Module 2: Git Assignment - 1

Tasks To Be Performed:

1. Based on what you have learnt in the class, do the following steps:
   a. Create a new folder
   b. Put the following files in the folder
      - Code.txt
      - Log.txt
      - Output.txt

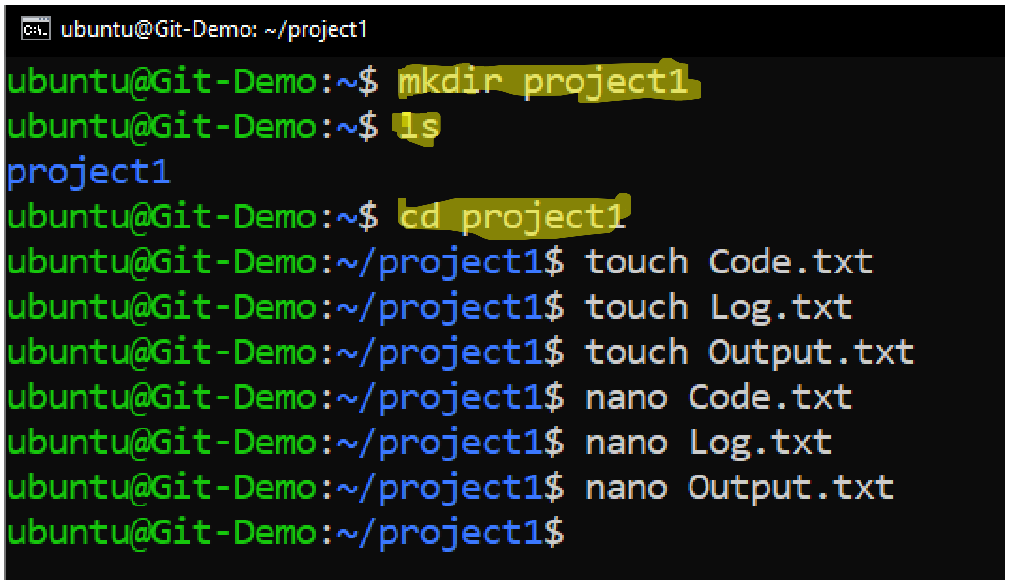
   c. Stage the Code.txt and Output.txt files
   d. Commit them
   e. And finally push them to GitHub

2. Please share the commands for the above points

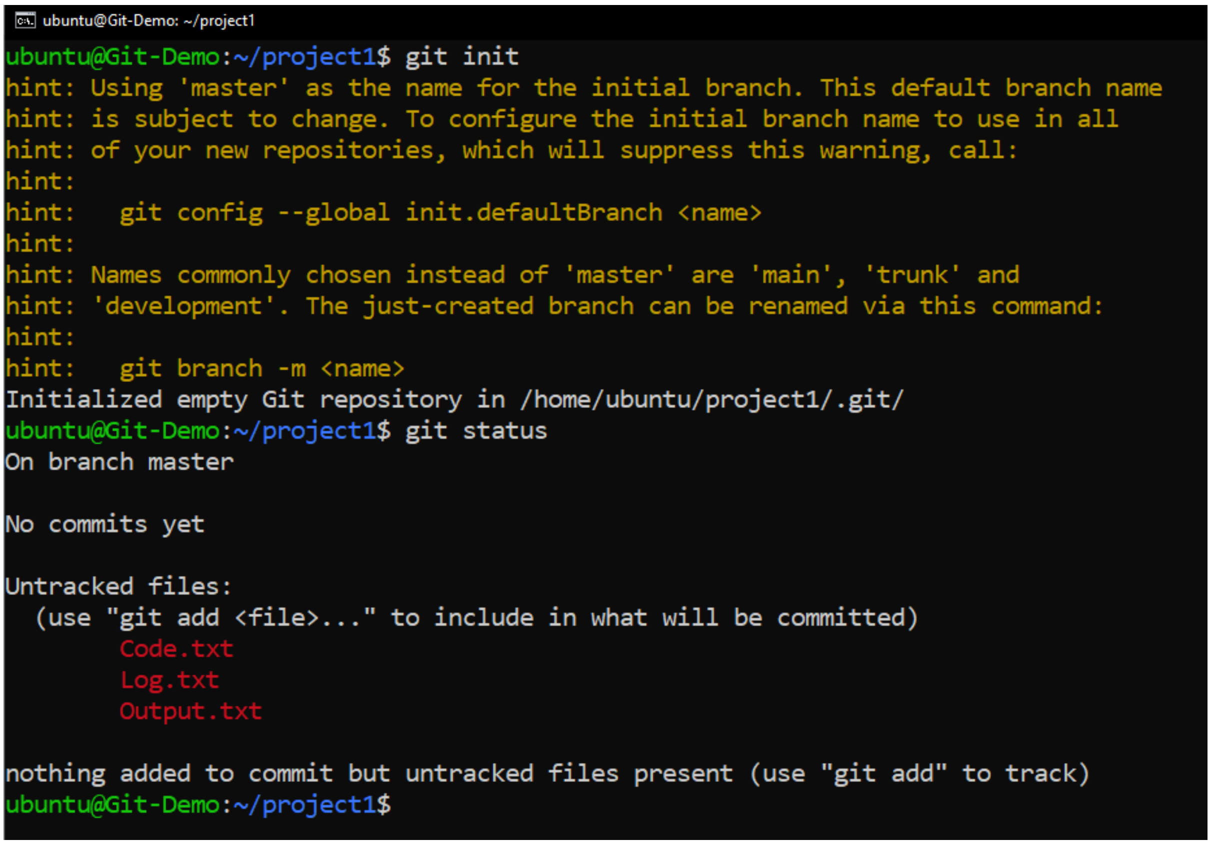
1. Go AWS > Launch Instances > SG > Allow SSH > Launch it > Connect SSH or Connect SSH client > I am using SSH client Copied that > and come to Local Maching Open CMD and Go Key Pair Location > and Paste that SSH Client Like this > ssh -i "Salman.pem" ubuntu@ec2-34-235-167-10.compute-1.amazonaws.com > After that it will connect

   Now if U want to change hostname then use the command > sudo hostnamectl set-hostname Git-Demo > logout > Again Connect it will change your Hostname ubuntu@Git-Demo

2. Now Follow the Commands and I will Type All Commands in the end Assignment

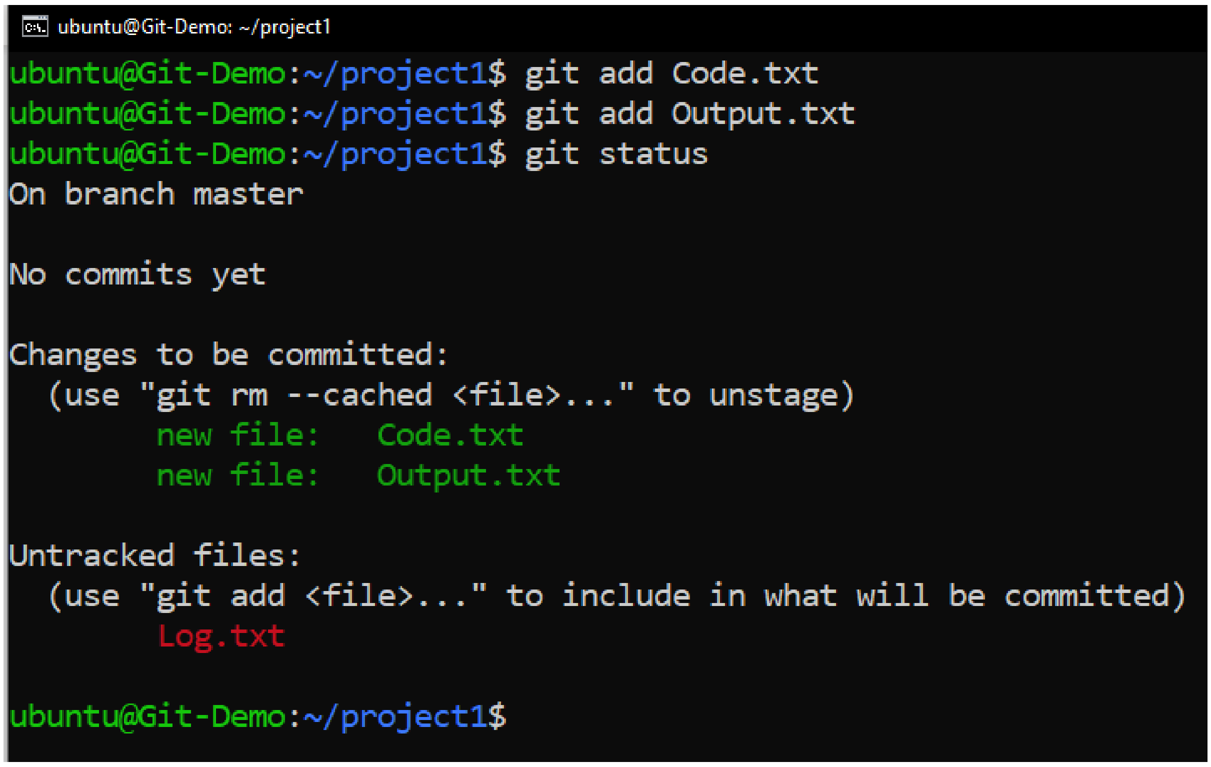


```
ubuntu@Git-Demo: ~/project1
ubuntu@Git-Demo:~$ mkdir project1
ubuntu@Git-Demo:~$ ls
project1
ubuntu@Git-Demo:~$ cd project1
ubuntu@Git-Demo:~/project1$ touch Code.txt
ubuntu@Git-Demo:~/project1$ touch Log.txt
ubuntu@Git-Demo:~/project1$ touch Output.txt
ubuntu@Git-Demo:~/project1$ nano Code.txt
ubuntu@Git-Demo:~/project1$ nano Log.txt
ubuntu@Git-Demo:~/project1$ nano Output.txt
ubuntu@Git-Demo:~/project1$
```



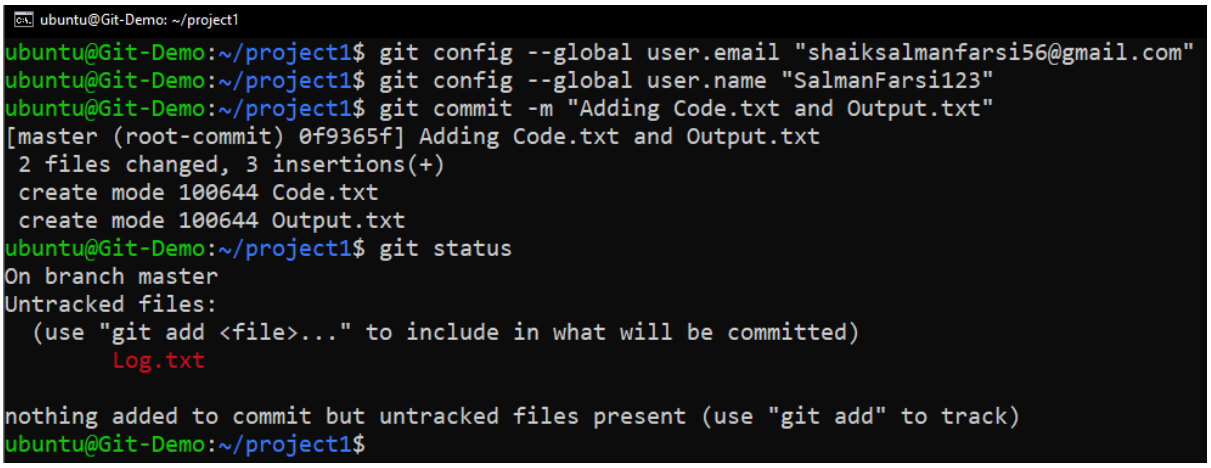
ubuntu@Git-Demo: ~/project1

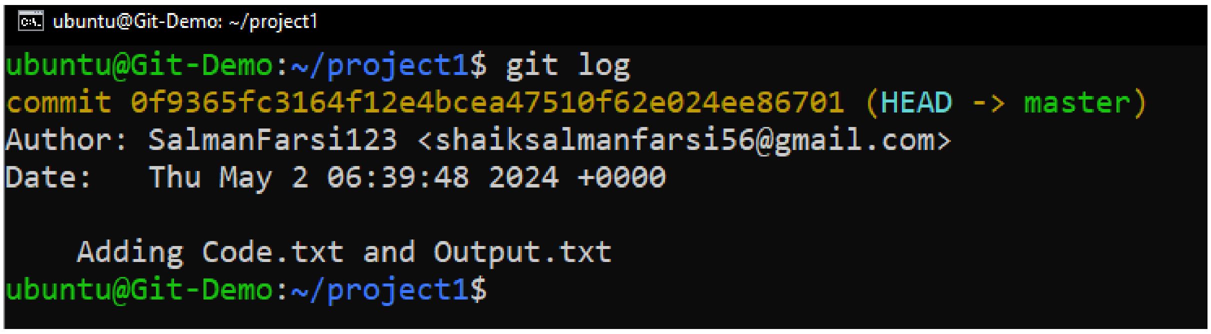
```
ubuntu@Git-Demo:~/project1$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global 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 /home/ubuntu/project1/.git/
ubuntu@Git-Demo:~/project1$ git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@Git-Demo:~/project1$
```

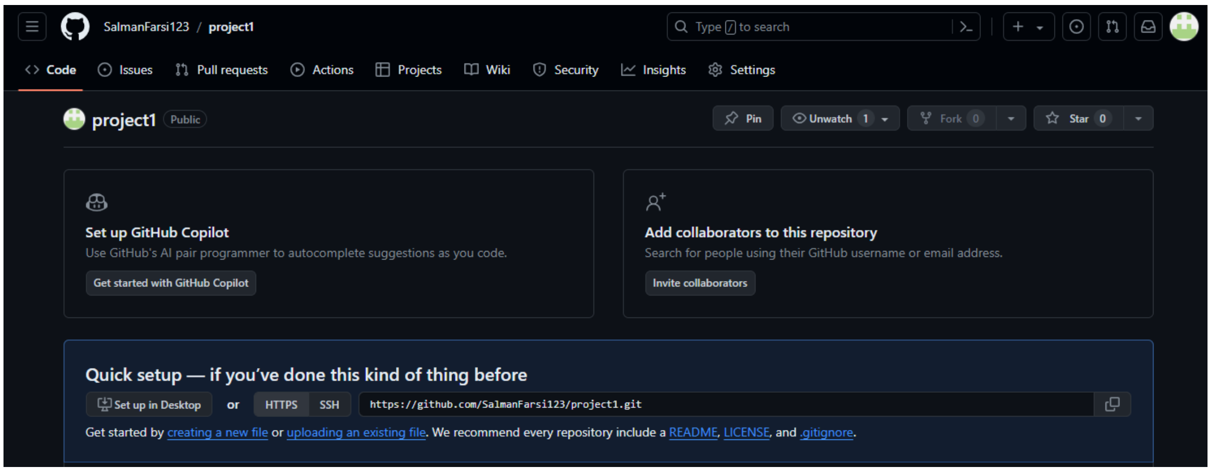


ubuntu@Git-Demo: ~/project1

```
ubuntu@Git-Demo:~/project1$ git add Code.txt
ubuntu@Git-Demo:~/project1$ git add Output.txt
ubuntu@Git-Demo:~/project1$ git status
On branch master

No commits yet

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

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

ubuntu@Git-Demo:~/project1$
```


```
ubuntu@Git-Demo: ~/project1
ubuntu@Git-Demo:~/project1$ git config --global user.email "shaiksalmanfarsi56@gmail.com"
ubuntu@Git-Demo:~/project1$ git config --global user.name "SalmanFarsi123"
ubuntu@Git-Demo:~/project1$ git commit -m "Adding Code.txt and Output.txt"
[master (root-commit) 0f9365f] Adding Code.txt and Output.txt
 2 files changed, 3 insertions(+)
 create mode 100644 Code.txt
 create mode 100644 Output.txt
ubuntu@Git-Demo:~/project1$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Log.txt

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@Git-Demo:~/project1$
```


```
ubuntu@Git-Demo: ~/project1
ubuntu@Git-Demo:~/project1$ git log
commit 0f9365fc3164f12e4bcea47510f62e024ee86701 (HEAD -> master)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 2 06:39:48 2024 +0000

    Adding Code.txt and Output.txt
ubuntu@Git-Demo:~/project1$
```



Set up GitHub Copilot

Use GitHub's AI pair programmer to autocomplete suggestions as you code.

Get started with GitHub Copilot

Add collaborators to this repository

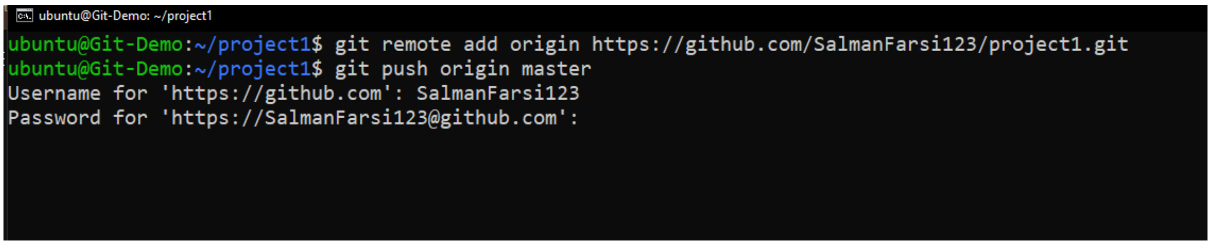
Search for people using their GitHub username or email address.

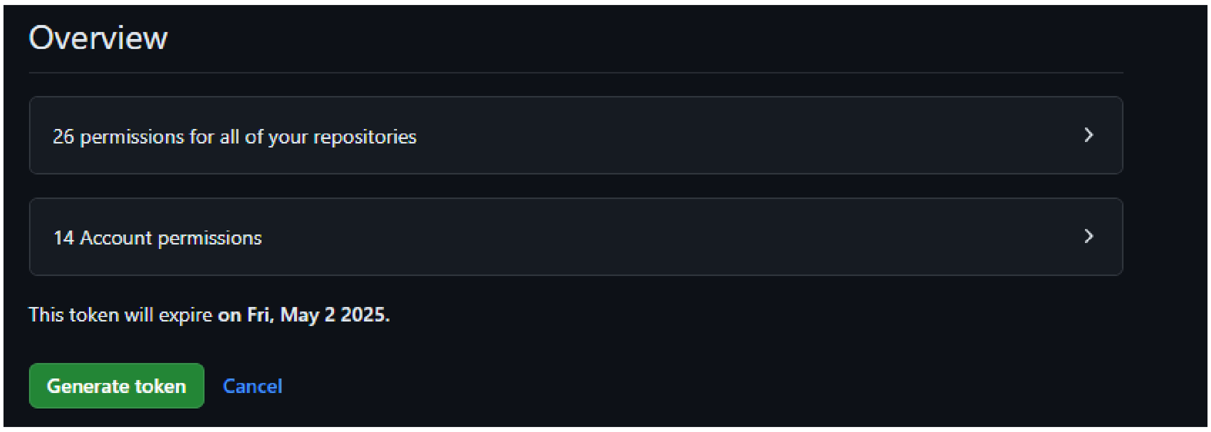
Invite collaborators

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/SalmanFarsi123/project1.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.


```
ubuntu@Git-Demo: ~/project1
ubuntu@Git-Demo:~/project1$ git remote add origin https://github.com/SalmanFarsi123/project1.git
ubuntu@Git-Demo:~/project1$ git push origin master
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
```

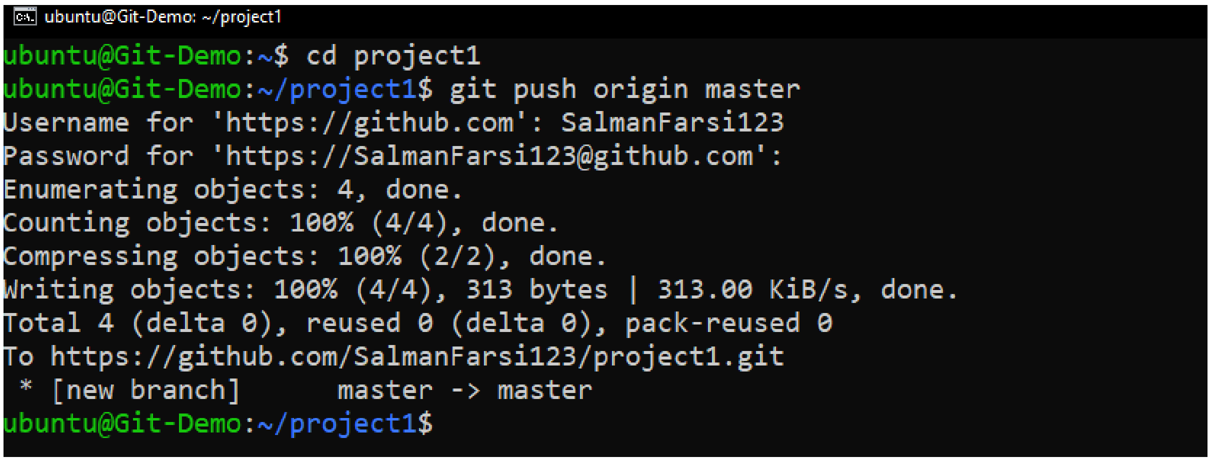


Overview

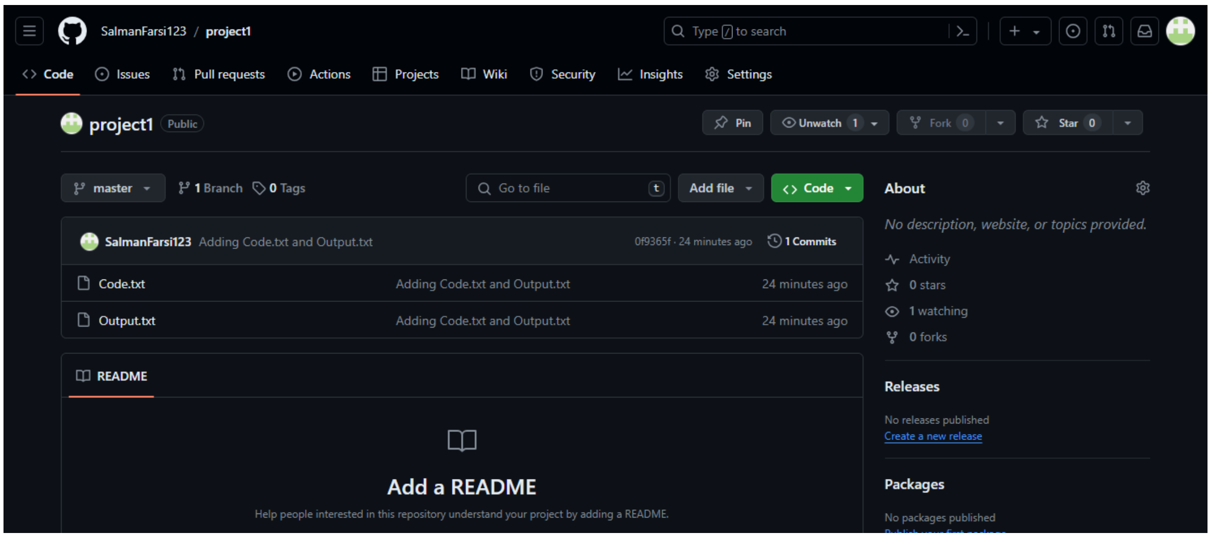
26 permissions for all of your repositories

14 Account permissions

This token will expire **on Fri, May 2 2025.**



```
ubuntu@Git-Demo: ~/project1
ubuntu@Git-Demo:~$ cd project1
ubuntu@Git-Demo:~/project1$ git push origin master
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 313 bytes | 313.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/SalmanFarsi123/project1.git
 * [new branch]      master -> master
ubuntu@Git-Demo:~/project1$
```



SalmanFarsi123 / project1

project1 Public

SalmanFarsi123 Adding Code.txt and Output.txt 0f9365f · 24 minutes ago 1 Commits

Code.txt	Adding Code.txt and Output.txt	24 minutes ago
Output.txt	Adding Code.txt and Output.txt	24 minutes ago

README

Add a README

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

About

No description, website, or topics provided.

Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

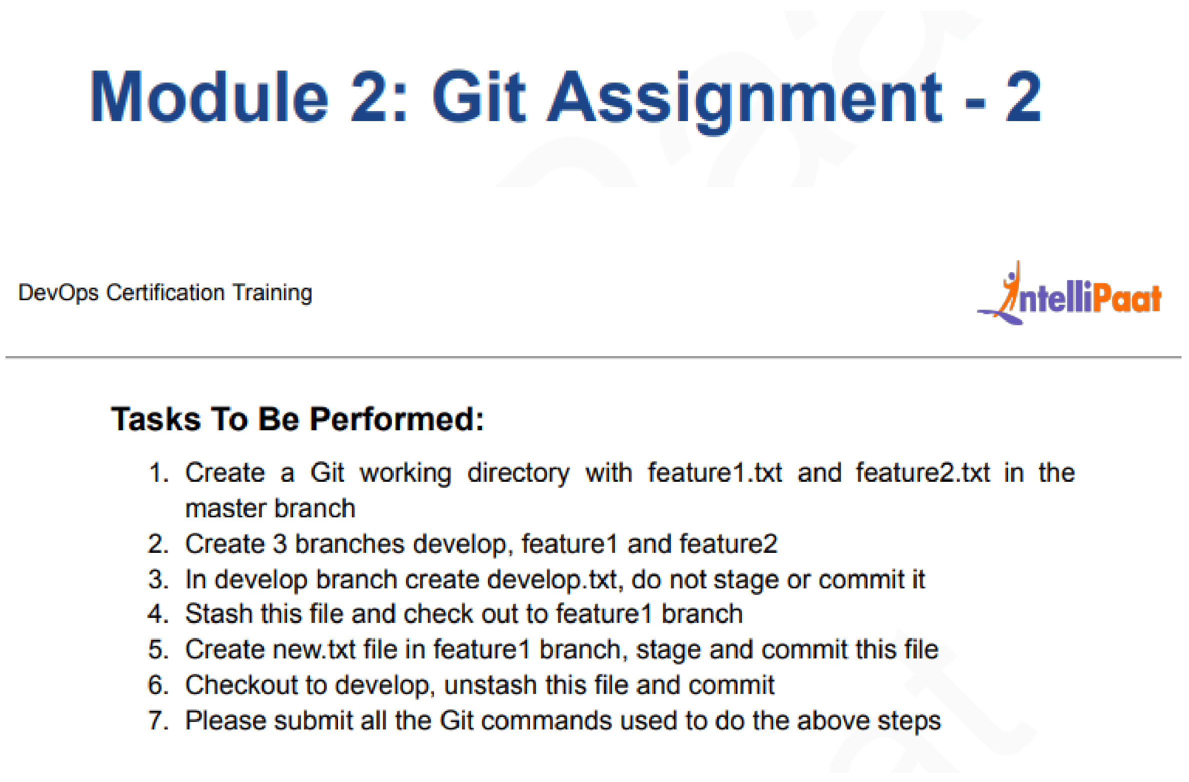
Packages

No packages published

Commands:

1. sudo hostnamectl set-hostname Git-Demo
2. logout
3. connect
4. mkdir project1
5. cd project1

6. touch Code.txt, touch Log.txt, touch Output.txt
7. nano Code.txt, Log.txt, Output.txt(Type within the editor like this code file, this log file, this Output)
8. git init(For initially we need to use this command)
9. git status(Now all this files in untracked)
10. git add Code.txt, git add Output.txt (Moving to Staging Area)
11. git status (to Check whether it is in Staging Area)
12. before commit it will ask the Credentials who are doing to move commit Area so Follow the below command
13. git config –global user.email "shaiksalmanfari56@gmail.com"
14. git config –global user.name "SalmanFarsi123"
15. git comit -m "Adding Code.txt and Output.txt"
16. git status (we can't able to see this 2 files so because its move on commit)
17. git log on there u can see the both files
18. Now U need to Create a GitHub Account(After Creation, Create a Repo In the Home)
19. After Creation of Repository Again Come to SSH Client(CMD)
20. git add remote origin https://github.com/SalmanFarsi123/project1.git
21. and again go to github right corner Click Profile > Settings > in the Last Bottom Click Developer Settings > Personal Access Tokens(PAT) > Fine-Generated Tokens > Click Generate Tokens > Allow 26 permisions and 14 Permisions and Click on Generate Token > Copy it and Save in Local Notepad because it will visible at only one time
22. Now Come to SSH and git push origin master it will Ask username and password
23. Username : SalmanFarsi123 and Password generate token paste it and check into the git hub will be push on there.



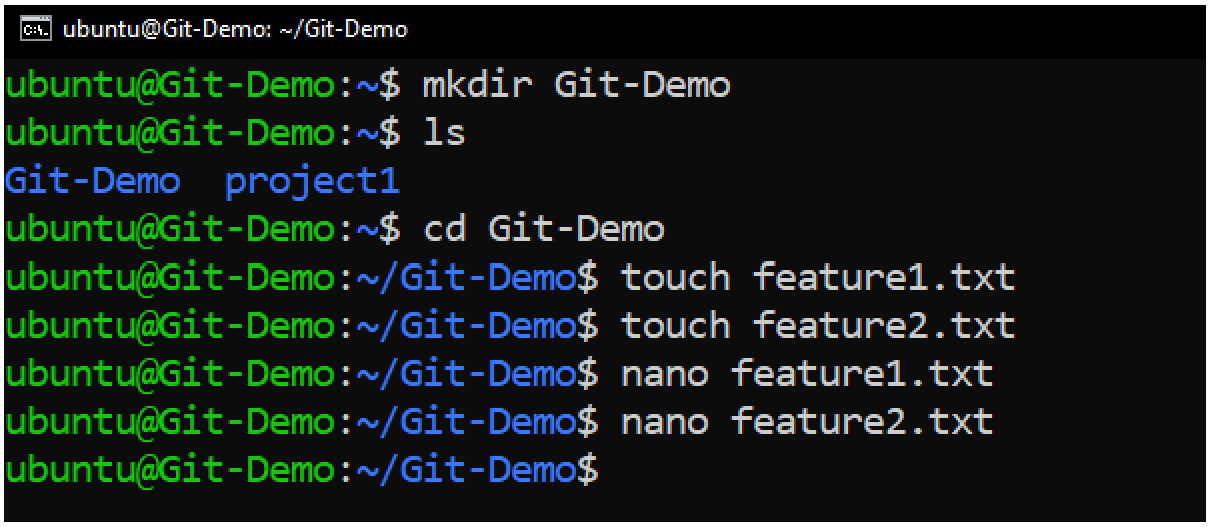
Module 2: Git Assignment - 2

DevOps Certification Training

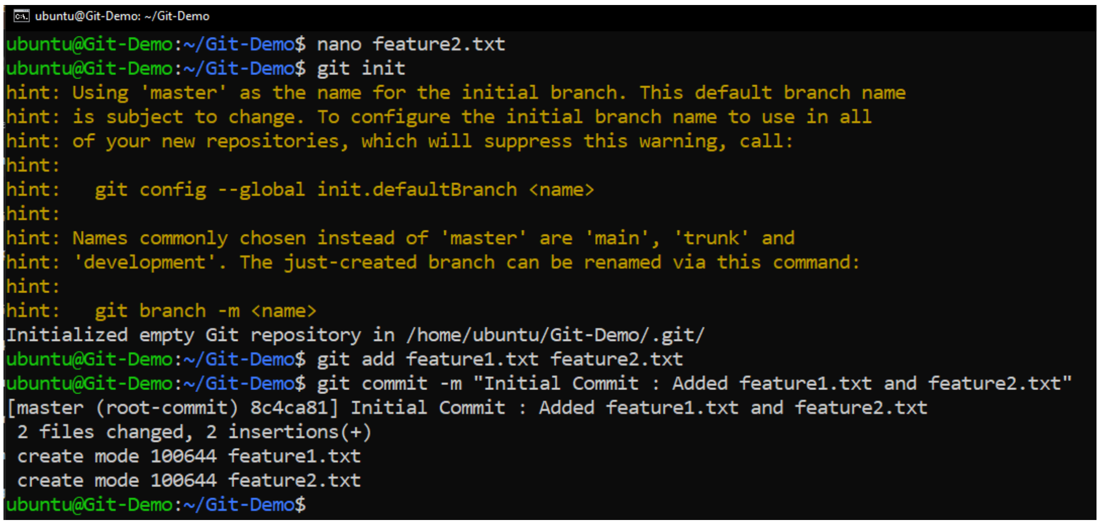
IntelliPaat

Tasks To Be Performed:

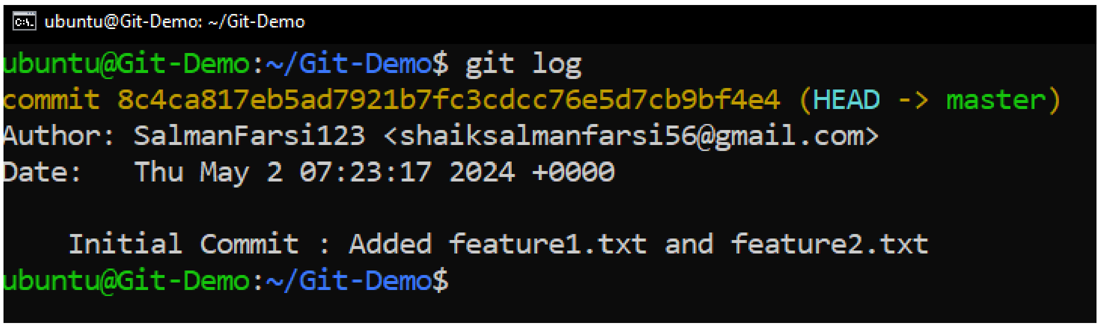
1. Create a Git working directory with feature1.txt and feature2.txt in the master branch
2. Create 3 branches develop, feature1 and feature2
3. In develop branch create develop.txt, do not stage or commit it
4. Stash this file and check out to feature1 branch
5. Create new.txt file in feature1 branch, stage and commit this file
6. Checkout to develop, unstash this file and commit
7. Please submit all the Git commands used to do the above steps



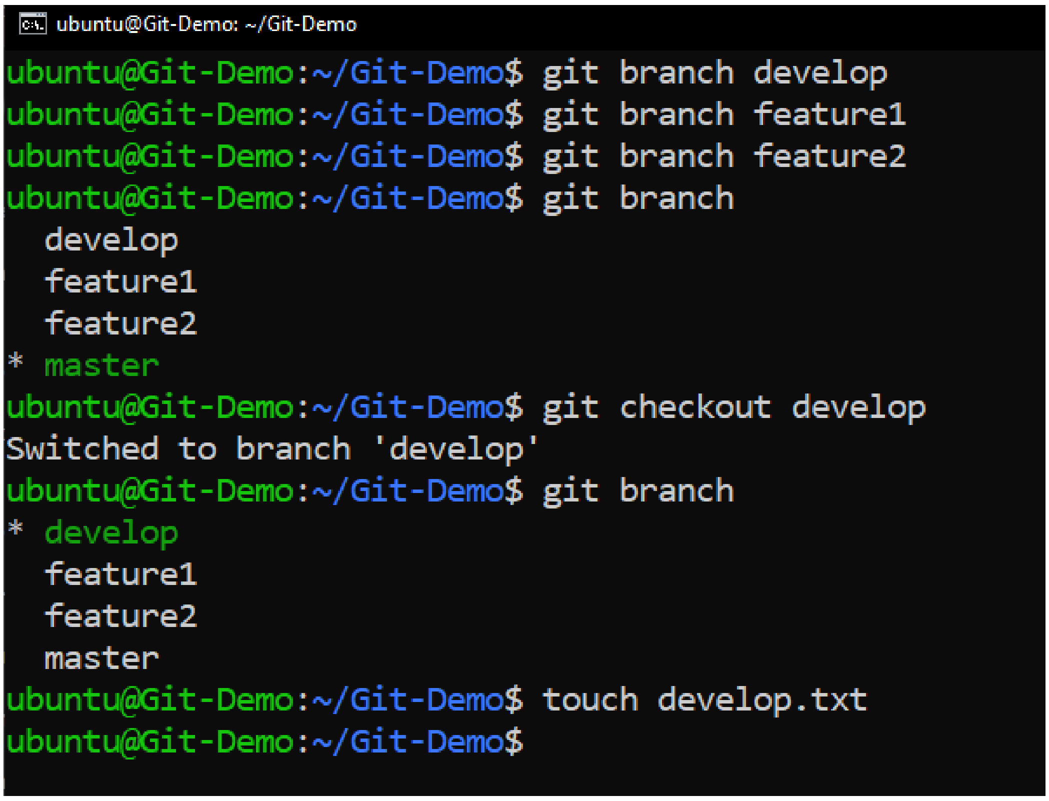
```
ubuntu@Git-Demo:~$ mkdir Git-Demo
ubuntu@Git-Demo:~$ ls
Git-Demo  project1
ubuntu@Git-Demo:~$ cd Git-Demo
ubuntu@Git-Demo:~/Git-Demo$ touch feature1.txt
ubuntu@Git-Demo:~/Git-Demo$ touch feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ nano feature1.txt
ubuntu@Git-Demo:~/Git-Demo$ nano feature2.txt
ubuntu@Git-Demo:~/Git-Demo$
```



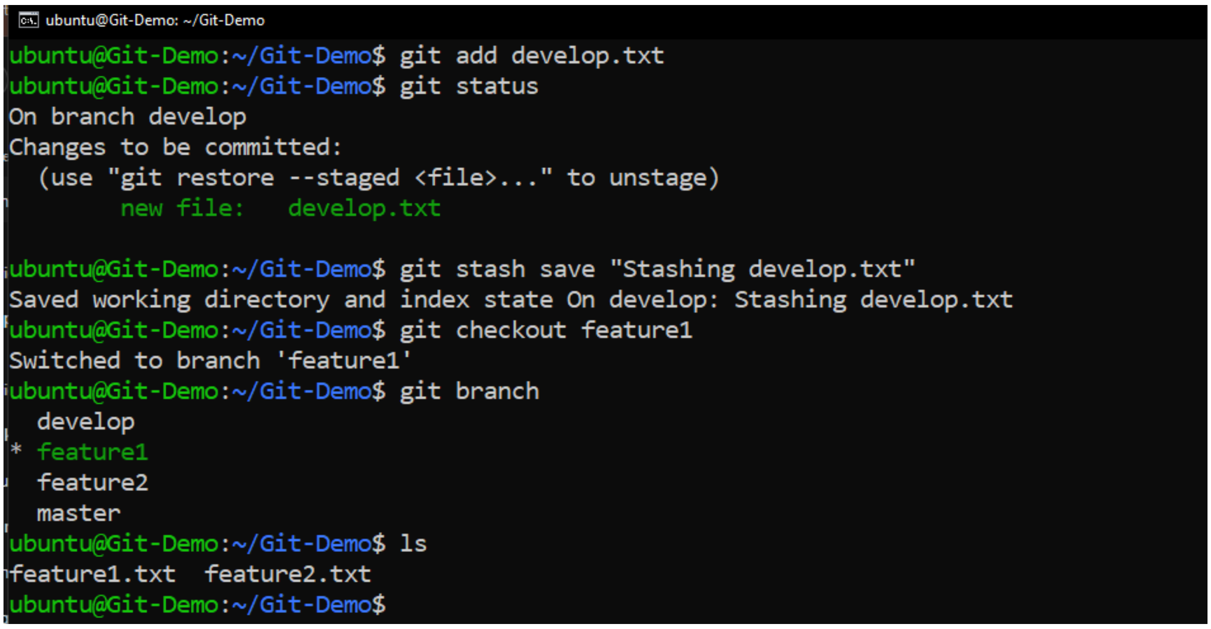
```
ubuntu@Git-Demo:~/Git-Demo$ nano feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this 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 /home/ubuntu/Git-Demo/.git/
ubuntu@Git-Demo:~/Git-Demo$ git add feature1.txt feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ git commit -m "Initial Commit : Added feature1.txt and feature2.txt"
[master (root-commit) 8c4ca81] Initial Commit : Added feature1.txt and feature2.txt
 2 files changed, 2 insertions(+)
 create mode 100644 feature1.txt
 create mode 100644 feature2.txt
ubuntu@Git-Demo:~/Git-Demo$
```



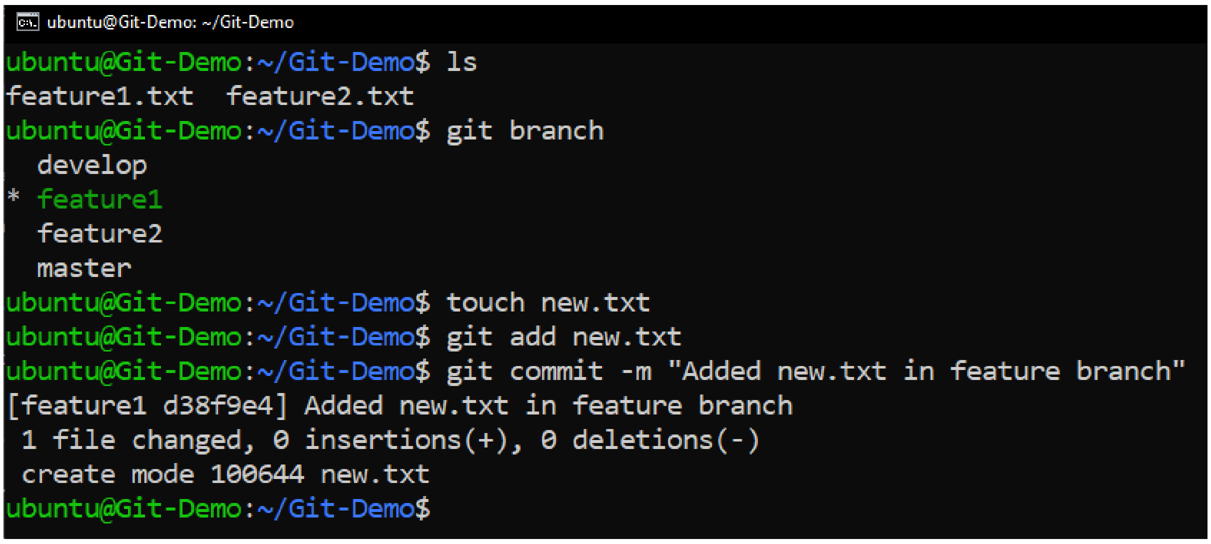
```
ubuntu@Git-Demo:~/Git-Demo$ git log
commit 8c4ca817eb5ad7921b7fc3cdcc76e5d7cb9bf4e4 (HEAD -> master)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 2 07:23:17 2024 +0000

    Initial Commit : Added feature1.txt and feature2.txt
ubuntu@Git-Demo:~/Git-Demo$
```



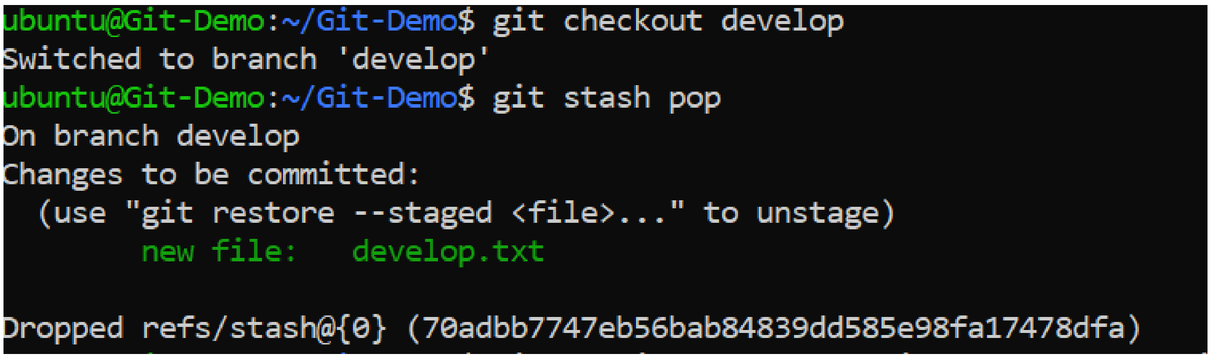
```
ubuntu@Git-Demo: ~/Git-Demo
ubuntu@Git-Demo:~/Git-Demo$ git branch develop
ubuntu@Git-Demo:~/Git-Demo$ git branch feature1
ubuntu@Git-Demo:~/Git-Demo$ git branch feature2
ubuntu@Git-Demo:~/Git-Demo$ git branch
  develop
  feature1
  feature2
* master
ubuntu@Git-Demo:~/Git-Demo$ git checkout develop
Switched to branch 'develop'
ubuntu@Git-Demo:~/Git-Demo$ git branch
* develop
  feature1
  feature2
  master
ubuntu@Git-Demo:~/Git-Demo$ touch develop.txt
ubuntu@Git-Demo:~/Git-Demo$
```



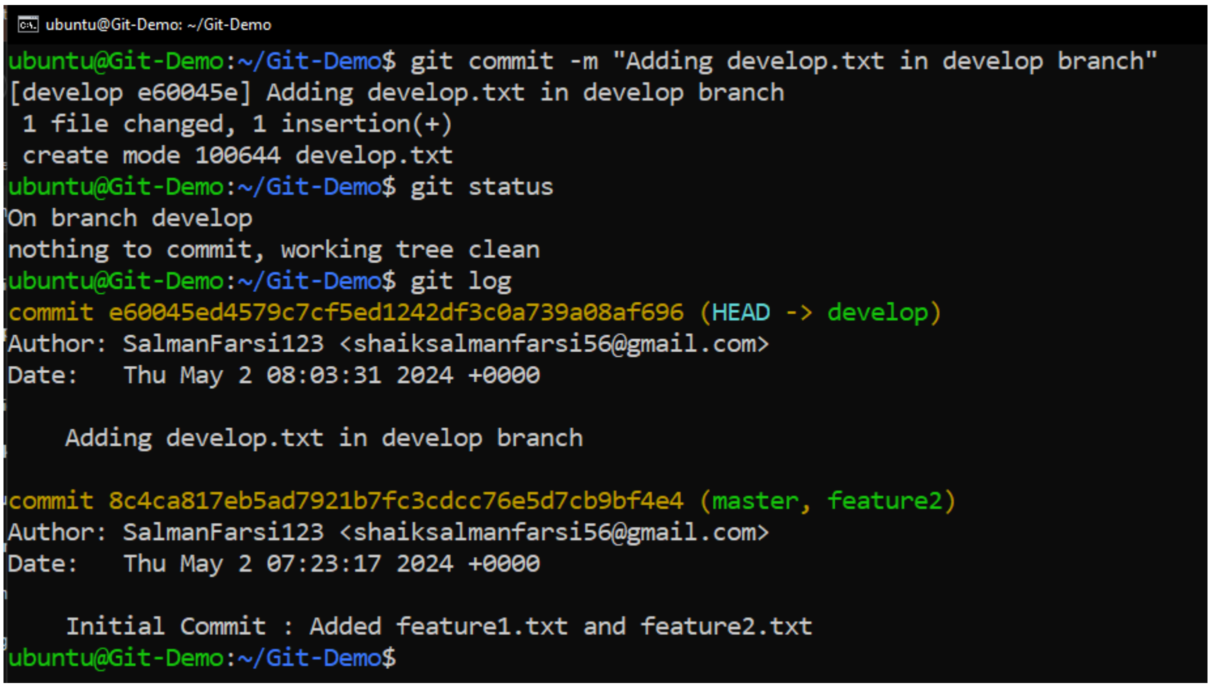
```
ubuntu@Git-Demo: ~/Git-Demo
ubuntu@Git-Demo:~/Git-Demo$ git add develop.txt
ubuntu@Git-Demo:~/Git-Demo$ git status
On branch develop
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   develop.txt

ubuntu@Git-Demo:~/Git-Demo$ git stash save "Stashing develop.txt"
Saved working directory and index state On develop: Stashing develop.txt
ubuntu@Git-Demo:~/Git-Demo$ git checkout feature1
Switched to branch 'feature1'
ubuntu@Git-Demo:~/Git-Demo$ git branch
  develop
* feature1
  feature2
  master
ubuntu@Git-Demo:~/Git-Demo$ ls
feature1.txt  feature2.txt
ubuntu@Git-Demo:~/Git-Demo$
```



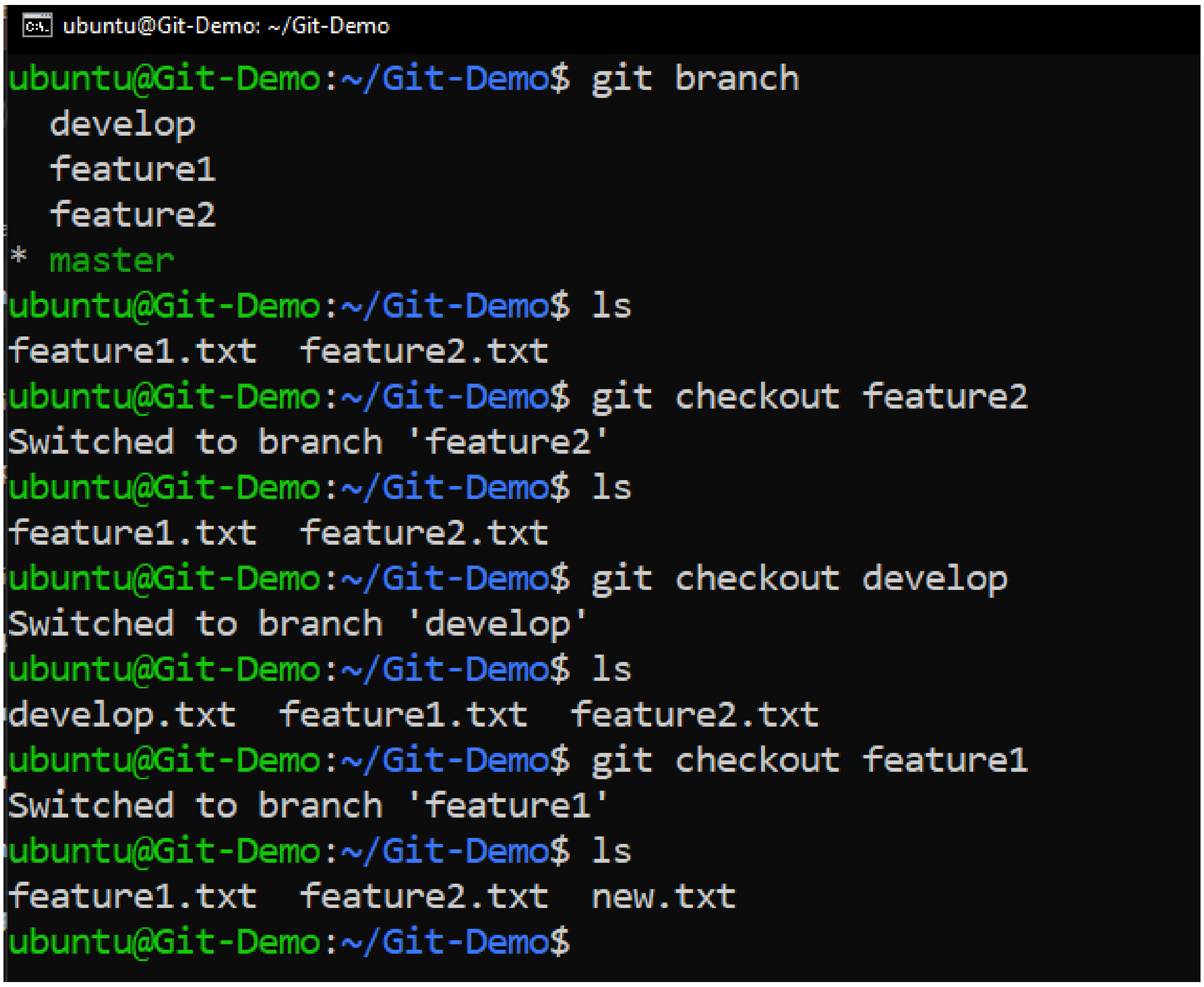
```
ubuntu@Git-Demo: ~/Git-Demo
ubuntu@Git-Demo:~/Git-Demo$ ls
feature1.txt  feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ git branch
  develop
* feature1
  feature2
  master
ubuntu@Git-Demo:~/Git-Demo$ touch new.txt
ubuntu@Git-Demo:~/Git-Demo$ git add new.txt
ubuntu@Git-Demo:~/Git-Demo$ git commit -m "Added new.txt in feature branch"
[feature1 d38f9e4] Added new.txt in feature branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 new.txt
ubuntu@Git-Demo:~/Git-Demo$
```



```
ubuntu@Git-Demo:~/Git-Demo$ git checkout develop
Switched to branch 'develop'
ubuntu@Git-Demo:~/Git-Demo$ git stash pop
On branch develop
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   develop.txt

Dropped refs/stash@{0} (70adbb7747eb56bab84839dd585e98fa17478dfa)
```



```
ubuntu@Git-Demo: ~/Git-Demo
ubuntu@Git-Demo:~/Git-Demo$ git commit -m "Adding develop.txt in develop branch"
[develop e60045e] Adding develop.txt in develop branch
 1 file changed, 1 insertion(+)
 create mode 100644 develop.txt
ubuntu@Git-Demo:~/Git-Demo$ git status
On branch develop
nothing to commit, working tree clean
ubuntu@Git-Demo:~/Git-Demo$ git log
commit e60045ed4579c7cf5ed1242df3c0a739a08af696 (HEAD -> develop)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 2 08:03:31 2024 +0000

    Adding develop.txt in develop branch

commit 8c4ca817eb5ad7921b7fc3cdcc76e5d7cb9bf4e4 (master, feature2)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 2 07:23:17 2024 +0000

    Initial Commit : Added feature1.txt and feature2.txt
ubuntu@Git-Demo:~/Git-Demo$
```



```
ubuntu@Git-Demo: ~/Git-Demo
ubuntu@Git-Demo:~/Git-Demo$ git branch
  develop
  feature1
  feature2
* master
ubuntu@Git-Demo:~/Git-Demo$ ls
feature1.txt  feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ git checkout feature2
Switched to branch 'feature2'
ubuntu@Git-Demo:~/Git-Demo$ ls
feature1.txt  feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ git checkout develop
Switched to branch 'develop'
ubuntu@Git-Demo:~/Git-Demo$ ls
develop.txt  feature1.txt  feature2.txt
ubuntu@Git-Demo:~/Git-Demo$ git checkout feature1
Switched to branch 'feature1'
ubuntu@Git-Demo:~/Git-Demo$ ls
feature1.txt  feature2.txt  new.txt
ubuntu@Git-Demo:~/Git-Demo$
```

Followed the Commands Step to Step

1. mkdir git-demo
2. cd git-demo
3. touch feature1.txt
4. touch feature2.txt
5. git init
6. git add feature1.txt feature2.txt
7. git commit -m "Initial commit: Added feature1.txt and feature2.txt"
8. git branch develop
9. git branch feature1
10. git branch feature2
11. git checkout develop
12. touch develop.txt
13. git add develop.txt
14. git stash save "Stashing develop.txt"
15. git checkout feature1
16. touch new.txt
17. git add new.txt

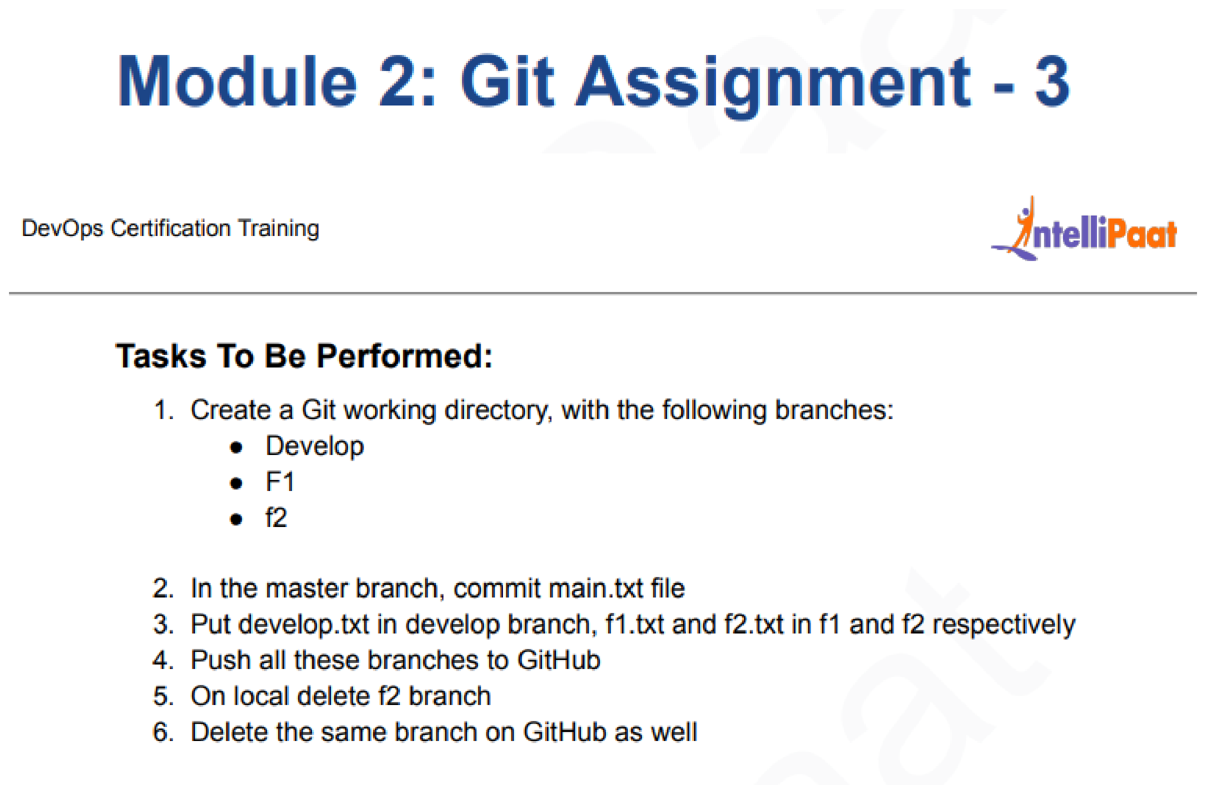
18. git commit -m "Added new.txt in feature1 branch"
19. git checkout develop
20. git stash pop
21. git commit -m "Added develop.txt in develop branch"

Concept:

when you initially commit files to the master branch, those commits are specific to the master branch. However, when you create a new branch from the master branch, the new branch starts with the same commits and files as the master branch at that point in time.

So, if you create a new branch after the initial commit on the master branch, that new branch will have all the files and commits present in the master branch at that time.

If you commit new changes or stash changes on the master branch after creating new branches, those changes won't automatically appear in the new branches. Each branch maintains its own separate history and changes. So, any new commits or stashed changes made after creating a branch will only be present in the branch where they were made.



Module 2: Git Assignment - 3

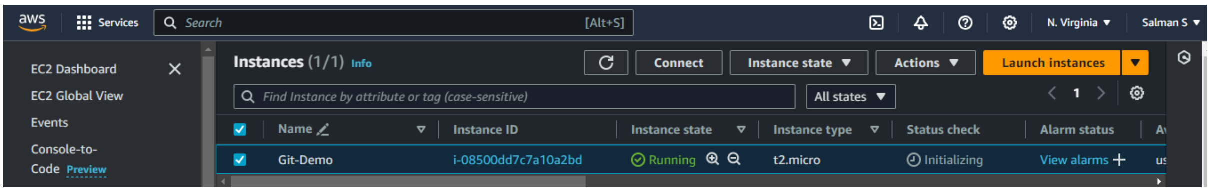
DevOps Certification Training

IntelliPaat

Tasks To Be Performed:

1. Create a Git working directory, with the following branches:
   - Develop
   - F1
   - f2
2. In the master branch, commit main.txt file
3. Put develop.txt in develop branch, f1.txt and f2.txt in f1 and f2 respectively
4. Push all these branches to GitHub
5. On local delete f2 branch
6. Delete the same branch on GitHub as well

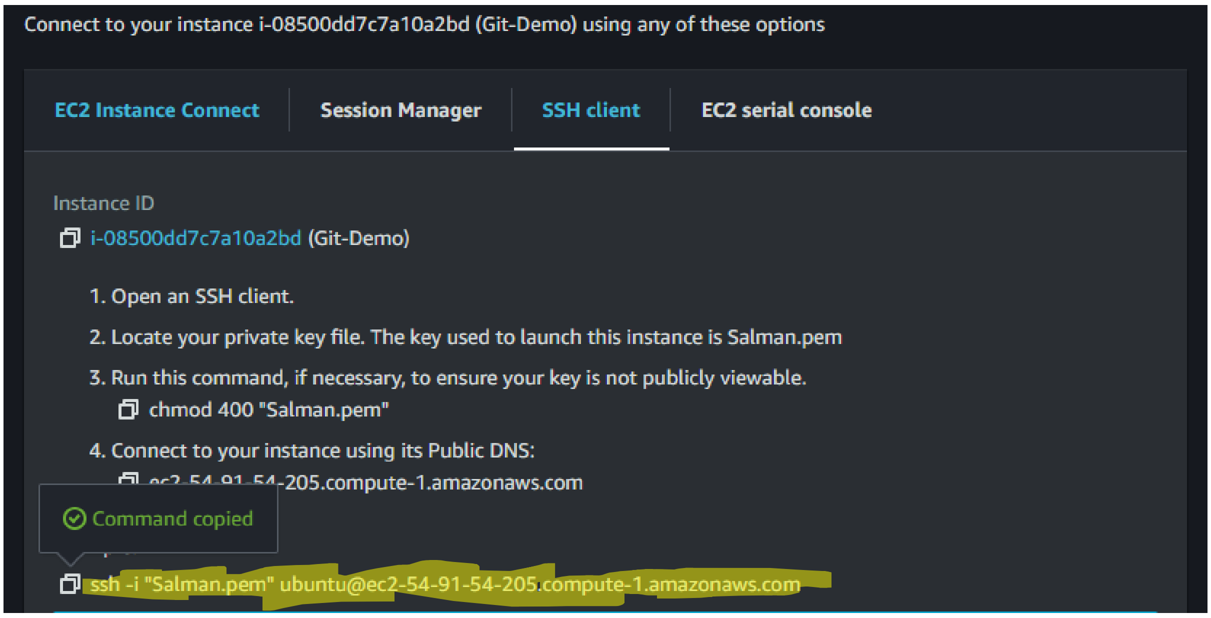
1. Launched Instances



Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
Git-Demo	i-08500dd7c7a10a2bd	Running	t2.micro	Initializing	View alarms

2. Connecting Through with ssh client



Connect to your instance i-08500dd7c7a10a2bd (Git-Demo) using any of these options

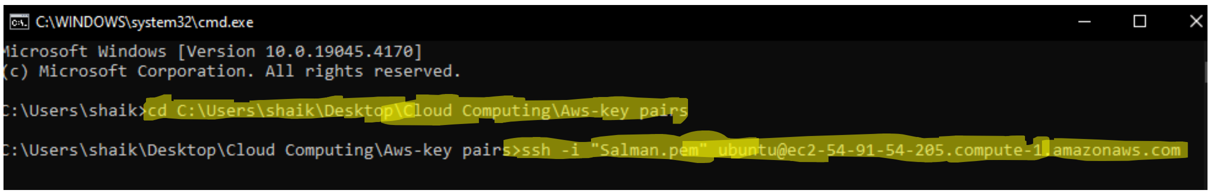
Instance ID

i-08500dd7c7a10a2bd (Git-Demo)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is Salman.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "Salman.pem"
4. Connect to your instance using its Public DNS:
   ec2-54-91-54-205.compute-1.amazonaws.com

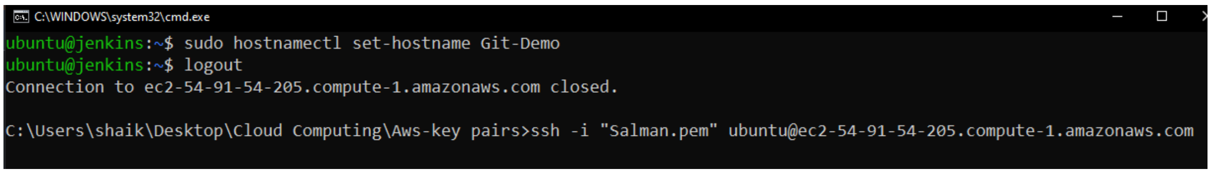
ssh -i "Salman.pem" ubuntu@ec2-54-91-54-205.compute-1.amazonaws.com

3. Change directory into your key location and paste the SSH Client Connect



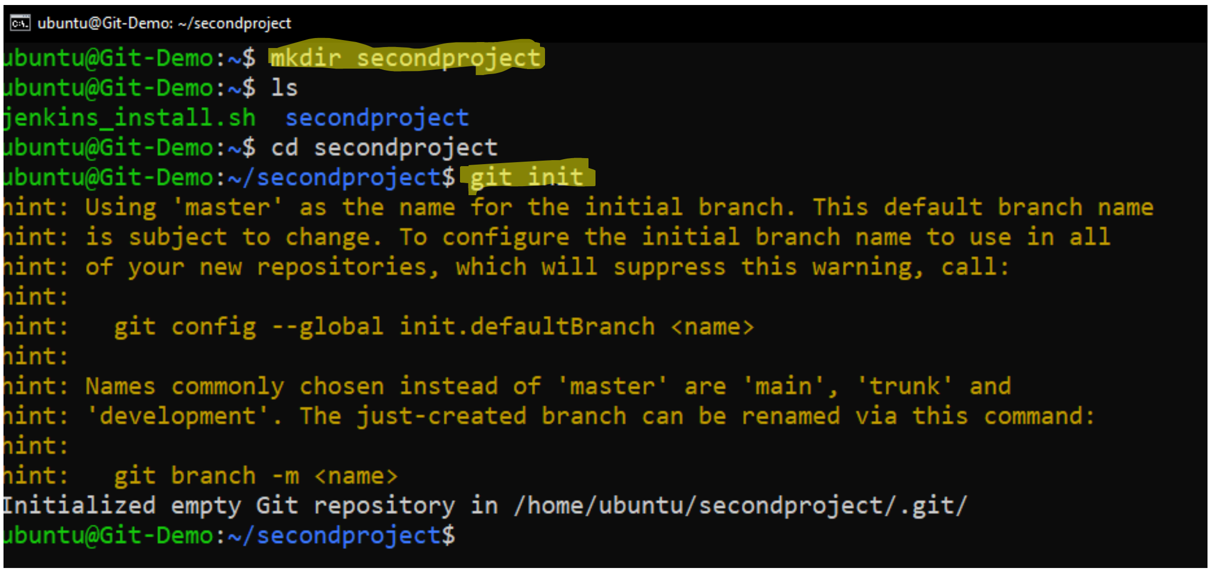
```
icrosoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

:\Users\shaik>cd C:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs

:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs>ssh -i "Salman.pem" ubuntu@ec2-54-91-54-205.compute-1.amazonaws.com
```

4. If u want to change Hostname run this command sudo hostnamectl set-hostname Git-Demo
   Logout and paste SSH Client Again and it will connect and your hostname will be changed



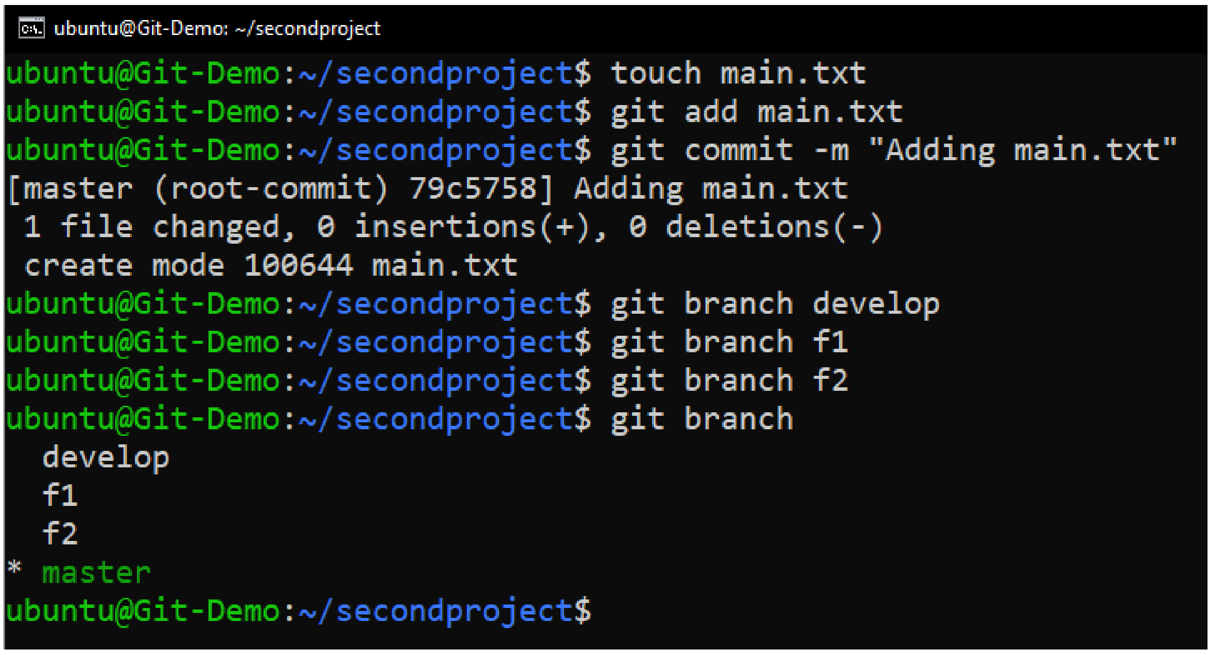
```
ubuntu@jenkins:~$ sudo hostnamectl set-hostname Git-Demo
ubuntu@jenkins:~$ logout
Connection to ec2-54-91-54-205.compute-1.amazonaws.com closed.

C:\Users\shaik\Desktop\Cloud Computing\Aws-key pairs>ssh -i "Salman.pem" ubuntu@ec2-54-91-54-205.compute-1.amazonaws.com
```

5. Created Directory and came to inside the directory initializing git(git init)



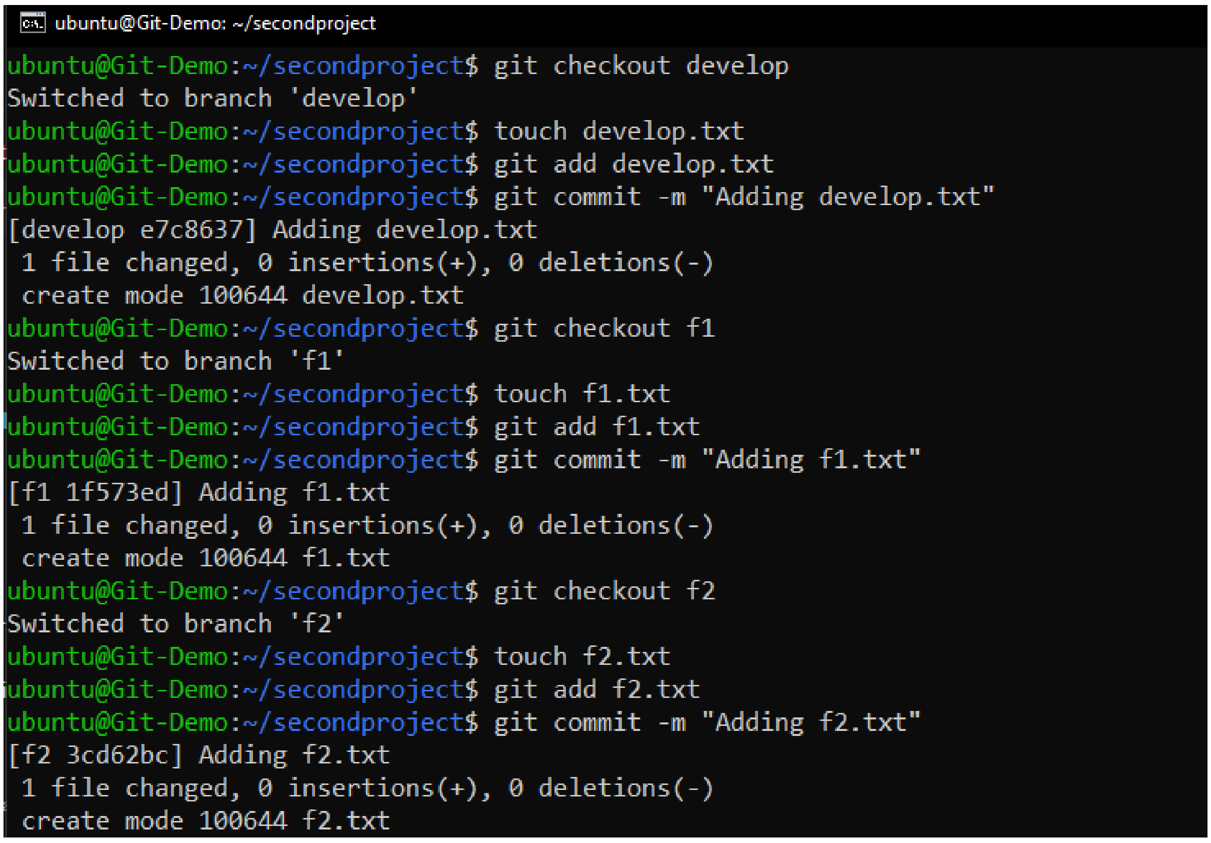
```
ubuntu@Git-Demo:~$ mkdir secondproject
ubuntu@Git-Demo:~$ ls
jenkins_install.sh  secondproject
ubuntu@Git-Demo:~$ cd secondproject
ubuntu@Git-Demo:~/secondproject$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the 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 /home/ubuntu/secondproject/.git/
ubuntu@Git-Demo:~/secondproject$
```

6. Create text file main.txt and add this file in staging and commit it and create git branch named as develop, f1, f2, now see git branch were we are in actually



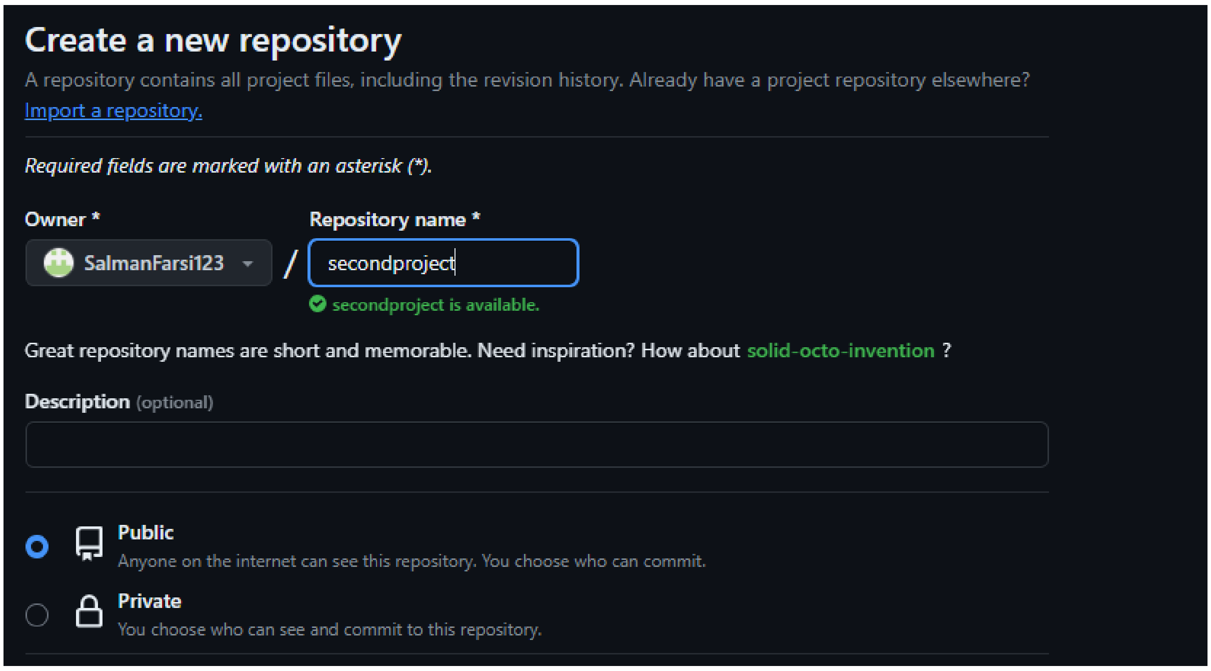
```
ubuntu@Git-Demo: ~/secondproject
ubuntu@Git-Demo:~/secondproject$ touch main.txt
ubuntu@Git-Demo:~/secondproject$ git add main.txt
ubuntu@Git-Demo:~/secondproject$ git commit -m "Adding main.txt"
[master (root-commit) 79c5758] Adding main.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 main.txt
ubuntu@Git-Demo:~/secondproject$ git branch develop
ubuntu@Git-Demo:~/secondproject$ git branch f1
ubuntu@Git-Demo:~/secondproject$ git branch f2
ubuntu@Git-Demo:~/secondproject$ git branch
  develop
  f1
  f2
* master
ubuntu@Git-Demo:~/secondproject$
```

7. git checkout develop it will switched to develop create a develop.txt file and add develop.txt in staging and commit it and respectively follow to f1 and f2



```
ubuntu@Git-Demo: ~/secondproject
ubuntu@Git-Demo:~/secondproject$ git checkout develop
Switched to branch 'develop'
ubuntu@Git-Demo:~/secondproject$ touch develop.txt
ubuntu@Git-Demo:~/secondproject$ git add develop.txt
ubuntu@Git-Demo:~/secondproject$ git commit -m "Adding develop.txt"
[develop e7c8637] Adding develop.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 develop.txt
ubuntu@Git-Demo:~/secondproject$ git checkout f1
Switched to branch 'f1'
ubuntu@Git-Demo:~/secondproject$ touch f1.txt
ubuntu@Git-Demo:~/secondproject$ git add f1.txt
ubuntu@Git-Demo:~/secondproject$ git commit -m "Adding f1.txt"
[f1 1f573ed] Adding f1.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 f1.txt
ubuntu@Git-Demo:~/secondproject$ git checkout f2
Switched to branch 'f2'
ubuntu@Git-Demo:~/secondproject$ touch f2.txt
ubuntu@Git-Demo:~/secondproject$ git add f2.txt
ubuntu@Git-Demo:~/secondproject$ git commit -m "Adding f2.txt"
[f2 3cd62bc] Adding f2.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 f2.txt
```

8. Create Repository and I keep it as public



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 * SalmanFarsi123 / Repository name * secondproject

secondproject is available.

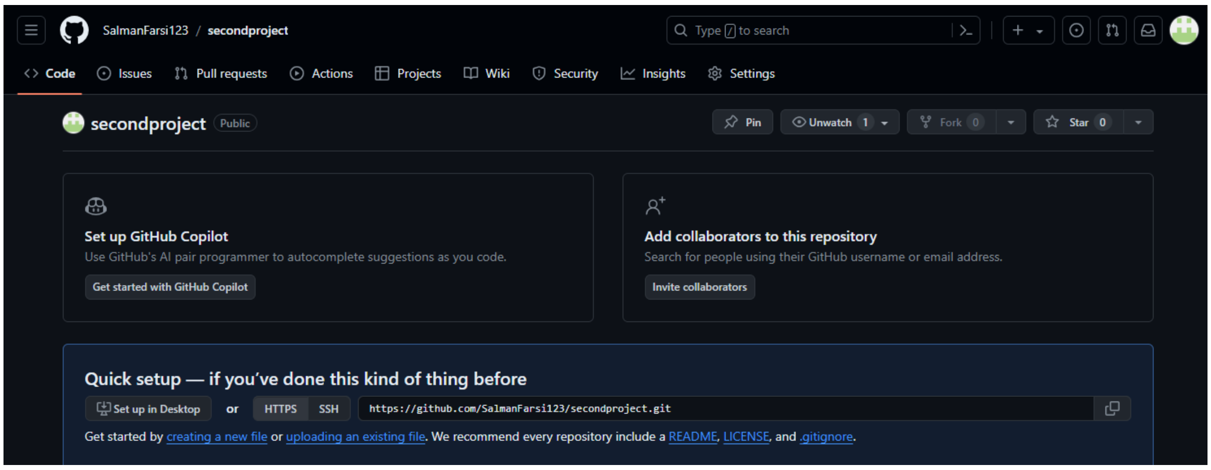
Great repository names are short and memorable. Need inspiration? How about solid-octo-invention ?

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.

9. Copy the link



Set up GitHub Copilot

Use GitHub's AI pair programmer to autocomplete suggestions as you code.

Add collaborators to this repository

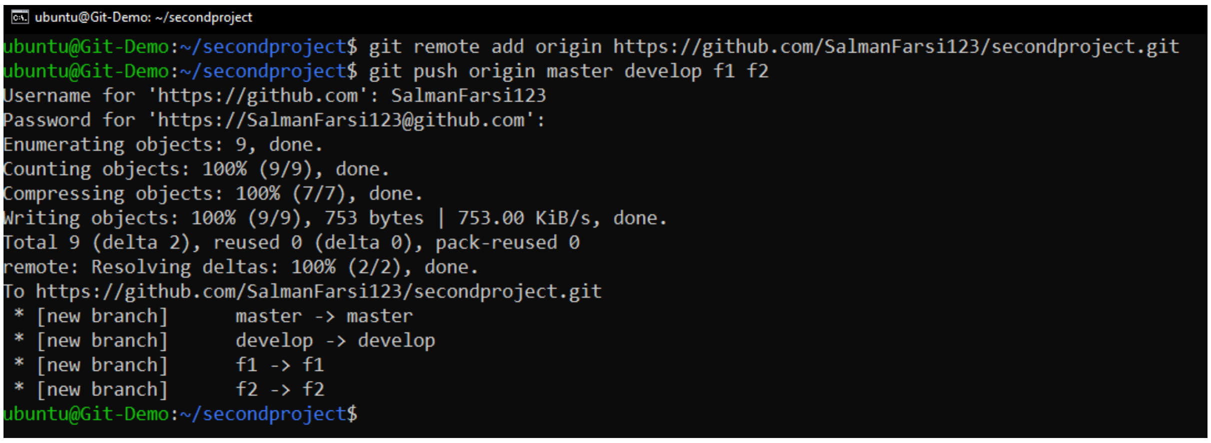
Search for people using their GitHub username or email address.

Quick setup — if you've done this kind of thing before

```
https://github.com/SalmanFarsi123/secondproject.git
```

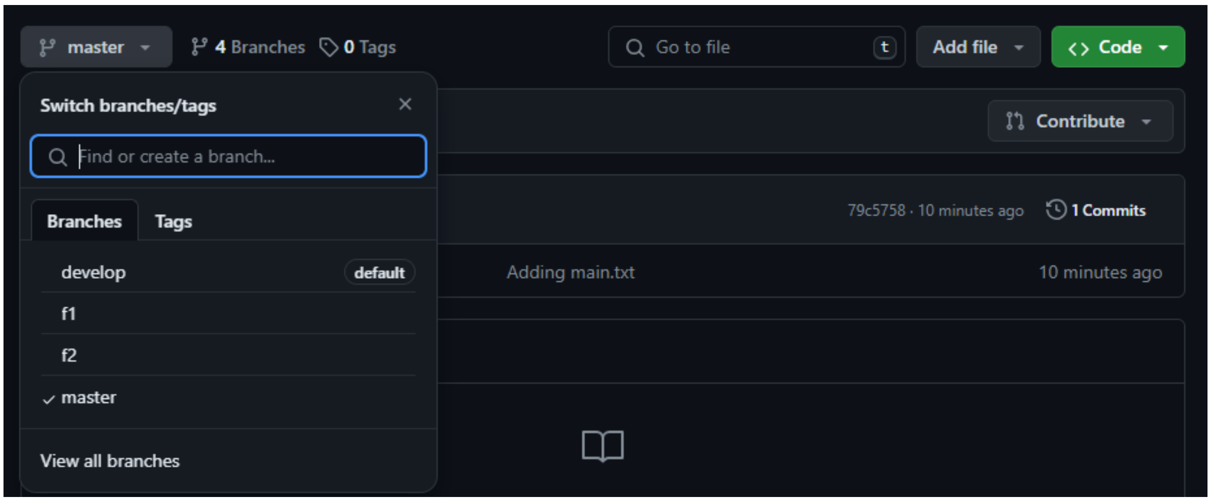
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

10. Run the command git remove add origin and paste it github location link and next push the branches to remote repo and it will ask user name and password and here Token is password which we previously created and after we successfully pushed to git repo.

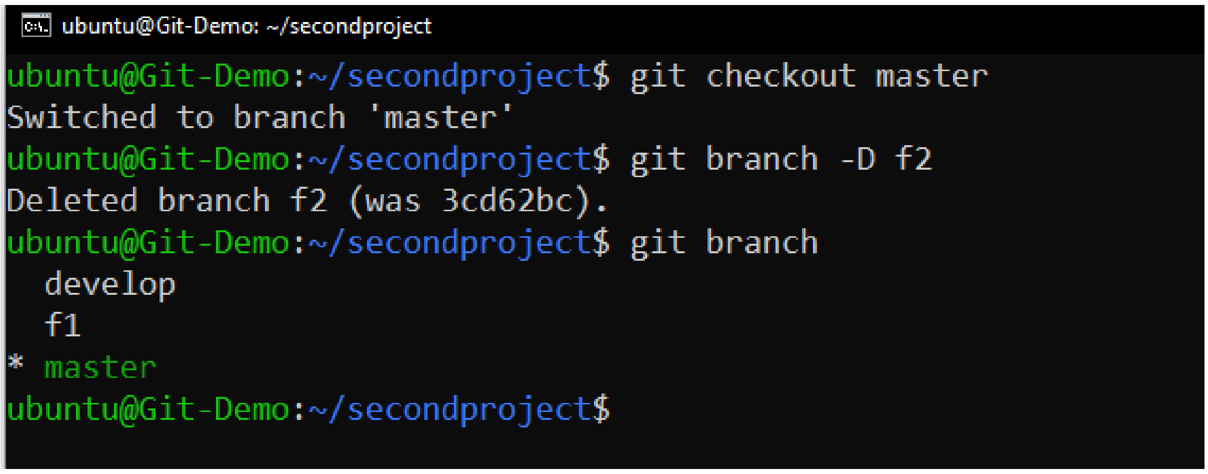


```
ubuntu@Git-Demo: ~/secondproject
ubuntu@Git-Demo:~/secondproject$ git remote add origin https://github.com/SalmanFarsi123/secondproject.git
ubuntu@Git-Demo:~/secondproject$ git push origin master develop f1 f2
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 753 bytes | 753.00 KiB/s, done.
Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/SalmanFarsi123/secondproject.git
 * [new branch]      master -> master
 * [new branch]      develop -> develop
 * [new branch]      f1 -> f1
 * [new branch]      f2 -> f2
ubuntu@Git-Demo:~/secondproject$
```

11. We can see the branches in Github repo

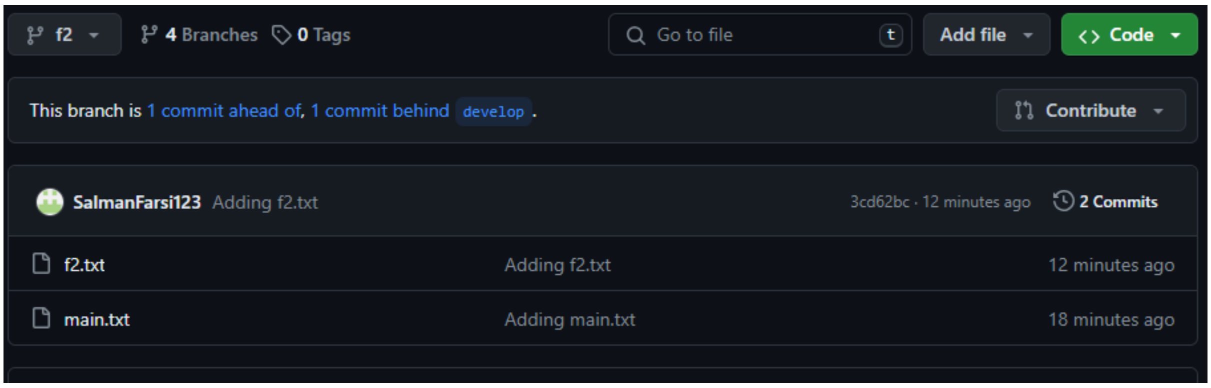


12. Now Need to Delete in Local run the command git branch -D f2 it will deleted successfully



```
ubuntu@Git-Demo: ~/secondproject
ubuntu@Git-Demo:~/secondproject$ git checkout master
Switched to branch 'master'
ubuntu@Git-Demo:~/secondproject$ git branch -D f2
Deleted branch f2 (was 3cd62bc).
ubuntu@Git-Demo:~/secondproject$ git branch
  develop
  f1
* master
ubuntu@Git-Demo:~/secondproject$
```

13. But in the remote repo its available

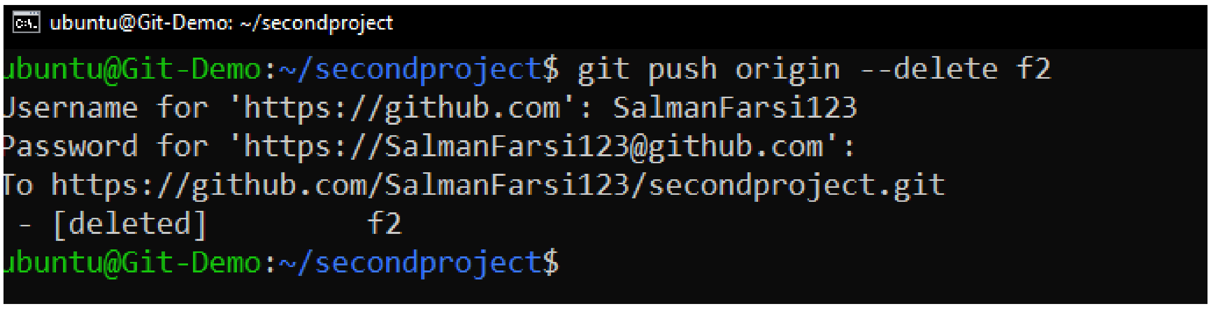


This branch is 1 commit ahead of, 1 commit behind develop .

SalmanFarsi123 Adding f2.txt 3cd62bc · 12 minutes ago

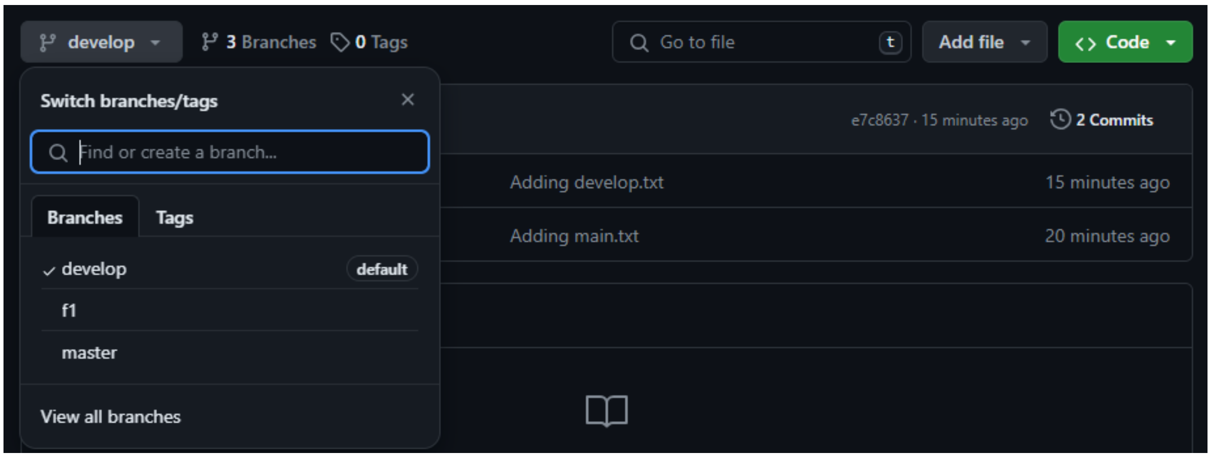
f2.txt	Adding f2.txt	12 minutes ago
main.txt	Adding main.txt	18 minutes ago

14. According to Task we need to delete in remote repo so run the command git push origin –delete f2 and it will delete.



```
ubuntu@Git-Demo: ~/secondproject
ubuntu@Git-Demo:~/secondproject$ git push origin --delete f2
Username for 'https://github.com': SalmanFarsi123
Password for 'https://SalmanFarsi123@github.com':
To https://github.com/SalmanFarsi123/secondproject.git
 - [deleted]         f2
ubuntu@Git-Demo:~/secondproject$
```

15. Deleted successfully.



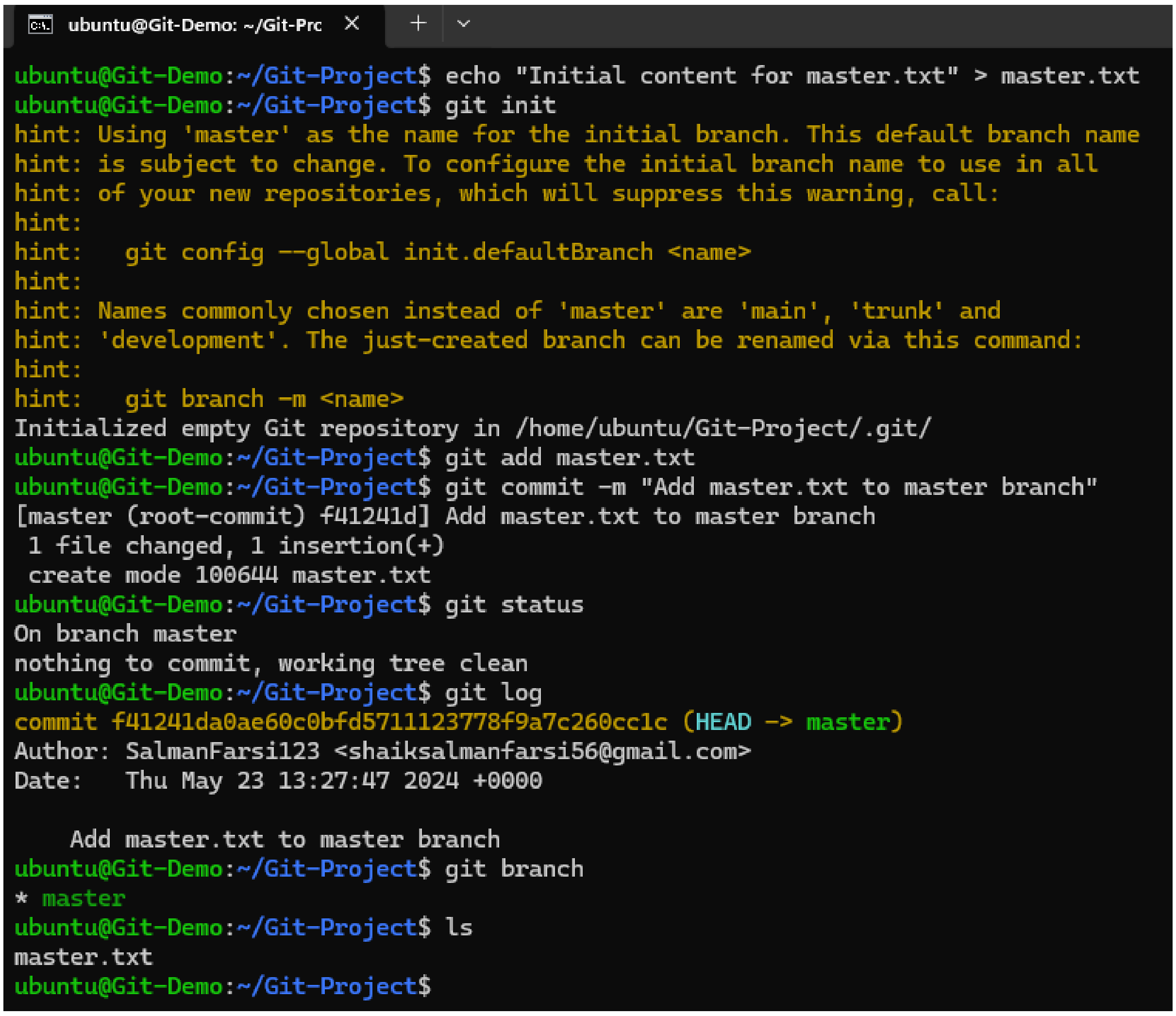
Module 2: Git Assignment - 4

DevOps Certification Training

Tasks To Be Performed:

1. Put master.txt on master branch, stage and commit
2. Create 3 branches: public 1, public 2 and private
3. Put public1.txt on public 1 branch, stage and commit
4. Merge public 1 on master branch
5. Merge public 2 on master branch
6. Edit master.txt on private branch, stage and commit
7. Now update branch public 1 and public 2 with new master code in private
8. Also update new master code on master
9. Finally update all the code on the private branch

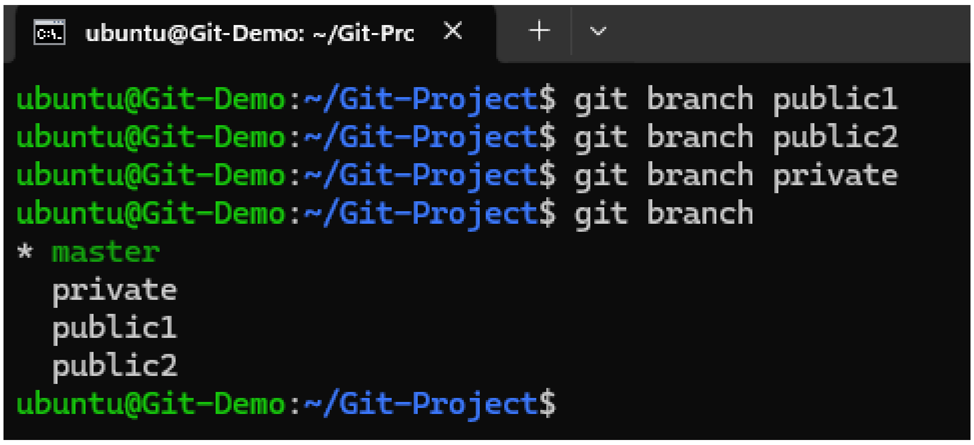
Step 1: **Put `master.txt` on master branch, stage, and commit**



ubuntu@Git-Demo: ~/Git-Prc

```
ubuntu@Git-Demo:~/Git-Project$ echo "Initial content for master.txt" > master.txt
ubuntu@Git-Demo:~/Git-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 /home/ubuntu/Git-Project/.git/
ubuntu@Git-Demo:~/Git-Project$ git add master.txt
ubuntu@Git-Demo:~/Git-Project$ git commit -m "Add master.txt to master branch"
[master (root-commit) f41241d] Add master.txt to master branch
 1 file changed, 1 insertion(+)
 create mode 100644 master.txt
ubuntu@Git-Demo:~/Git-Project$ git status
On branch master
nothing to commit, working tree clean
ubuntu@Git-Demo:~/Git-Project$ git log
commit f41241da0ae60c0bfd5711123778f9a7c260cc1c (HEAD -> master)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 13:27:47 2024 +0000

    Add master.txt to master branch
ubuntu@Git-Demo:~/Git-Project$ git branch
* master
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt
ubuntu@Git-Demo:~/Git-Project$
```

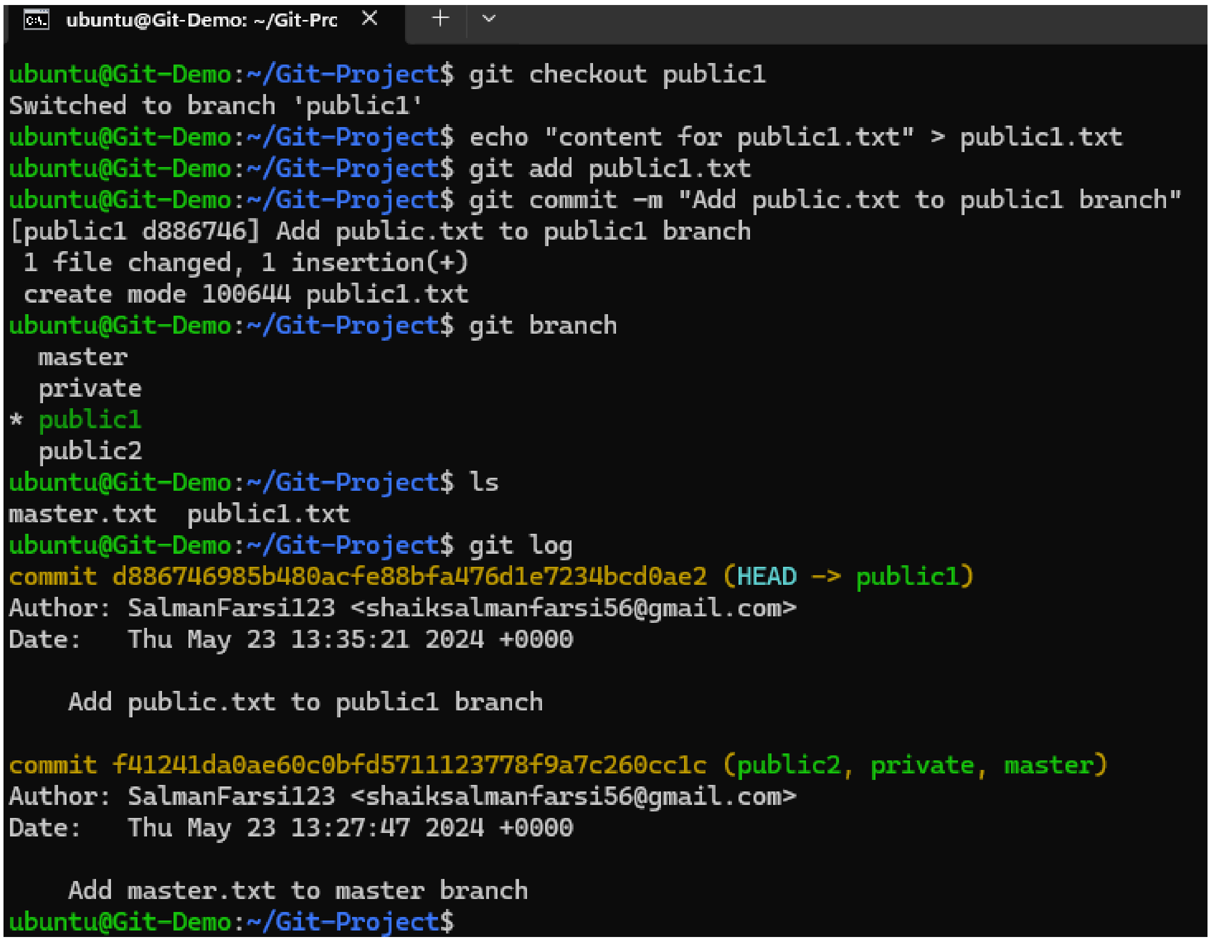
Step 2 **Create 3 branches: `public1`, `public2`, and `private`**



ubuntu@Git-Demo: ~/Git-Prc

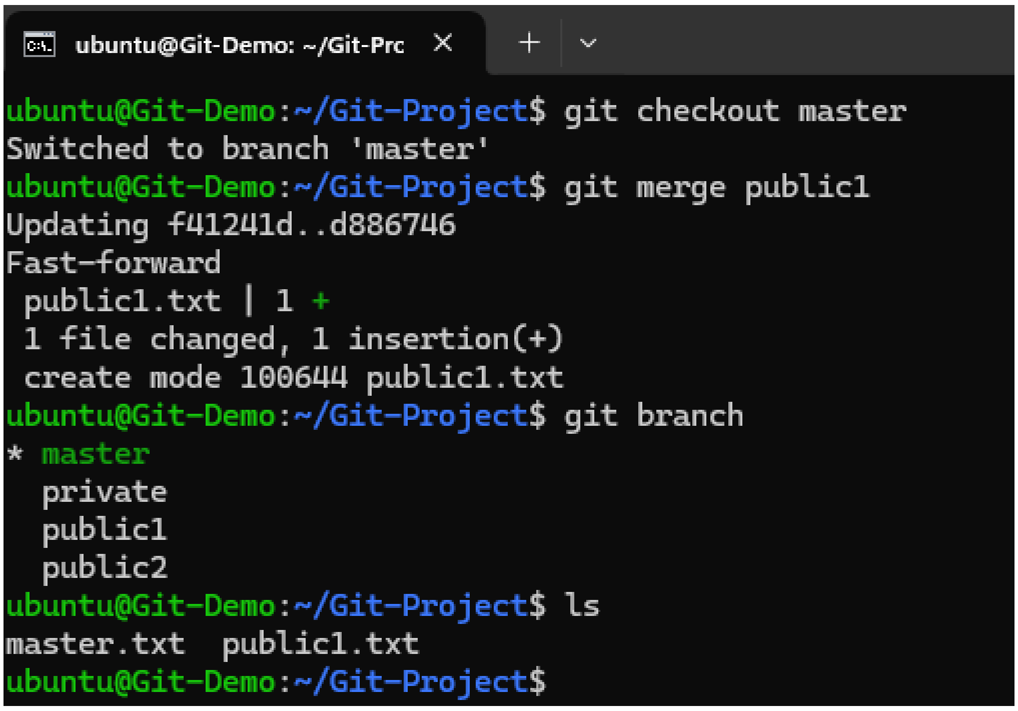
```
ubuntu@Git-Demo:~/Git-Project$ git branch public1
ubuntu@Git-Demo:~/Git-Project$ git branch public2
ubuntu@Git-Demo:~/Git-Project$ git branch private
ubuntu@Git-Demo:~/Git-Project$ git branch
* master
  private
  public1
  public2
ubuntu@Git-Demo:~/Git-Project$
```

Step 3: **Put `public1.txt` on `public1` branch, stage, and commit**



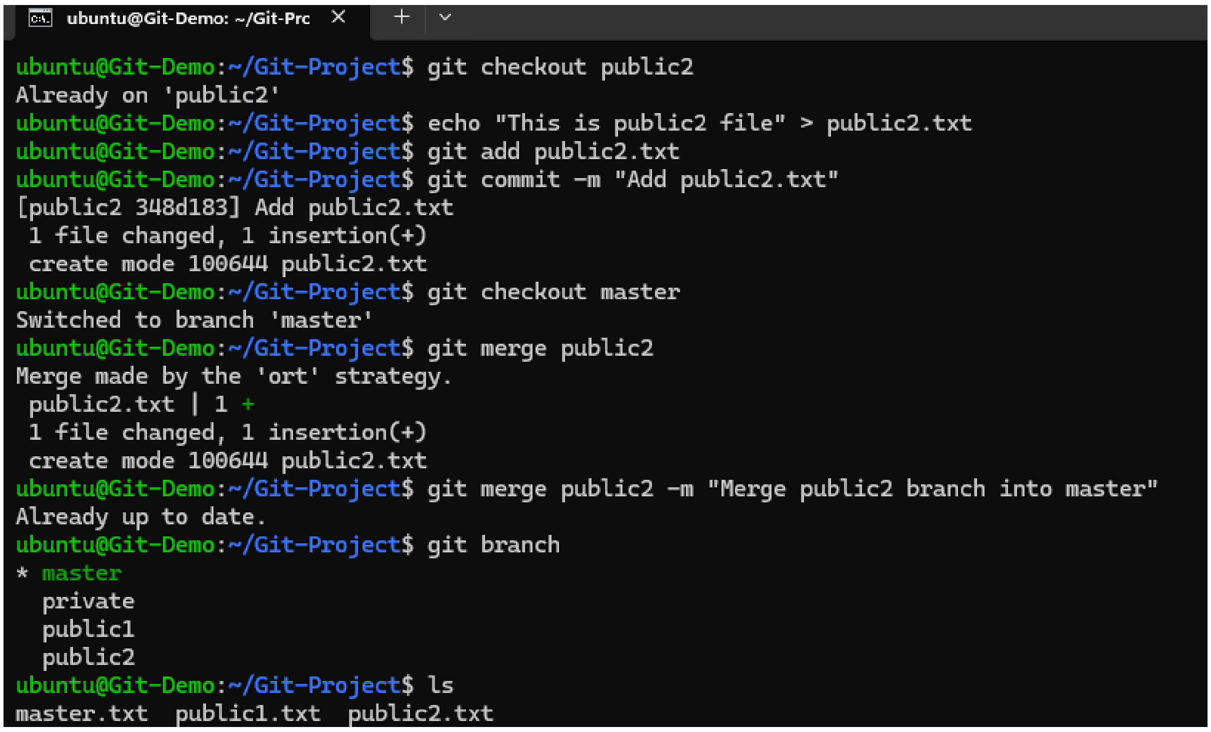
```
ubuntu@Git-Demo:~/Git-Project$ git checkout public1
Switched to branch 'public1'
ubuntu@Git-Demo:~/Git-Project$ echo "content for public1.txt" > public1.txt
ubuntu@Git-Demo:~/Git-Project$ git add public1.txt
ubuntu@Git-Demo:~/Git-Project$ git commit -m "Add public.txt to public1 branch"
[public1 d886746] Add public.txt to public1 branch
 1 file changed, 1 insertion(+)
 create mode 100644 public1.txt
ubuntu@Git-Demo:~/Git-Project$ git branch
  master
  private
* public1
  public2
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt
ubuntu@Git-Demo:~/Git-Project$ git log
commit d886746985b480acfe88bfa476d1e7234bcd0ae2 (HEAD -> public1)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 13:35:21 2024 +0000

    Add public.txt to public1 branch

commit f41241da0ae60c0bfd5711123778f9a7c260cc1c (public2, private, master)
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 13:27:47 2024 +0000

    Add master.txt to master branch
ubuntu@Git-Demo:~/Git-Project$
```

Step 4: Merge `public1` on master branch



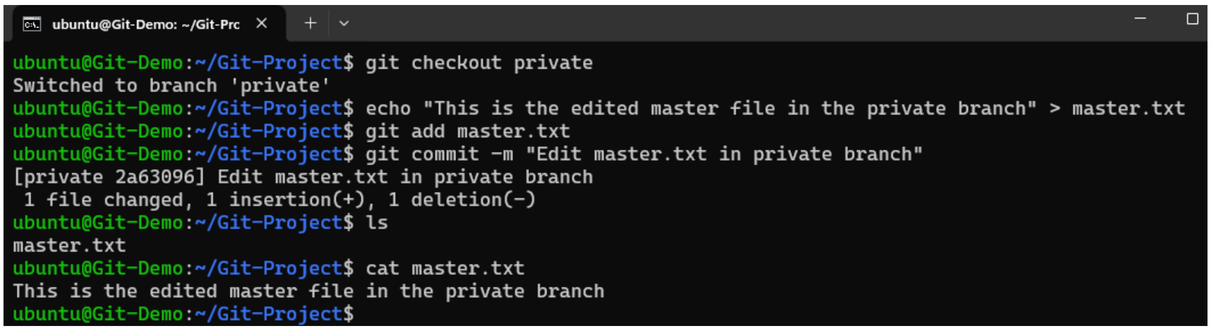
```
ubuntu@Git-Demo:~/Git-Project$ git checkout master
Switched to branch 'master'
ubuntu@Git-Demo:~/Git-Project$ git merge public1
Updating f41241d..d886746
Fast-forward
 public1.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 public1.txt
ubuntu@Git-Demo:~/Git-Project$ git branch
* master
  private
  public1
  public2
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt
ubuntu@Git-Demo:~/Git-Project$
```

Step 5: **Merge `public2` on the master branch**



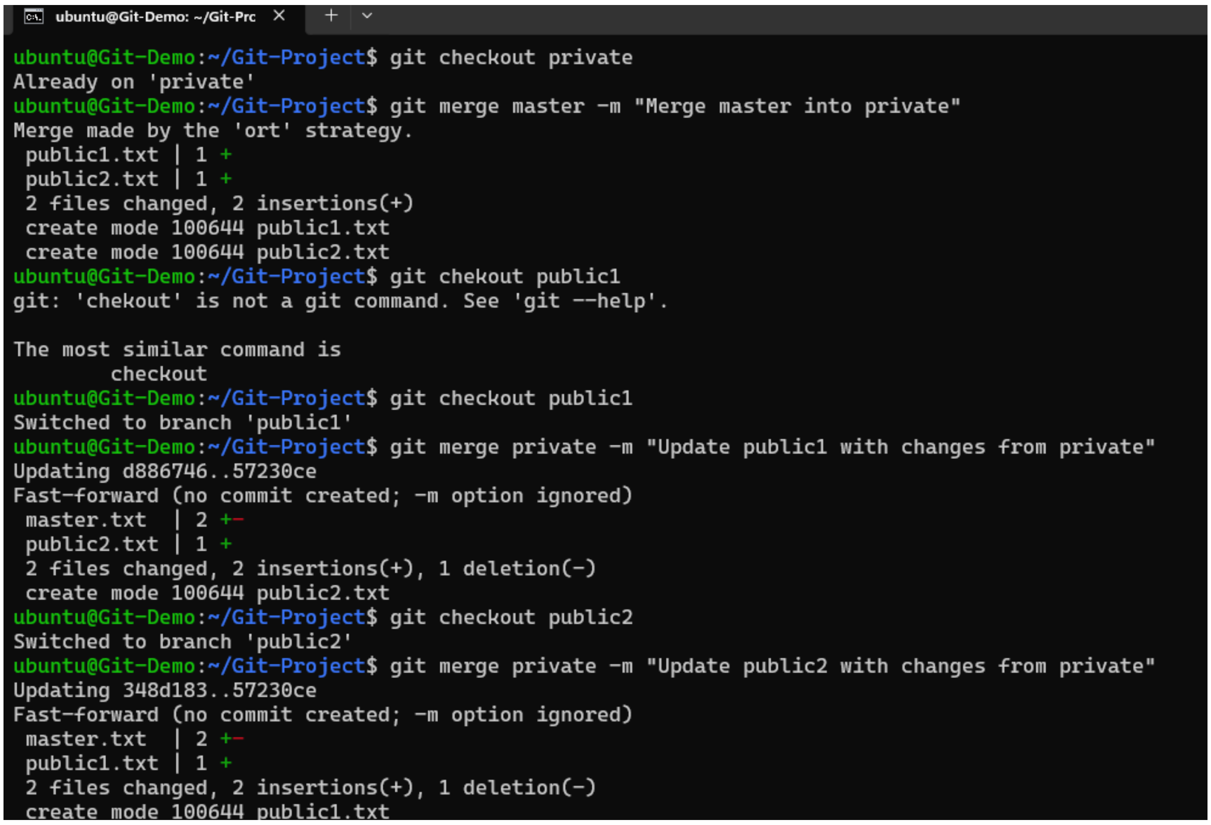
```
ubuntu@Git-Demo:~/Git-Project$ git checkout public2
Already on 'public2'
ubuntu@Git-Demo:~/Git-Project$ echo "This is public2 file" > public2.txt
ubuntu@Git-Demo:~/Git-Project$ git add public2.txt
ubuntu@Git-Demo:~/Git-Project$ git commit -m "Add public2.txt"
[public2 348d183] Add public2.txt
 1 file changed, 1 insertion(+)
 create mode 100644 public2.txt
ubuntu@Git-Demo:~/Git-Project$ git checkout master
Switched to branch 'master'
ubuntu@Git-Demo:~/Git-Project$ git merge public2
Merge made by the 'ort' strategy.
 public2.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 public2.txt
ubuntu@Git-Demo:~/Git-Project$ git merge public2 -m "Merge public2 branch into master"
Already up to date.
ubuntu@Git-Demo:~/Git-Project$ git branch
* master
  private
  public1
  public2
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt  public2.txt
```

Step 6: **Edit master.txt on the private branch, stage and commit**



```
ubuntu@Git-Demo:~/Git-Project$ git checkout private
Switched to branch 'private'
ubuntu@Git-Demo:~/Git-Project$ echo "This is the edited master file in the private branch" > master.txt
ubuntu@Git-Demo:~/Git-Project$ git add master.txt
ubuntu@Git-Demo:~/Git-Project$ git commit -m "Edit master.txt in private branch"
[private 2a63096] Edit master.txt in private branch
 1 file changed, 1 insertion(+), 1 deletion(-)
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
This is the edited master file in the private branch
ubuntu@Git-Demo:~/Git-Project$
```

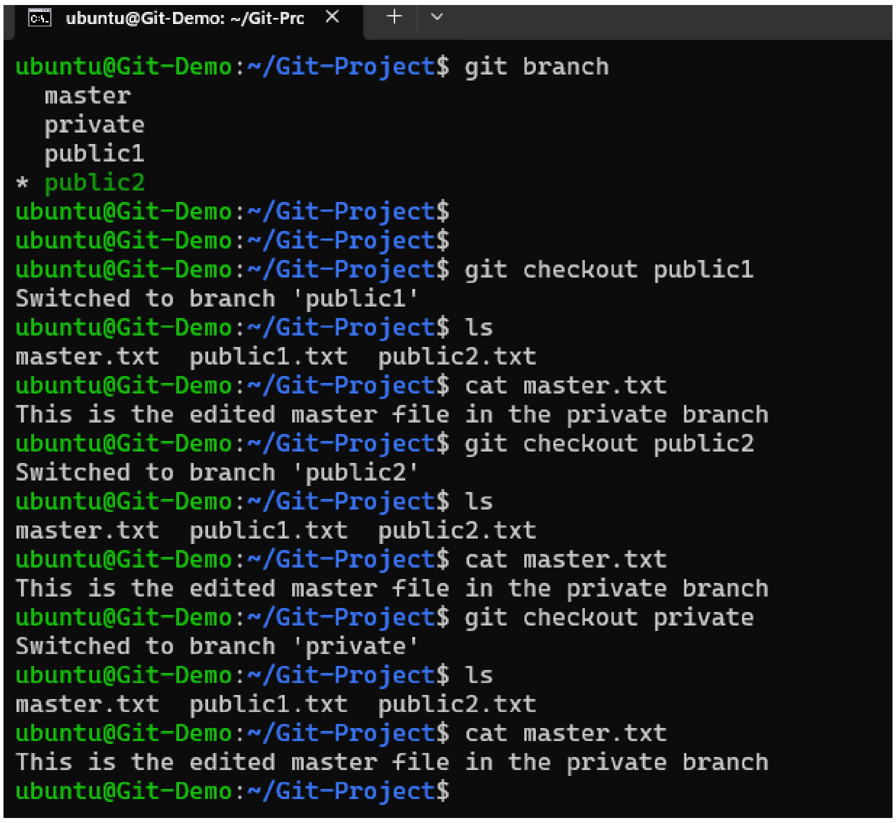
Step 7: Update branches public1 and public2 with the new master code in private



ubuntu@Git-Demo: ~/Git-Prc

```
ubuntu@Git-Demo:~/Git-Project$ git checkout private
Already on 'private'
ubuntu@Git-Demo:~/Git-Project$ git merge master -m "Merge master into private"
Merge made by the 'ort' strategy.
 public1.txt | 1 +
 public2.txt | 1 +
 2 files changed, 2 insertions(+)
 create mode 100644 public1.txt
 create mode 100644 public2.txt
ubuntu@Git-Demo:~/Git-Project$ git chekout public1
git: 'chekout' is not a git command. See 'git --help'.

The most similar command is
        checkout
ubuntu@Git-Demo:~/Git-Project$ git checkout public1
Switched to branch 'public1'
ubuntu@Git-Demo:~/Git-Project$ git merge private -m "Update public1 with changes from private"
Updating d886746..57230ce
Fast-forward (no commit created; -m option ignored)
 master.txt  | 2 +-
 public2.txt | 1 +
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 public2.txt
ubuntu@Git-Demo:~/Git-Project$ git checkout public2
Switched to branch 'public2'
ubuntu@Git-Demo:~/Git-Project$ git merge private -m "Update public2 with changes from private"
Updating 348d183..57230ce
Fast-forward (no commit created; -m option ignored)
 master.txt  | 2 +-
 public1.txt | 1 +
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 public1.txt
```

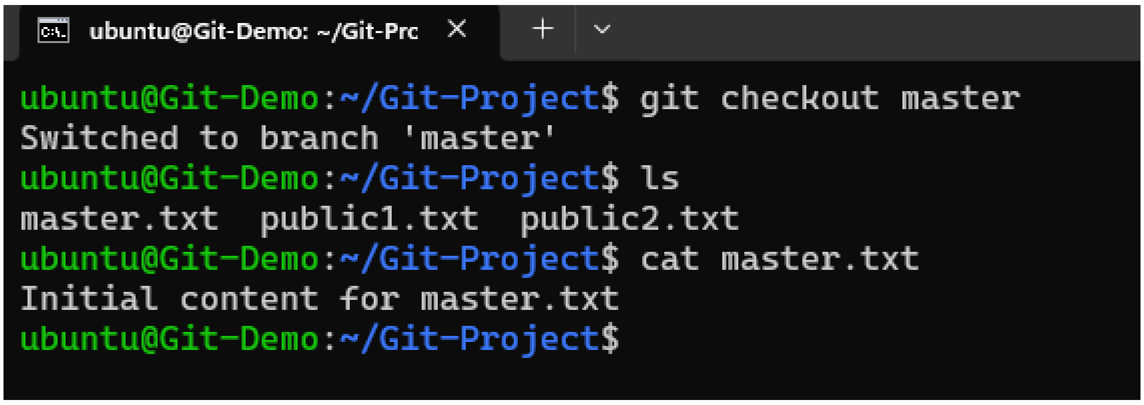
Now we can See successfully merged private master code into public1 and public2



ubuntu@Git-Demo: ~/Git-Prc

```
ubuntu@Git-Demo:~/Git-Project$ git branch
  master
  private
  public1
* public2
ubuntu@Git-Demo:~/Git-Project$
ubuntu@Git-Demo:~/Git-Project$
ubuntu@Git-Demo:~/Git-Project$ git checkout public1
Switched to branch 'public1'
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt  public2.txt
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
This is the edited master file in the private branch
ubuntu@Git-Demo:~/Git-Project$ git checkout public2
Switched to branch 'public2'
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt  public2.txt
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
This is the edited master file in the private branch
ubuntu@Git-Demo:~/Git-Project$ git checkout private
Switched to branch 'private'
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt  public2.txt
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
This is the edited master file in the private branch
ubuntu@Git-Demo:~/Git-Project$
```

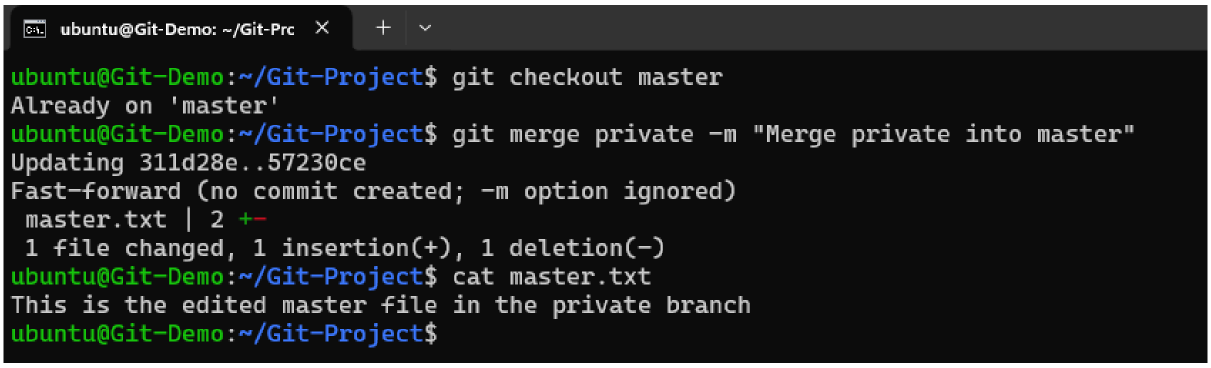
Before Starting step 8 we just checking the master branch, see in the below its initially data is available as of now



ubuntu@Git-Demo: ~/Git-Prc

```
ubuntu@Git-Demo:~/Git-Project$ git checkout master
Switched to branch 'master'
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt  public2.txt
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
Initial content for master.txt
ubuntu@Git-Demo:~/Git-Project$
```

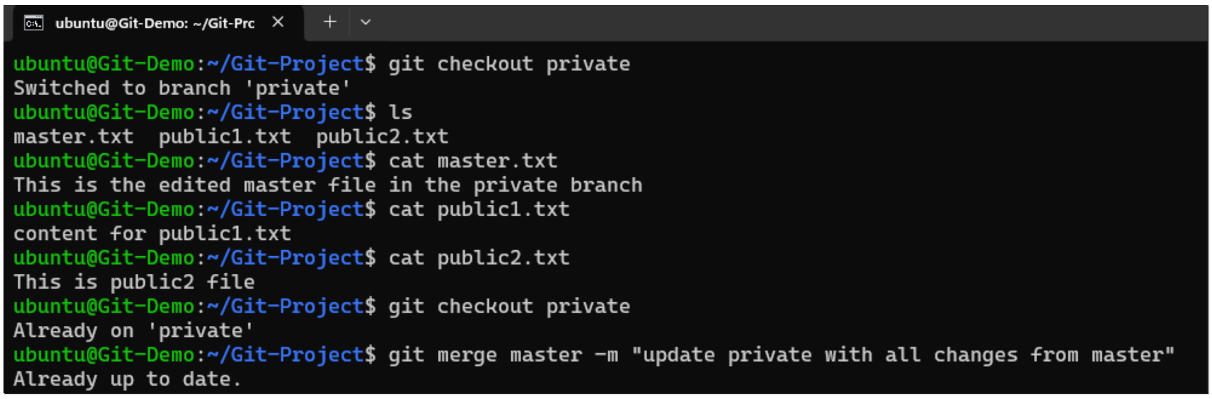
Step 8: **Also update the new master code on master**



ubuntu@Git-Demo: ~/Git-Prc

```
ubuntu@Git-Demo:~/Git-Project$ git checkout master
Already on 'master'
ubuntu@Git-Demo:~/Git-Project$ git merge private -m "Merge private into master"
Updating 311d28e..57230ce
Fast-forward (no commit created; -m option ignored)
 master.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
This is the edited master file in the private branch
ubuntu@Git-Demo:~/Git-Project$
```

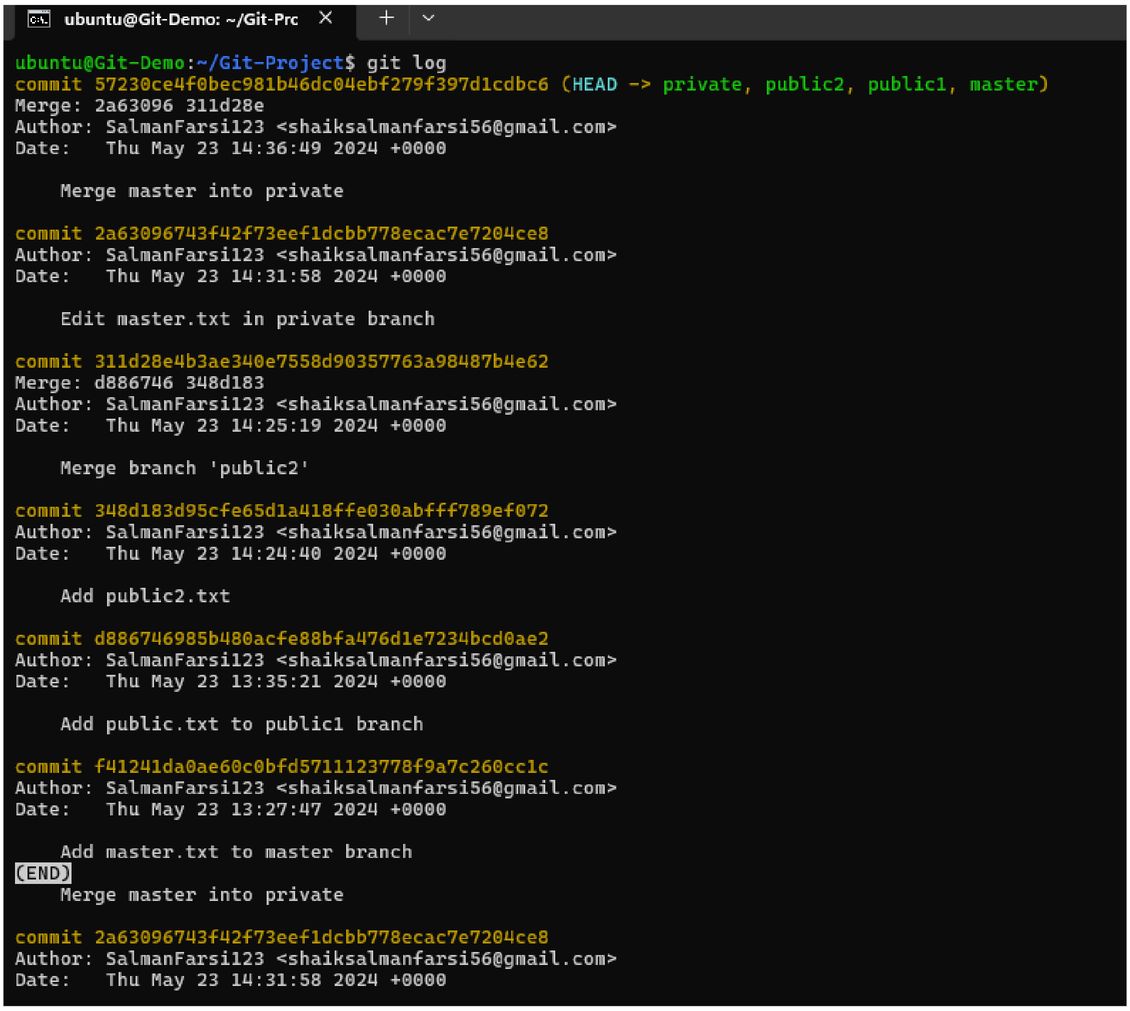
Step 9: **Finally update all the code on the private branch, its just updating from the master branch we, can its already up to date**



ubuntu@Git-Demo: ~/Git-Prc

```
ubuntu@Git-Demo:~/Git-Project$ git checkout private
Switched to branch 'private'
ubuntu@Git-Demo:~/Git-Project$ ls
master.txt  public1.txt  public2.txt
ubuntu@Git-Demo:~/Git-Project$ cat master.txt
This is the edited master file in the private branch
ubuntu@Git-Demo:~/Git-Project$ cat public1.txt
content for public1.txt
ubuntu@Git-Demo:~/Git-Project$ cat public2.txt
This is public2 file
ubuntu@Git-Demo:~/Git-Project$ git checkout private
Already on 'private'
ubuntu@Git-Demo:~/Git-Project$ git merge master -m "update private with all changes from master"
Already up to date.
```

Git Logs:


```
ubuntu@Git-Demo: ~/Git-Prc
```

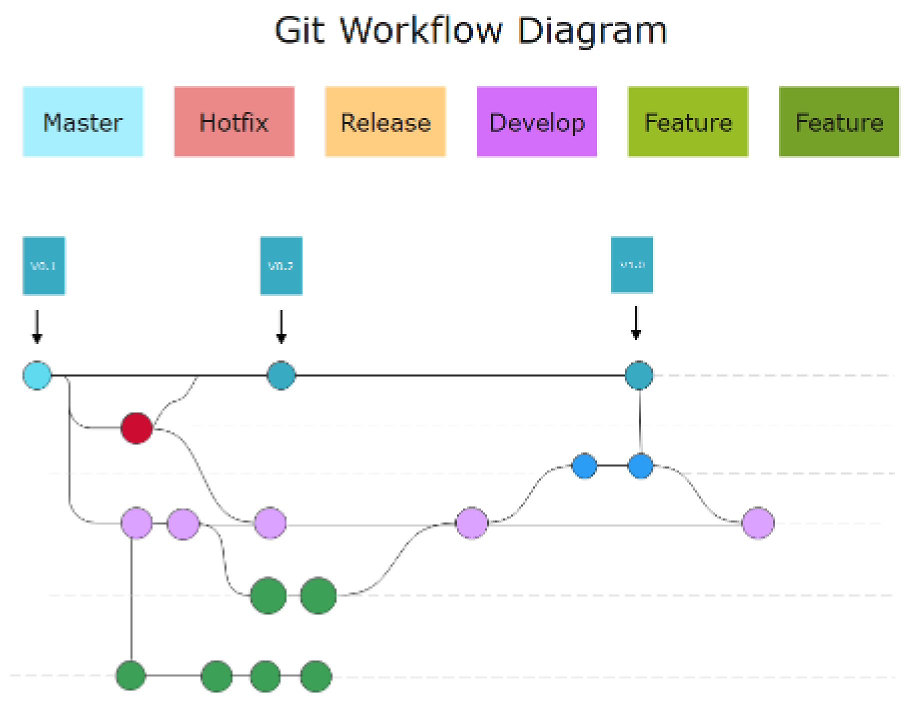
```
ubuntu@Git-Demo:~/Git-Project$ git log
commit 57230ce4f0bec981b46dc04ebf279f397d1cdbc6 (HEAD -> private, public2, public1, master)
Merge: 2a63096 311d28e
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 14:36:49 2024 +0000

    Merge master into private

commit 2a63096743f42f73eef1dcbb778ecac7e7204ce8
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 14:31:58 2024 +0000

    Edit master.txt in private branch

commit 311d28e4b3ae340e7558d90357763a98487b4e62
Merge: d886746 348d183
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 14:25:19 2024 +0000

    Merge branch 'public2'

commit 348d183d95cfe65d1a418ffe030abfff789ef072
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 14:24:40 2024 +0000

    Add public2.txt

commit d886746985b480acfe88bfa476d1e7234bcd0ae2
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 13:35:21 2024 +0000

    Add public.txt to public1 branch

commit f41241da0ae60c0bfd5711123778f9a7c260cc1c
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 13:27:47 2024 +0000

    Add master.txt to master branch
(END)
    Merge master into private

commit 2a63096743f42f73eef1dcbb778ecac7e7204ce8
Author: SalmanFarsi123 <shaiksalmanfarsi56@gmail.com>
Date:   Thu May 23 14:31:58 2024 +0000
```

Module 2: Git Assignment - 5

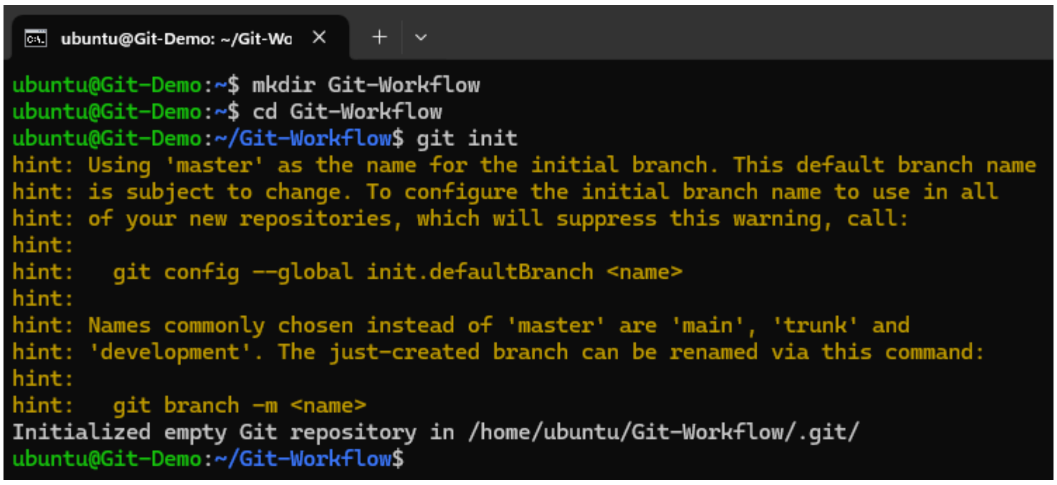
DevOps Certification Training

Tasks To Be Performed:

1. Create a Git Flow workflow architecture on Git
2. Create all the required branches
3. Starting from the feature branch, push the branch to the master, following the architecture
4. Push a urgent.txt on master using hotfix

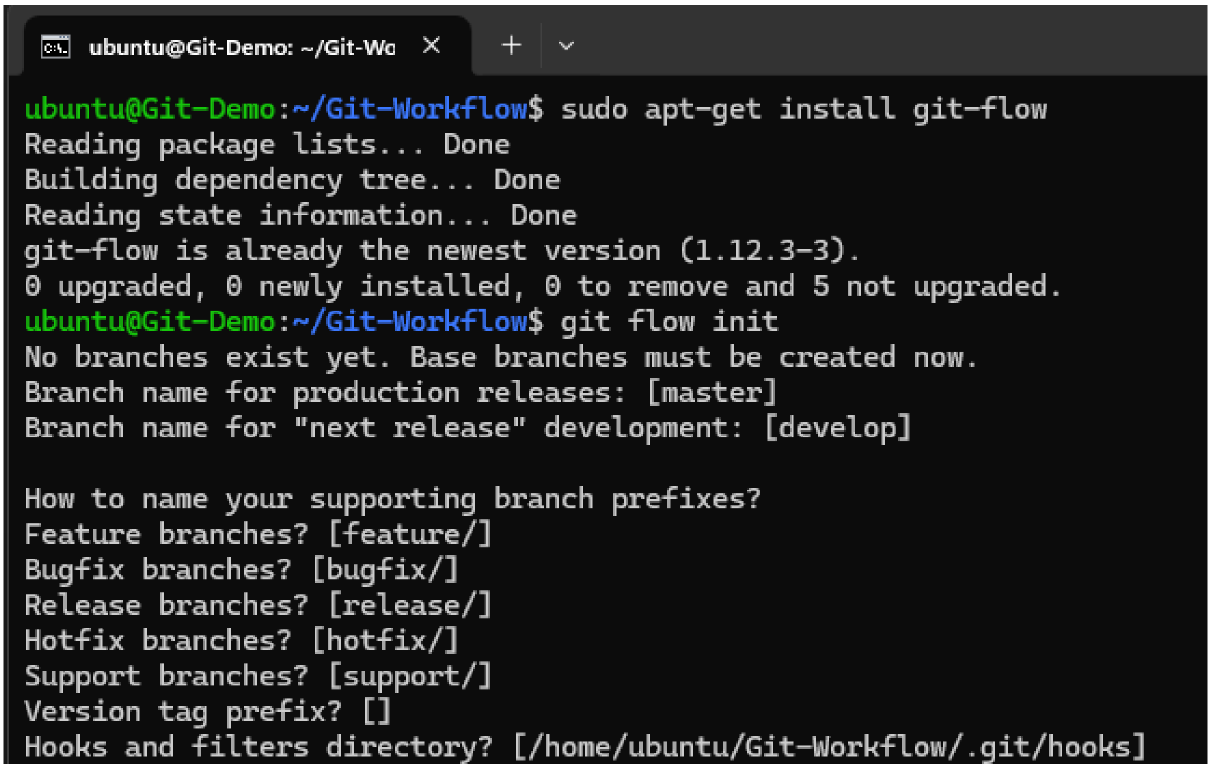


Step 1: **Create a Git Workflow architecture on Git**



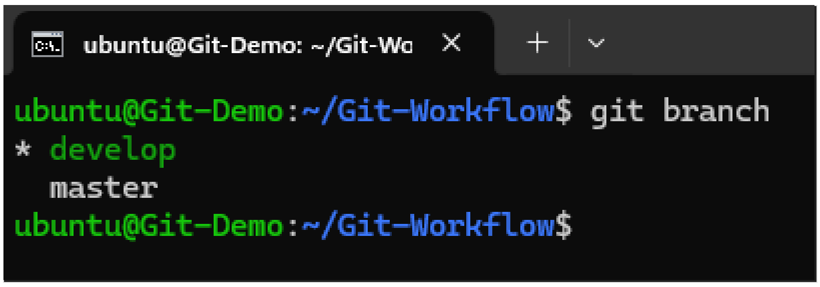
```
ubuntu@Git-Demo:~$ mkdir Git-Workflow
ubuntu@Git-Demo:~$ cd Git-Workflow
ubuntu@Git-Demo:~/Git-Workflow$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, 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 /home/ubuntu/Git-Workflow/.git/
ubuntu@Git-Demo:~/Git-Workflow$
```

Step2: Create all the required branches



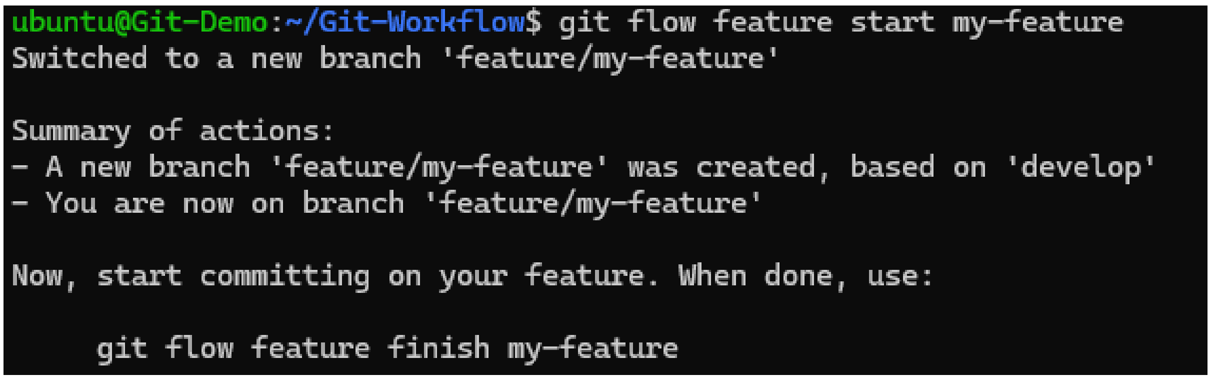
```
ubuntu@Git-Demo:~/Git-Workflow$ sudo apt-get install git-flow
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git-flow is already the newest version (1.12.3-3).
0 upgraded, 0 newly installed, 0 to remove and 5 not upgraded.
ubuntu@Git-Demo:~/Git-Workflow$ git flow init
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [/home/ubuntu/Git-Workflow/.git/hooks]
```

After Initializing git-flow, you will have the master and develop branches created automatically



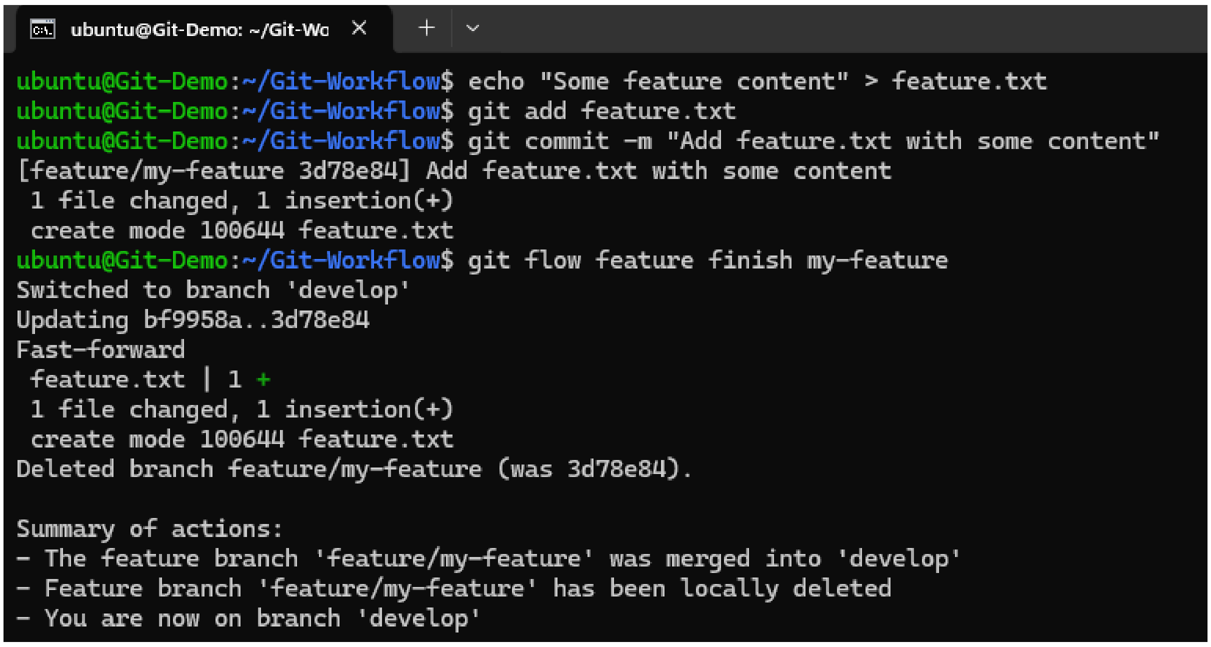
```
ubuntu@Git-Demo:~/Git-Workflow$ git branch
* develop
  master
ubuntu@Git-Demo:~/Git-Workflow$
```

Step 3: **Create a feature branch**



```
ubuntu@Git-Demo:~/Git-Workflow$ git flow feature start my-feature
Switched to a new branch 'feature/my-feature'

Summary of actions:
- A new branch 'feature/my-feature' was created, based on 'develop'
- You are now on branch 'feature/my-feature'

Now, start committing on your feature. When done, use:

     git flow feature finish my-feature
```

Finish the Feature branch

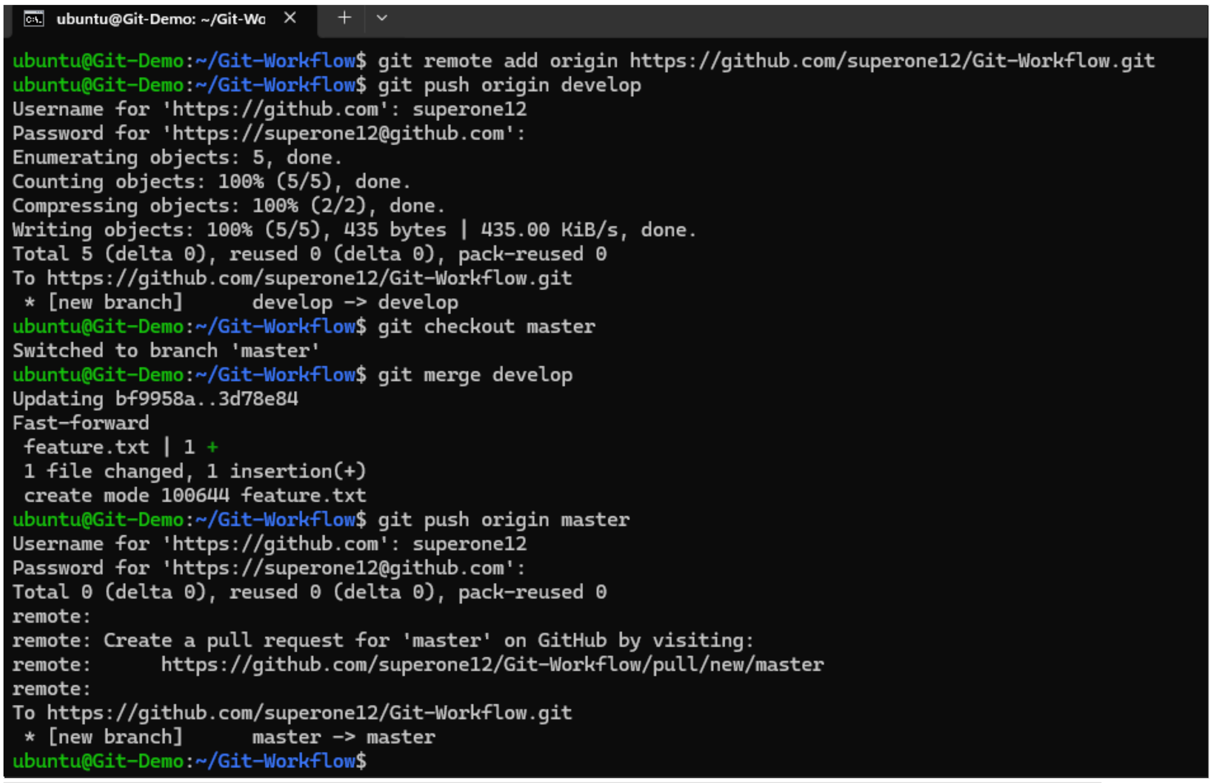


```
ubuntu@Git-Demo:~/Git-Workflow$ echo "Some feature content" > feature.txt
ubuntu@Git-Demo:~/Git-Workflow$ git add feature.txt
ubuntu@Git-Demo:~/Git-Workflow$ git commit -m "Add feature.txt with some content"
[feature/my-feature 3d78e84] Add feature.txt with some content
 1 file changed, 1 insertion(+)
 create mode 100644 feature.txt
ubuntu@Git-Demo:~/Git-Workflow$ git flow feature finish my-feature
Switched to branch 'develop'
Updating bf9958a..3d78e84
Fast-forward
 feature.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 feature.txt
Deleted branch feature/my-feature (was 3d78e84).

Summary of actions:
- The feature branch 'feature/my-feature' was merged into 'develop'
- Feature branch 'feature/my-feature' has been locally deleted
- You are now on branch 'develop'
```

This Command Will:

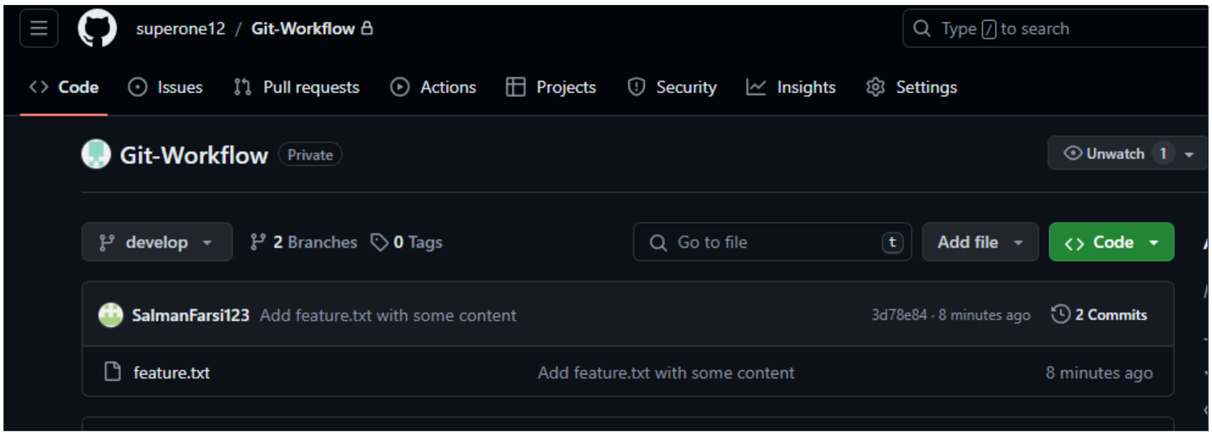
- Merge `feature/my-feature` into `develop`
- Delete `feature/my-feature` branch
- Switch back to the `develop` branch

3. Starting from the feature branch, push the branch to the master, following the architecture



```
ubuntu@Git-Demo:~/Git-Workflow$ git remote add origin https://github.com/superone12/Git-Workflow.git
ubuntu@Git-Demo:~/Git-Workflow$ git push origin develop
Username for 'https://github.com': superone12
Password for 'https://superone12@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 435 bytes | 435.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/superone12/Git-Workflow.git
 * [new branch]      develop -> develop
ubuntu@Git-Demo:~/Git-Workflow$ git checkout master
Switched to branch 'master'
ubuntu@Git-Demo:~/Git-Workflow$ git merge develop
Updating bf9958a..3d78e84
Fast-forward
 feature.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 feature.txt
ubuntu@Git-Demo:~/Git-Workflow$ git push origin master
Username for 'https://github.com': superone12
Password for 'https://superone12@github.com':
Total 0 (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/superone12/Git-Workflow/pull/new/master
remote:
To https://github.com/superone12/Git-Workflow.git
 * [new branch]      master -> master
ubuntu@Git-Demo:~/Git-Workflow$
```

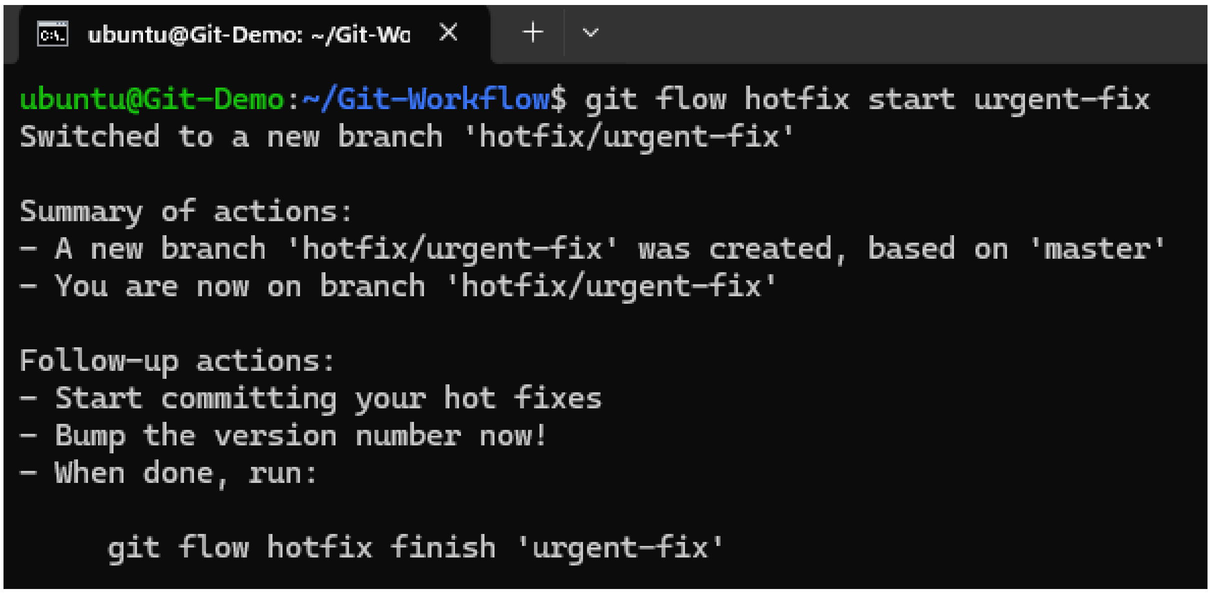
we can see in the remote repo, its successfully push the things and we can see 2 branches



SalmanFarsi123 Add feature.txt with some content 3d78e84 · 8 minutes ago 2 Commits

feature.txt Add feature.txt with some content 8 minutes ago

Step 4: Push an urgent.txt on Master Using Hotfix



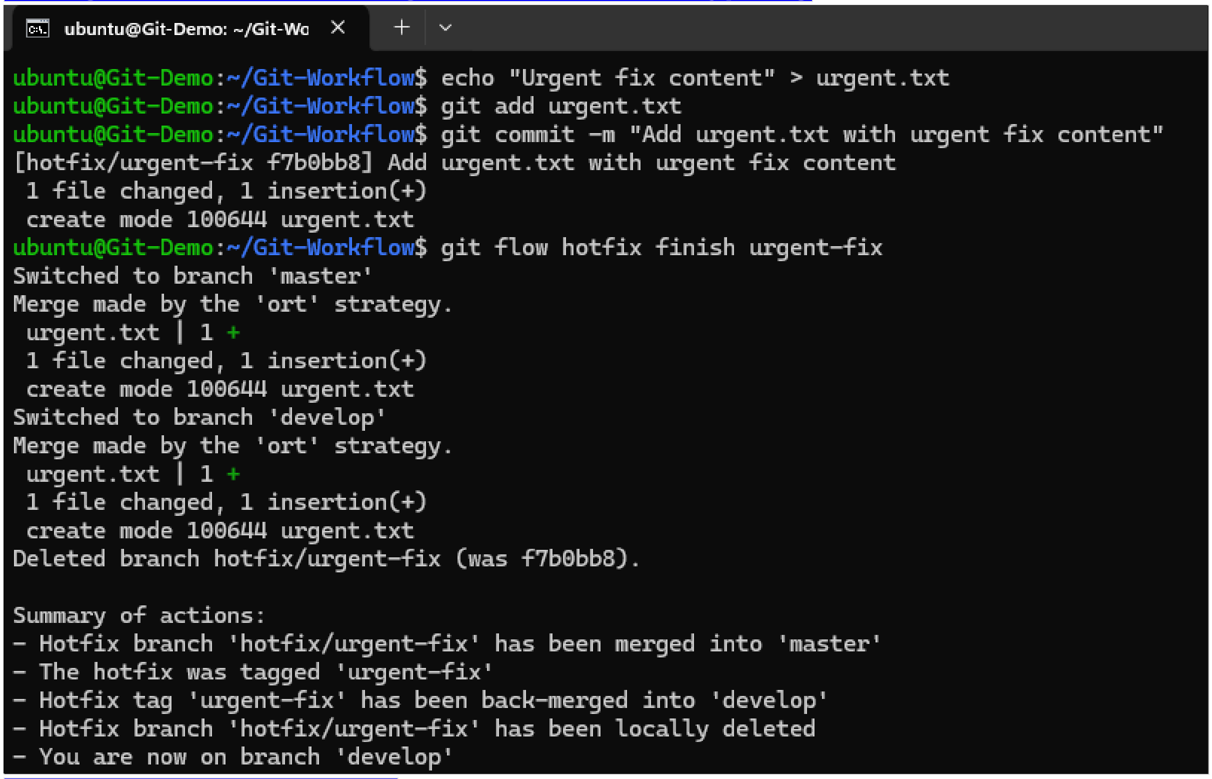
```
ubuntu@Git-Demo:~/Git-Workflow$ git flow hotfix start urgent-fix
Switched to a new branch 'hotfix/urgent-fix'

Summary of actions:
- A new branch 'hotfix/urgent-fix' was created, based on 'master'
- You are now on branch 'hotfix/urgent-fix'

Follow-up actions:
- Start committing your hot fixes
- Bump the version number now!
- When done, run:

     git flow hotfix finish 'urgent-fix'
```

Add and commit the urgent.txt file:

This command will:

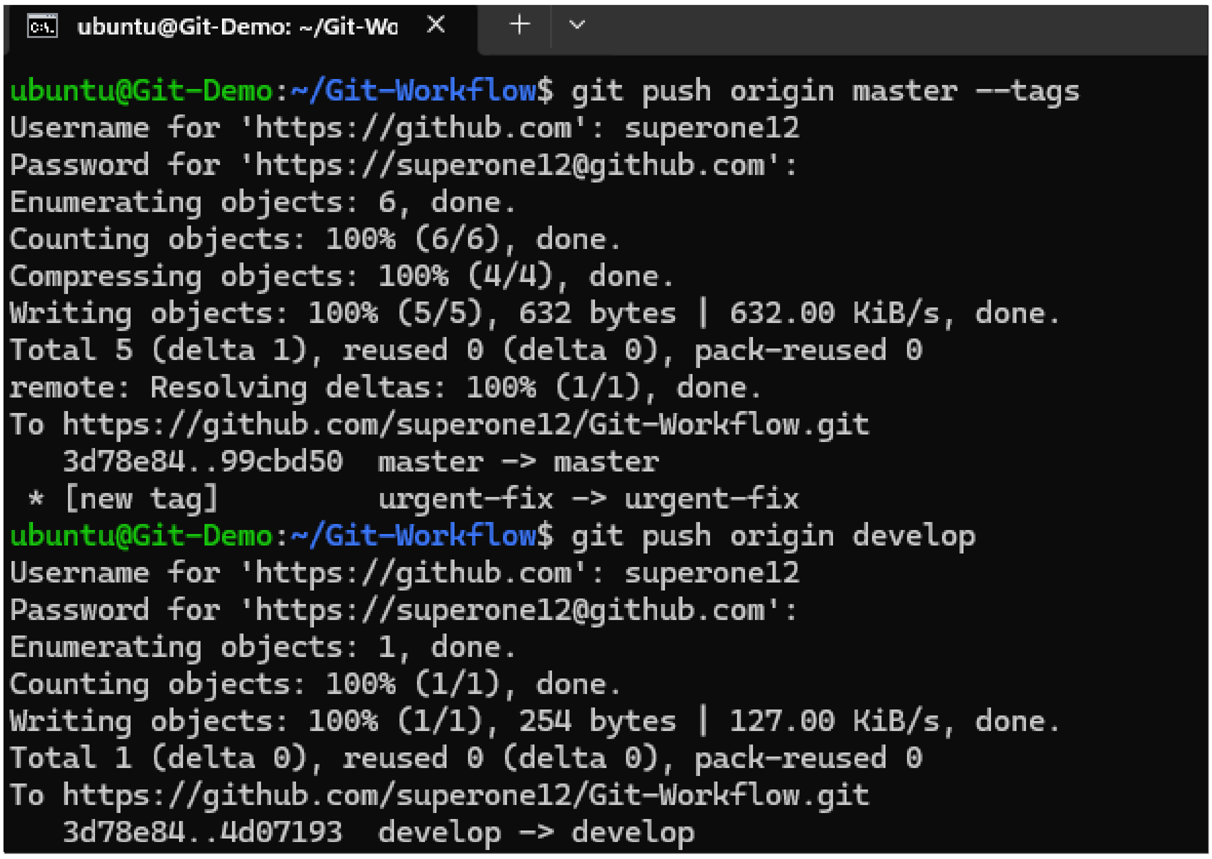
- Merge `hotfix/urgent-fix` into `master` and `develop`
- Tag the master branch
- Delete `hotfix/urgent-fix` branch
- Switch back to the `master` branch

After git flow hotfix finish urgent-fix it will ask to type tag



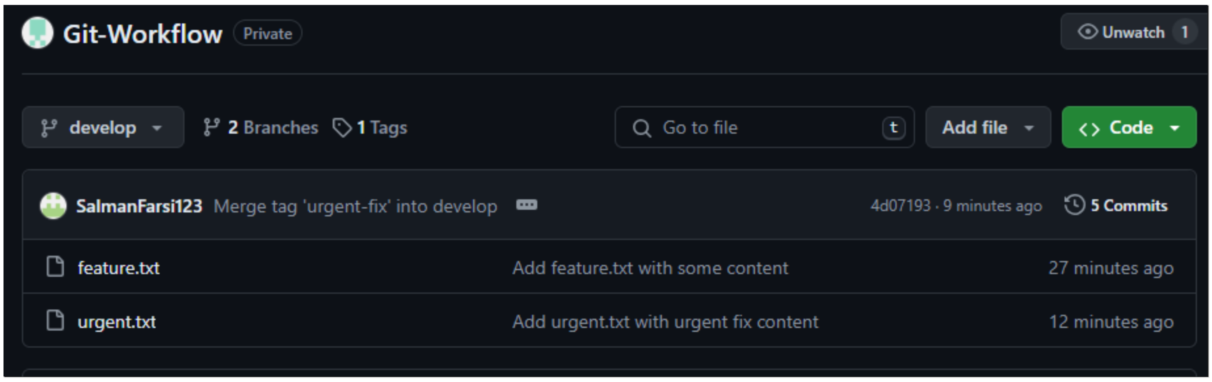
```
ubuntu@Git-Demo:~/Git-Workflow$ echo "Urgent fix content" > urgent.txt
ubuntu@Git-Demo:~/Git-Workflow$ git add urgent.txt
ubuntu@Git-Demo:~/Git-Workflow$ git commit -m "Add urgent.txt with urgent fix content"
[hotfix/urgent-fix f7b0bb8] Add urgent.txt with urgent fix content
 1 file changed, 1 insertion(+)
 create mode 100644 urgent.txt
ubuntu@Git-Demo:~/Git-Workflow$ git flow hotfix finish urgent-fix
Switched to branch 'master'
Merge made by the 'ort' strategy.
 urgent.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 urgent.txt
Switched to branch 'develop'
Merge made by the 'ort' strategy.
 urgent.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 urgent.txt
Deleted branch hotfix/urgent-fix (was f7b0bb8).

Summary of actions:
- Hotfix branch 'hotfix/urgent-fix' has been merged into 'master'
- The hotfix was tagged 'urgent-fix'
- Hotfix tag 'urgent-fix' has been back-merged into 'develop'
- Hotfix branch 'hotfix/urgent-fix' has been locally deleted
- You are now on branch 'develop'
```

git push master and develop



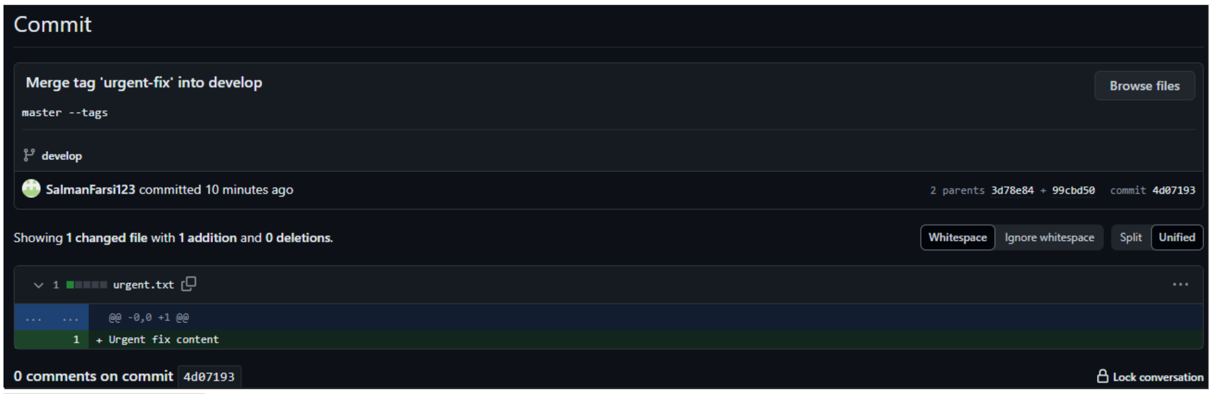
```
ubuntu@Git-Demo:~/Git-Workflow$ git push origin master --tags
Username for 'https://github.com': superone12
Password for 'https://superone12@github.com':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 632 bytes | 632.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/superone12/Git-Workflow.git
   3d78e84..99cbd50  master -> master
 * [new tag]         urgent-fix -> urgent-fix
ubuntu@Git-Demo:~/Git-Workflow$ git push origin develop
Username for 'https://github.com': superone12
Password for 'https://superone12@github.com':
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 254 bytes | 127.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/superone12/Git-Workflow.git
   3d78e84..4d07193  develop -> develop
```

Develop Branch


Git-Workflow Private
Unwatch 1
develop 2 Branches 1 Tags Go to file Add file Code
SalmanFarsi123 Merge tag 'urgent-fix' into develop 4d07193 · 9 minutes ago 5 Commits

feature.txt	Add feature.txt with some content	27 minutes ago
urgent.txt	Add urgent.txt with urgent fix content	12 minutes ago

Its showing the I written tag content in the develop branch


Commit

Merge tag 'urgent-fix' into develop

```
master --tags
```

develop

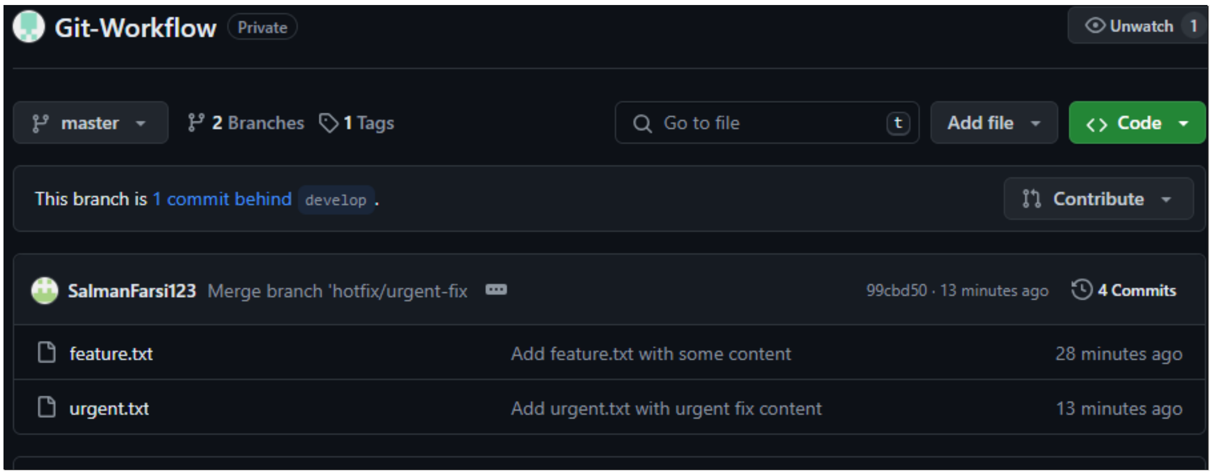
SalmanFarsi123 committed 10 minutes ago 2 parents 3d78e84 + 99cbd50 commit 4d07193

Showing 1 changed file with 1 addition and 0 deletions.

```
1 urgent.txt
@@ -0,0 +1 @@
1  + Urgent fix content
```

0 comments on commit 4d07193 Lock conversation

Master Branch

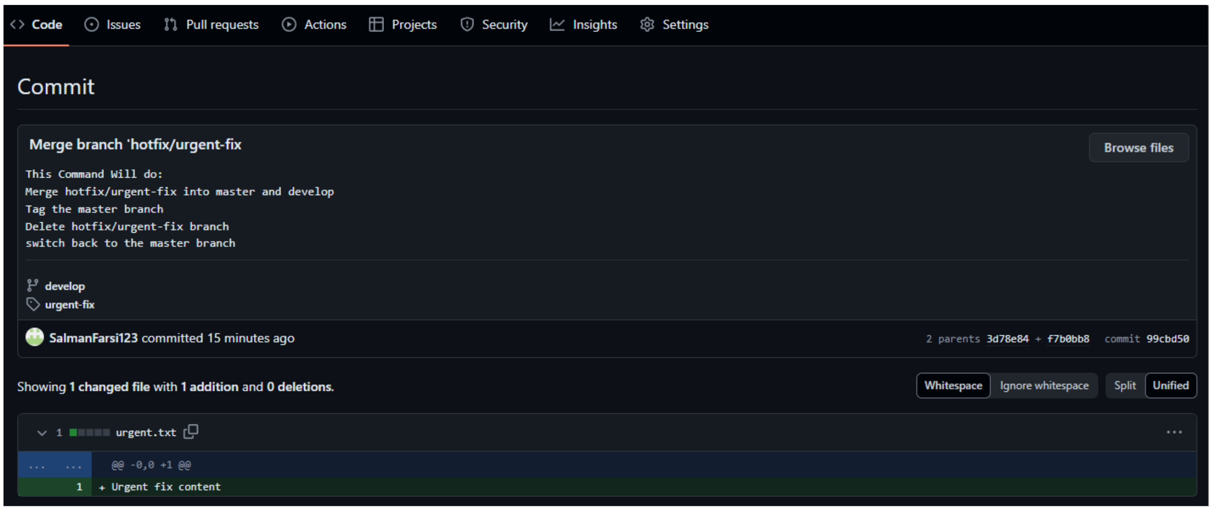

Git-Workflow Private
Unwatch 1
master 2 Branches 1 Tags Go to file Add file Code

This branch is 1 commit behind develop. Contribute

SalmanFarsi123 Merge branch 'hotfix/urgent-fix 99cbd50 · 13 minutes ago 4 Commits

feature.txt	Add feature.txt with some content	28 minutes ago
urgent.txt	Add urgent.txt with urgent fix content	13 minutes ago

Its showing the what we written the content to Develop to urgent-fix(Bug Fix)



Commit

Merge branch 'hotfix/urgent-fix

```
This Command Will do:
Merge hotfix/urgent-fix into master and develop
Tag the master branch
Delete hotfix/urgent-fix branch
switch back to the master branch
```

develop
urgent-fix

SalmanFarsi123 committed 15 minutes ago

2 parents 3d78e84 + f7b0bb8 commit 99cbd50

Showing **1 changed file** with **1 addition** and **0 deletions.**

```
1 urgent.txt
@@ -0,0 +1 @@
1 + Urgent fix content
```