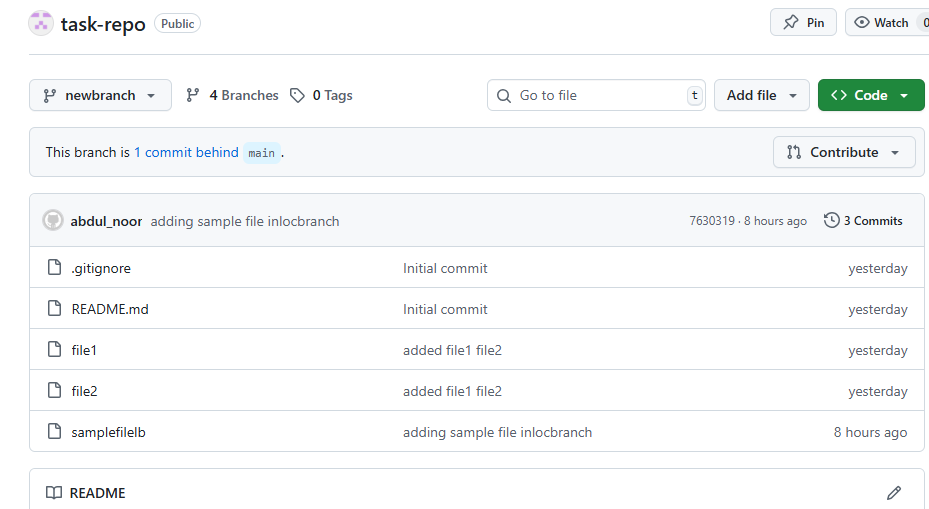




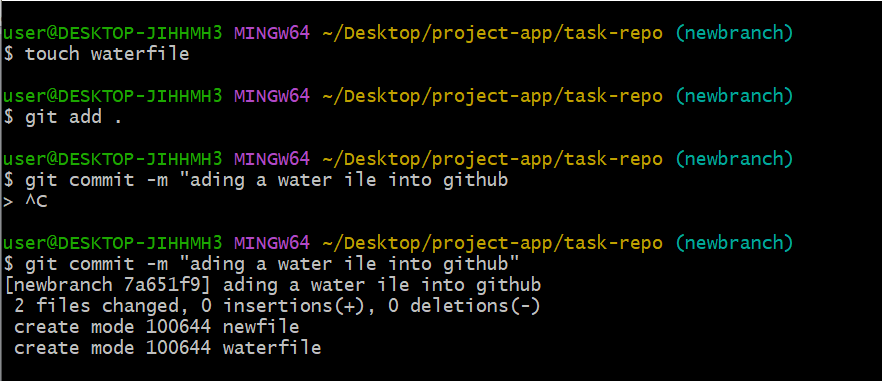
Just for information

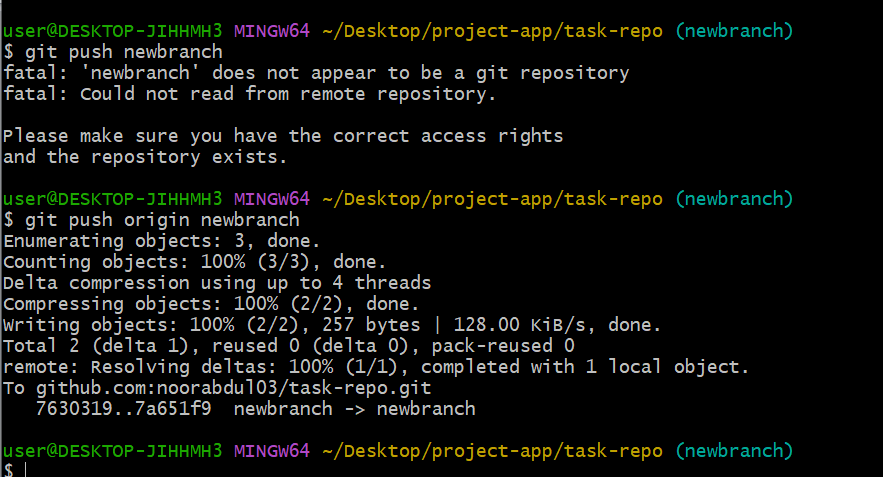


10)create a file in local and send that to branch in github.



Before it was

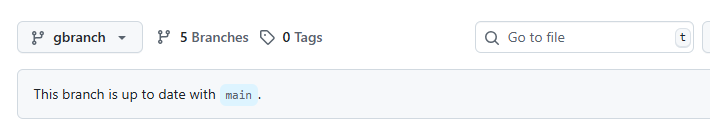


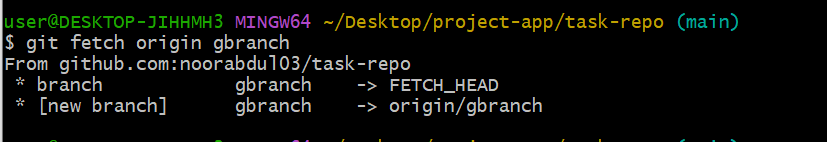


11) clone only a branch from github to local.

As we already cloned repo and the newly created branch doesn’t have any files in that branch so the (I any files then it might give display as 100%) steps are:

First step : created a branch naming gbranch in github







Cloded the task …………………..

**Below is error analysed and learning notes. Which I got beore doing the above activity**.

Actually I created a Abcbranch in github and I cloned being in cd task-repo/

What system did , system created a branch in another folder naming again task-repo . so that’s why I am able see my Abcbranch in the branch list and I am not able checkout to this branch. The error was coming : **fatal: destination path 'task-repo' already exists and is not an empty directory.**

**So I need deted the task-repo/ by seeing the path**

**Desktop/project-app/task-repo/**

**Ls**

**It is showing here one more folder task-repo/**

**So I deleted first the newly added task-repo/**

**Rm –rf task-repo/**

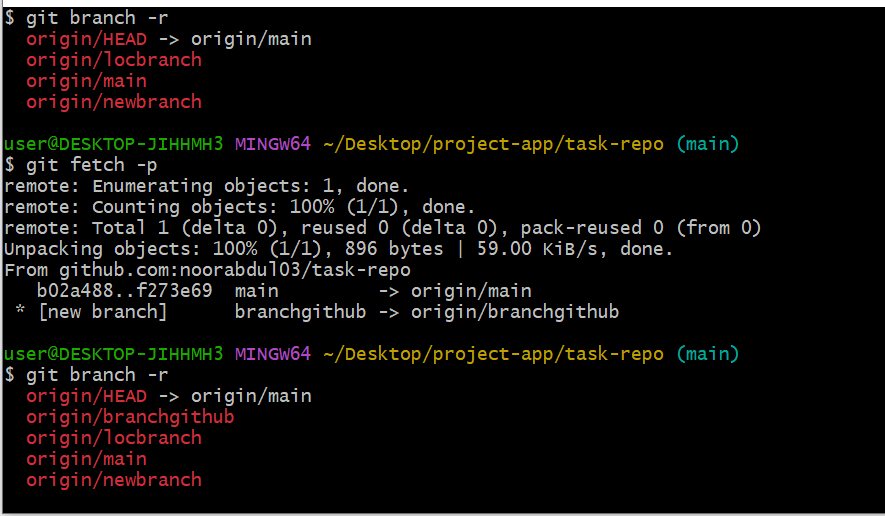
Again just for smooth work I deleted the branch created ABC from github

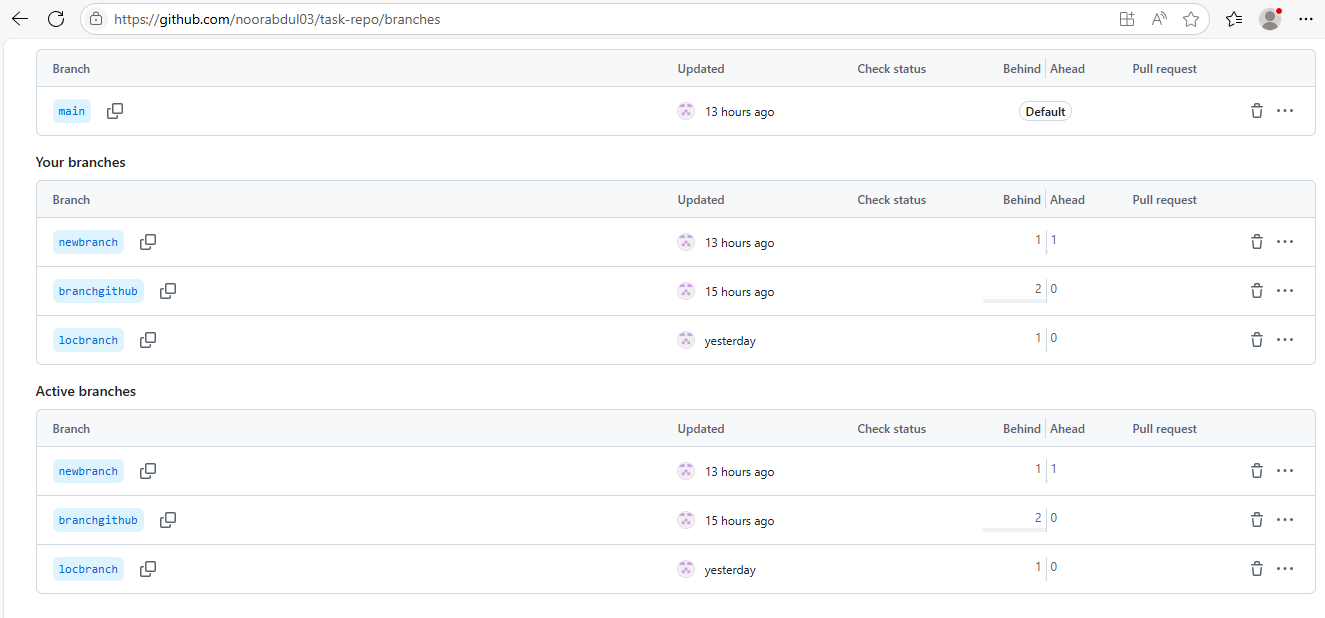
The link I used <https://github.com/noorabdul03/task-repo/branches>

From here I deleted the Abc branch from your branches the I need to deleted from below Active branches also then It was couldn’t delete the brach rom here. So I did

From git by using the below command in git

After that I just refreshed the github it removed from the active branches.





Details:

git fetch –p

fetch updates the list of remote branches

-p (prune) removes branches that were deleted from GitHub

git branch -r

lists all remote branches your local Git is aware of.

**What are remote branches?**

Remote branches are **branches stored on the remote repo (e.g., GitHub)** that your local Git knows about after fetching.

They usually appear as:

bash

CopyEdit

origin/main

origin/dev

origin/feature1

* origin is the default name for your GitHub repo (the "remote")
* origin/main means the main branch on GitHub

**Why We Use git fetch**

git fetch is used to **update your local Git with the latest changes from the remote repository** — without merging them into your current working branch.

**🔍 In Simple Terms:**

* It **downloads** information about new branches, deleted branches, new commits, etc., **from GitHub** (or any remote).
* But it **does NOT change your files** or current branch — just updates your .git data.

Q: git clone -b gbranch --single-branch git@github.com:noorabdul03/task-repo.git can i use this command where i already cloned repo

No — you **should NOT** use this command if you’ve **already cloned the repo**.

### Why?

bash

CopyEdit

git clone -b gbranch --single-branch git@github.com:noorabdul03/task-repo.git

* This tries to **create a new folder** named task-repo.
* Since you **already cloned the repo**, you'll get an error like:

pgsql

CopyEdit

fatal: destination path 'task-repo' already exists and is not an empty directory.

### What to do instead:

Since the repo is already cloned and gbranch was created **after** cloning:

1. **Go to your repo folder:**

bash

CopyEdit

cd task-repo

1. **Fetch the branch:**

bash

CopyEdit

git fetch origin gbranch

1. **Checkout the branch locally:**

bash

CopyEdit

git checkout -b gbranch origin/gbranch

Q: yes repo is already cloned before creating a gbranch in github

Got it! Since your repo was **already cloned before gbranch was created on GitHub**, you need to **fetch the new branch** and then check it out locally.

**Steps to get gbranch locally:**

1. **Fetch the new branch from GitHub:**

bash

CopyEdit

git fetch origin gbranch

1. **Create a local branch tracking the remote gbranch:**

bash

CopyEdit

git checkout -b gbranch origin/gbranch

**🔍 What this does:**

* git fetch origin gbranch: pulls the new branch metadata from GitHub.
* git checkout -b gbranch origin/gbranch: creates and switches to a local branch that tracks the remote one.

**🛠 Optional: List all remote branches**

To see if gbranch exists on GitHub:

bash

CopyEdit

git branch -r

Look for:

bash

CopyEdit

origin/gbranch

**Before cloning Repo need to branch newly created steps as below**

Q: i have created one branch naming gbranch and can i want to clone this branch by using command git clone -b gbranch [git@github.com:noorabdul03/task-repo.git](mailto:git@github.com:noorabdul03/task-repo.git)

You're **almost correct**, but one small detail is missing for **cloning only gbranch**. You should include --single-branch to avoid cloning all branches.

**Correct command to clone only gbranch:**

bash

CopyEdit

git clone -b gbranch --single-branch git@github.com:noorabdul03/task-repo.git

**❗If you run without --single-branch:**

bash

CopyEdit

git clone -b gbranch git@github.com:noorabdul03/task-repo.git

It will:

* Checkout gbranch after cloning
* But **still download all other branches' history** (slower/larger)

**💡 Final Tip:**

If you **already have** the repo cloned and want to get the gbranch:

bash

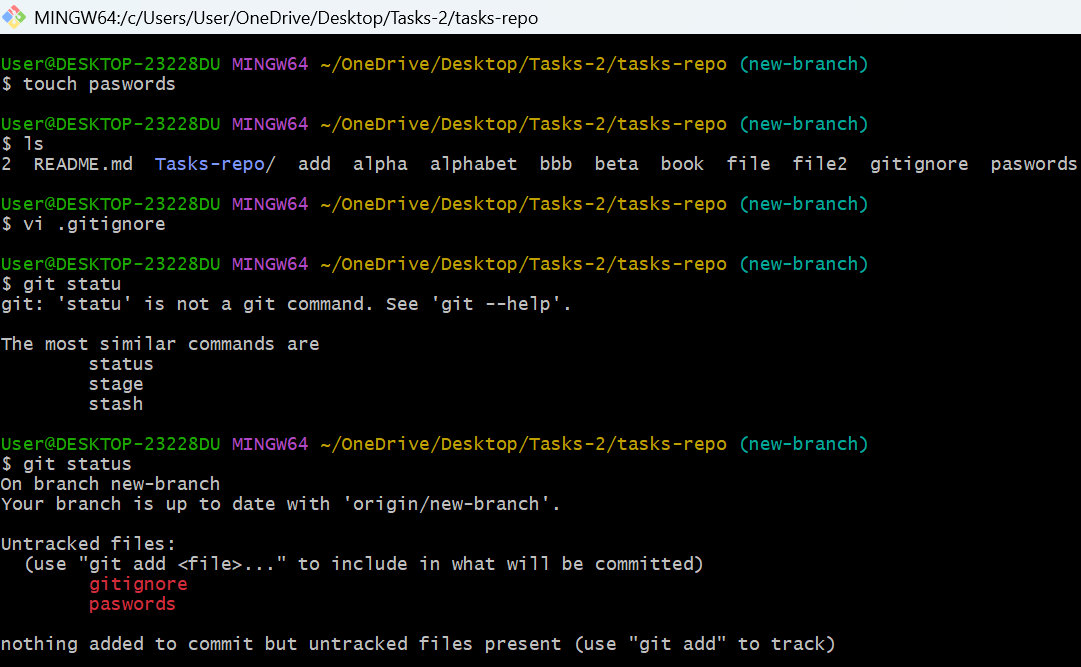
CopyEdit

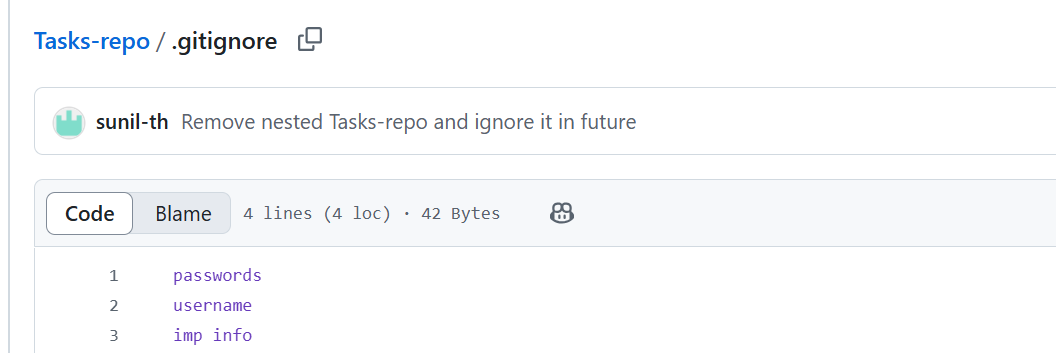
git fetch origin gbranch

git checkout gbranch

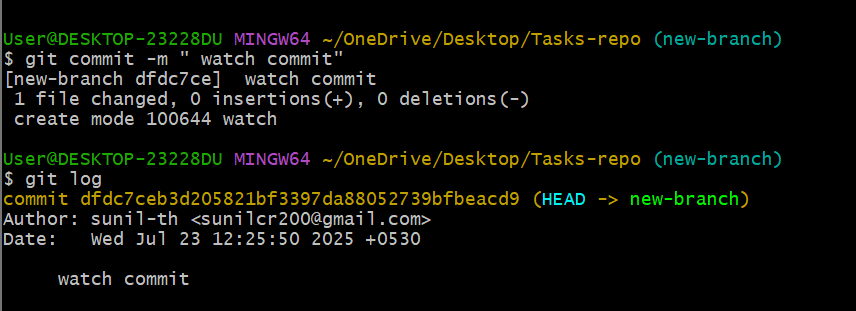
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

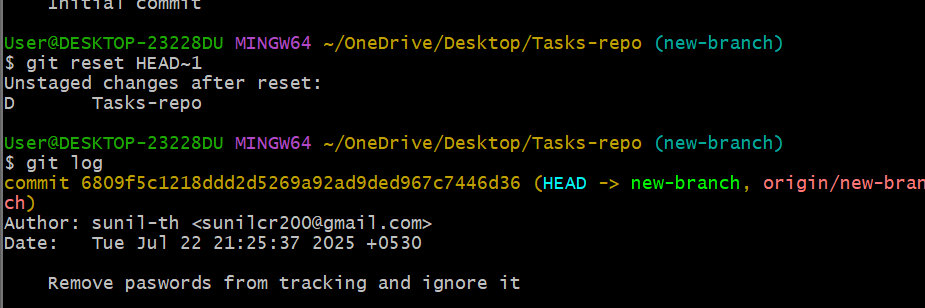
12) create a file with all passwords and make that untrackable with git.



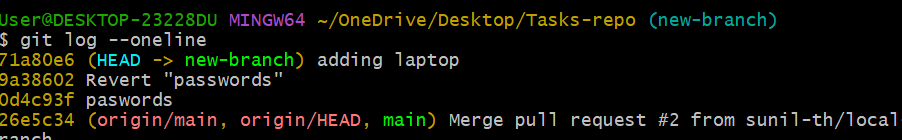


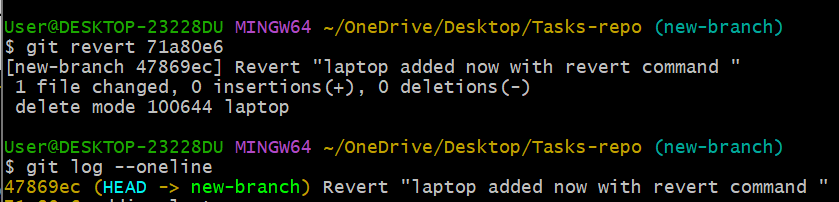
13)make a commit and make that commit reset without savings changes.



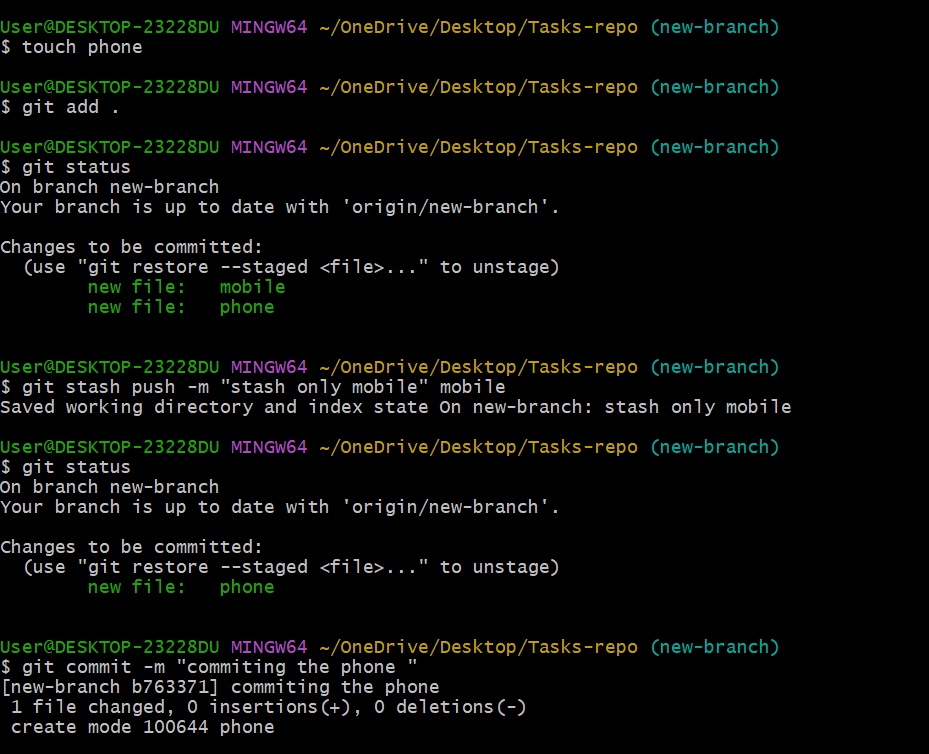


14)Revert a commited commit to the older version.

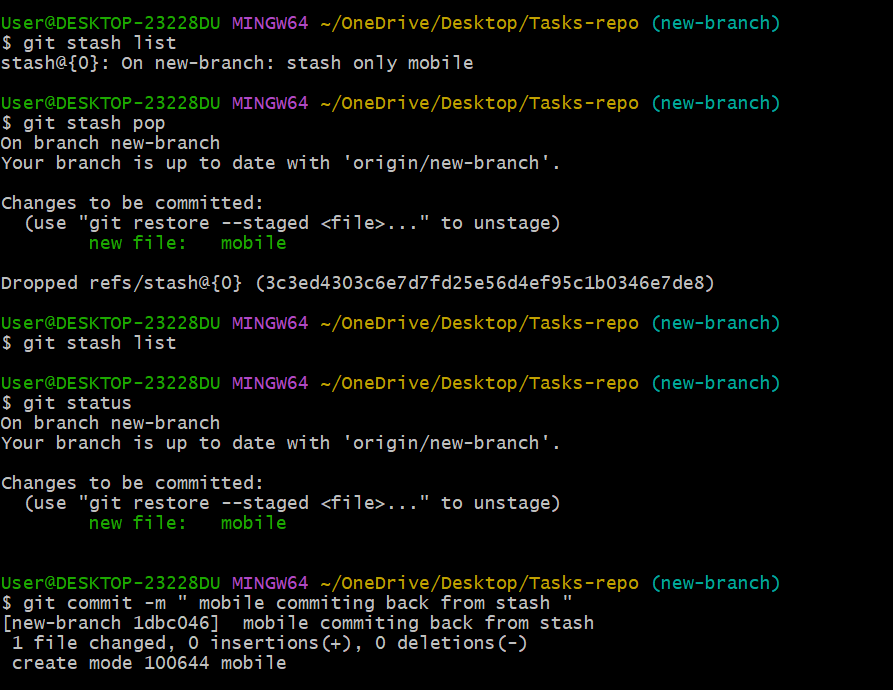




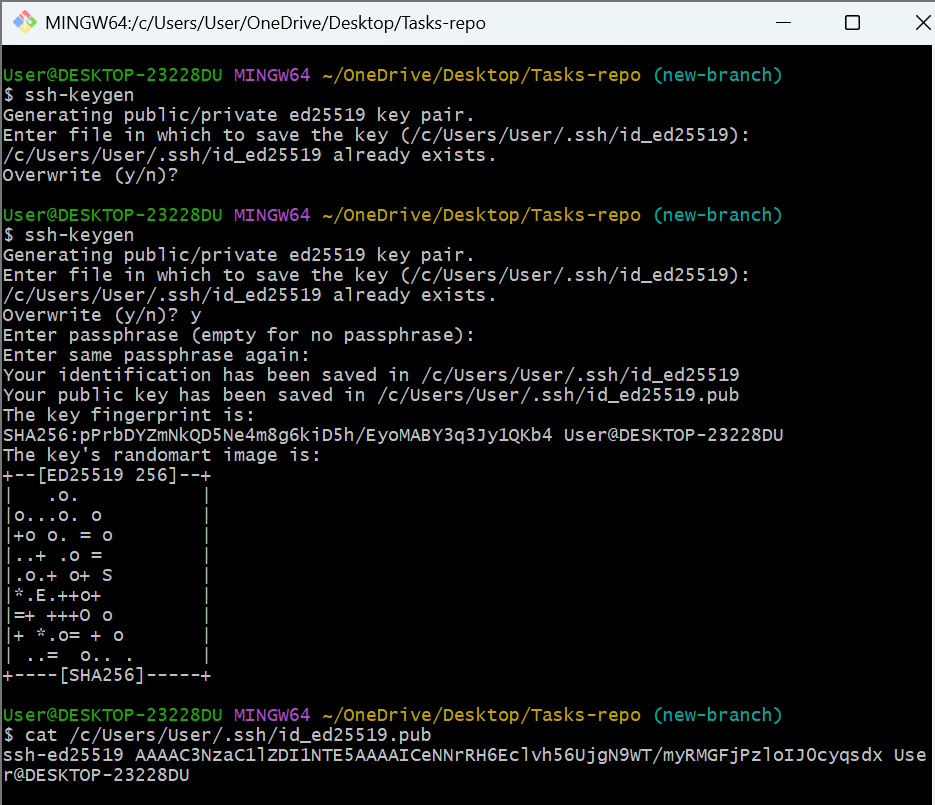
15)push a file to stash without savings the changes and work on another file.



16)undo the stash file and start working on that again.



17)generate a ssh-keygen and configure into github.

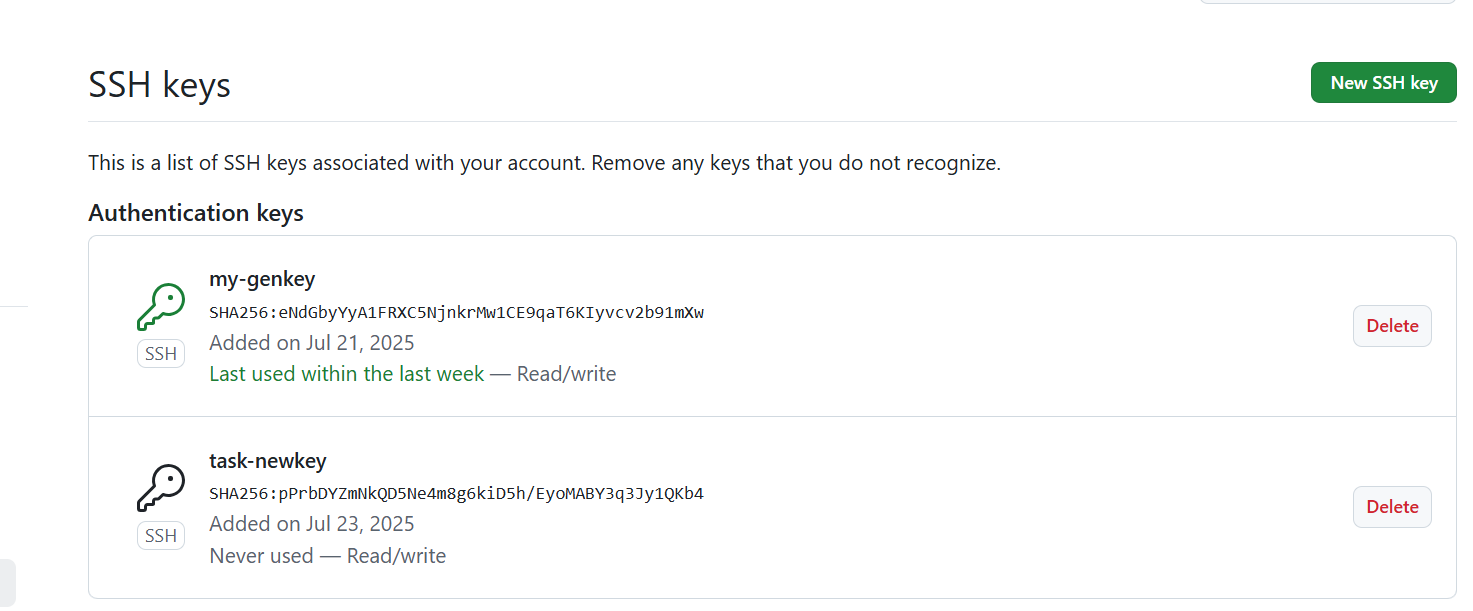


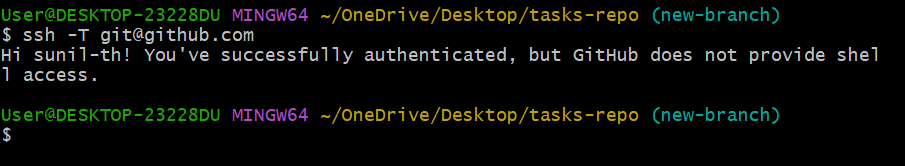
Add the key to GitHub

Go to [GitHub SSH Keys page](https://github.com/settings/ssh/new)

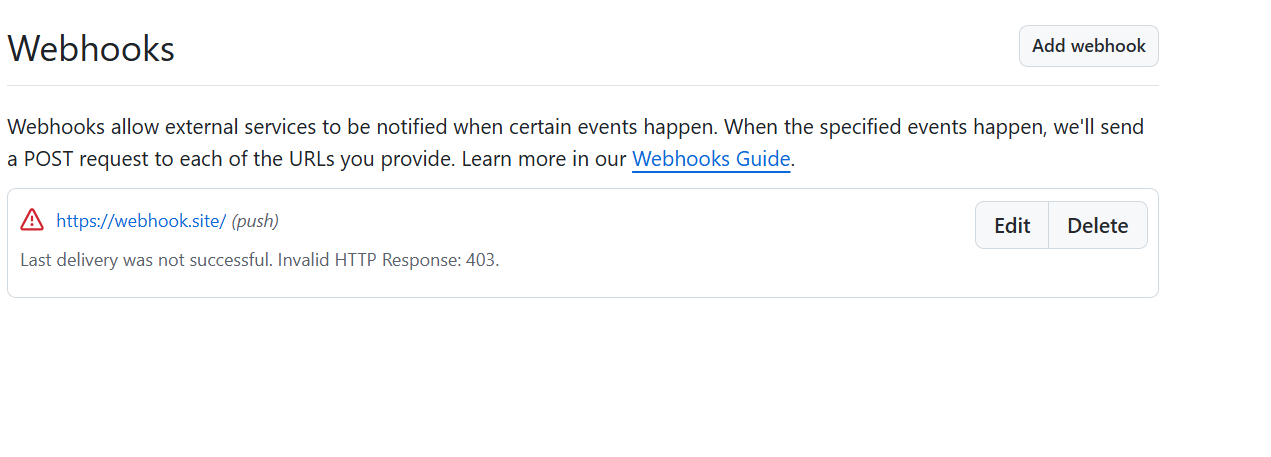
**Key**: Paste the copied public key.

Click **"Add SSH key"**





18)configure webhooks to github.



19)basic understanding of .git file.

The .git directory is the **heart of a Git repository**. When you run:

git init

Git creates a hidden .git/ folder inside your project. This folder contains everything Git needs to track changes, branches, history, remotes, and more.

20)Check all the logs of git.

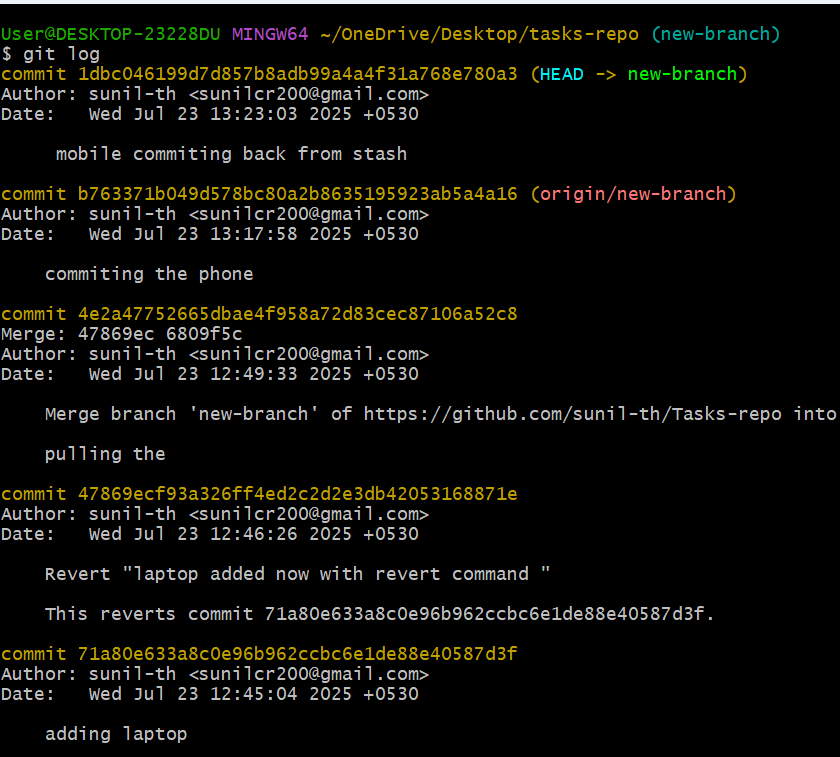
git log

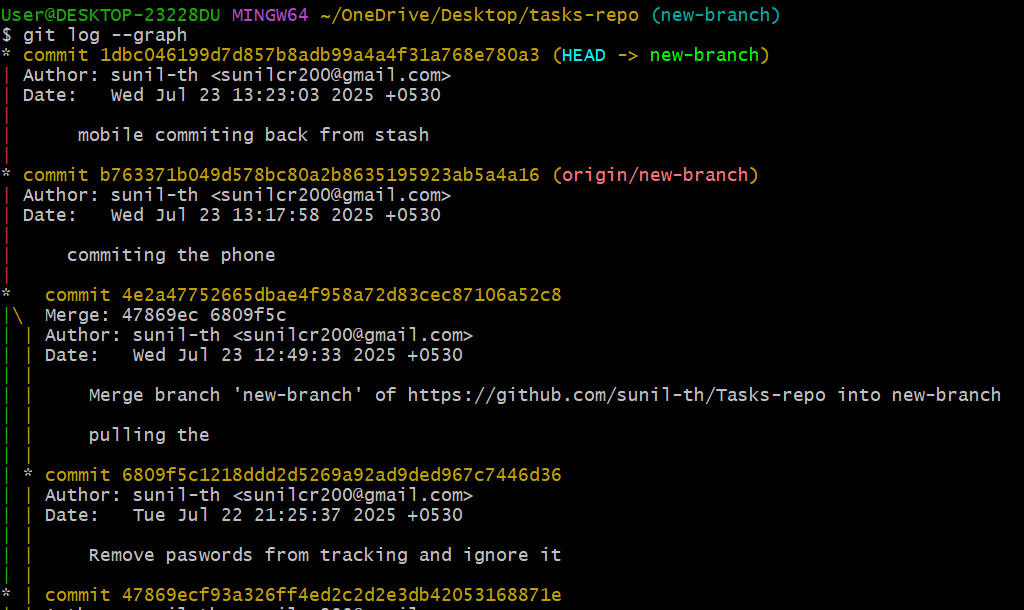
git log --oneline # One-line per commit

git log --graph # Shows branches and merges

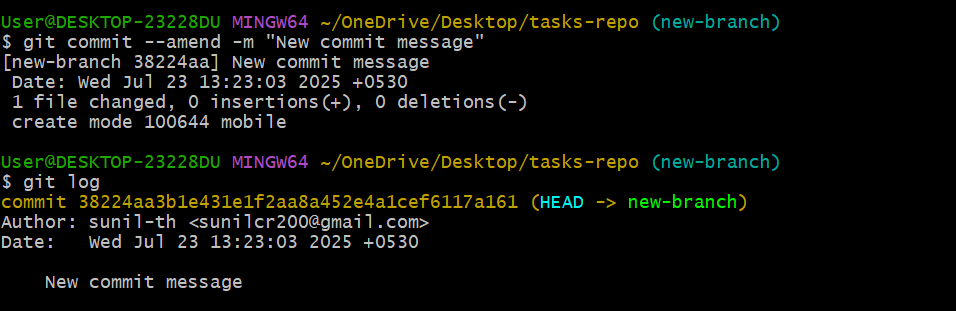
git log --all # Shows commits from all branches

git log --stat # Shows file-level changes in each commit





21)Rename the commit message.



22)Merge multiple commits into single commit.

