

CASE STUDY - GIT WORKFLOW

Problem Statement:

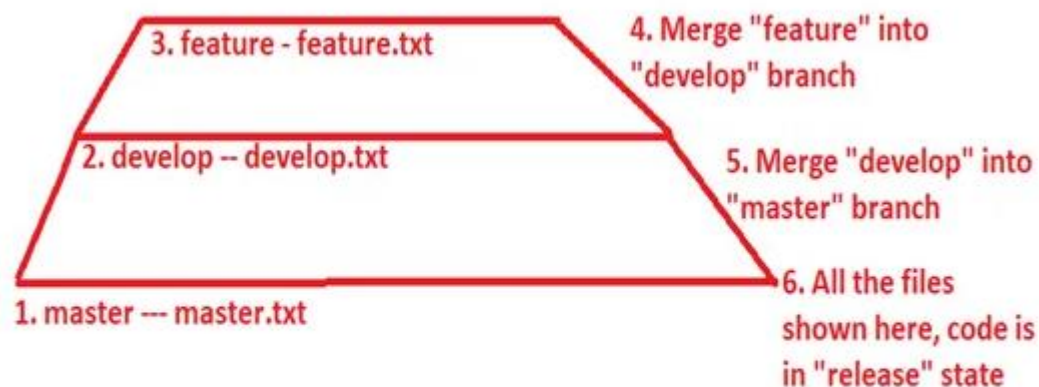
You work as a Devops Architect in Zendriix Softwares. The company has been struggling to manage their product releases. The releases should happen on 25th of every month. Suggest a Git Workflow Architecture for this requirement.

Simulate this workflow, by creating a pseudo code files and branches, and upload the same to your GitHub Account.

As a part of solution, share the link to your GitHub repository.

Solution:

Git- Workflow



Git Case Study 1 — Workflow Architecture

In this case, we will create three separate branches and files

Master - master.txt

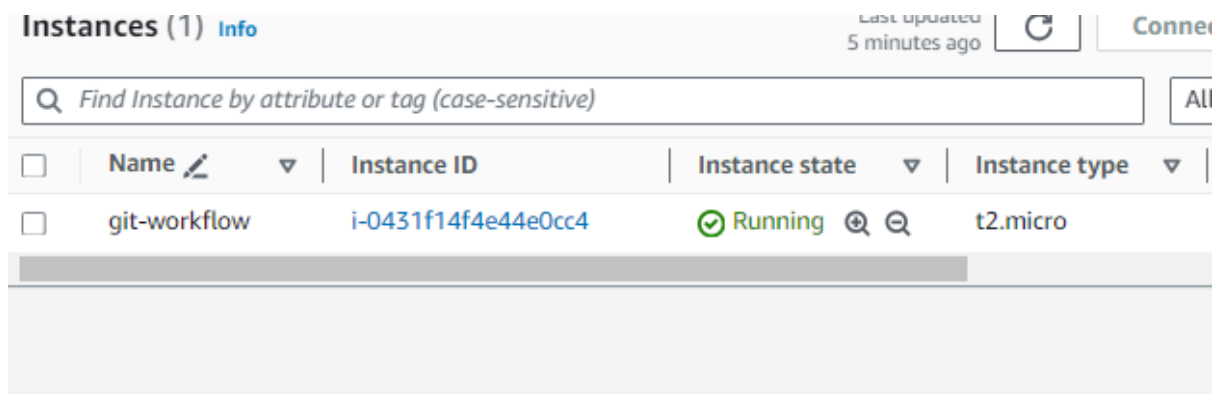
Develop – develop.txt

Feature – feature.txt

After creating all the branches & files, we will merge the “feature” branch into the “develop” branch & after merging the “feature” branch, we will merge the “develop” branch into the “master” branch.

Merging all branches into a master means the final code is ready to release & company will easily release the final product on the 25th of every month using this strategy.

Create Ec2 instance



The screenshot shows the AWS Management Console 'Instances' page. It displays one instance named 'git-workflow' with ID 'i-0431f14f4e44e0cc4', which is in a 'Running' state and of type 't2.micro'. The page includes a search bar, a table with columns for Name, Instance ID, Instance state, and Instance type, and a 'Connect' button.

	Name	Instance ID	Instance state	Instance type
<input type="checkbox"/>	git-workflow	i-0431f14f4e44e0cc4	Running	t2.micro

Update the machine

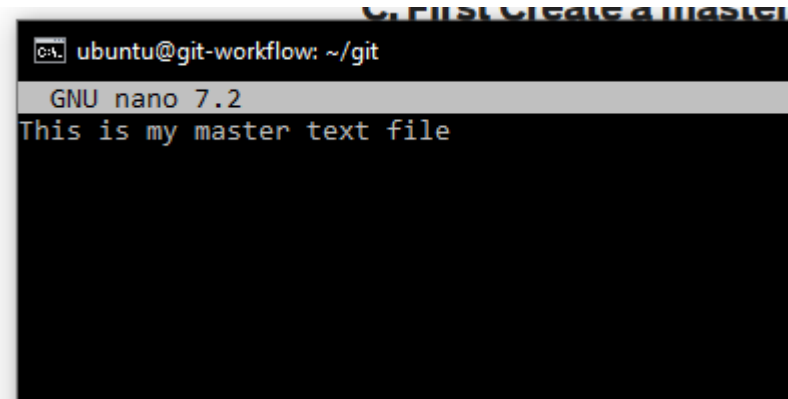
```
ubuntu@git-workflow: ~  
ubuntu@git-workflow:~$ sudo apt-get update
```

Create one folder for git and initialize the git

```
ubuntu@git-workflow:~$ cd git  
ubuntu@git-workflow:~/git$ 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/.git/
```

Create master text file in master branch

```
ubuntu@git-workflow:~/git$ sudo nano master.txt
ubuntu@git-workflow:~/git$
```



```
ubuntu@git-workflow: ~/git
GNU nano 7.2
This is my master text file
```

Add master.txt file and check status

```
ubuntu@git-workflow:~/git$ git add master.txt
ubuntu@git-workflow:~/git$ git status
On branch master

No commits yet

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

Commit them

```
ubuntu@git-workflow:~/git$ git commit -m "committing master.txt file"
[master (root-commit) 1ff191e] committing master.txt file
 1 file changed, 1 insertion(+)
 create mode 100644 master.txt
ubuntu@git-workflow:~/git$
```

Now master branch will be reflected

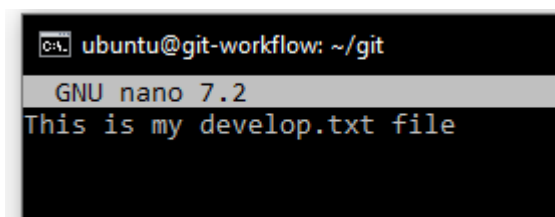
```
ubuntu@git-workflow:~/git$ git branch
* master
ubuntu@git-workflow:~/git$
```

Next create develop branch with develop.txt file

Before creating file checkout to the develop branch

```
ubuntu@git-workflow:~/git$ git branch develop
ubuntu@git-workflow:~/git$ git branch
  develop
* master
ubuntu@git-workflow:~/git$ git checkout develop
Switched to branch 'develop'
ubuntu@git-workflow:~/git$ git branch
* develop
  master
ubuntu@git-workflow:~/git$
```

```
ubuntu@git-workflow:~/git$ nano develop.txt
```



The screenshot shows the nano text editor interface. The title bar reads 'ubuntu@git-workflow: ~/git'. The status bar at the bottom indicates 'GNU nano 7.2' and the current file is 'This is my develop.txt file'.

Add develop.txt file and check status

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

Commit the develop.txt file

```
ubuntu@git-workflow:~/git$ git commit -m "committing develop.txt file"
[develop d3d6517] committing develop.txt file
 1 file changed, 1 insertion(+)
 create mode 100644 develop.txt
ubuntu@git-workflow:~/git$
```

List the files, here master.txt will also be present

```
ubuntu@git-workflow:~/git$ ls
develop.txt  master.txt
```

Create another branch feature

```
ubuntu@git-workflow:~/git$ git branch feature
ubuntu@git-workflow:~/git$ git checkout feature
Switched to branch 'feature'
ubuntu@git-workflow:~/git$ git branch
  develop
* feature
  master
```

Create feature.txt file

```
ubuntu@git-workflow:~/git$ nano feature.txt
ubuntu@git-workflow:~/git$
```

```
ubuntu@git-workflow: ~/git
GNU nano 7.2
This is my feature.txt file
```

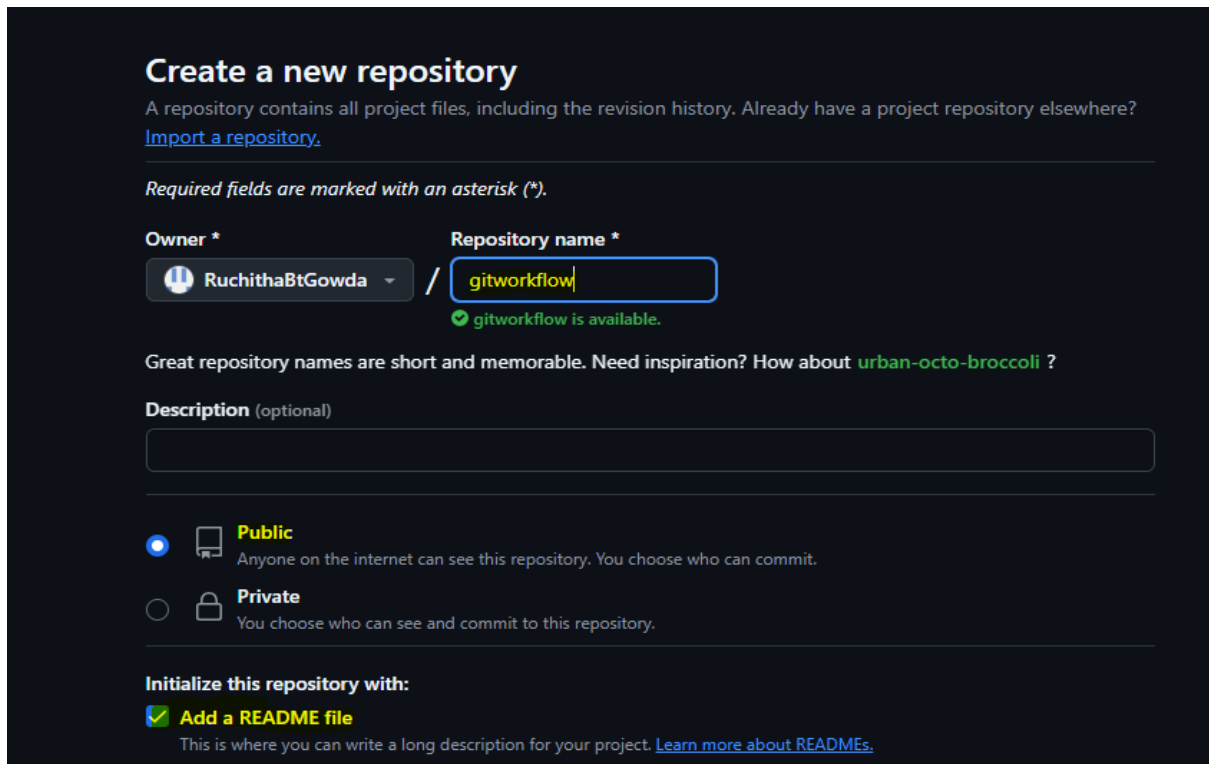
Add feature.txt file

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

Commit it

```
ubuntu@git-workflow:~/git$ git commit -m "committing feature.txt file"
[feature e00d057] committing feature.txt file
 1 file changed, 1 insertion(+)
 create mode 100644 feature.txt
ubuntu@git-workflow:~/git$
```

Go to your git hub repository and add new repo, click on add README file



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 * RuchithaBtGowda / **Repository name ***

✔ gitworkflow is available.

Great repository names are short and memorable. Need inspiration? How about [urban-octo-broccoli](#) ?

Description (optional)

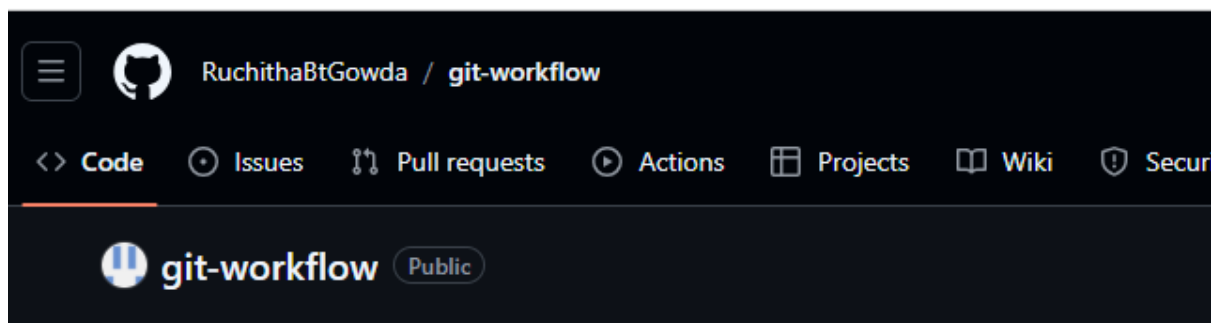
☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

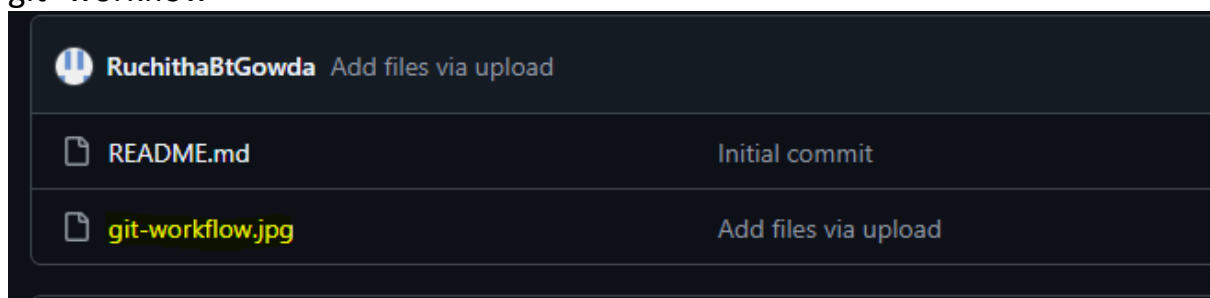
Initialize this repository with:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

New repository is created



README.md file is also present along with that add new file of your git- workflow



Go to .ssh path and generate the key using the command -> ssh-keygen

```
ubuntu@git-workflow:~$ cd .ssh
ubuntu@git-workflow:~/ssh$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:00Y8WCLqaEFPFnNzox3gtFIofsQM3m7D9MK6SfbJWrI ubuntu@git-workflow
The key's randomart image is:
+--[ED25519 256]--+
| .+o.*.+          |
| ...=B *.o         |
| .oo*.*+o.+        |
| ..Ooo. O          |
| .oO .. S          |
| o+ o              |
| *.               |
| + O              |
| E.+              |
+----[SHA256]-----+
ubuntu@git-workflow:~/ssh$
```

Go to the public key folder and copy it to paste on your remote location.

```
ubuntu@git-workflow:~/ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
ubuntu@git-workflow:~/ssh$ sudo cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAINc7xzHcoCBECn6NR0w7SZiFvff0leZIMR83KFNgVEn9 ubuntu@git-workflow:~/ssh$
```

Go to your remote repository > settings > SSH and CPG keys > add new SSH key

Add new SSH Key

Title: SSH keys

Key type: Authentication Key

Key: ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAINc7xzHcoCBECn6NR0w7SZiFvff0leZIMR83KFNgVEn9 ubuntu@git-workflow:~/ssh\$

Add SSH key

Now clone your remote repository using SSH method

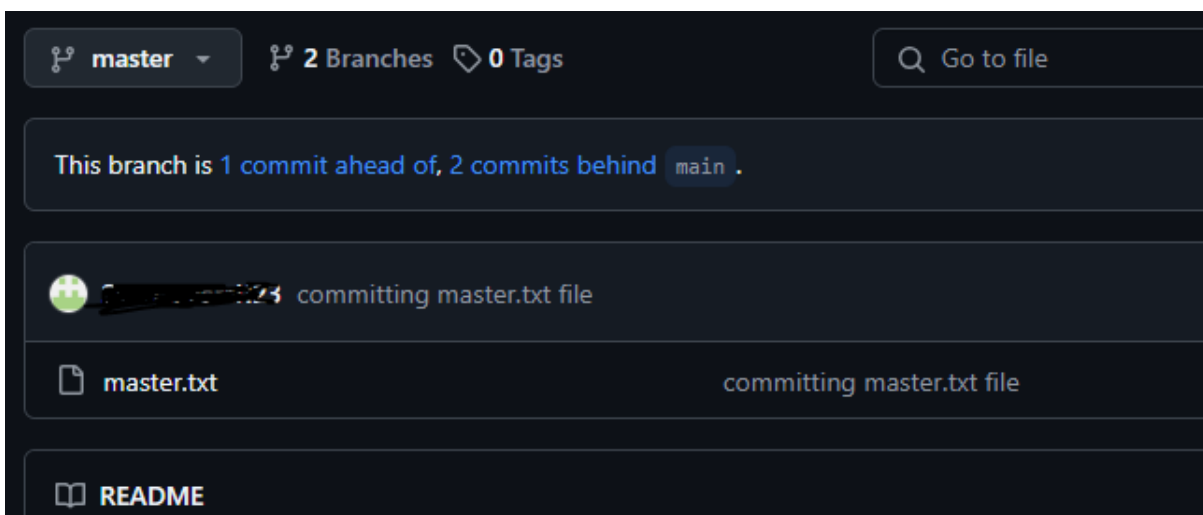
And confirm it by git remote -v command.

```
ubuntu@git-workflow:~/git$ git remote add gitworkflow git@github.com:RuchithaBtGowda/gitworkflow.git
ubuntu@git-workflow:~/git$ git remote -v
gitworkflow      git@github.com:RuchithaBtGowda/gitworkflow.git (fetch)
gitworkflow      git@github.com:RuchithaBtGowda/gitworkflow.git (push)
ubuntu@git-workflow:~/git$
```

Push your master branch from your Ubuntu machine to your remote location

```
ubuntu@git-workflow:~/git$ git push gitworkflow master
The authenticity of host 'github.com (20.205.243.166)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 252 bytes | 252.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/RuchithaBtGowda/gitworkflow/pull/new/master
remote:
To github.com:RuchithaBtGowda/gitworkflow.git
 * [new branch]      master -> master
ubuntu@git-workflow:~/git$
```

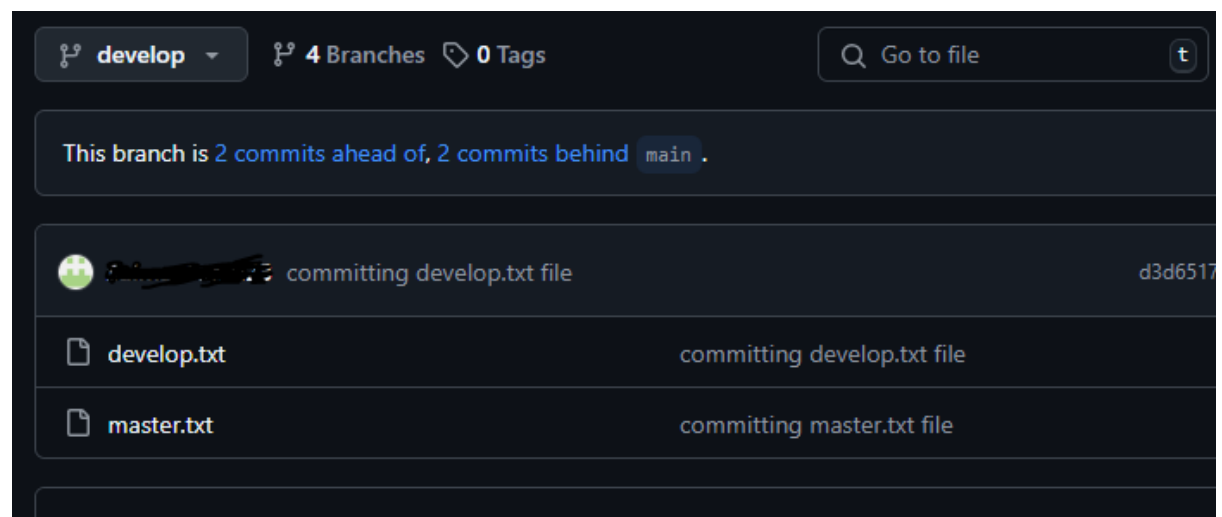
Your master branch will be reflected in remote directory.



Similarly push yours develop branch

```
ubuntu@git-workflow:~/git$ git push gitworkflow develop
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 317 bytes | 317.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/RuchithaBtGowda/gitworkflow/pull/new/develop
remote:
To github.com:RuchithaBtGowda/gitworkflow.git
 * [new branch]      develop -> develop
ubuntu@git-workflow:~/git$ git push gitworkflow feature
```

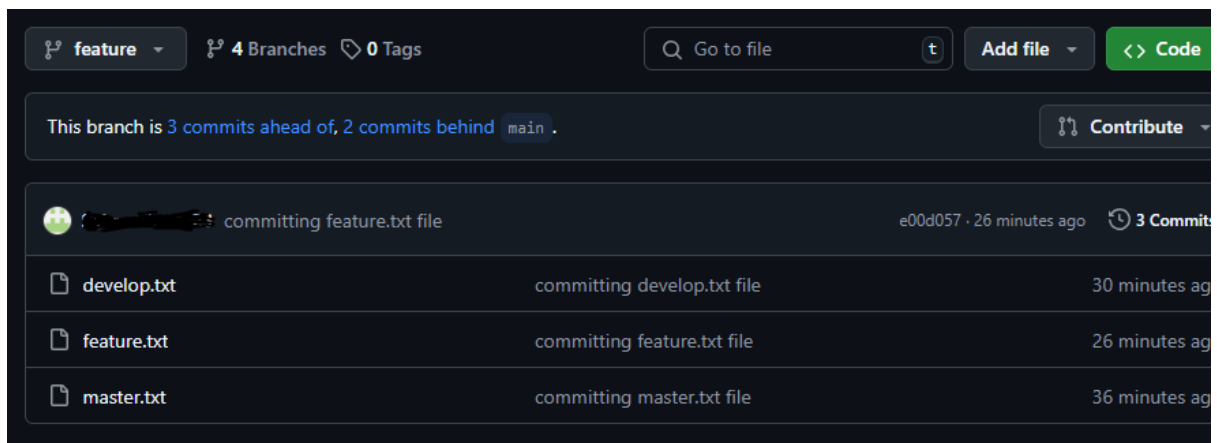
Check on the git hub by selecting develop



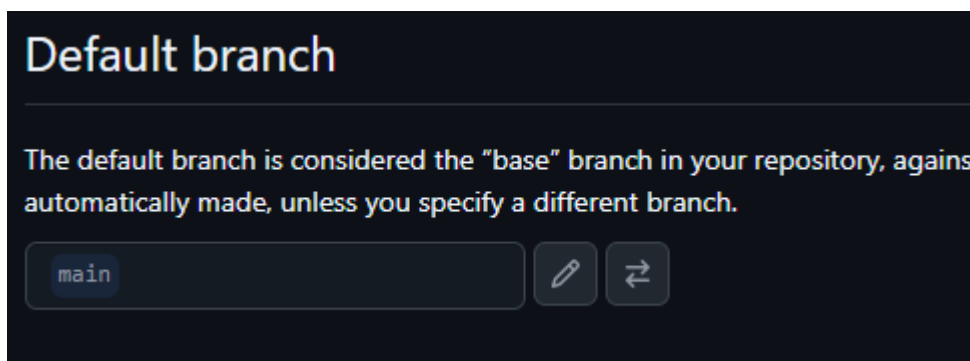
Push feature branch too

```
ubuntu@git-workflow:~/git$ git push gitworkflow feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 351 bytes | 351.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:   https://github.com/RuchithaBtGowda/gitworkflow/pull/new/feature
remote:
To github.com:RuchithaBtGowda/gitworkflow.git
 * [new branch]      feature -> feature
ubuntu@git-workflow:~/git$
```

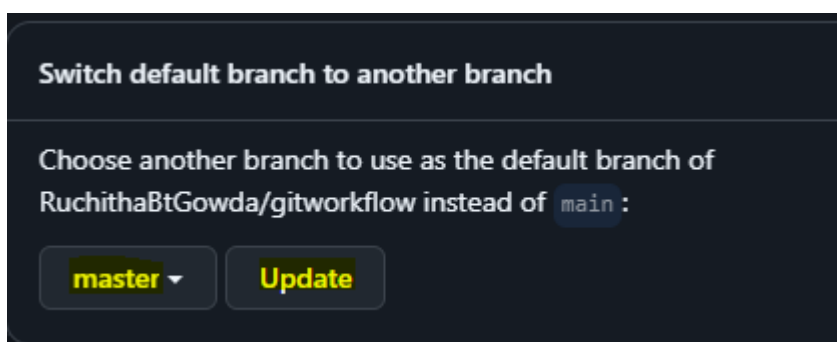
Check on git hub



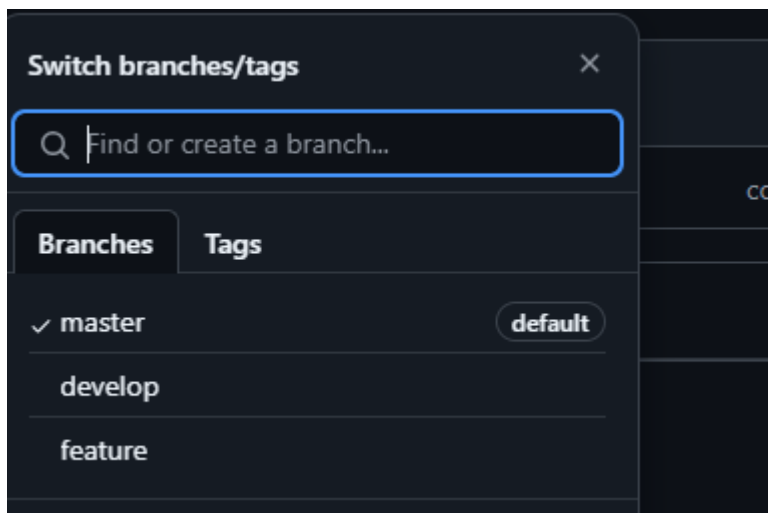
Change the default branch from main to master, also delete main branch



Select master and click on Update



You can only 3 of your branch is present



Now merge all the branch to master branch, first merge feature branch to develop and then develop branch to master

Checkout to develop branch and merge feature branch

```
ubuntu@git-workflow:~/git$ git merge feature
Updating d3d6517..e00d057
Fast-forward
 feature.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 feature.txt
ubuntu@git-workflow:~/git$
```

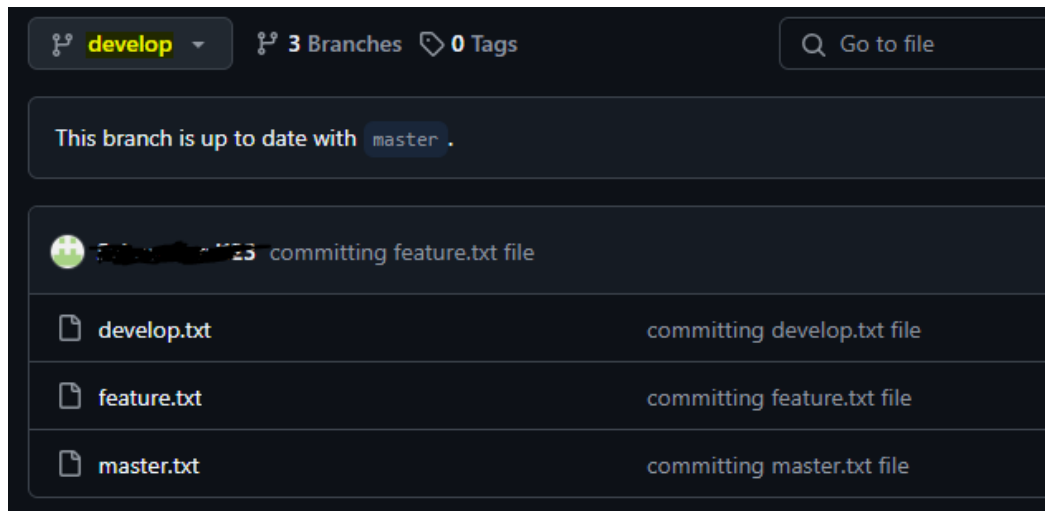
Next check out to master branch to merge develop branch

```
ubuntu@git-workflow:~/git$ git checkout master
Switched to branch 'master'
ubuntu@git-workflow:~/git$ ls
master.txt
ubuntu@git-workflow:~/git$ git merge develop
Updating 1ff191e..e00d057
Fast-forward
 develop.txt | 1 +
 feature.txt | 1 +
 2 files changed, 2 insertions(+)
 create mode 100644 develop.txt
 create mode 100644 feature.txt
ubuntu@git-workflow:~/git$ ls
develop.txt feature.txt master.txt
```

Again push develop branch to remote directory

```
ubuntu@git-workflow:~/git$ git push gitworkflow develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:RuchithaBtGowda/gitworkflow.git
d3d6517..e00d057 develop -> develop
```

Check the all the text files present



Push master branch finally

```
ubuntu@git-workflow:~/git$ git push gitworkflow master
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:RuchithaBtGowda/gitworkflow.git
1ff191e..e00d057 master -> master
```

Check them in the github.

