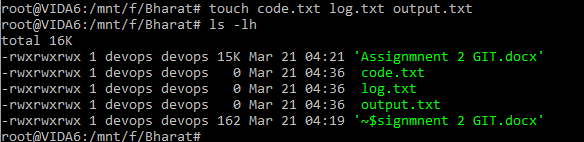
**1 Based on what you have learnt in the class, do the following steps:**

1. Create a new folder
2. b. Put the following files in the folder

 Code.txt

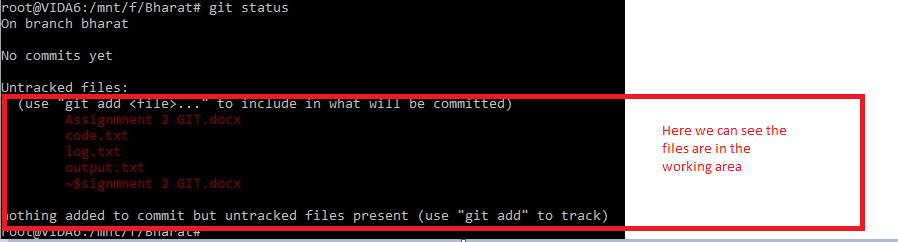
Log.txt

Output.txt

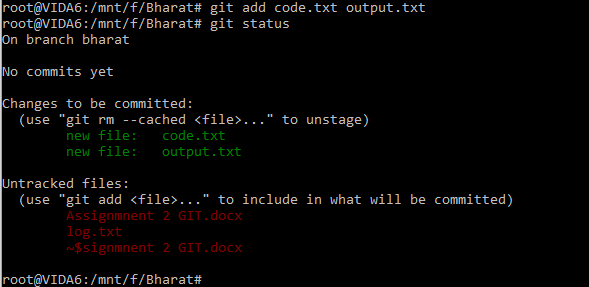


1. Stage the Code.txt and Output.txt files

For staging we need to use git add command, using this command in git we can move form working area to staging area. As shown in below command:



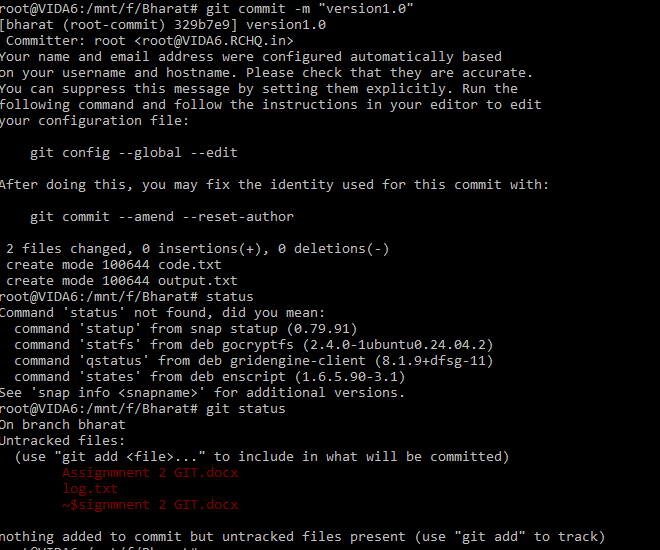
After then using **git** **add** code.txt and output.txt



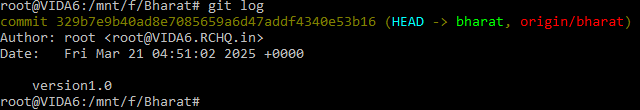
1. Commit them

Command used for commit : **git commit -m “commit name”**

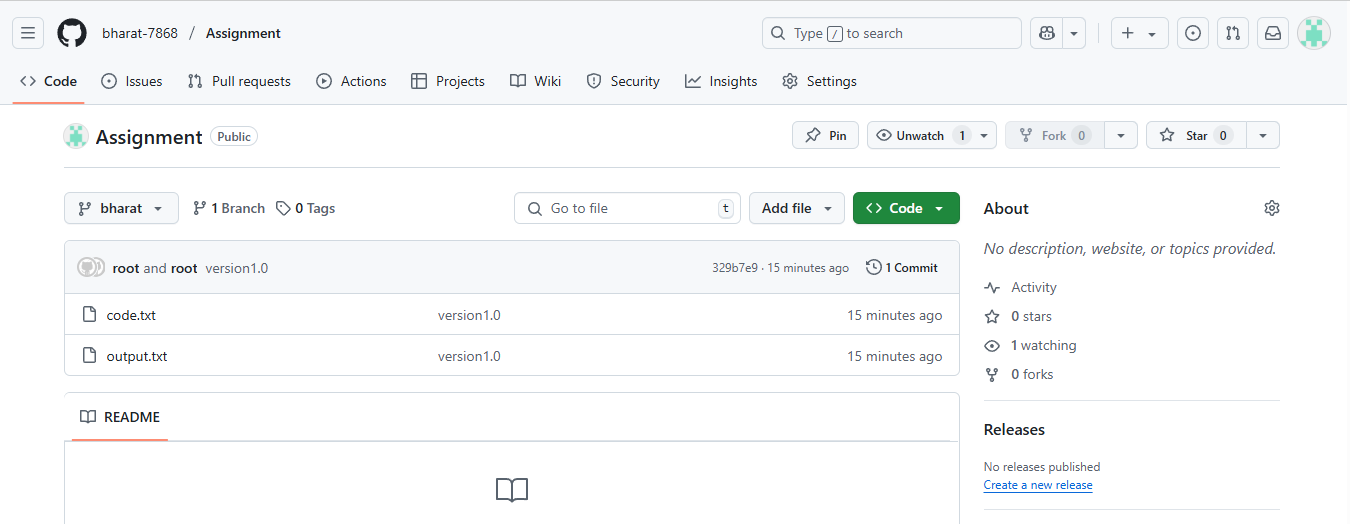
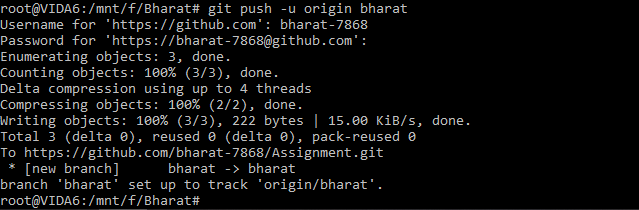
**We can also call commit at version or check point.**

****

From git log we can check the commit details

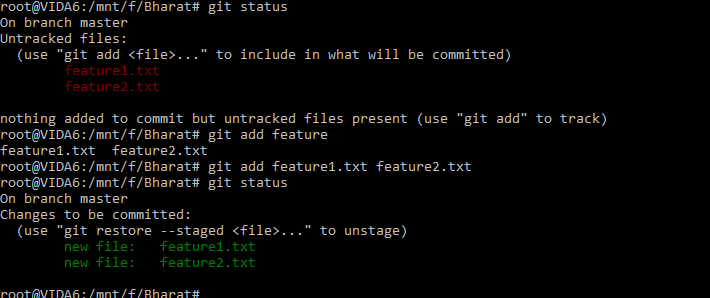


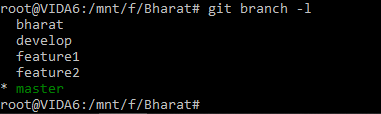
E: And finally push them to GitHub

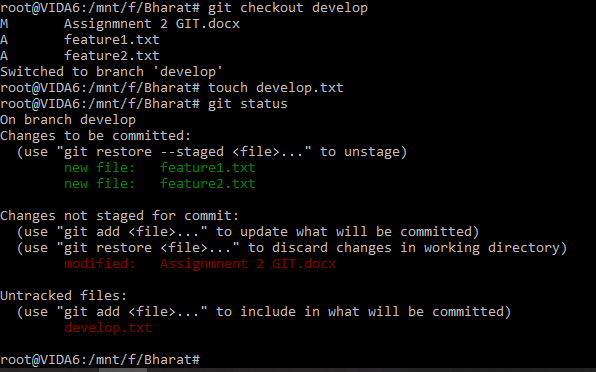
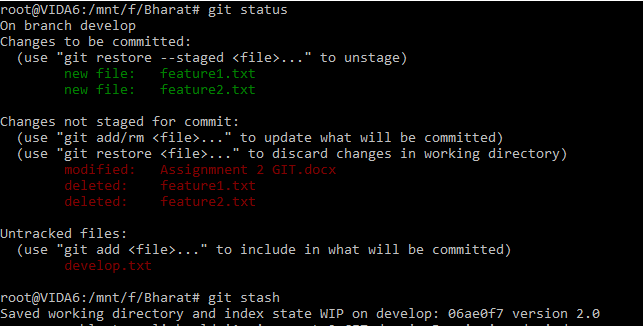
For push any data to git hub repository **git push -u origin brnachname**

**­**

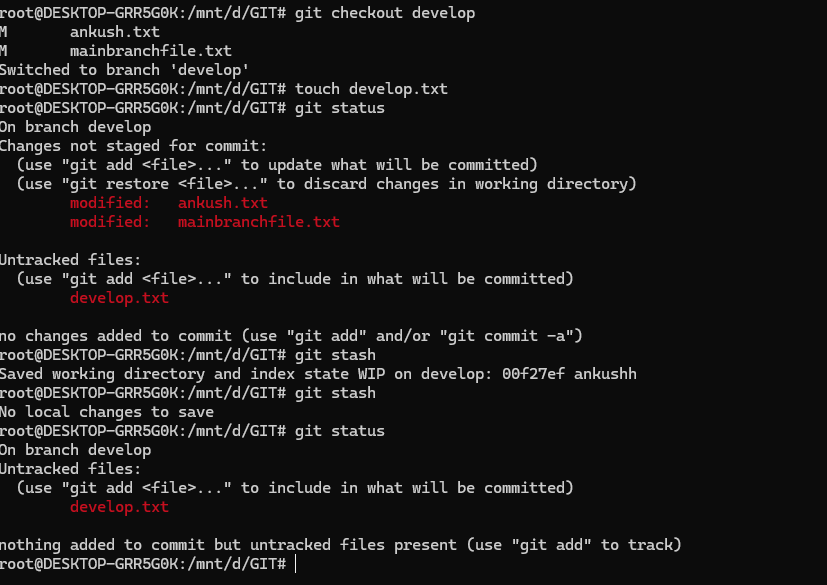
**Q 2: Tasks to Be Performed:**

1. Create a Git working directory with feature1.txt and feature2.txt in the master branch 
2. Create 3 branches develop, feature1 and feature2

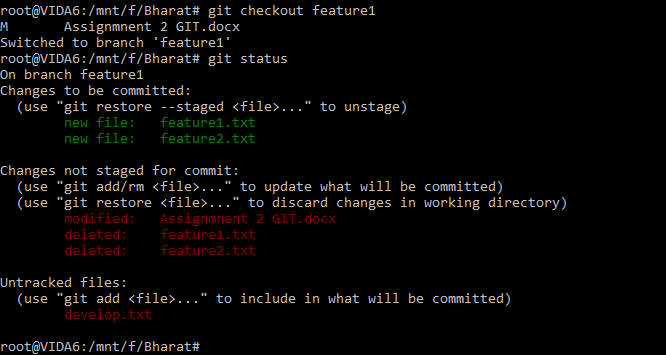


1. In develop branch create develop.txt, do not stage or commit it 
2. Stash this file and check out to feature1 branch 

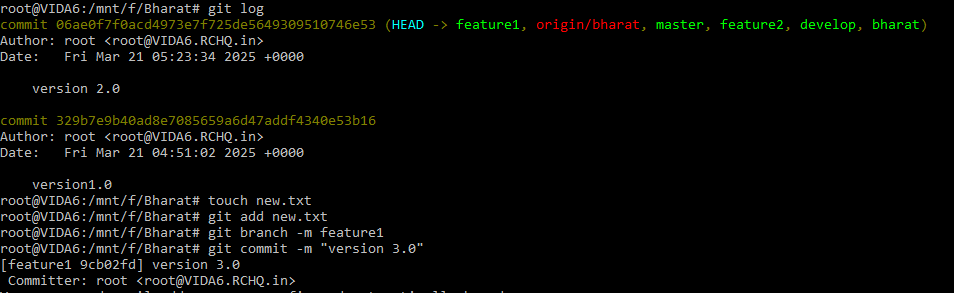
Now we will stash the file as shown in below image

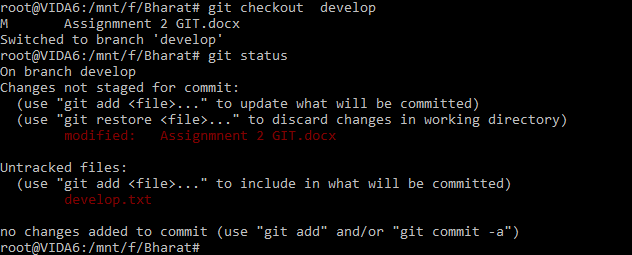


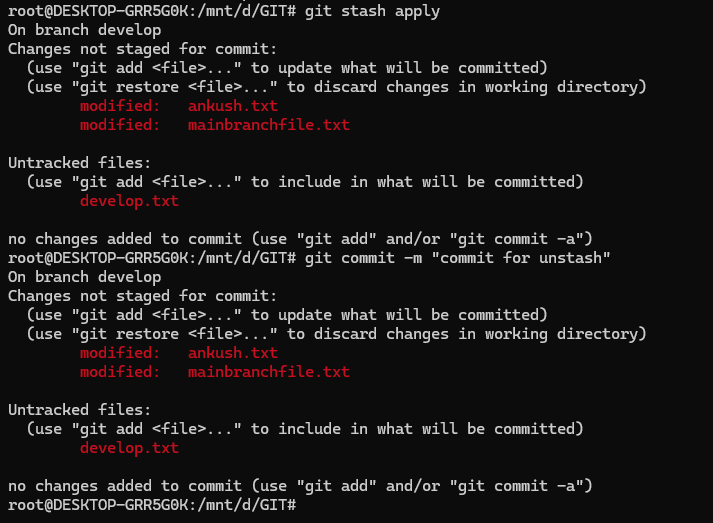
After then we will switch to feature1 branch.



1. Create new.txt file in feature1 branch, stage and commit this file



1. Checkout to develop, unstash this file and commit 

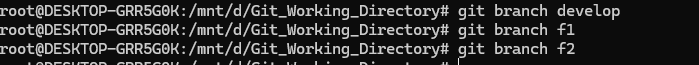
For unstash the file we use command “**git stash apply**” as shown in below image

1. Please submit all the Git commands used to do the above step>

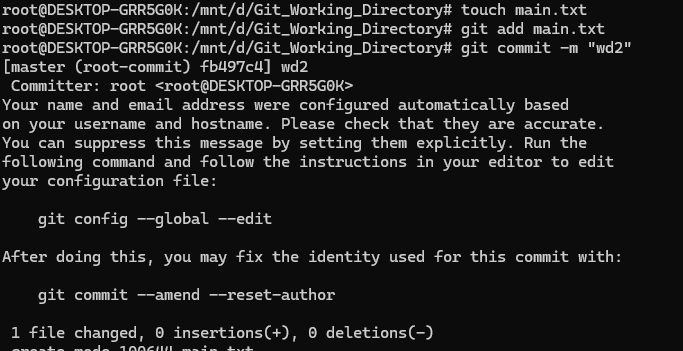
You can check all the commands in the screen shots it self.

**Q3. Tasks to be performed.**

**1. Create a Git working directory, with the following branches:                 Develop                 F1                 f2**

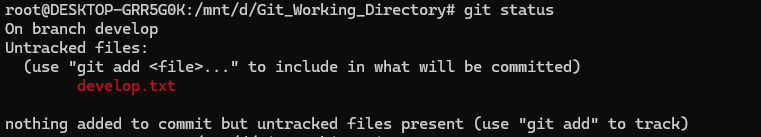
****

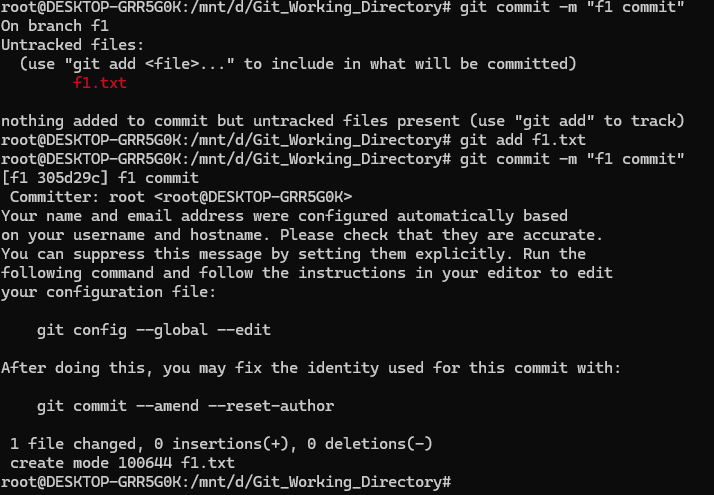
2. In the master branch commit main.txt as shown below



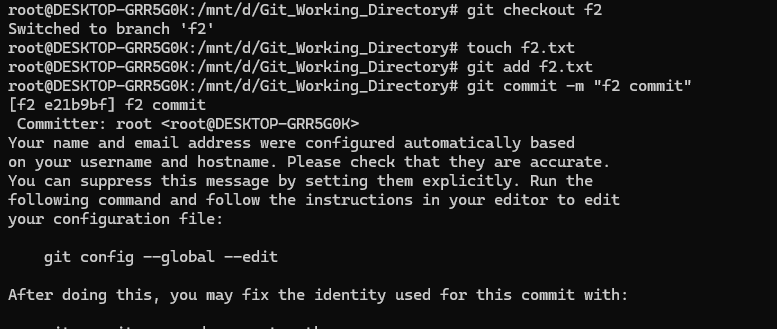
3. Put develop.txt in develop branch, f1.txt and f2.txt in f1 and f2 respectively

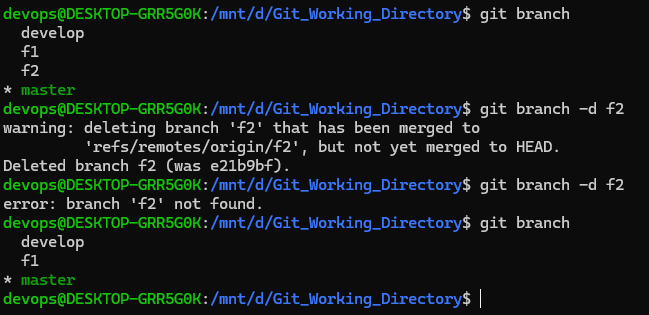
For develop branch



For f1 brnach

For f2 brnach

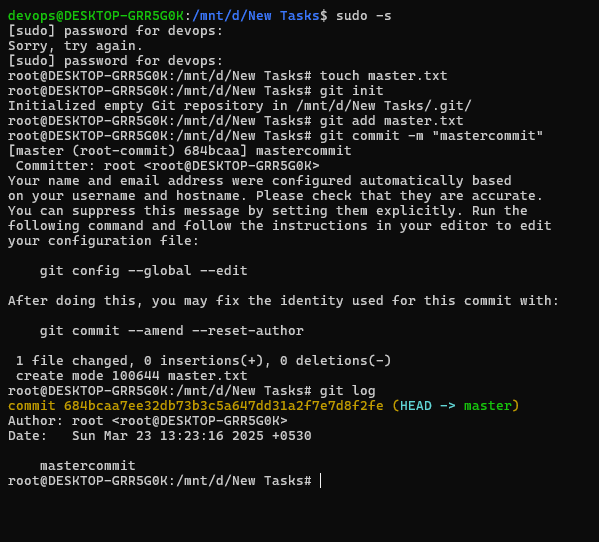


Now we will push all branches to git hub repository Also we have deleted f2 brnachA screenshot of a computer

AI-generated content may be incorrect.

**Q4 Tasks to Be Performed:**

1. Put master.txt on master branch, stage and commit



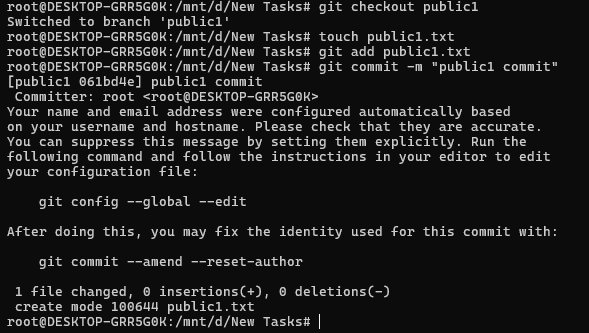
2 Create 3 branches: public 1, public 2 and private

A computer screen with white text

AI-generated content may be incorrect.

3 Put public1.txt on public 1 branch, stage and commit

For public1

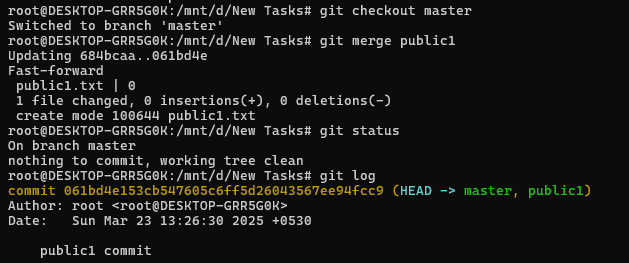


For public2

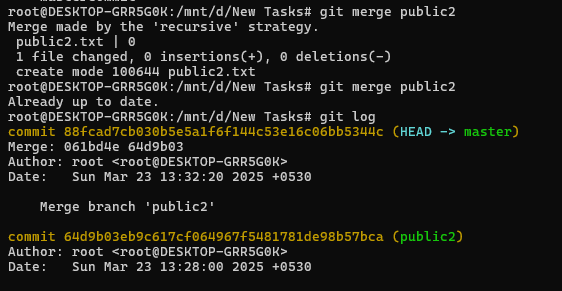
A screenshot of a computer program

AI-generated content may be incorrect.

4 Merge public 1 on master branch

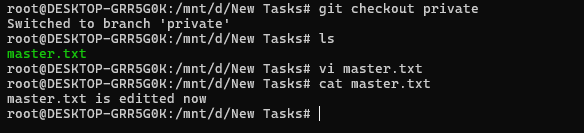


5 Merge public 2 on master branch



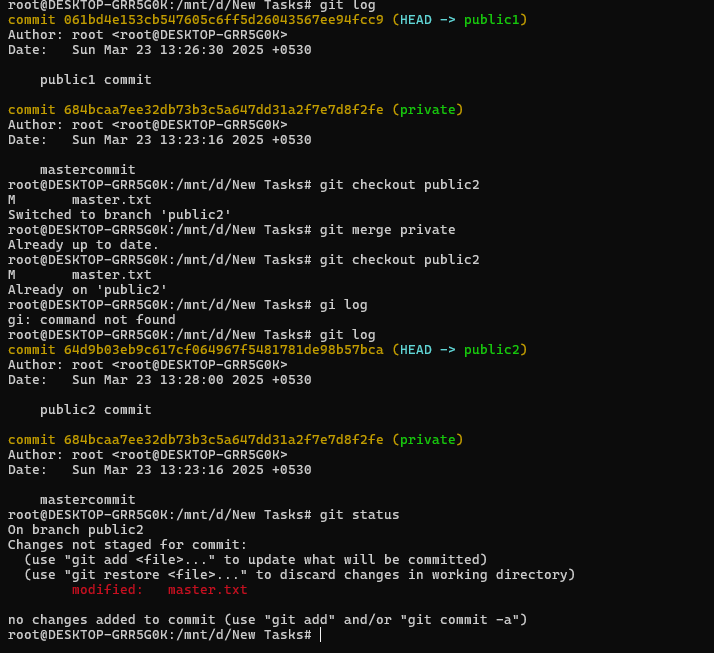
6 Edit master.txt on private branch, stage and commit

For this now we will switch to private branch and edit the file



7 Now update branch public 1 and public 2 with new master code in private

For this we will merge private branch in both public1 and public2



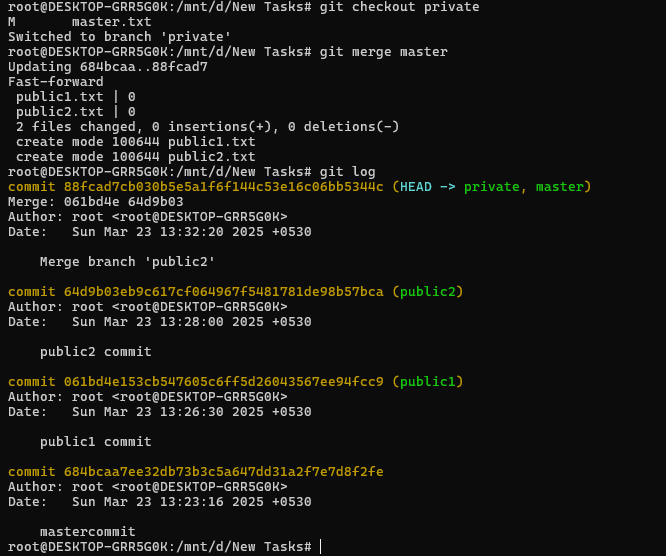
8 Also update new master code on master

git checkout master

git merge private

9 Finally update all the code on the private branch

For this we will not checkout to private branch and merge the master branch



**Q5. Tasks to Be Performed:**

**1. Create a Git Flow workflow architecture on Git**

**2. Create all the required branches**

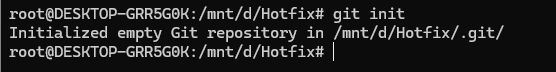
**3. starting from e feature branch, push the branch to the master, following the architecture**

**4. Push an urgent.txt on master using hotfix**

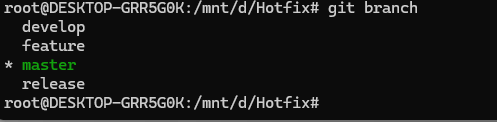
Git Flow is a branching model for Git that helps manage development, releases, and hotfixes systematically. The main branches are:

* **master** → The production branch, only updated through releases or hotfixes.
* **develop** → The main branch for ongoing development.
* **Feature branches (feature/\*)** → For new features.
* **Release branches (release/\*)** → For preparing stable releases.
* **Hotfix branches (hotfix/\*)** → For urgent fixes directly applied to master.

First, initialize Git Flow in your repository:



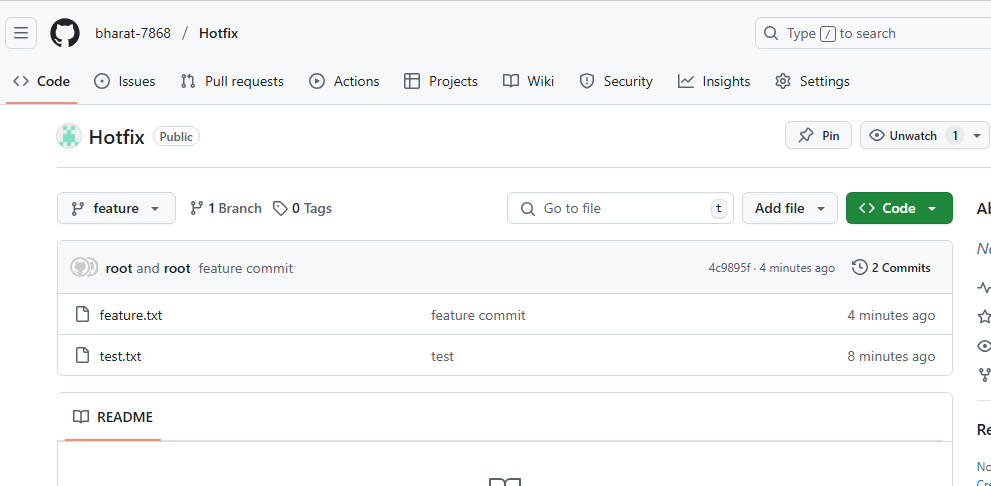
Create Required Branches



Push a Feature Branch to Master (Following Git Flow)

A screen shot of a computer

AI-generated content may be incorrect.

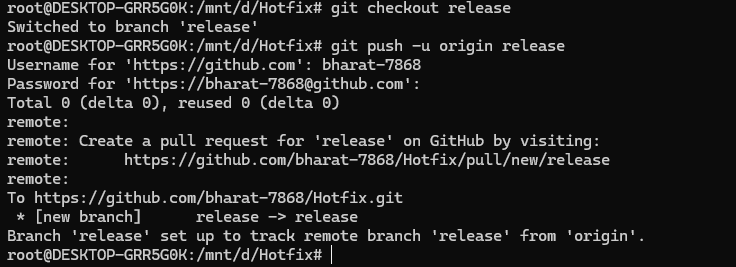


Merge feature branch to develop branch then push



Prepare a Release

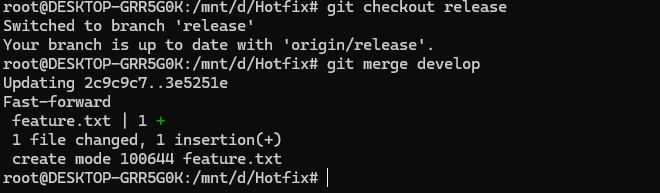
Now we will switch to release branch then we will push to git hub repository



A screenshot of a computer

AI-generated content may be incorrect.

Now we will merge develop branch to the release branch so that we can get the files in release branch as well.



A screenshot of a computer

AI-generated content may be incorrect.

Hence we can see that merging is happening as shown in the above picture.

Thankyou.