**What is GIT?**

Free and open source version control system.

**What is Version Control?**Management of changes to documents, computer programs, large websites and other collections of information.

**Github**A website to host your repositories online.

**Git Commands**1. Clone: bring a repo that is hosted somewhere like github into a folder on your local machine.  
2. Add: track your files and changes in git.  
3. Commit: save your files in git  
4. Push: upload git commits to a remote repo like Github  
5. Pull: download changes from remote repo to your local machine the opposite of push.  
6. Status: gives the status of all of the files that have some changes before the git commit.  
7. Git add . : all of the folders and files tracked by git  
8. Git commit -m “” : enters the commit message before the changes are pushed into github  
9. Git branch -M main: deletes the current branch and creates a new branch with the name given.  
10. Git branch master: creates a new branch  
11. Git branch: gives all of the branch names present   
12. Git checkout master: moves from main to master  
13. Git branch -d master: deletes the branch mentioned  
14. Git push origin main: pushes the committed changes to the branch main.  
15. Git init: initialize git  
16. Git remote add origin [copy from repo]  
17. Git checkout -b feature-11: -b here means creating a new branch and switching as well at the same time.  
18. Git diff: provides the changes made in a branch

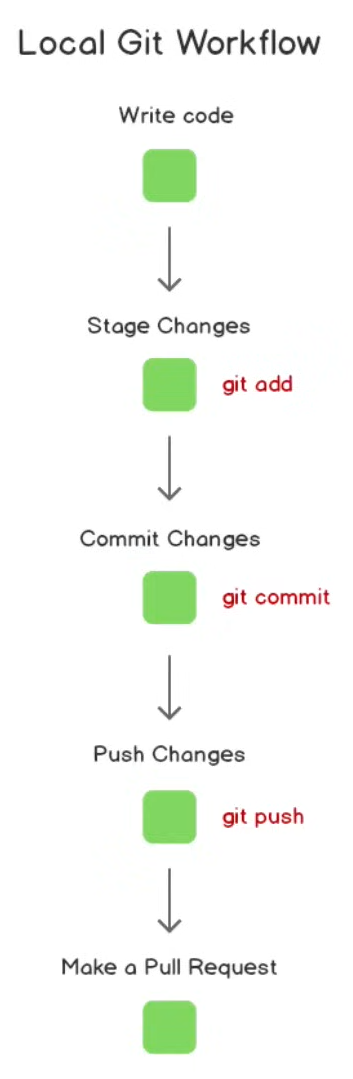
**STEPS TO BE FOLLOWED IF REPO ALREADY EXISTS.**

1. Git clone [from the repo on github]
2. Git add .
3. Git commit -m “”
4. Git push origin master

**STEPS TO BE FOLLOWED IF NEW REPO CREATED.**

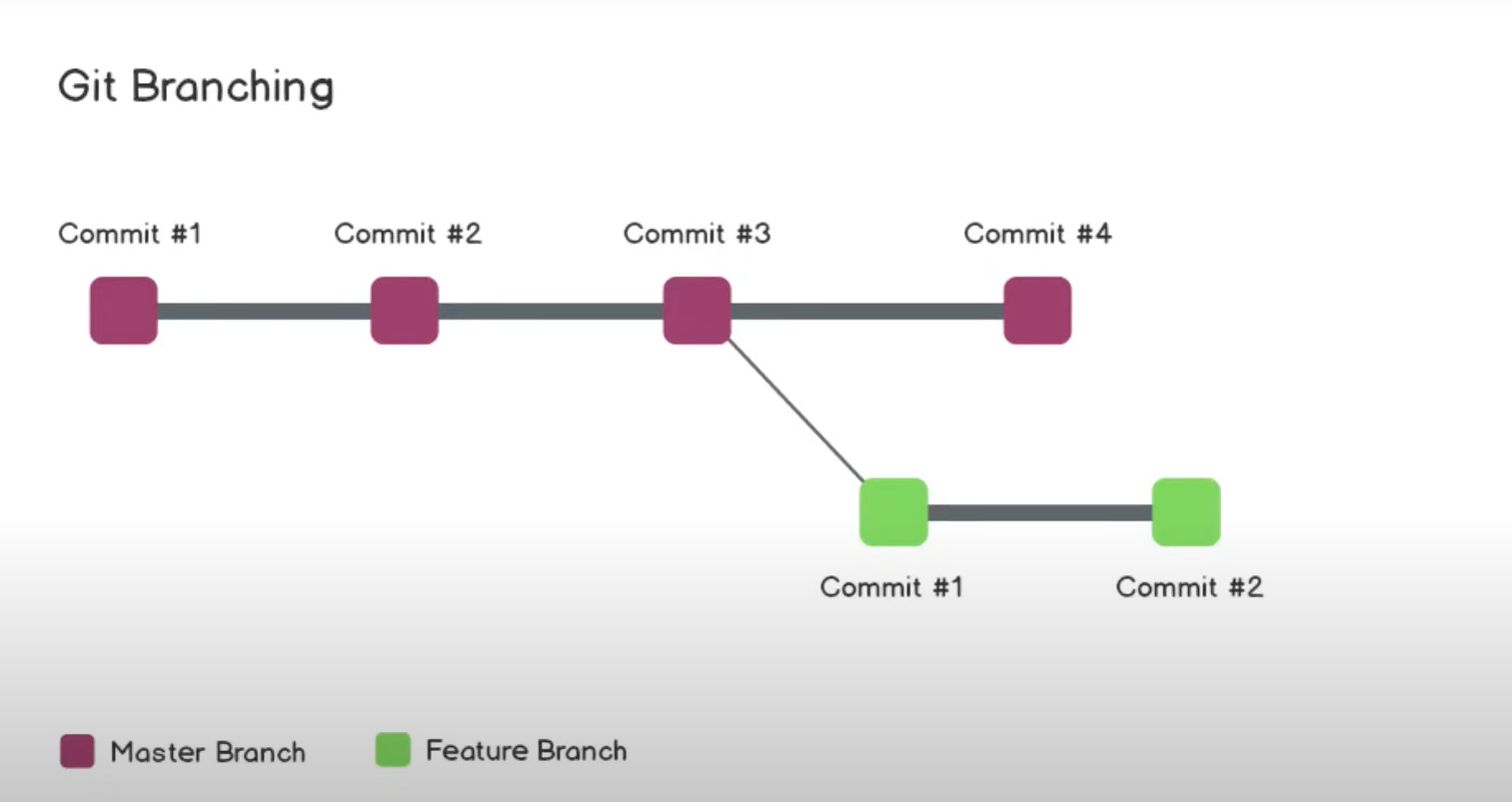
1. Git init
2. Git remote add origin [copy from repo]
3. Git add .
4. Git commit -m “”
5. Git branch -M main
6. Git push origin main {two options we can put -u to set an upstream so that everytime it is pushed its pushed into main by default and we can just type git push instead of whole thing everytime]

**GitHub Workflow**  
Local git workflow:



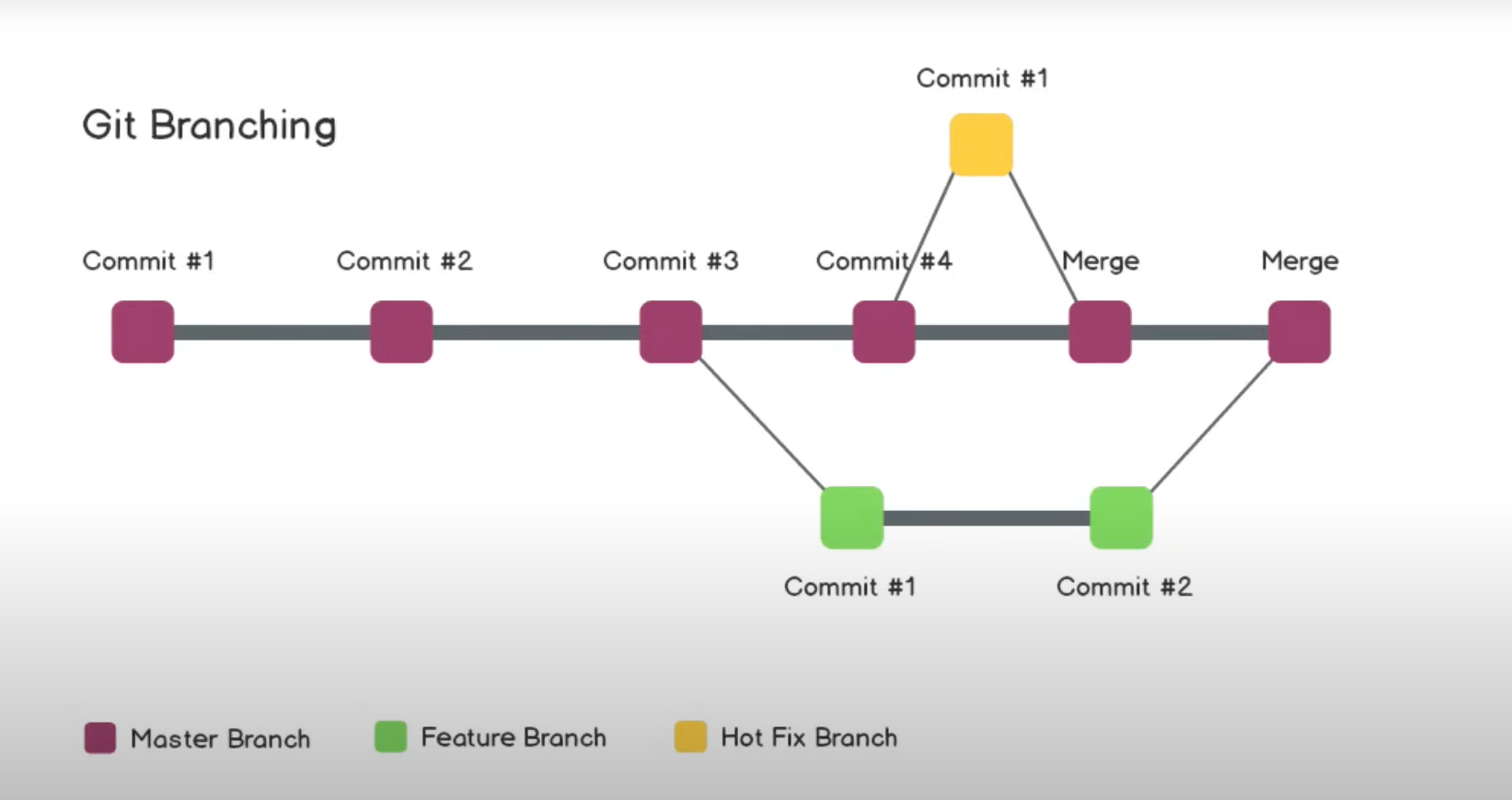
**Git Branching**

Creating multiple branches helps us inc ase we are developing an application and don’t want to add it to main branch as it might break the app so we just create a new feature branch and make some commits there which wont be visible in the main branch thus we can switch everytime .



Then they can be merged.

We find that there is a bug then we can create a hot fix branch where we can make the changes and then merge it back to the main branch.



1. Git branch -M main: deletes the current branch and creates a new branch with the name given.  
   2. Git branch master: creates a new branch  
   3. Git branch: gives all of the branch names present   
   4. Git checkout master: moves from main to master  
   5. Git branch -d master: deletes the branch mentioned  
   6. Git checkout -b feature-11: -b here means creating a new branch and switching as well at the same time.

Git commit -am “message” : it adds and commits with a message at the same time but it only works for modified files.

**GIT CONFLICTS**

1. Local to local conflict: can occur when trying to merge two branches on your local system. Need to be resolved manually.
2. Remote to local conflict: occurs when you try to push changes onto your repo but it already has some changes so one of the best possible ways is to before coding is to pull the changes from the day before onto your local system or if you have made some changes on your local system and they are important then create a new folder and clone the repo and then manually edit out the changes into the cloned folder. This is better than using a merge editor.