GIT TIPS YOU MUST KNOW AS EARLY AS POSSIBLE

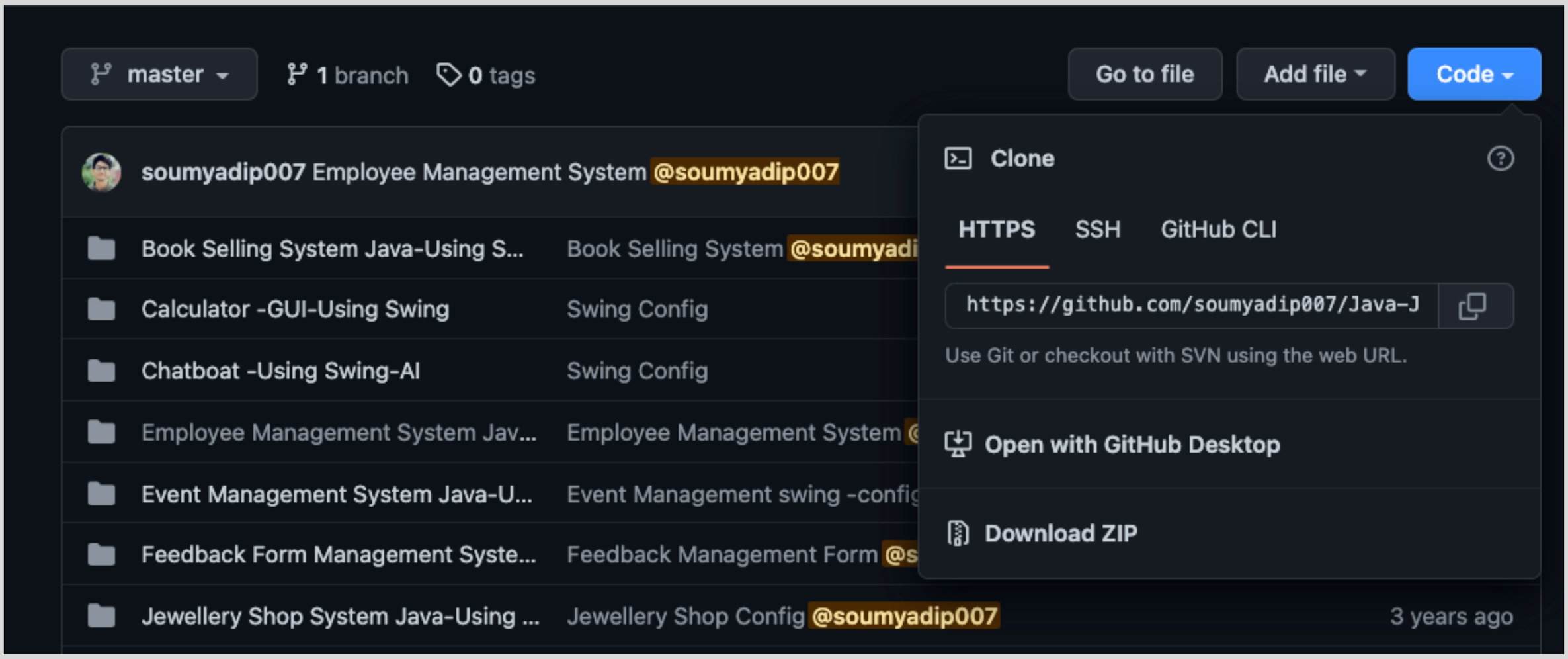
[MY KNOWLEDGE IN A NUTSHEEL]

/soumyadip-chowdhury

Clone a Project

Navigate to the location where you want your repository to be located after copying the URL.

Type the command below, replacing "repoUrl" with the URL you just copied.




```
git clone <repoUrl>
```

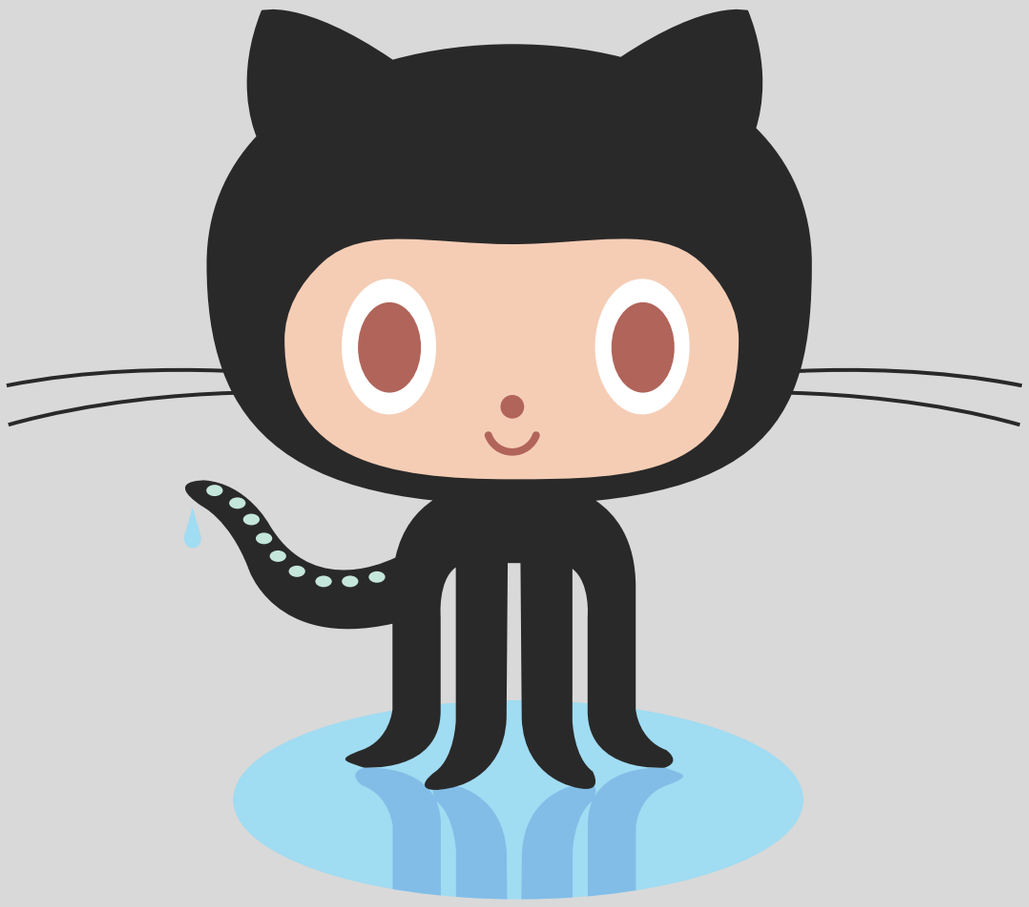
Changes in Git

Branch Create :

```
git checkout -b <branchName>
```

Stage all necessary modifications after making them :

```
git add .
```



Changes in Git

Make these modifications :

```
git commit
```

Add modifications to the main branch :

```
git push -u origin <branchName>
```

How to pull updates to your branch from the "main branch" ***

Use **git merge** or **rebase** to update the branch if you need the most recent modifications from the main branch included on your local copy.

Using Merge :

```
git stash -u //stash
git checkout <main_branch>
git pull
git checkout <your_branch>
git merge <main_branch>
git stash pop //re-applying changes
```

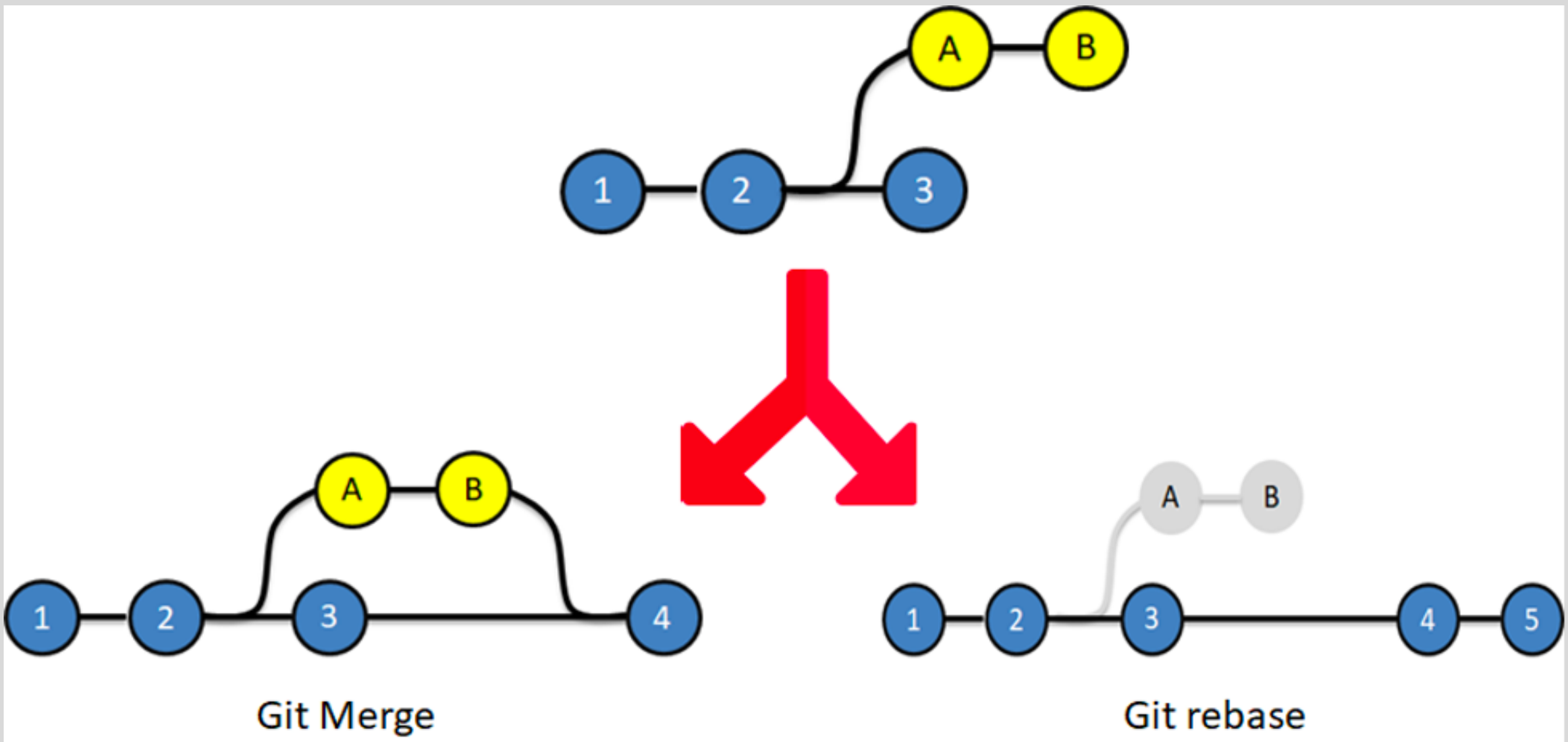
How to pull updates to your branch from the "main branch" ***

Using Stash :

```
git stash -u //stash
git checkout <main_branch>
git pull
git checkout <your_branch>
git rebase <main_branch>
git stash pop //re-applying changes
```

| Soumyadip Chowdhury

Rebasing & Merge



| Soumyadip Chowdhury

Fix "merge conflicts"

Do not be concerned **if you notice merging problems**. All you have to do is go through each conflicted file and sort out the issues. Just choose whether you want to maintain the recent or incoming changes.

```
<<<<<<< HEAD

=======

>>>>>>> branch
```

Branch & Commit

Edit commit message :

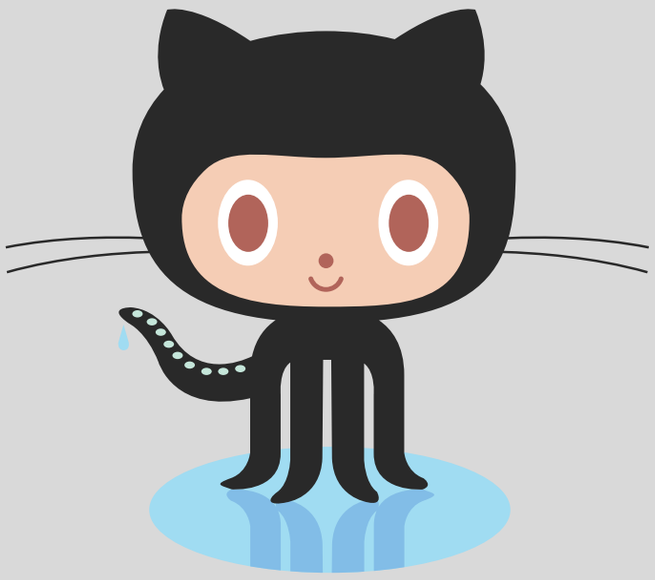
```
git commit --amend
```

Rename your branch :

```
git branch -m <branch-name>
```

Find your branch name :

```
git branch
```



Branch & Commit

Revert your last commit :

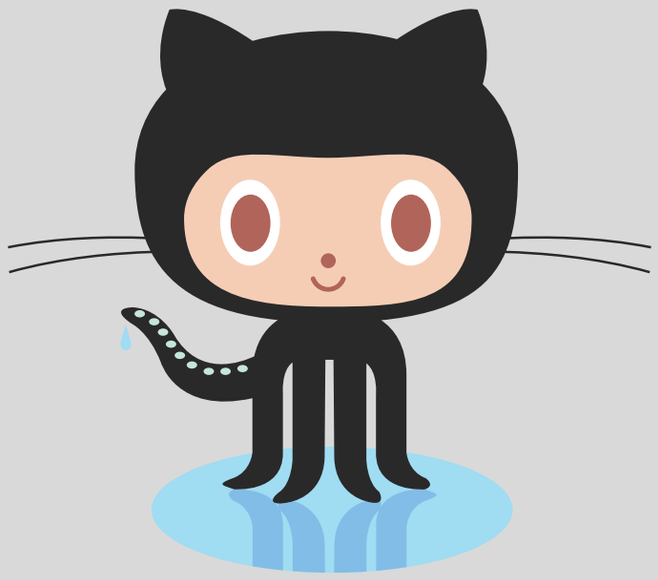
You can be in a position where you've pushed a commit but then discover you forgot to make certain changes. How do you behave?

Thankfully, you can easily undo your most recent commit.

```
git reset HEAD ~1

// or

git reset --soft HEAD ~1
```



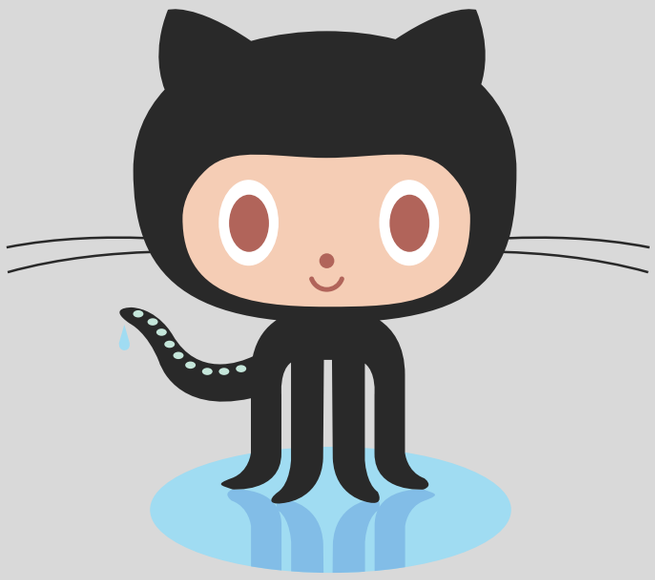
Cherry Pick

One of Git's helpful tools, **git cherry-pick**, allows you to copy and paste commits from one branch to another.

The majority of developers will view cherry-picking as a negative approach that can result in issues such as duplicate commits in several branches, messing up the git history, and other issues.

However, cherry-picking is one of the most potent tools, and if you know how it operates and utilize it carefully, it may be quite helpful.

You can cherry-pick changes that were made to the incorrect branch and move them to the correct branch in specific situations.



Cherry Pick

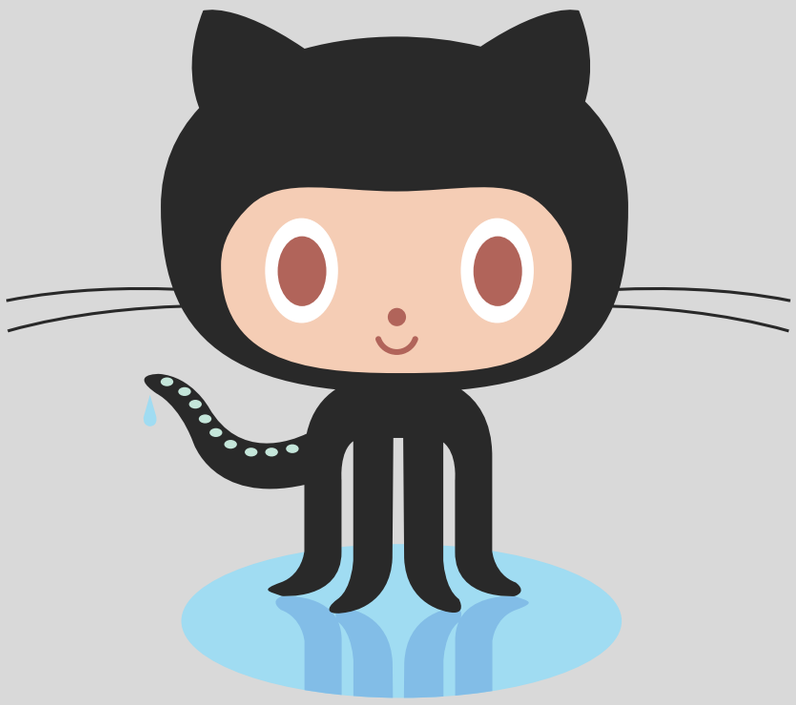
```
git cherry-pick <commit hash>

git cherry-pick -n <commit hash>

git cherry-pick -continue

git cherry-pick -abort

git cherry-pick A..B
```



Best Practice for Commits

- **feat**: A new user feature (not a feature to the code)
- **fix**: A user-friendly bug fix (not a fix to build)
- **docs**: Documentation updates
- **style**: Could refer to coding style or more general styling adjustments. has no impact on functioning.
- **chore**: Has no effect on output.

And for amazing stuff you can follow me

Soumyadip Chowdhury

soumyadip-chowdhury

@soumyadip007

@s_oumyadip

@println