GIT WORKFLOW

CONTENTS

INTRODUCTION

Git is a distributed version control system accessible through a command line. Git is free and open-source software.Git helps keep track of changes in code and allows collaboration on projects of all scale. Github has a plethora of unique features that enhance the git experience and make collaborative coding simple.

GIT OPERATIONS

1.Creating Repository

Creating local repositoy is same as creating a folder in our system while creating a remote repository means creating a folder in git hub account.

2.Initialising

`git` init creates an empty Git repository or re-initializes an existing one. It basically creates a `.git` directory with subdirectories and template files. Running a `git init` in an existing repository will not overwrite things that are already there. It rather picks up the newly added templates.

3.Adding files

This command updates the index using the current content found in the working tree and then prepares the content in the staging area for the next commit.Thus, after making changes to the working tree, and before running the `commit` command, you must use the `add` command to add any new or modified files to the index. For that, use the commands below:

```
git add <directory_name>
```

or

```
git add <file_name>
```

4.Commit files

This will commit the staged snapshot and will launch a text editor prompting you for a commit message.
you can commit by using the command `git commit -m "message"`.

5.Push files

This command transfer commits from your local repository to your remote repository. It is the opposite of the pull operation.pushing exports commits to the remote repositories.The use of `git push` is to publish your local changes to a central repository. After you've accumulated several local commits and are ready to share them with the rest of the team, you can then push them to the central repository by using the following command:

```
git push <remote_repo>
```

We can use the command `git push origin master` to reflect these files in the master branch of my central repository.

6.Pull files

The `git pull` command fetches changes from a remote repository to a local repository. It merges upstream changes in your local repository, which is a common task in Git-based collaborations. before you affect changes to the central repository, you should always pull changes from the central repository to your local repository to get updated with the work of all the collaborators that have been contributing to the central repository. For that, we will use the pull command.

But first, you need to set your central repository as origin using the command:

```
git remote add origin <link of your central repository>
```

```
git pull origin master
```

This command will copy all the files from the master branch of the remote repository to your local repository.Since my local repository was already updated with files from the master branch, hence the message is already up-to-date.
One can also try pulling files from a different branch using the following command:

```
git pull origin <branch-name>
```

Your local Git repository is now updated with all the recent changes. It is time you make changes in the central repository by using the `push` command

7.Branching

Branches in Git are pointers to a specific commit.
There are basically two types of branches viz. *local branches* and *remote-tracking* branches.
They link your work from the local repository to the work on the remote repository.
we can check what your current branch is by using the command :

```
git branch
```

To create a new branch, we use the following command:

`git branch <branch-name>`

I have created a new branch named*anandu* and switched on to the new branch using the command `git checkout`.

Or we can use the command `git checkout -b anandu`. This command will create a new branch named*anandu* and check out the new branch at the same time.
while we are in the branch *anandu*, add and commit a new file using the following commands:

`git add <file name>`

`git commit -m "message”`

8.Merging

Merging is the way to combine the work of different branches together. This will allow us to branch off, develop a new feature, and then combine it back in.Let us merge the two branches with the command:

`git merge branch_name`.

The branch name in the above command should be the branch you want to merge into the branch you are currently checking out.Let us merge all of the work of the branch *anandu* into the master branch. For that, I will first check out the master branch with the command `git checkout master` and merge *anandu* with the command `git merge anandu master`.All the data from the branch*anandu* are merged to the master branch.

9.Rebasing

This is also a way of combining the work between different branches. Rebasing takes a set of commits, copies them, and stores them outside the local repository. To rebase master,
`git rebase master`.This command will move all our work from the current branch to the master.

10.Cherry Picking

This is the method of bringing other contributor’s commit from their branch to our current branch.This can be done by the command:

`git cherry-pick <commit>`