

Module 2

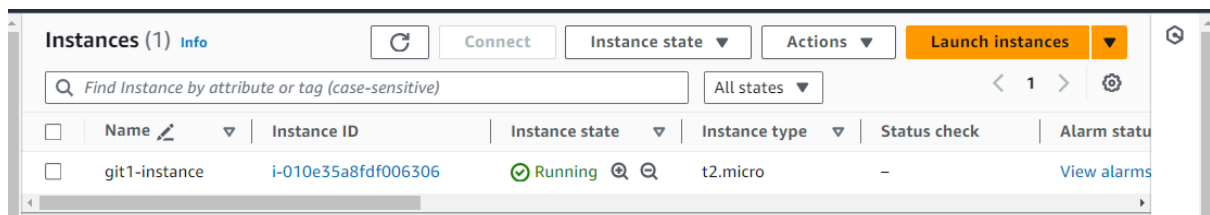
Git Assignment – 1

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

Solutions:

Create 1 Ec2 instance with a proper name and connect to your local terminal



Run the following command:

Cd downloads

git1-kp.pem

```
ssh -i "git1-kp.pem" ubuntu@ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com
```

```
ubuntu@ip-172-31-23-166: ~  
Microsoft Windows [Version 10.0.19045.4291]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Ruchitha>cd downloads  
  
C:\Users\Ruchitha\Downloads>git1-kp.pem  
  
C:\Users\Ruchitha\Downloads>ssh -i "git1-kp.pem" ubuntu@ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com  
ssh: Could not resolve hostname i: No such host is known.  
  
C:\Users\Ruchitha\Downloads>ssh -i "git1-kp.pem" ubuntu@ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com  
The authenticity of host 'ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com (54.169.131.62)' can't be established.  
ECDSA key fingerprint is SHA256:tD5X4Pcz1H4pj/D3kC+0MJiJ2Q5M6Aj6u7JC4pSVMg.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com,54.169.131.62' (ECDSA) to the list of  
known hosts.  
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1008-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/pro  
  
System information as of Thu May  2 16:12:30 UTC 2024  
  
System load:  0.0          Processes:      108  
Usage of /:   23.3% of 6.71GB  Users logged in:  1  
Memory usage: 19%          IPv4 address for enX0: 172.31.23.166  
Swap usage:   0%
```

```
ubuntu@ip-172-31-23-166: ~  
Usage of /:   23.3% of 6.71GB  Users logged in:      1  
Memory usage: 19%          IPv4 address for enX0: 172.31.23.166  
Swap usage:   0%  
  
Expanded Security Maintenance for Applications is not enabled.  
  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
Last login: Thu May  2 16:05:35 2024 from 3.0.5.37  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-23-166:~$
```

Run the command: `sudo hostnamectl set-hostname git1-instance` to change the name of the EC2 instance then, `exit`

```
ubuntu@ip-172-31-23-166:~$ sudo hostnamectl set-hostname git1-instance  
ubuntu@ip-172-31-23-166:~$ exit  
logout  
Connection to ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com closed.  
  
C:\Users\Ruchitha\Downloads>
```

Connect to the Ec2 instance again by running following command

```
ssh -i "git1-kp.pem" ubuntu@ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com
```

```
Connection to ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com closed.

C:\Users\Ruchitha\Downloads>ssh -i "git1-kp.pem" ubuntu@ec2-54-169-131-62.ap-southeast-1.compute.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1008-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Thu May  2 16:30:02 UTC 2024

System load:  0.0          Processes:      109
Usage of /:   23.2% of 6.71GB Users logged in: 1
Memory usage: 19%         IPv4 address for enX0: 172.31.23.166
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu May  2 16:12:31 2024 from 203.92.62.246
ubuntu@git1-instance:~$
```

Successfully connected to the Ec2 instance and the name is also changed.

Check the git version as if it is already installed

```
Last login: Thu May  2 16:12:31 2024 from 203.92.62.246
ubuntu@git1-instance:~$ git --version
git version 2.43.0
ubuntu@git1-instance:~$
```

a) To create a new folder

Create a directory basically means a folder

```
mkdir project1
```

```
ls
```

```
cd project1
```

```
ubuntu@git1-instance:~$ mkdir project1
ubuntu@git1-instance:~$ ls
project1
ubuntu@git1-instance:~$ cd project1
ubuntu@git1-instance:~/project1$
```

You can give git init, init means initialization, it gives some information before starting use git.

```
ubuntu@git1-instance:~/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@git1-instance:~/project1$ ls --a
ls: option '--a' is ambiguous; possibilities: '--all' '--almost-all' '--author'
Try 'ls --help' for more information.
ubuntu@git1-instance:~/project1$ ls -a
.  ..  .git
ubuntu@git1-instance:~/project1$ ls
ubuntu@git1-instance:~/project1$ cd .git
cd.git: command not found
ubuntu@git1-instance:~/project1$ cd .git
ubuntu@git1-instance:~/project1/.git$ ls
HEAD  branches  config  description  hooks  info  objects  refs
ubuntu@git1-instance:~/project1/.git$ cd ..
ubuntu@git1-instance:~/project1$
```

b. Put the following files in the folder

- Code.txt
- Log.txt
- Output.txt

```
ubuntu@git1-instance:~/project1$ touch Code.txt
ubuntu@git1-instance:~/project1$ touch Log.txt
ubuntu@git1-instance:~/project1$ touch Output.txt
ubuntu@git1-instance:~/project1$ nano Code.txt
ubuntu@git1-instance:~/project1$ ubuntu@git1-instance:~/project1$
```

Check the git status

```
ubuntu@git1-instance:~/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@git1-instance:~/project1$
```

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

```
ubuntu@git1-instance:~/project1$ git add Code.txt
ubuntu@git1-instance:~/project1$ git add Output.txt
ubuntu@git1-instance:~/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@git1-instance:~/project1$
```

d. Commit them

```
ubuntu@git1-instance:~/project1$ git commit -m "Code.txt"
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'ubuntu@git1-instance.(none)')

ubuntu@git1-instance:~/project1$ git config --global user.email "ruchitharuthu123@gmail.com"
ubuntu@git1-instance:~/project1$ git config --global user.name "RuchithaBtGowda"
ubuntu@git1-instance:~/project1$ git commit -m "Adding Code.txt and Output.txt"
[master (root-commit) 1959664] Adding Code.txt and Output.txt
 2 files changed, 2 insertions(+)
 create mode 100644 Code.txt
 create mode 100644 Output.txt
ubuntu@git1-instance:~/project1$ git log
commit 1959664570a895fecalcee53ab3ff3ecfe4a5921 (HEAD -> master)
Author: RuchithaBtGowda <ruchitharuthu123@gmail.com>
Date: Thu May 2 17:50:47 2024 +0000

    Adding Code.txt and Output.txt
ubuntu@git1-instance:~/project1$ git log --oneline
1959664 (HEAD -> master) Adding Code.txt and Output.txt
ubuntu@git1-instance:~/project1$ git status
```

```

ubuntu@git1-instance:~/project1$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
       Code.txt
       Log.txt

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@git1-instance:~/project1$ git log
commit 5a410f84a0369a845b138463858b5d9e46acaa77 (HEAD -> master)
Author: RuchithaBtGowda <ruchitharuthu123@gmail.com>

```

e. And finally push them to GitHub



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/RuchithaBtGowda/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@ip-172-31-30-29:~/project1$ git remote add origin https://github.com/RuchithaBtGowda/project1.git
ubuntu@ip-172-31-30-29:~/project1$ git push origin master
Username for 'https://github.com': RuchithaBtGowda
Password for 'https://RuchithaBtGowda@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 326 bytes | 326.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/RuchithaBtGowda/project1.git
 * [new branch]      master -> master
ubuntu@ip-172-31-30-29:~/project1$

```



project1 Public



Pin



Unwatch

1



master



1 Branch

0 Tags

Go to file



Add file

Code



Ubuntu Initial commit: Code.txt and Output.txt

f1c6709 · 8 minutes ago

1 Commits



Code.txt

Initial commit: Code.txt and Output.txt

8 minutes ago



Output.txt

Initial commit: Code.txt and Output.txt

8 minutes ago

2. Please share the commands for the above points

```
mkdir project1    #make directory which means to create the directory
ls                #list
cd project1       #to go inside the directory
touch Code.txt    #to create the file
touch Log.txt
touch Output.txt
nano Code.txt     #to edit the file
nano Output.txt
:wq              #save and exit
ls
git init          #git initialization before starting git commands
git add Code.txt Output.txt
git status
git commit -m "Initial commit: Code.txt and Output.txt"
git log
git log --oneline #to create log in one line
git remote add origin URL    #add the files to remote repository
git push origin master
username
password: token
```

Module 2

Git Assignment – 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
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

Solutions:

1. Create a Git working directory with feature1.txt and feature2.txt in the master branch

```
ubuntu@ip-172-31-30-29:~$ mkdir project2
ubuntu@ip-172-31-30-29:~$ ls
project1  project2
ubuntu@ip-172-31-30-29:~$ cd project2
ubuntu@ip-172-31-30-29:~/project2$ touch feature1.txt feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ ls
feature1.txt  feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ nano feature1.txt
ubuntu@ip-172-31-30-29:~/project2$ nano feature2.txt
```

Git init

```
ubuntu@ip-172-31-30-29:~/project2$ 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/project2/.git/
```


Create any other branch because master branch is already used

```
ubuntu@ip-172-31-30-29:~/project2$ git branch -m main
ubuntu@ip-172-31-30-29:~/project2$ git branch
ubuntu@ip-172-31-30-29:~/project2$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    feature1.txt
    feature2.txt

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

Add and commit the created files

```
ubuntu@ip-172-31-30-29:~/project2$ git add feature1.txt feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ git commit -m "added feature1.txt and feature2.txt"
[main (root-commit) eefbc8d] added feature1.txt and feature2.txt
Committer: Ubuntu <ubuntu@ip-172-31-30-29.ap-southeast-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

2 files changed, 2 insertions(+)
create mode 100644 feature1.txt
create mode 100644 feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ git status
```

Check git status and git log

```
ubuntu@ip-172-31-30-29:~/project2$ git status
On branch main
nothing to commit, working tree clean
ubuntu@ip-172-31-30-29:~/project2$ git log
commit eefbc8da36cf1803f729712e4b7e5ea18cf045d0 (HEAD -> main)
Author: Ubuntu <ubuntu@ip-172-31-30-29.ap-southeast-1.compute.internal>
Date:   Sun May 5 07:14:07 2024 +0000

    added feature1.txt and feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ git log --oneline
eefbc8d (HEAD -> main) added feature1.txt and feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ git status
On branch main
nothing to commit, working tree clean
```

2. Create 3 branches develop, feature1 and feature2

```
ubuntu@ip-172-31-30-29:~/project2$ git branch
* main
ubuntu@ip-172-31-30-29:~/project2$ git branch develop
ubuntu@ip-172-31-30-29:~/project2$ git branch feature1
ubuntu@ip-172-31-30-29:~/project2$ git branch feature2
ubuntu@ip-172-31-30-29:~/project2$ git branch
  develop
  feature1
  feature2
* main
```

3. In develop branch create develop.txt, do not stage or commit it

```
ubuntu@ip-172-31-30-29:~/project2$ git checkout develop
Switched to branch 'develop'
ubuntu@ip-172-31-30-29:~/project2$ git branch
* develop
  feature1
  feature2
  main
```

```
ubuntu@ip-172-31-30-29:~/project2$ touch develop.txt
ubuntu@ip-172-31-30-29:~/project2$ nano develop.txt
```

```
ubuntu@ip-172-31-30-29:~/project2$ ls
develop.txt  feature1.txt  feature2.txt
```

4. Stash this file and check out to feature1 branch

```
ubuntu@ip-172-31-30-29:~/project2$ git add develop.txt
ubuntu@ip-172-31-30-29:~/project2$ git stash save "stashing develop.txt"
Saved working directory and index state On develop: stashing develop.txt
ubuntu@ip-172-31-30-29:~/project2$ ls
feature1.txt  feature2.txt
```

```
ubuntu@ip-172-31-30-29:~/project2$ git checkout feature1
Switched to branch 'feature1'
ubuntu@ip-172-31-30-29:~/project2$ git branch
  develop
* feature1
  feature2
  main
ubuntu@ip-172-31-30-29:~/project2$ touch new.txt
```

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

```
ubuntu@ip-172-31-30-29:~/project2$ git add new.txt
ubuntu@ip-172-31-30-29:~/project2$ git commit -m "Added new.txt file to feature1 branch"
[feature1 e86dc9f] Added new.txt file to feature1 branch
Committer: Ubuntu <ubuntu@ip-172-31-30-29.ap-southeast-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 new.txt
ubuntu@ip-172-31-30-29:~/project2$
```

6. Checkout to develop, unstash this file and commit

```
ubuntu@ip-172-31-30-29:~/project2$ git branch
  develop
* feature1
  feature2
  main
ubuntu@ip-172-31-30-29:~/project2$ git checkout develop
Switched to branch 'develop'
```

Unstashing the file

```
ubuntu@ip-172-31-30-29:~/project2$ 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} (2e060bee493fb3da3c740a812c00e98bc19bc0fe)
ubuntu@ip-172-31-30-29:~/project2$ ls
develop.txt  feature1.txt  feature2.txt
ubuntu@ip-172-31-30-29:~/project2$
```

Ls command

```
ubuntu@ip-172-31-30-29:~/project2$ ls
develop.txt  feature1.txt  feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ git status
On branch develop
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   develop.txt
```

Commit develop.txt file in develop branch

```
ubuntu@ip-172-31-30-29:~/project2$ git commit -m "Added develop.txt file to develop branch"
[develop 89b1e7b] Added develop.txt file to develop branch
Committer: Ubuntu <ubuntu@ip-172-31-30-29.ap-southeast-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+)
create mode 100644 develop.txt
ubuntu@ip-172-31-30-29:~/project2$
```

Check git status branch

```
ubuntu@ip-172-31-30-29:~/project2$ git status
On branch develop
nothing to commit, working tree clean
ubuntu@ip-172-31-30-29:~/project2$ git log
commit 89b1e7b371fd7d4c71ba76f5e98703531ae5b818 (HEAD -> develop)
Author: Ubuntu <ubuntu@ip-172-31-30-29.ap-southeast-1.compute.internal>
Date:   Sun May 5 08:02:16 2024 +0000

    Added develop.txt file to develop branch

commit eefbc8da36cf1803f729712e4b7e5ea18cf045d0 (main, feature2)
Author: Ubuntu <ubuntu@ip-172-31-30-29.ap-southeast-1.compute.internal>
Date:   Sun May 5 07:14:07 2024 +0000

    added feature1.txt and feature2.txt
ubuntu@ip-172-31-30-29:~/project2$
```

```
ubuntu@ip-172-31-30-29:~/project2$ git branch
* develop
  feature1
  feature2
  main
ubuntu@ip-172-31-30-29:~/project2$ git checkout main
Switched to branch 'main'
ubuntu@ip-172-31-30-29:~/project2$ ls
feature1.txt  feature2.txt
ubuntu@ip-172-31-30-29:~/project2$ git checkout feature1
Switched to branch 'feature1'
ubuntu@ip-172-31-30-29:~/project2$ ls
feature1.txt  feature2.txt  new.txt
ubuntu@ip-172-31-30-29:~/project2$ git checkout feature2
Switched to branch 'feature2'
ubuntu@ip-172-31-30-29:~/project2$ ls
feature1.txt  feature2.txt
```

7. Please submit all the Git commands used to do the above steps

```
mkdir project2
```

```
ls
```

```
cd project2
```

```
touch feature1.txt feature2.txt
```

```
nano feature1.txt
```

```
nano feature2.txt
```

```
git init
```

```
git branch -m main
```

```
git add feature1.txt feature2.txt
```

```
git commit -m "added feature1.txt and feature2.txt"
```

```
git status
```

```
git branch develop
```

```
git branch feature1
```

```
git branch feature2
```

```
git checkout develop
```

```
touch develop.txt
```

```
nano develop.txt
```

```
git add develop.txt
```

```
git stash save "stashing develop.txt"
```

```
git checkout feature1
```

```
touch new.txt
```

```
git add new.txt
```

```
git commit -m "Added new.txt file to feature1 branch"
```

```
git checkout develop
```

ls

git stash pop

git status

git commit -m "Added develop.txt file to develop branch"

=====

Module 2

Git Assignment – 3

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

Solution:

Launch an Instance

Name: Git-demo

Instances (10) Info

Connect

Instance state ▾

Actions ▾

Launch instances

Find Instance by attribute or tag (case-sensitive)

All states ▾






<

1

>

<input type="checkbox"/>	Name <div></div> ▾	Instance ID	Instance state <div></div> ▲	Instance type ▾	Status check	Alarm status	Availa
<input type="checkbox"/>	Git-demo	i-01327ba8cea8d55a4	<div><div></div>Running</div> <div><div></div><div></div></div>	t2.micro	<div><div></div>2/2 checks passed</div> View alarms +		ap-so

Login to the client server through SSH client

EC2 Instance Connect	Session Manager	SSH client	EC2 serial console
<p>Instance ID</p> <p> i-01327ba8cea8d55a4 (Git-demo)</p> <ol style="list-style-type: none">1. Open an SSH client.2. Locate your private key file. The key used to launch this instance is git1-kp.pem3. Run this command, if necessary, to ensure your key is not publicly viewable.  <code>chmod 400 "git1-kp.pem"</code>4. Connect to your instance using its Public DNS:  <code>ec2-18-139-219-142.ap-southeast-1.compute.amazonaws.com</code> <p> Command copied</p> <p> <code>ssh -i "git1-kp.pem" ubuntu@ec2-18-139-219-142.ap-southeast-1.compute.amazonaws.com</code></p>			

Change hostname by using command –

`sudo hostname set-hostname git-demo`

```
ubuntu@git-demo: ~  
ubuntu@git-demo:~$
```

Create a Directory and go inside the directory initializing git (git init)

```
ubuntu@git-demo:~$ mkdir gitproject  
ubuntu@git-demo:~$ ls  
gitproject  
ubuntu@git-demo:~$ cd gitproject  
ubuntu@git-demo:~/gitproject$ 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/gitproject/.git/  
ubuntu@git-demo:~/gitproject$
```


Create main.txt file in the master branch and commit it

```
ubuntu@git-demo:~/gitproject$ touch main.txt
ubuntu@git-demo:~/gitproject$ git add main.txt
ubuntu@git-demo:~/gitproject$ git commit -m "Adding main.txt file"
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'ubuntu@git-demo.(none)')
ubuntu@git-demo:~/gitproject$ git config --global user.email "ruchitharuthu123@gmail.com"
ubuntu@git-demo:~/gitproject$ git config --global user.name "RuchithaBtGowda"
ubuntu@git-demo:~/gitproject$ git commit -m "Adding main.txt file"
[master (root-commit) eca2fd5] Adding main.txt file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 main.txt
ubuntu@git-demo:~/gitproject$
```

Perform task 1 creating the branches develop, f1 and f2

```
ubuntu@git-demo:~/gitproject$ git branch
* master
ubuntu@git-demo:~/gitproject$ git branch develop
ubuntu@git-demo:~/gitproject$ git branch f1
ubuntu@git-demo:~/gitproject$ git branch f2
ubuntu@git-demo:~/gitproject$ git branch
  develop
  f1
  f2
* master
```

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

```
ubuntu@git-demo:~/gitproject$ git checkout develop
Switched to branch 'develop'
ubuntu@git-demo:~/gitproject$ touch develop.txt
ubuntu@git-demo:~/gitproject$ git add develop.txt
ubuntu@git-demo:~/gitproject$ git commit -m "Adding develop.txt file"
[develop e5f0cb5] Adding develop.txt file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 develop.txt
```

```
ubuntu@git-demo:~/gitproject$ git checkout f1
Switched to branch 'f1'
ubuntu@git-demo:~/gitproject$ touch f1.txt
ubuntu@git-demo:~/gitproject$ git add f1.txt
ubuntu@git-demo:~/gitproject$ git commit -m "Adding f1.txt file"
[f1 c111338] Adding f1.txt file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 f1.txt
```

```
ubuntu@git-demo:~/gitproject$ git checkout f2
Switched to branch 'f2'
ubuntu@git-demo:~/gitproject$ touch f2.txt
ubuntu@git-demo:~/gitproject$ git add f2.txt
ubuntu@git-demo:~/gitproject$ git commit -m "Adding f2.txt file"
[f2 f52a373] Adding f2.txt file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 f2.txt
ubuntu@git-demo:~/gitproject$ git branch
  develop
  f1
* f2
  master
```


Create new repository and 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 * Repository name *


 RuchithaBtGowda /

✔ gitdemo is available.

Great repository names are short and memorable. Need inspiration? How about super-parakeet ?



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.

To perform task 4: Push all these branches to GitHub, Copy the generated link.

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

 Set up in Desktop or HTTPS SSH 

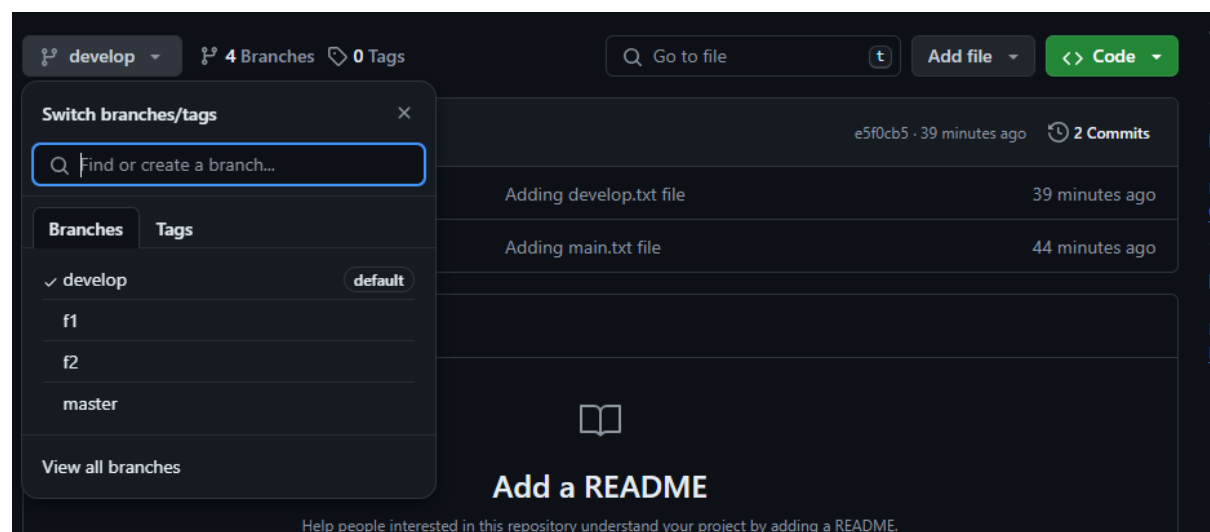
Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

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

You can also give `git push origin --all`

```
ubuntu@git-demo:~/gitproject$ git push origin master develop f1 f2
Username for 'https://github.com': RuchithaBtGowda
Password for 'https://RuchithaBtGowda@github.com':
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 765 bytes | 765.00 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/RuchithaBtGowda/gitdemo.git
 * [new branch]      master -> master
 * [new branch]      develop -> develop
 * [new branch]      f1 -> f1
 * [new branch]      f2 -> f2
```

We can see the branches in git hub

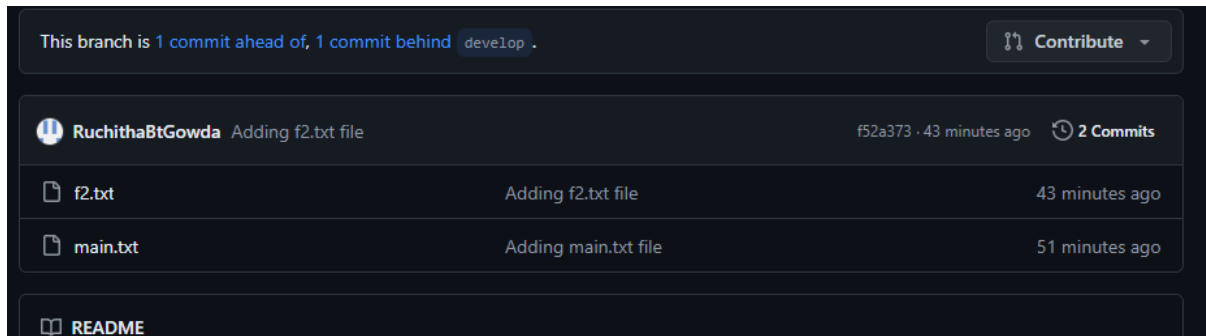


As per the task 5 :On local delete f2 branch,

We need to delete f2 branch in local so give the cmd: `git branch -D f2` it will get deleted successfully

```
ubuntu@git-demo:~/gitproject$ git checkout master
Switched to branch 'master'
ubuntu@git-demo:~/gitproject$ git branch -D f2
Deleted branch f2 (was f52a373).
ubuntu@git-demo:~/gitproject$
```

But it is not deleted in the remote repository

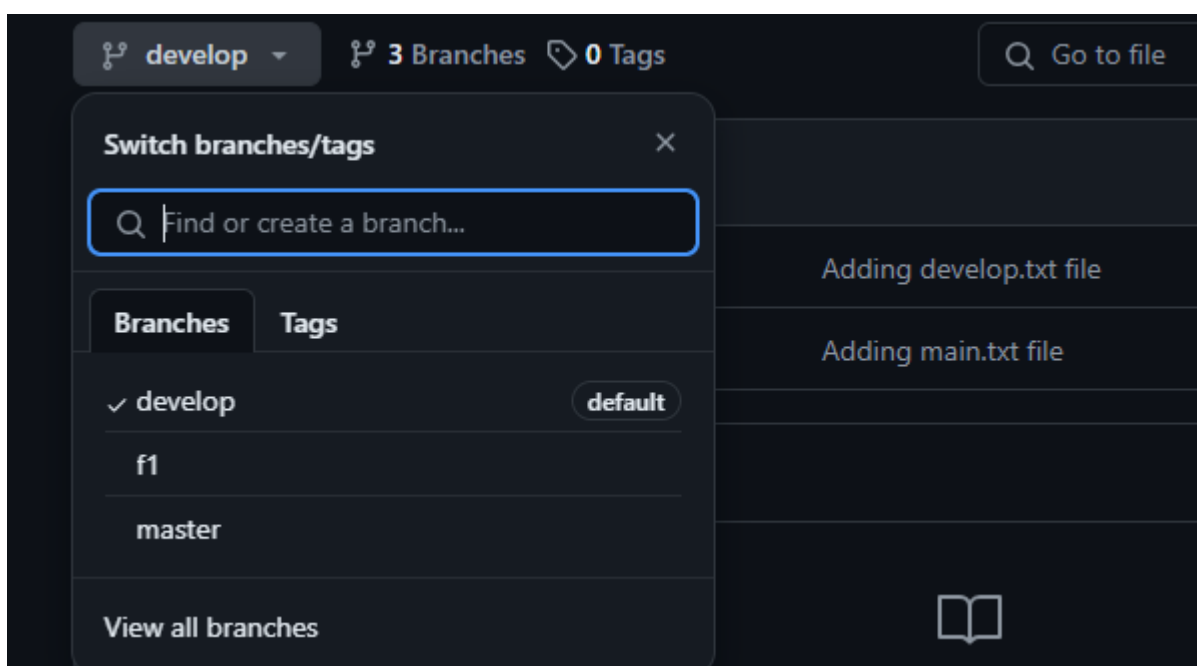


As per task 6: Delete the same branch on GitHub as well

Give the command: `git push origin --delete f2`, to delete in the git hub as well.

```
ubuntu@git-demo:~/gitproject$ git push origin --delete f2
Username for 'https://github.com': RuchithaBtGowda
Password for 'https://RuchithaBtGowda@github.com':
To https://github.com/RuchithaBtGowda/gitdemo.git
- [deleted]          f2
```

F2 branch is deleted successfully.



Module 2

Git Assignment – 4

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

Solutions:

Launch an instance → Git-Asgn

Instances (1) Info							Refresh	Connect	Instance state ▼	Actions
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>									All states ▼	
<input type="checkbox"/>	Name ✎ ▼	Instance ID	Instance state ▼	Instance type ▼	Status check	Al				
<input type="checkbox"/>	Git-Asgn	i-00f786500cde297ad	✓ Running ⓘ 🔍	t2.micro	🕒 Initializing	Vi				

Connect to ssh client

```
System load: 0.34          Processes: 106
Usage of /: 23.2% of 6.71GB Users logged in: 0
Memory usage: 21%          IPv4 address for enX0: 172.31.25.198
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Jun 23 05:27:52 2024 from 114.79.174.77
ubuntu@gitasgn:~$
```

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

Create new directory git4 \$ mkdir git4

go inside git4 and Initialize the git

```
ubuntu@gitasgn:~$ mkdir git4
ubuntu@gitasgn:~$ ls
git4
ubuntu@gitasgn:~$ cd git4
ubuntu@gitasgn:~/git4$ 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/git4/.git/
ubuntu@gitasgn:~/git4$
```

Create master.txt, stage and commit it

```
ubuntu@gitasgn:~/git4$ touch master.txt
ubuntu@gitasgn:~/git4$ ls
master.txt
ubuntu@gitasgn:~/git4$ git add .
ubuntu@gitasgn:~/git4$ git status
On branch master

No commits yet

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

```
ubuntu@gitasgn:~/git4$ git commit -m "m file"
[master (root-commit) 6f45c3b] m file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 master.txt
```

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

```
ubuntu@gitasgn:~/git4$ git branch
* master
ubuntu@gitasgn:~/git4$ git branch p1
ubuntu@gitasgn:~/git4$ git branch p2
ubuntu@gitasgn:~/git4$ git branch p
ubuntu@gitasgn:~/git4$ git branch
* master
  p
  p1
  p2
ubuntu@gitasgn:~/git4$
```

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

```
ubuntu@gitasgn:~/git4$ git checkout p1
Switched to branch 'p1'
ubuntu@gitasgn:~/git4$ touch p1.txt
ubuntu@gitasgn:~/git4$ ls
master.txt  p1.txt
ubuntu@gitasgn:~/git4$ git add .
ubuntu@gitasgn:~/git4$ git commit -m "p1.txt"
[p1 5e185ce] p1.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 p1.txt
ubuntu@gitasgn:~/git4$
```

4. Merge public 1 on master branch

When you need to merge, you have to checkout on which branch you are merging, so here we are merging on master branch

Checkout to master branch and merge p1

```
ubuntu@gitasgn:~/git4$ git checkout master
Switched to branch 'master'
ubuntu@gitasgn:~/git4$ git merge p1
Updating 6f45c3b..5e185ce
Fast-forward
 p1.txt | 0
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 p1.txt
```

5. Merge public 2 on master branch

Whenever you are merging of the branch without any new commit it shows up to date.

```
ubuntu@gitasgn:~/git4$ git merge p2
Already up to date.
```

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

Now checkout private branch to edit master branch

```
ubuntu@gitasgn:~/git4$ git checkout p
Switched to branch 'p'
ubuntu@gitasgn:~/git4$ ls
master.txt
ubuntu@gitasgn:~/git4$ sudo nano master.txt
ubuntu@gitasgn:~/git4$ sudo cat master.txt
This is the file on private branch
```

Stage and commit it.

```
ubuntu@gitasgn:~/git4$ git add .
ubuntu@gitasgn:~/git4$ git commit -m "p file"
[p 55ad09f] p file
1 file changed, 1 insertion(+)
```

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

We are merging private on public1 so checkout to p1

```
ubuntu@gitasgn:~/git4$ git checkout p1
Switched to branch 'p1'
ubuntu@gitasgn:~/git4$ ls
master.txt  p1.txt
ubuntu@gitasgn:~/git4$ sudo cat master.txt
```

Master.txt is empty now

When you give cmd “\$ git merge p” to merge private to public1, Merge conflict arises give a message we are merging on p1

Save and exit

```
Merge branch 'p' into p1
We are merging on p1
# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
```

Successfully merged private to public1

```
ubuntu@gitasgn:~/git4$ git merge p
Merge made by the 'ort' strategy.
Merge made by the 'ort' strategy.
 master.txt | 1 +
 1 file changed, 1 insertion(+)
ubuntu@gitasgn:~/git4$
```

Updated master file present in public1 branch

```
ubuntu@gitasgn:~/git4$ sudo cat master.txt
This is the file on private branch
```

Similarly, checkout to p2 branch and merge private branch

```
ubuntu@gitasgn:~/git4$ git checkout p2
Switched to branch 'p2'
ubuntu@gitasgn:~/git4$ ls
master.txt
ubuntu@gitasgn:~/git4$ sudo cat master.txt
ubuntu@gitasgn:~/git4$
```

Here merge conflict did not arrive because we already gave the reason for merging from private branch.

```
ubuntu@gitasgn:~/git4$ git merge p
Updating 6f45c3b..55ad09f
Fast-forward
 master.txt | 1 +
 1 file changed, 1 insertion(+)
ubuntu@gitasgn:~/git4$ sudo cat master.txt
This is the file on private branch
ubuntu@gitasgn:~/git4$
```

8. Also update new master code on master

We need to checkout master and merge private branch

```
ubuntu@gitasgn:~/git4$ git checkout master
Switched to branch 'master'
ubuntu@gitasgn:~/git4$ ls
master.txt  p1.txt
ubuntu@gitasgn:~/git4$ sudo cat master.txt
```

Here we will get the merge conflict because it is parent branch, as master.txt file is created in master branch

```
Merge branch 'p'
We want this merge
# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
```

You can the edited file present in master branch

```
ubuntu@gitasgn:~/git4$ sudo cat master.txt
This is the file on private branch
ubuntu@gitasgn:~/git4$
```

9. Finally update all the code on the private branch

You need to merge branch to private where all the codes are present.

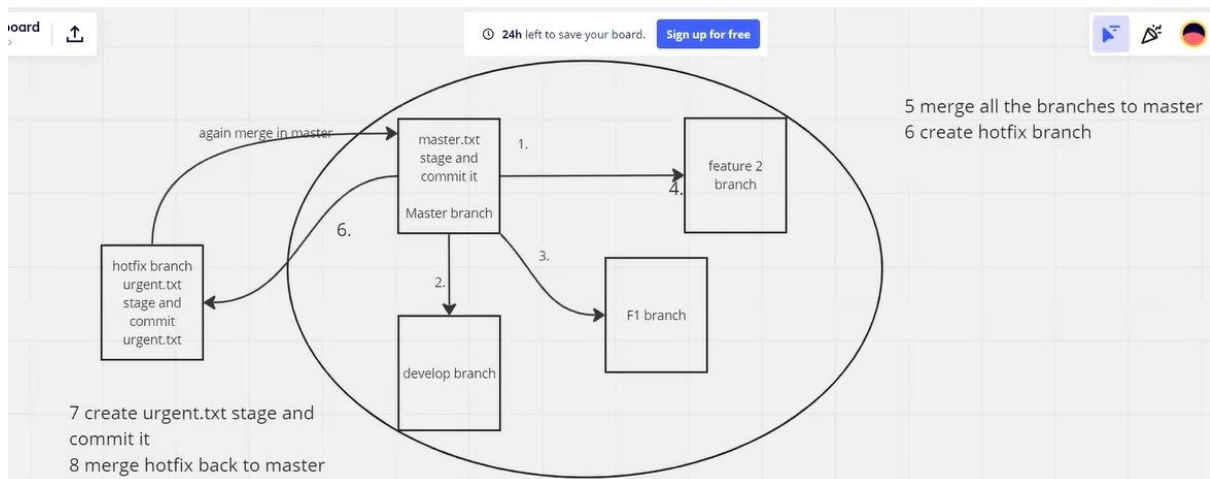
```
ubuntu@gitasgn:~/git4$ git checkout p
Switched to branch 'p'
ubuntu@gitasgn:~/git4$ ls
master.txt
ubuntu@gitasgn:~/git4$ git merge master
Updating 55ad09f..be65205
Fast-forward
 p1.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 p1.txt
ubuntu@gitasgn:~/git4$ ls
master.txt  p1.txt
```

Git Assignment – 5

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

Solutions:



Create new directory git5

Create master.txt file in that

```
ubuntu@gitasgn:~$ mkdir git5
ubuntu@gitasgn:~$ ls
git5
ubuntu@gitasgn:~$ cd git5
ubuntu@gitasgn:~/git5$ touch master.txt
ubuntu@gitasgn:~/git5$ ls
master.txt
```

Initialize the git before starting the git command

```
ubuntu@gitasgn:~/git5$ 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/git5/.git/
```

Stage and commit master.txt file

```
ubuntu@gitasgn:~/git5$ git add .
ubuntu@gitasgn:~/git5$ git status
On branch master

No commits yet

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

ubuntu@gitasgn:~/git5$ git commit -m "master.txt"
[master (root-commit) 2e36be7] master.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 master.txt
```

Master branch is created and created all other branches – develop, feature1 and feaure2

```
ubuntu@gitasgn:~/git5$ git branch
* master
ubuntu@gitasgn:~/git5$ git branch develop
ubuntu@gitasgn:~/git5$ git branch feature1
ubuntu@gitasgn:~/git5$ git branch feature2
```

Add some files in develop branch a.txt and b.txt

Add and commit them

```
ubuntu@gitasgn:~/git5$ git checkout develop
Switched to branch 'develop'
ubuntu@gitasgn:~/git5$ touch a.txt
ubuntu@gitasgn:~/git5$ touch b.txt
```

Add files in feature1 branch p.txt q.txt r.txt s.txt

add and commit them

```
ubuntu@gitasgn:~/git5$ git checkout feature1
Already on 'feature1'
ubuntu@gitasgn:~/git5$ touch p.txt
ubuntu@gitasgn:~/git5$ touch q.txt
ubuntu@gitasgn:~/git5$ touch r.txt
ubuntu@gitasgn:~/git5$ touch s.txt
```

Add files in feature2 branch x.txt y.txt and z.txt

Add and commit them all

```
ubuntu@gitasgn:~/git5$ git checkout feature2
Switched to branch 'feature2'
ubuntu@gitasgn:~/git5$ touch x.txt
ubuntu@gitasgn:~/git5$ touch y.txt
ubuntu@gitasgn:~/git5$ touch z.txt
ubuntu@gitasgn:~/git5$ git add .
ubuntu@gitasgn:~/git5$ git status
On branch feature2
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   x.txt
    new file:   y.txt
    new file:   z.txt
```

```
ubuntu@gitasgn:~/git5$ git commit .
[feature2 4d54f72] I need to commit this file
3 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 x.txt
create mode 100644 y.txt
create mode 100644 z.txt
ubuntu@gitasgn:~/git5$ ls
master.txt  x.txt  y.txt  z.txt
```

Checkout to master branch and merge develop branch

```
ubuntu@gitasgn:~/git5$ git merge develop
Updating 2e36be7..321652f
Fast-forward
 a.txt | 0
 b.txt | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 a.txt
 create mode 100644 b.txt
ubuntu@gitasgn:~/git5$ ls
a.txt b.txt master.txt
```

Then merge feature1 branch to master branch

```
ubuntu@gitasgn:~/git5$ git merge feature1
Merge made by the 'ort' strategy.
 p.txt | 0
 q.txt | 0
 r.txt | 0
 s.txt | 0
 4 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 p.txt
 create mode 100644 q.txt
 create mode 100644 r.txt
 create mode 100644 s.txt
ubuntu@gitasgn:~/git5$ ls
a.txt b.txt master.txt p.txt q.txt r.txt s.txt
```

Merge feature2 branch to master branch

```
ubuntu@gitasgn:~/git5$ git merge feature2
Merge made by the 'ort' strategy.
 x.txt | 0
 y.txt | 0
 z.txt | 0
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x.txt
 create mode 100644 y.txt
 create mode 100644 z.txt
ubuntu@gitasgn:~/git5$ ls
a.txt b.txt master.txt p.txt q.txt r.txt s.txt x.txt y.txt z.txt
ubuntu@gitasgn:~/git5$
```

Create Hotfix branch and create urgent.txt file

Stage and commit that file

```
ubuntu@gitasgn:~/git5$ git checkout hotfix
Switched to branch 'hotfix'
ubuntu@gitasgn:~/git5$ ls
a.txt b.txt master.txt p.txt q.txt r.txt s.txt x.txt y.txt z.txt
ubuntu@gitasgn:~/git5$ nano urgent.txt
ubuntu@gitasgn:~/git5$ git add urgent.txt
ubuntu@gitasgn:~/git5$ git commit -m "urgent.txt"
[hotfix e78fb90] urgent.txt
 1 file changed, 1 insertion(+)
 create mode 100644 urgent.txt
```

Merge hotfix branch to master branch

```
ubuntu@gitasgn:~/git5$ git checkout master
Switched to branch 'master'
ubuntu@gitasgn:~/git5$ git merge hotfix
Updating 51a98fb..e78fb90
Fast-forward
 urgent.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 urgent.txt
```

All the files will be available in the master branch after merging.

```
ubuntu@gitasgn:~/git5$ git checkout master
M      urgent.txt
Switched to branch 'master'
ubuntu@gitasgn:~/git5$ ls
a.txt b.txt master.txt p.txt q.txt r.txt s.txt urgent.txt x.txt y.txt z.txt
ubuntu@gitasgn:~/git5$
```
