Git Commands

Follow the documents to get understanding of git commands

Here is a list of some common Git commands, along with a brief explanation of what each one does:

1. git init - This command initializes a new Git repository. It creates a new subdirectory called .git that contains all of the necessary files for the repository.

2. git clone - This command creates a copy of an existing Git repository on your local machine. It is typically used to download a copy of a remote repository from a hosting service like GitHub.

3. git add - This command adds files to the staging area, which is a temporary holding area where changes are kept before they are committed to the repository.

4. git commit - This command saves changes to the repository. It takes a snapshot of the current state of the files in the staging area and stores it permanently in the repository's history.

5. git status - This command shows the current status of the repository, including which files have been modified, added, or deleted.

6. git diff - This command shows the differences between the current version of the files in the repository and the most recent commit.

7. git log - This command shows a history of all the commits that have been made to the repository.

8. git branch - This command shows a list of all the branches in the repository, including the current branch.

9. git merge - This command combines changes from one branch into another branch.

10. git pull - This command updates the local copy of a repository with changes from a remote repository.

11. git push - This command uploads changes from the local repository to a remote repository.

These are just a few basic Git commands, there are many more advanced git commands which can be used as per requirement. These commands are more advanced and may require more understanding 😇 of git and version control to use effectively.

1. git stash - This command temporarily saves changes that have been made to the working directory, but haven't been committed yet. This allows you to switch to a different branch without losing your changes.

2. git rebase - This command modifies the existing commit history by integrating changes from one branch into another. It is often used to clean up a messy commit history or to merge changes from multiple branches into a single branch.

3. git cherry-pick - This command allows you to select specific commits from one branch and apply them to another branch.

4. git reset - This command undoes changes that have been made to the repository. It can be used to remove commits, unmodify files, or return to a specific point in the repository's history.

5. git tag - This command is used to label specific commits in the repository's history. It can be used to mark a specific version of the code, such as a release version.

6. git remote - This command allows you to manage remote repositories that are connected to your local repository. It can be used to add, remove, or view remote repositories.

7. git fetch - This command retrieves new commits from a remote repository, but doesn't merge them with the local repository.

8. git submodule - This command allows you to include one repository inside another repository as a subdirectory. It's often used when you want to include a third-party library or module in your project.

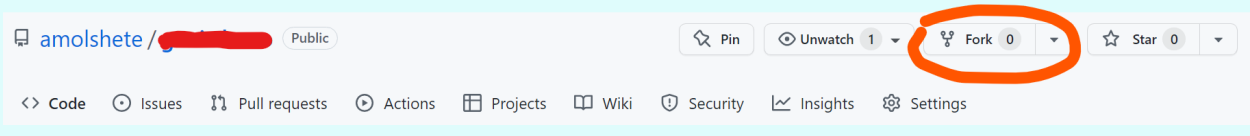
9. git bisect - This command is used to find the commit that introduced a specific bug. It works by binary search in the commit history

10. git blame - This command shows the last modification to each line in a file, along with the associated commit and author. This can be useful for tracking down who made specific changes or for identifying bugs.

How to raise Pull request. Lets understand it from scratch.

A pull request is a way to propose changes to a project on GitHub. If you're a beginner, here's how you can raise a pull request:

1. Fork the repository: The first step is to create a copy of the original repository you want to contribute to. To do this, click on the "Fork" button on the top right corner of the repository page.



2. Clone the repository: Next, you need to clone the forked repository to your local machine. You can do this by clicking the "Clone or download" button and copying the URL. Then, open your terminal and type the following command: git clone <URL>

```
Amol@DESKTOP-2MVQBON MINGW64 ~
$ git clone git@github.com:amolshete/
Cloning into '...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 23 (delta 4), reused 16 (delta 1), pack-reused 0
Receiving objects: 100% (23/23), done.
Resolving deltas: 100% (4/4), done.
```

3. Create a new branch: Once the repository is cloned, create a new branch for your changes. You can do this by using the following command: git checkout -b <branch-name> or git branch <new-branch-name>. Lets assume in this example

we are creating branch with name "phonepay". To switch the branch use following command: git checkout <new-branch-name>

```
Amol@DESKTOP-2MVQBON MINGW64 ~/Desktop[redacted] (master)
$ git branch phonepay
```

```
Amol@DESKTOP-2MVQBON MINGW64 ~/Des[redacted] (master)
$ git checkout phonepay
Switched to branch 'phonepay'
```

4. Make changes: Now you can make the changes you want to propose to the original repository. You can add new files, modify existing files, or delete files as needed.

```
Amol@DESKTOP-2MVQBON MINGW64 ~[redacted] (phonepay)
$ touch test.txt

Amol@DESKTOP-2MVQBON MINGW64 ~/Desktop[redacted] (phonepay)
$ nano test.txt
```

5. Commit changes: After you've made the changes, you need to commit them to the branch you created. Use the following commands:
   git add . to stage all changes
   git commit -m "your commit message" to commit the changes

```
Amol@DESKTOP-2MVQBON MINGW64 ~/Desktop[redacted] (phonepay)
$ git status
On branch phonepay
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        test.txt
```

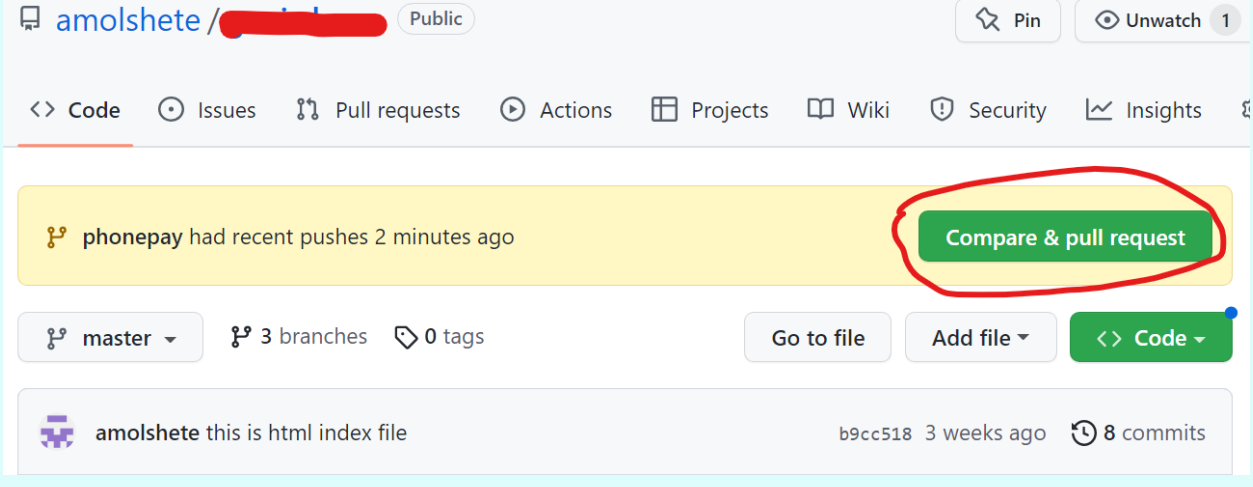
```
Amol@DESKTOP-2MVQBON MINGW64 ~[redacted] (phonepay)
$ git add test.txt

Amol@DESKTOP-2MVQBON MINGW64 ~/Desktop[redacted] (phonepay)
$ git status
On branch phonepay
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   test.txt
```

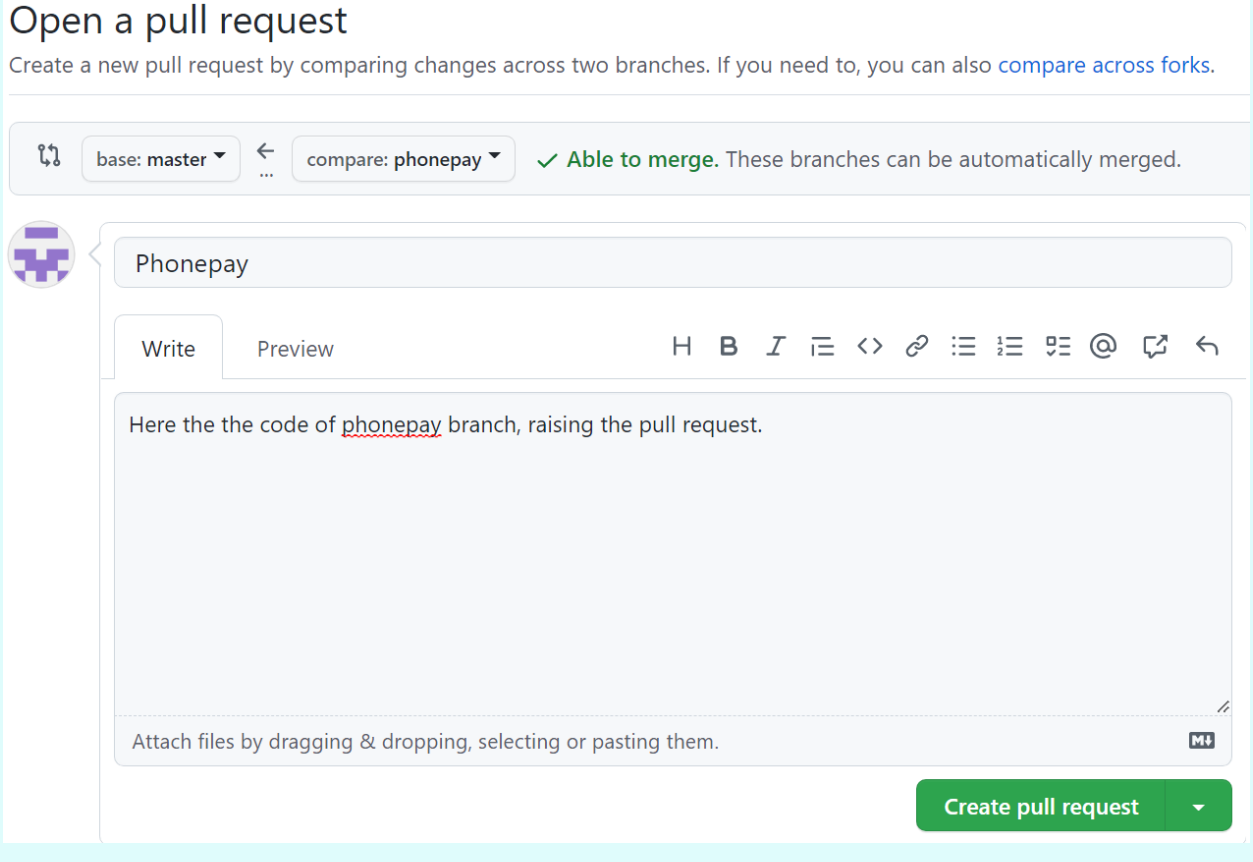
6. Push changes: Now that your changes are committed, you need to push them to your forked repository on GitHub. You can do this using the following command: git push origin <branch-name>

```
Amol@DESKTOP-2MVQBON MINGW64 ~/Desktop[redacted] (phonepay)
$ git push origin phonepay
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 448 bytes | 448.00 KiB/s, done.
```

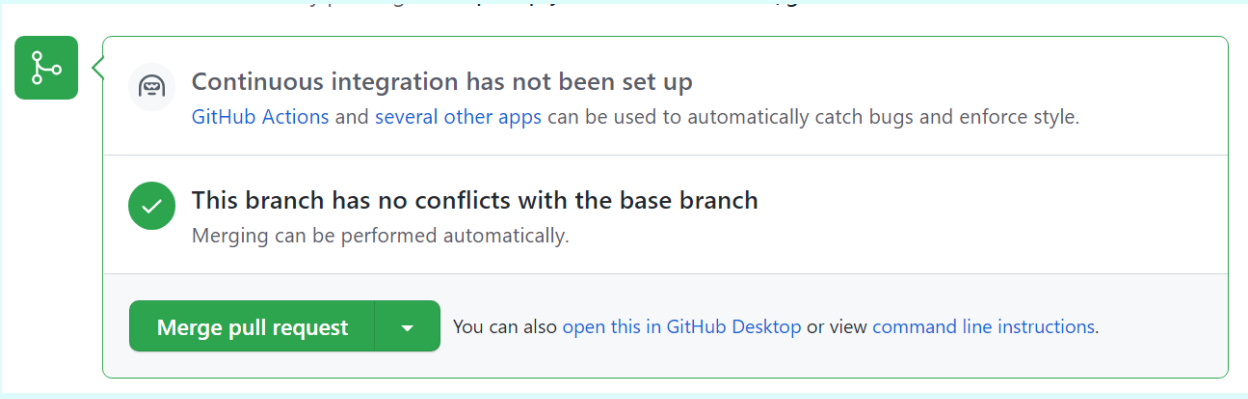
```
Writing objects: 100% (3/3), 448 bytes | 448.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
remote:
remote: Create a pull request for 'phonepay' on GitHub by visiting:
remote:      https://github.com/amolshete/[redacted]/pull/new/phonepay
remote:
To github.com:amolshete/[redacted].git
 * [new branch]      phonepay -> phonepay
```

7. Raise a pull request: Finally, you need to raise a pull request on GitHub. Go to the original repository on GitHub, and you should see a "Compare & pull request" button. Click on it, and then click on the "Create pull request" button. Fill in the pull request title and a description of your changes, and then submit the pull request.





8. Your changes will now be reviewed by the maintainers of the original repository, and they will either merge your changes or provide feedback for you to make further changes.



9. Once pull merge is done then you may delete the newly created branch.