Work with Git and GitHub

What is Git

Git is a tool you install on your computer. It helps you ***track changes*** *to your files.*

If you make a mistake, you can ***go back to an earlier version.***

Git is a DVCS (Distributed Version Control System), ***everyone has their own copy*** *of the file. You can work on your copy, even without the internet. Later, everyone shares their changes, and the project gets updated.*

How to install Git :

For Ubuntu:

- ✓ ***sudo apt install git***

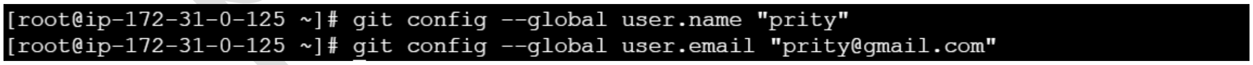
For RHEL/CentOS:

- ✓ ***sudo yum install git***

Configure Git

Run these commands to configure Git on you local:

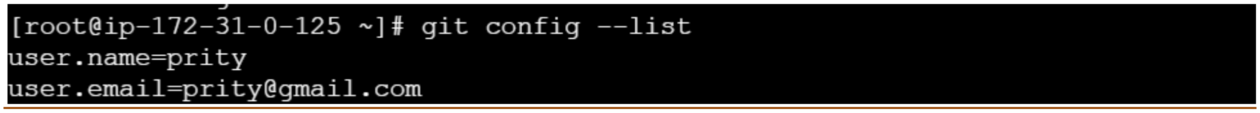
- ✓ ***git config --global user.name "Your Name"***
- ✓ ***git config --global user.email "youremail@example.com"***



```
[root@ip-172-31-0-125 ~]# git config --global user.name "prity"
[root@ip-172-31-0-125 ~]# git config --global user.email "prity@gmail.com"
```

Check Your Configuration

- ✓ ***git config --list***



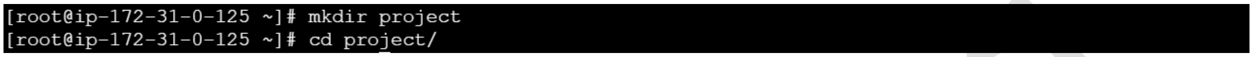
```
[root@ip-172-31-0-125 ~]# git config --list
user.name=prity
user.email=prity@gmail.com
```

To initialize a Git repository in a folder to be tracked, follow these steps:

- *Use the* ***cd*** *command to go to the folder you want to track with Git.*

Example: If your folder is named ***project****, run command*

- ✓ ***cd my-project***

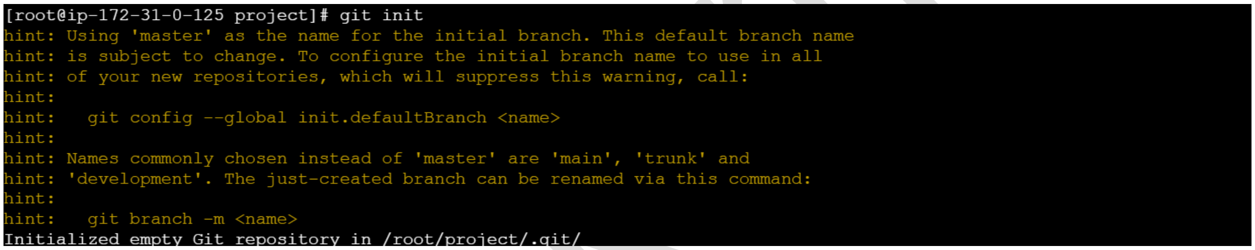


```
[root@ip-172-31-0-125 ~]# mkdir project
[root@ip-172-31-0-125 ~]# cd project/
```

Initialize a Git repository:

Run the following command to make the folder a Git repository:

- ✓ ***git init***

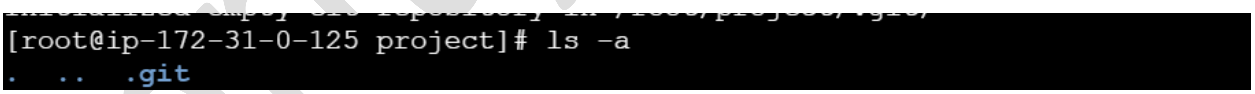


```
[root@ip-172-31-0-125 project]# git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /root/project/.git/
```

This command sets up a new Git repository in the folder by creating a ***.git*** *directory.*

To confirm the folder is now a Git repository, run:

- ✓ ***ls -a***



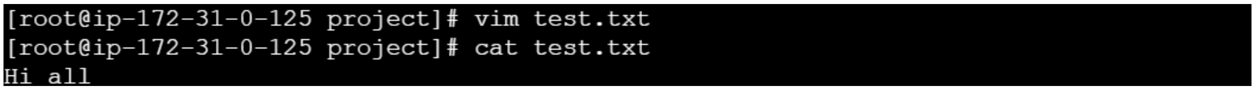
```
Initialized empty Git repository in /root/project/.git/
[root@ip-172-31-0-125 project]# ls -a
.   ..   .git
```

You should see a hidden ***.git*** *folder. This indicates that the directory is now tracked by Git.*

Create a New File:

Example: vim test.txt

Then type some content in the file and save it.

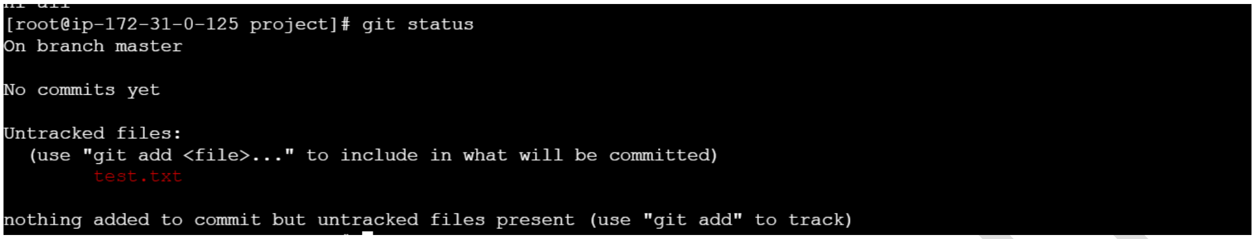


```
[root@ip-172-31-0-125 project]# vim test.txt
[root@ip-172-31-0-125 project]# cat test.txt
Hi all
```

Check File Status :

To see the current state of files in the repository (tracked or untracked), run:

- ✓ ***git status***

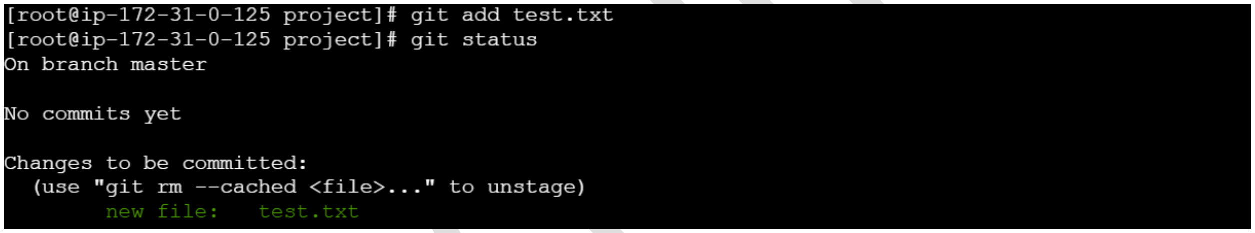


```
[root@ip-172-31-0-125 project]# git status
On branch master

No commits yet

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

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

*New files will appear under "**Untracked files**".*

Add the File to Git Tracking:

- ✓ ***git add test.txt***



```
[root@ip-172-31-0-125 project]# git add test.txt
[root@ip-172-31-0-125 project]# git status
On branch master

No commits yet

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

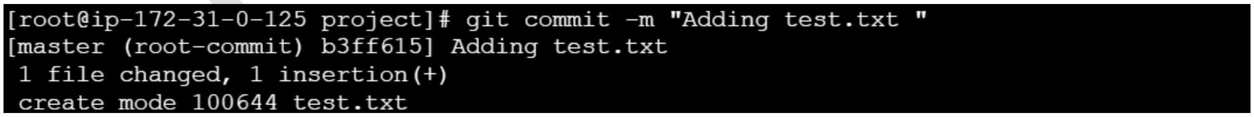
Or, to add all new/modified files in the folder:

- ✓ ***git add .***

Commit the File:

After adding the file, commit it to save the changes locally with a message:

git commit -m "Adding test.txt "



```
[root@ip-172-31-0-125 project]# git commit -m "Adding test.txt "
[master (root-commit) b3ff615] Adding test.txt
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt
```

Push the File to Remote Repository:

*If your repository is linked to a remote (e.g., GitHub), push the changes****

Before that lets learn about GitHub

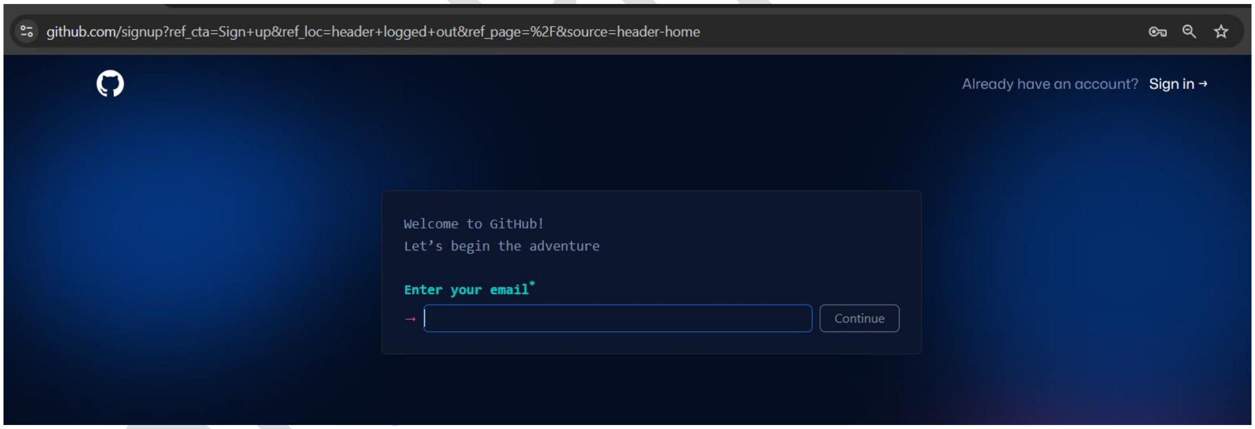
What is GitHub

*GitHub is a **website**. It's a place where you can **upload your Git projects** to the internet. This lets you **share your work** with others or work together on the same project.*

*GitHub is a **hosting service for Git repositories**. Git and GitHub work together, but they are not the same thing.*

How to create GitHub Account

- *Go to GitHub.com.*
- *Sign up for free.*

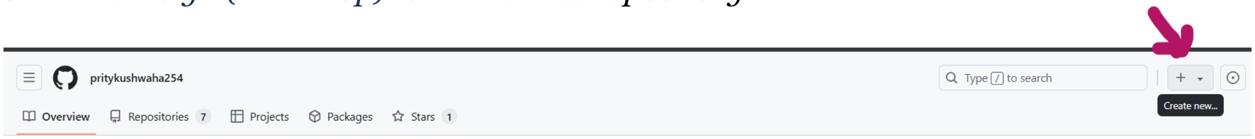


github.com/signup?ref_cta=Sign+up&ref_loc=header+logged+out&ref_page=%2F&source=header-home

```
Welcome to GitHub!
Let's begin the adventure

Enter your email*
→
```

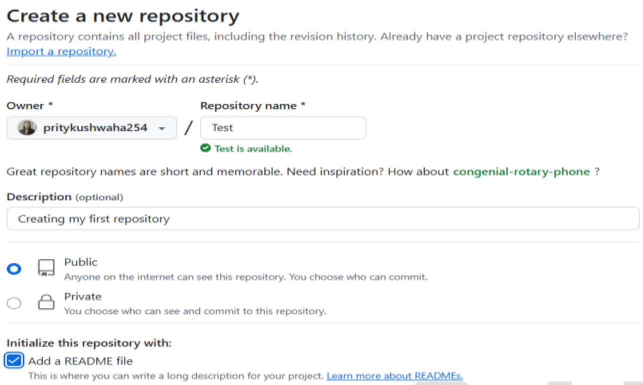
Create a GitHub Repository:

- *Login to [GitHub](https://github.com)*
- *Click the + sign (at the top) to create new repository.*



- *Enter a **repository name** (any name you wish)*

- *Add a description (it is optional).*
- *Choose **Public** (for everyone to see) or **Private** (only you can see).*
- *Tick **"Initialize this repository with a README"** (optional).*



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * pritykushwaha254 / **Repository name *** Test

✓ Test is available.

Great repository names are short and memorable. Need inspiration? How about congenial-rotary-phone ?

Description (optional)

Creating my first repository

● 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.

- *Click **Create repository**. One repository will create on GitHub.*

Connect to your GitHub repository from your local (Git)

There is two way to connect with your GitHub repository from local

1) SSH URL

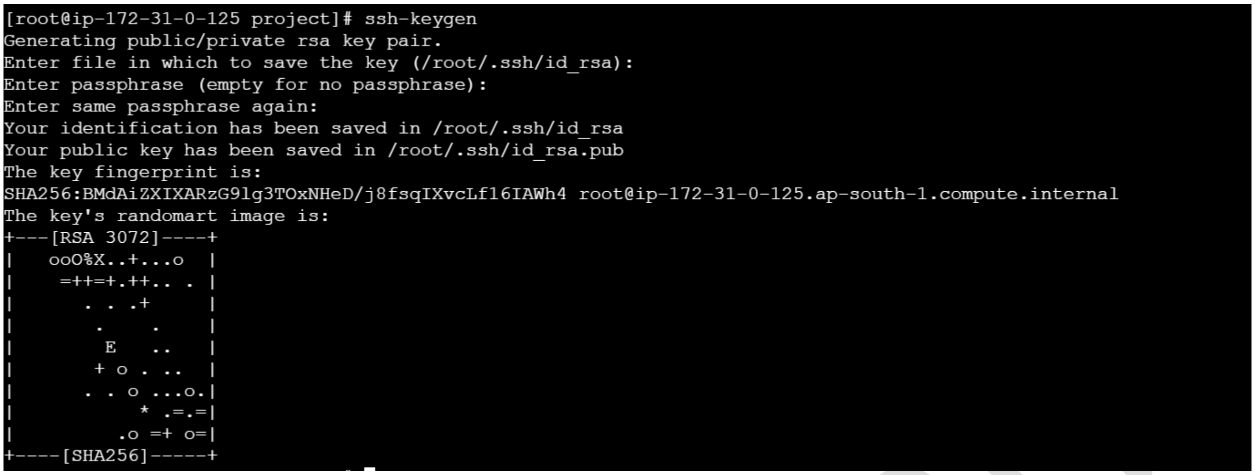
2) HTTPS URL

- ***SSH**: More secure, no need to enter credentials repeatedly.*
- ***HTTPS**: Simpler setup but requires a token/password when pushing or pulling.*

Connect to GitHub using SSH URL :

Generate an SSH key on your computer:

✓ ***ssh-keygen***

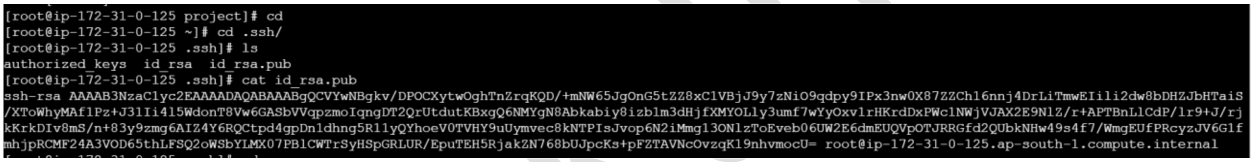


```
[root@ip-172-31-0-125 project]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:BMdAiZXIXARzG9lg3TOxNHeD/j8fsqIXvcLf16IAWh4 root@ip-172-31-0-125.ap-south-1.compute.internal
The key's randomart image is:
+---[RSA 3072]----+
|   ooO%X..+...o  |
|    =++=+.++.. . |
|      . . .+     |
|       .    .    |
|        E    ..  |
|       + o . ..  |
|      . . o ...o.|
|           *  .=.=|
|         .o =+ o=|
+----[SHA256]-----+
```

Add the SSH key to your GitHub account:

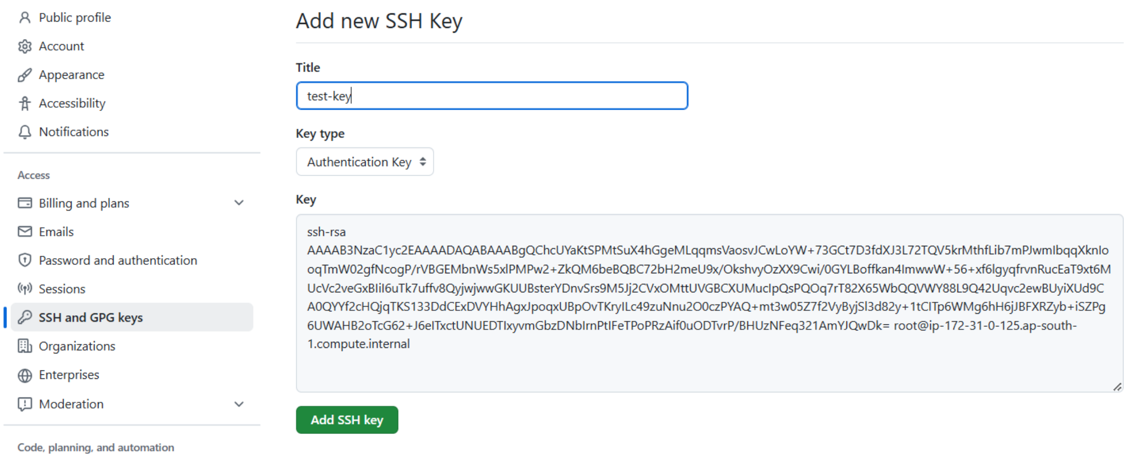
Copy the publickey:

- ✓ ***cat id_rsa.pub***



```
[root@ip-172-31-0-125 project]# cd
[root@ip-172-31-0-125 ~]# cd .ssh/
[root@ip-172-31-0-125 .ssh]# ls
authorized_keys  id_rsa  id_rsa.pub
[root@ip-172-31-0-125 .ssh]# cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCVYwNBgkv/DPOCXytwOghTnZrqKQD/+mNW65JgOnG5tZZ8xC1VBjJ9y7zNiO9qdpy9IPx3nw0X87ZZCh16nnj4DrLiTmwEIili2dw8bDHZJbHTaiS
/XToWhyMAf1Pz+J31Ii415WdonT8Vw6GASbVVqpzmoIqngDT2QrUtdutKBxgQ6NMYgN8Abkabiy8izb1m3dHjfXMYOLly3umf7wYyOxv1rHKrdDxPWc1NWjVJAX2E9N1Z/r+APTBnL1CdP/1r9+J/rj
kKrkDIv8mS/n+83y9zmg6AIZ4Y6RQCtpd4gpDn1dhng5R11yQYhoeV0TVHY9uUymvec8kNTPIsJvop6N2iMmg13ON1zToEveb06UW2E6dmEUQVpOTJRRGfd2QUbkNHw49s4f7/WmgEUfPRcyzJV6G1f
mhjpRCMF24A3VOD65thLFSQ2oWSbYLMX07PB1CWTrSyHSpGRLUR/EpuTEH5RjakZN768bUJpcKs+pFZTAVNcOvzqK19nhvmocU= root@ip-172-31-0-125.ap-south-1.compute.internal
```

Go to ***GitHub > Settings > SSH and GPG keys > New SSH key****, paste the key, and save it.*



Add new SSH Key

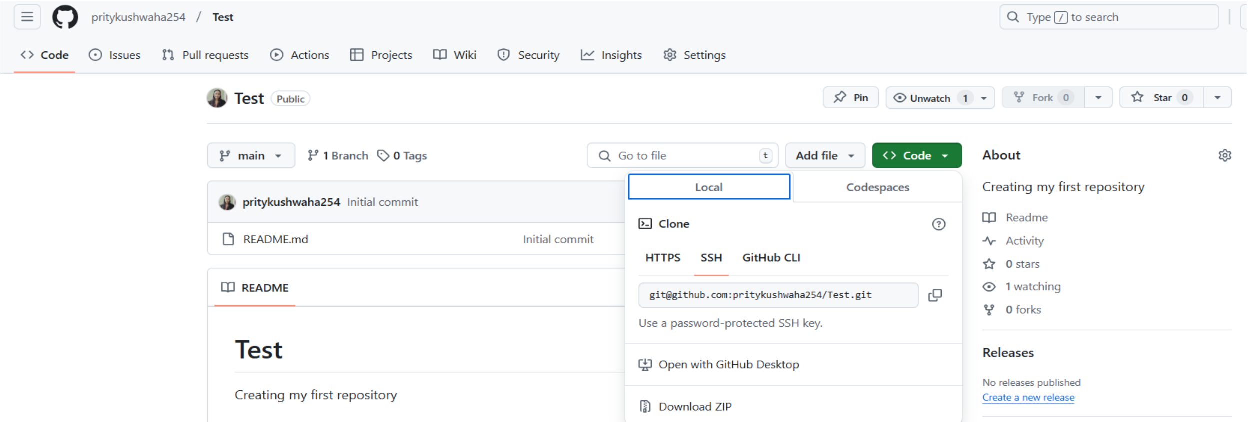
Title: test-key

Key type: Authentication Key

Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQChcUYaKtSPMtSuX4hGgeMLqqmsVaosvJCwLoYW+73GCt7D3fdXJ3L72TQV5krMthfLib7mPJwmIbqqXknIo
oqTmW02gfNcogP/rVBGEMbnWs5xlPMPw2+ZkQM6beBQBC72bH2meU9x/OkshvyOzXX9Cwi/0GYLBoffkan4ImwwW+56+xf6lgyqfrvnRucEaT9xt6M
UcVc2veGxBIiI6uTk7uffv8QyjwjwwGKUUBsterYDnvSrs9M5Jj2CVxOMttUVGBCXUMucIpQsPQOq7rT82X65WbQQVWY88L9Q42Uqvc2ewBUyiXUd9C
A0QYYf2cHQjqTKS133DdCExDVYHhAgxJpoqxUBpOvTKryILc49zuNnu2O0czPYAQ+mt3w05Z7f2VyByjSl3d82y+1tCITp6WMg6hH6jJBFXRZyb+iSZPg
6UWAHB2oTcG62+J6eITxctUNUEDTIxyvmGbzDNbIrnPtIFeTPoPRzAif0uODTvrP/BHUzNFeq321AmYJQwDk= root@ip-172-31-0-125.ap-south-
1.compute.internal
```

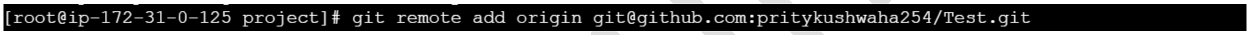
Copy the SSH URL of your GitHub repository



Go to your local Git and Run the following command in your local Git repository:

git remote add origin <SSH URL of your GitHub repository>

- ✓ ***git remote add origin git@github.com:pritykushwaha254/Test.git***

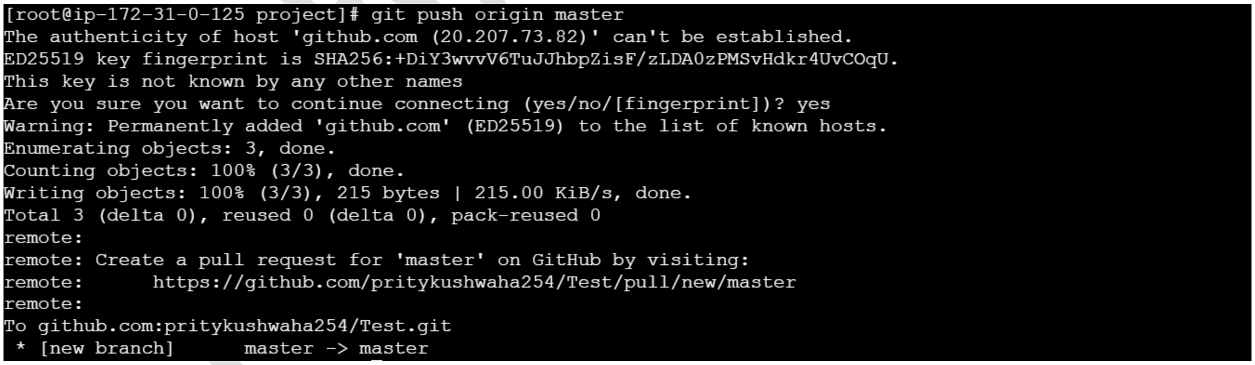


```
[root@ip-172-31-0-125 project]# git remote add origin git@github.com:pritykushwaha254/Test.git
```

Push the File to Remote Repository from local (Git)

*If your repository is linked to a remote (e.g., GitHub), push the changes****

- ✓ ***git push origin master***



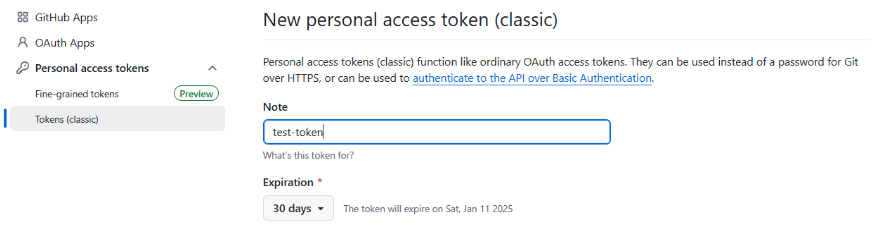
```
[root@ip-172-31-0-125 project]# git push origin master
The authenticity of host 'github.com (20.207.73.82)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 215 bytes | 215.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/pritykushwaha254/Test/pull/new/master
remote:
To github.com:pritykushwaha254/Test.git
 * [new branch]      master -> master
```

Now your code pushed to GitHub from your local.

Connect to GitHub using HTTPS URL:

To connect using HTTPS we need to generate personal access token in GitHub

Go to ***GitHub > Settings > Developer settings > Personal access tokens****. Use it as a password.*



New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

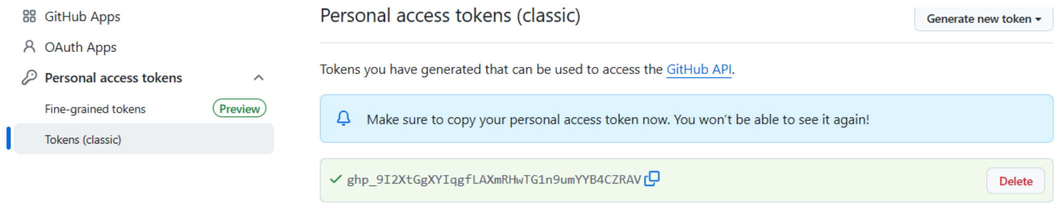
Note

test-token

What's this token for?

Expiration *

30 days ▾ The token will expire on Sat, Jan 11 2025



Personal access tokens (classic)

Tokens you have generated that can be used to access the GitHub API.

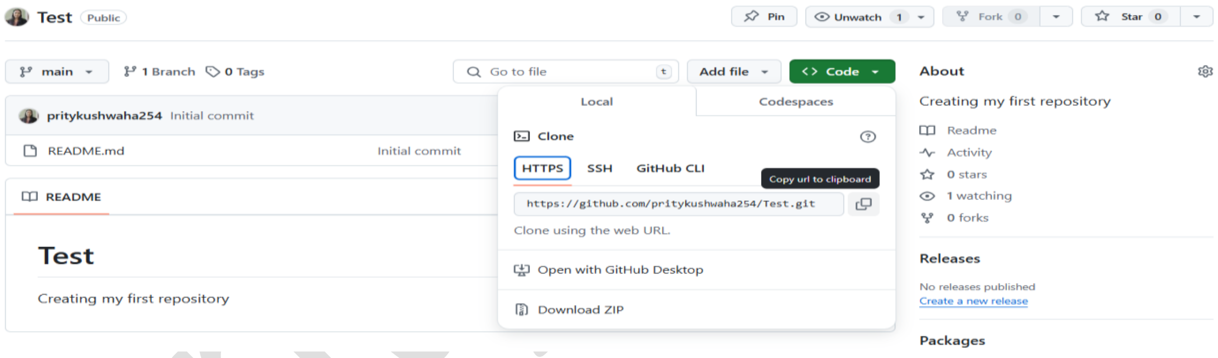
Make sure to copy your personal access token now. You won't be able to see it again!

✓ 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

Copy the token and save for further use it won't be visible again.

Copy the HTTPS URL of your GitHub repository.

Go to your local Git and Run the command in your local Git repository:



pritykushwaha254 Initial commit

README.md Initial commit

Test

Creating my first repository

https://github.com/pritykushwaha254/Test.git

Clone using the web URL.

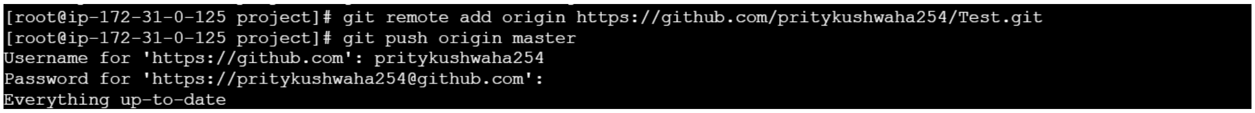
Creating my first repository

git remote add origin <HTTPS URL of your GitHub repository>

- ✓ *git remote add origin https://github.com/pritykushwaha254/Test.git*

When you push your code to GitHub it will ask:

- ➢ *Enter your **GitHub username**.*
- ➢ *Enter your **Personal Access Token** as the password.*



```
[root@ip-172-31-0-125 project]# git remote add origin https://github.com/pritykushwaha254/Test.git
[root@ip-172-31-0-125 project]# git push origin master
Username for 'https://github.com': pritykushwaha254
Password for 'https://pritykushwaha254@github.com':
Everything up-to-date
```

Here it is saying everything up-to-date because I have already push code to GitHub in previous example of SSH URL.

- *Always ensure you use the correct URL format based on your connection method (SSH or HTTPS).*

Now your file is being tracked by Git and, it will be available in your remote repository(GitHub).

Full Workflow summery:

```
cd project                          # Go to your project folder in local
git init                            # Initialize Git repository
vim test.txt                         # Create a file
git add test.txt                     # Add file to Git tracking
git commit -m "Add test.txt"         # Commit the file
git add origin URL                  # Connect with your GitHub repository
git push origin master               # Push changes to the remote repository
```

Git clone

git clone is a command used to ***copy a repository from GitHub*** *to your computer. When you clone a repository, you get a complete copy of all the files, commit history, and branches. Think of it like downloading a project so you can work on it locally.*

Cloning a Public Repository:

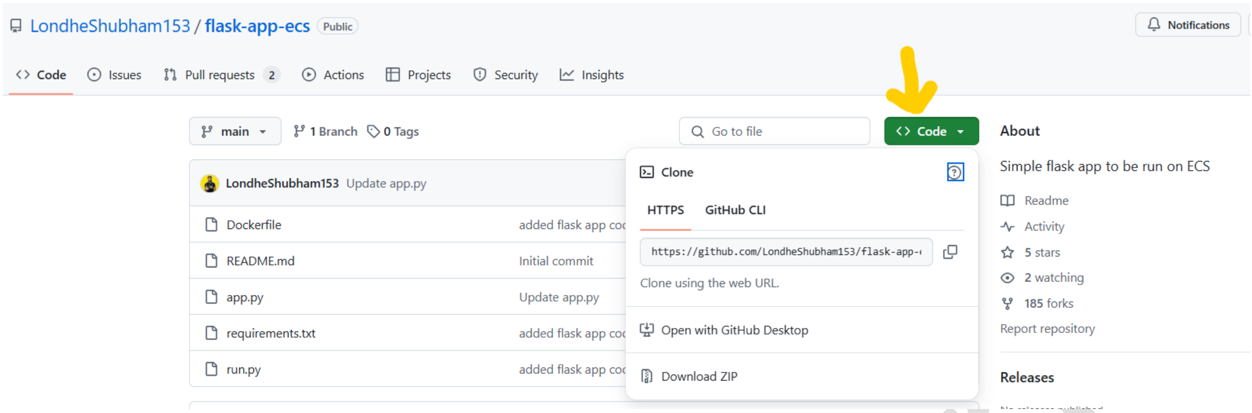
Public repositories are open to everyone, so you don't need any special authentication to clone them.

Go to GitHub and open the repository you want to clone.

Copy the Repository URL

Click on the green ***Code*** *button.*

Copy the URL (e.g., https://github.com/LondheShubham153/flask-app-ecs.git).



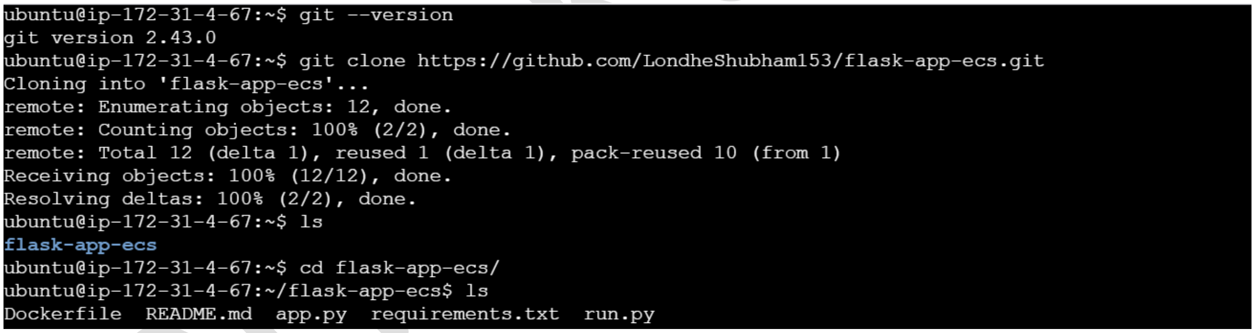
***Open a Terminal** go to the directory where you want to clone the repository.*

Run the git clone Command

- ✓ ***git clone <URL>***

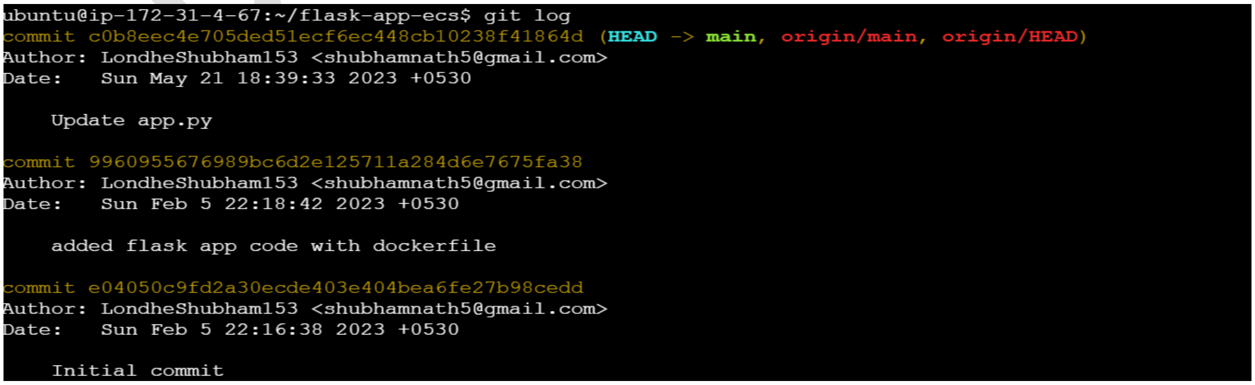
Example: git clone https://github.com/LondheShubham153/flask-app-ecs.git

This will create a new folder with the same name as the repository, containing all the files.



```
ubuntu@ip-172-31-4-67:~$ git --version
git version 2.43.0
ubuntu@ip-172-31-4-67:~$ git clone https://github.com/LondheShubham153/flask-app-ecs.git
Cloning into 'flask-app-ecs'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (2/2), done.
remote: Total 12 (delta 1), reused 1 (delta 1), pack-reused 10 (from 1)
Receiving objects: 100% (12/12), done.
Resolving deltas: 100% (2/2), done.
ubuntu@ip-172-31-4-67:~$ ls
flask-app-ecs
ubuntu@ip-172-31-4-67:~$ cd flask-app-ecs/
ubuntu@ip-172-31-4-67:~/flask-app-ecs$ ls
Dockerfile  README.md  app.py  requirements.txt  run.py
```

*You can also check commit history using **git log** command*



```
ubuntu@ip-172-31-4-67:~/flask-app-ecs$ git log
commit c0b8eec4e705ded51ecf6ec448cb10238f41864d (HEAD -> main, origin/main, origin/HEAD)
Author: LondheShubham153 <shubhamnath5@gmail.com>
Date:   Sun May 21 18:39:33 2023 +0530

    Update app.py

commit 9960955676989bc6d2e125711a284d6e7675fa38
Author: LondheShubham153 <shubhamnath5@gmail.com>
Date:   Sun Feb 5 22:18:42 2023 +0530

    added flask app code with dockerfile

commit e04050c9fd2a30ecde403e404bea6fe27b98cedd
Author: LondheShubham153 <shubhamnath5@gmail.com>
Date:   Sun Feb 5 22:16:38 2023 +0530

    Initial commit
```

Cloning a Private Repository:

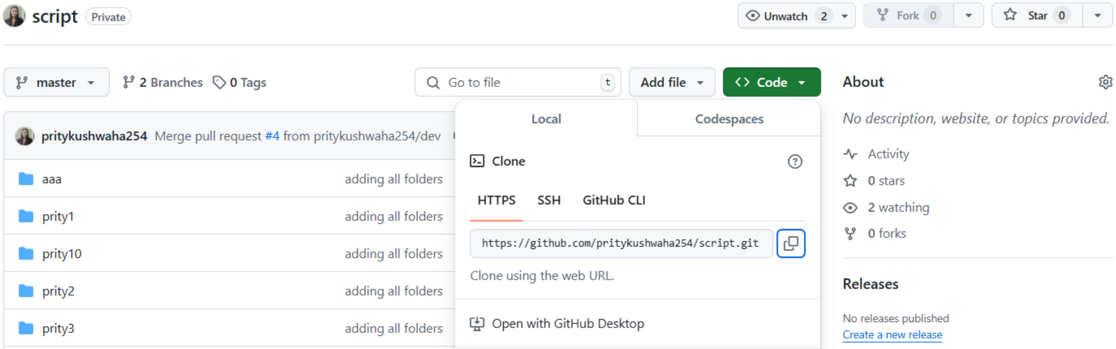
Private repositories require authentication since they are not open to everyone.

Using HTTPS:

*Go to GitHub and open the private repository you want to clone.**Copy the Repository HTTPS URL***

*Click the green **Code** button.*

Copy the HTTPS URL

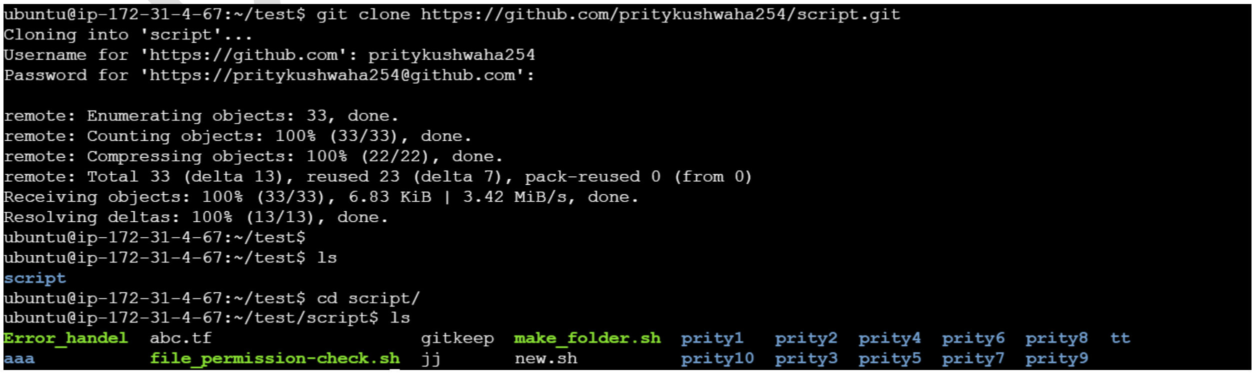


Run the git clone Command

- ✓ ***git clone <repository HTTPS URL>***

Example :** **git clone https://github.com/pritykushwaha254/script.git

*When it ask for, enter your GitHub **username** and **personal access token** instead of a password. (GitHub no longer supports password authentication; you must generate a **personal access token**.)*

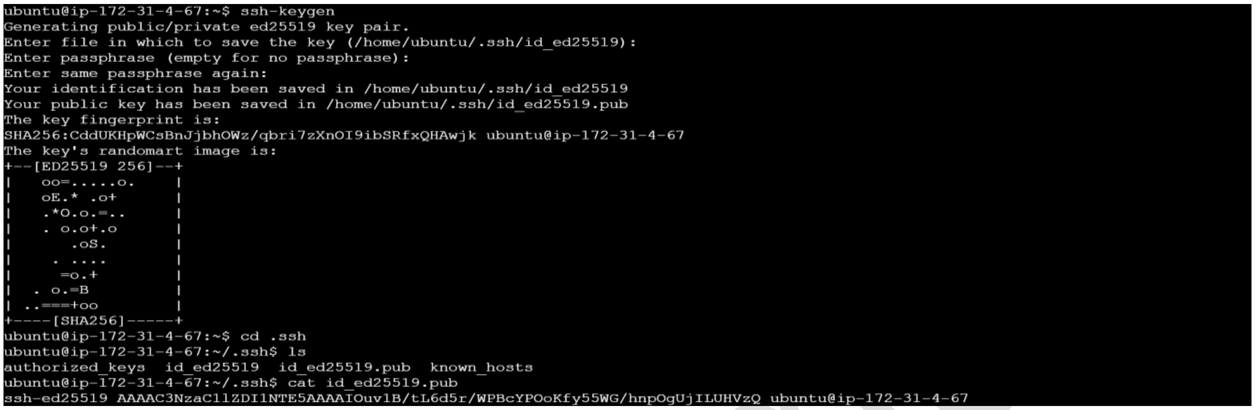


```
ubuntu@ip-172-31-4-67:~/test$ git clone https://github.com/pritykushwaha254/script.git
Cloning into 'script'...
Username for 'https://github.com': pritykushwaha254
Password for 'https://pritykushwaha254@github.com':

remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 33 (delta 13), reused 23 (delta 7), pack-reused 0 (from 0)
Receiving objects: 100% (33/33), 6.83 KiB | 3.42 MiB/s, done.
Resolving deltas: 100% (13/13), done.
ubuntu@ip-172-31-4-67:~/test$
ubuntu@ip-172-31-4-67:~/test$ ls
script
ubuntu@ip-172-31-4-67:~/test$ cd script/
ubuntu@ip-172-31-4-67:~/test/script$ ls
Error_handel  abc.tf                    gitkeep  make_folder.sh  prity1   prity2  prity4  prity6  prity8  tt
aaa           file_permission-check.sh  jj       new.sh          prity10  prity3  prity5  prity7  prity9
```

Now your private gitHub repository gets cloned in your local using HTTPS.

Using SSH:

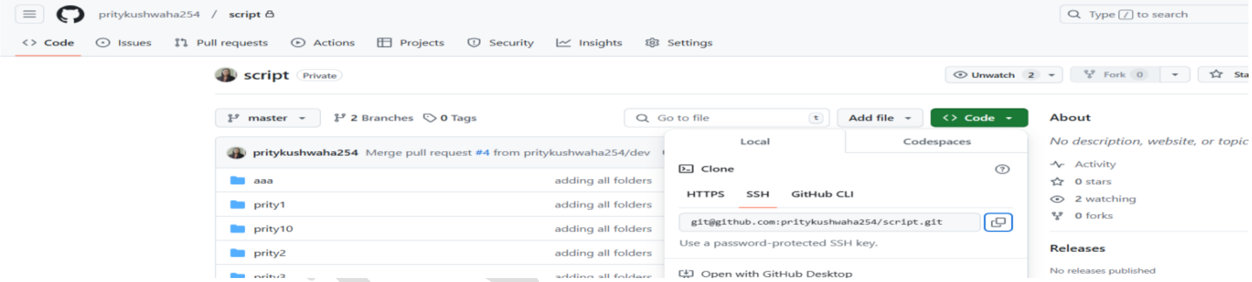
Generate an SSH key pair using ssh-keygen.



```
ubuntu@ip-172-31-4-67:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:CddUKHpWCsBnJjbhOWz/qbri7zXnOI9ibSRfxQHAwjk ubuntu@ip-172-31-4-67
The key's randomart image is:
+--[ED25519 256]--+
|    oo=.....o.   |
|    oE.* .o+     |
|    .*O.o.=..    |
|    . o.o+.o     |
|       .oS.      |
|     . ....      |
|      =o.+       |
|  . o.=B         |
| ..===+oo        |
+----[SHA256]-----+
ubuntu@ip-172-31-4-67:~$ cd .ssh
ubuntu@ip-172-31-4-67:~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub  known_hosts
ubuntu@ip-172-31-4-67:~/.ssh$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOuv1B/tL6d5r/WPBcYPOoKfy55WG/hnpOgUjILUHVzQ ubuntu@ip-172-31-4-67
```

Add the public key to your GitHub account as explained before

Copy the Repository SSH URL



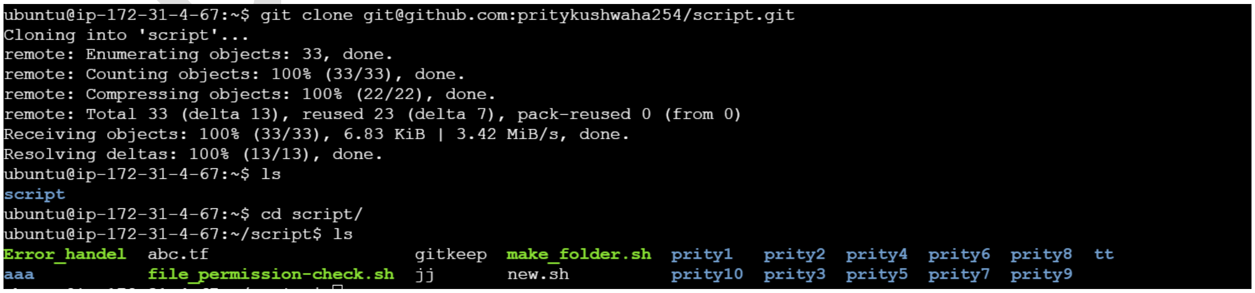
git@github.com:pritykushwaha254/script.git

Use a password-protected SSH key.

Run the git clone Command

- ✓ ***git clone < repository SSH URL>***

Example : git clone git@github.com:pritykushwaha254/script.git



```
ubuntu@ip-172-31-4-67:~$ git clone git@github.com:pritykushwaha254/script.git
Cloning into 'script'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 33 (delta 13), reused 23 (delta 7), pack-reused 0 (from 0)
Receiving objects: 100% (33/33), 6.83 KiB | 3.42 MiB/s, done.
Resolving deltas: 100% (13/13), done.
ubuntu@ip-172-31-4-67:~$ ls
script
ubuntu@ip-172-31-4-67:~$ cd script/
ubuntu@ip-172-31-4-67:~/script$ ls
Error_handel  abc.tf                    gitkeep  make_folder.sh  prity1   prity2  prity4  prity6  prity8  tt
aaa           file_permission-check.sh  jj       new.sh          prity10  prity3  prity5  prity7  prity9
```

Now your private gitHub repository gets cloned in your local using SSH.

Is git init needed for git clone?

*Answer is No, **git init is NOT needed before using git clone** because:*

- ***git clone automatically initializes a Git repository** in the folder it creates.*
- *It also downloads all the files and commit history, so there's no need for an additional git init.*

we use git init when starting a new repository from scratch (not cloning). For example, if you want to create a fresh project and track it with Git, you would first run:git init.

git init   *For creating a new, empty repository locally.*

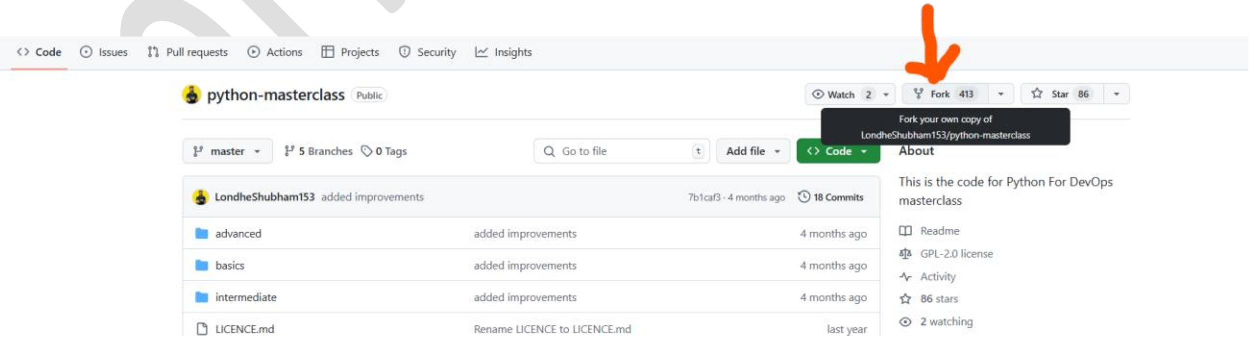
***git clone** For copying an existing repository (public or private).*

Git Fork

*A **fork** is like creating your own personal copy of someone else's GitHub repository. This copy is saved in **your GitHub account**, and you can make changes to it without affecting the original repository.*

Forking is useful when you want to contribute to an open-source project or create your custom version of a repository.

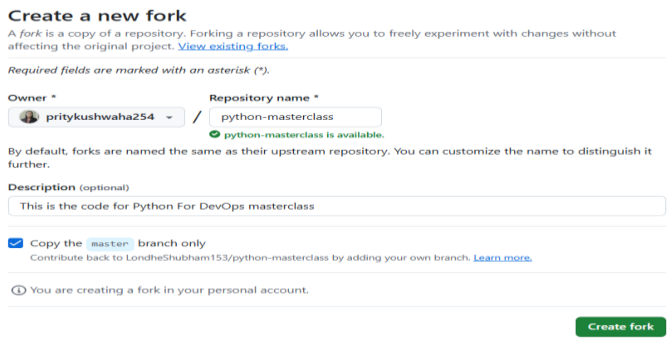
*Click the **Fork** button at the top right of the repository page on GitHub.*



Fork your own copy of LondheShubham153/python-masterclass

This is the code for Python For DevOps masterclass

LondheShubham153	added improvements	7b1caf3 · 4 months ago · 18 Commits
advanced	added improvements	4 months ago
basics	added improvements	4 months ago
intermediate	added improvements	4 months ago
LICENCE.md	Rename LICENCE to LICENCE.md	last year



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. View existing forks.

Required fields are marked with an asterisk (*).

Owner * — pritykushwaha254 / Repository name * — python-masterclass

python-masterclass is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

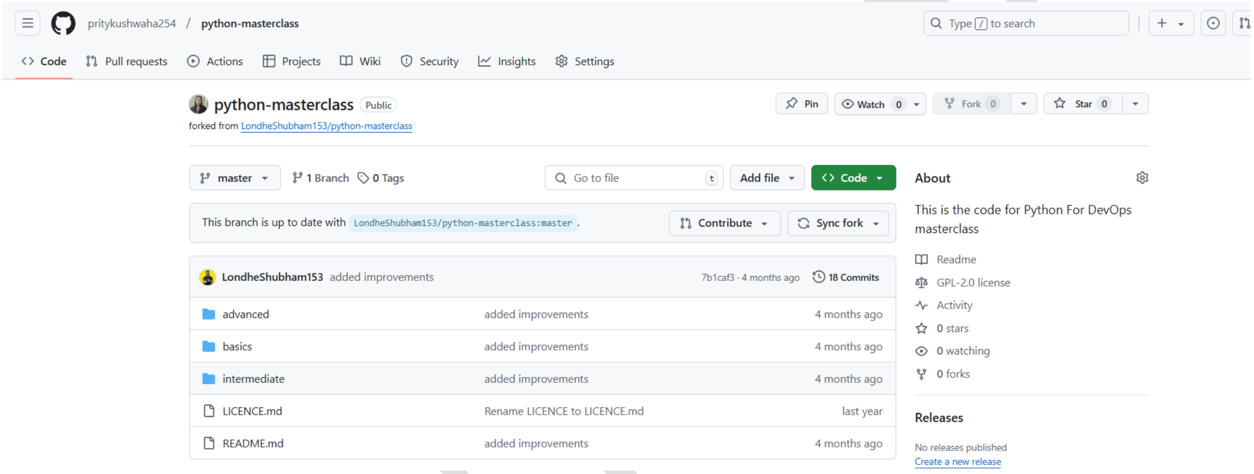
Description (optional): This is the code for Python For DevOps masterclass

☑ Copy the `master` branch only

Contribute back to LondheShubham153/python-masterclass by adding your own branch. Learn more.

ⓘ You are creating a fork in your personal account.

This creates a copy of the repository in your GitHub account.



python-masterclass Public

forked from LondheShubham153/python-masterclass

This branch is up to date with `LondheShubham153/python-masterclass:master`.

LondheShubham153 added improvements — 7b1caf3 · 4 months ago — 18 Commits

advanced	added improvements	4 months ago
basics	added improvements	4 months ago
intermediate	added improvements	4 months ago
LICENCE.md	Rename LICENCE to LICENCE.md	last year
README.md	added improvements	4 months ago

About

This is the code for Python For DevOps masterclass

Readme
GPL-2.0 license
Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release
