Git Hub classes

Github

- Folders that we upload on github is called repository.

#Repository

- Readme file - stores relative info
- commit - change (github tracks your changes)
- To use readme.md file learn basic html.

Git ⇒ It is installed on the computer

#Configuring Git

- git config --global user.name "My name"
- " " --global user.email "Someone@email.com"
- git " --list

→ you can also use local if you want to use a different account to make changes in the repo

```
PS D:\programming\git_bootcamp> git config --global user.name "Aayush Kumar"
PS D:\programming\git_bootcamp> git config --global user.email "kumaraayush835@gmail.com"
PS D:\programming\git_bootcamp> git config --list
```

Clone & status

[remote] [local]

Clone :- Cloning a repository on our local machine — github → on laptop

git clone <- some link ->

→ https link of the repo

Status :- display status of the codes

Different status shown are

1. Untracked ⇒ new file that git doesn't yet track
2. Modified ⇒ changed
3. staged ⇒ file is ready to be committed
4. unmodified ⇒ unchanged.

change (modified) / newfile (untracked)
↓
add (staged)
↓
commit (unchanged)
↓
push

add & commit

add ⇒ adds new or changed files in your working directory to the git staging area.

git add <file name>

git add -A → This will add across all folders

git add . → this is to add all the files, if there is a lot. and it only works for the folder which is open

commit ⇒ It is a record of changes

```
PS D:\programming\git_bootcamp> git commit -m "first commit"
[main (root-commit) 117af36] first commit
 3 files changed, 1 insertion(+)
 create mode 100644 GitHub notes.pdf
 create mode 100644 git-cheat-sheet-education.pdf
 create mode 100644 repo1.txt
```

```
PS D:\programming\git_bootcamp> git commit -am "commit in practice branch"
[practice d48addd] commit in practice branch
 1 file changed, 2 insertions(+), 1 deletion(-)
```

u can use -am if u r commiting a file for the first time

git commit -m "some message" (for message)

git commit -am "some message" → for both add and message

even after all this the files won't show on github .. we use git push

```
PS D:\programming\git_bootcamp> git branch -M main
PS D:\programming\git_bootcamp> git remote add origin https://github.com/A-K001/git_bootcamp.git
PS D:\programming\git_bootcamp> git push -u origin main
```

push command

push :- upload local repo content to remote repo.

git push origin main

init command

init :- used to creat new git repos

```
PS D:\programming\git_bootcamp> git init
Reinitialized existing Git repository in D:/programming/git_bootcamp/.git/
```

```
PS D:\programming\git_bootcamp> git remote rm origin
```

different commands → git remote rm origin → to delete origin

git init → to creat new repo → link of the new repo created on github
git remote add origin <link> → This is the name of new remote repo
git remote -v → to verify remote
git branch → to check branch
git branch -M name of new branch (main) → to rename branch
git push origin main

whatever name you use here will be used here

Note if you use ⇒ git push -u origin main then from next time you only need to type (git push) and it will push in origin main only

to push branch → git push -u origin <branch name>

```
PS D:\programming\git_bootcamp> git push -u origin practice
Enumerating objects: 5, done.
```

Git Branches

- Branches are made so that different people can work on the same project at the same time and no one has to wait for the other person to complete since they are working on their own branch.

Branch commands

- git branch → to check branch
- git branch -M main → To rename branch (Here name is changed to main)
- git checkout <branch name> → to navigate to different branch.
- git checkout -b <new branch name> → to creat new and switch branch. / git branch <name of new branch> → only to creat new branch
- git branch -d <branch name> → to delete branch

```
PS D:\programming\git_bootcamp> git branch -d practice1
Deleted branch practice1 (was 0246409).
```

You wont be able to delete the branch you are currently working on

```
PS D:\programming\git_bootcamp> git branch practice
PS D:\programming\git_bootcamp> git branch
* main
  practice
PS D:\programming\git_bootcamp> git checkout -b practice1
Switched to a new branch 'practice1'
PS D:\programming\git_bootcamp> git branch
  main
  practice
* practice1
```

Merging code

Way 1

- git diff <branch name> → to compare commits, branches, files & more
- git merge <branch name> → to merge 2 branch

Way 2

- Create a PR → PR is a pull request

Pull request

It lets you tell others about changes you've pushed to a branch in a repo on github

process of pull request is from 57:00 - 1:00 in the video

https://www.youtube.com/watch?v=Ez8F0nW6S-w&t=1911s

Pull command

- git pull origin main

This is used to fetch and download content from remote repo and immediately update the local repo to match that content.

Merge conflicts

An event that takes place when git is unable to automatically resolve difference in code between 2 commits

- If you have pushed already then perform "git reset -- Hard HEAD~1" and then git push --Forced origin main

Undoing Changes

Case 1 : staged changes (add)

- git reset <- file name ->
- git reset → for multiple files

Case 2 : commited changes (for one commit)

- git reset -- Hard HEAD ~1 → Undo and erase the changes
- git reset --soft HEAD ~1 → Undo but keep the changes at staged status.

Case 3 : commited changes (for many commits)

- git reset <- commit hash ->
- git reset --hard <- commit hash ->
- git reset --soft <commit hash>

These hashes are different for every commits and can be checked by using the command

- git log / git log --oneline

press shift + Q to get out of git log.

Case 4 : If not used git add as well

- git checkout <file name> → 1 file
- git checkout -F → multi file

```
PS D:\programming\git_bootcamp> git checkout repo.txt
PS D:\programming\git_bootcamp> git checkout -f
```

Fork

A fork is a new repository that shares code and visibility settings with the original "upstream" repository. The repo from another account can be cloned directly into your account using fork.

Fork is a rough copy.

Deleting a file

```
PS D:\programming\git_bootcamp> git rm waste.c++
```

- git rm <file name> → to delete a file from local as well as remote repo
- git rm --cached <file name> → to delete a file from remote repo only.

```
PS D:\programming\git_bootcamp> git rm --cached waste1.c++
```

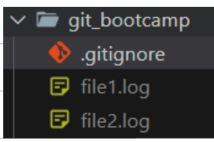
Renaming a file

- git mv <current file name> <name you want to change to>

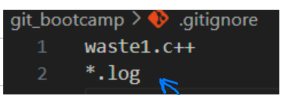
```
PS D:\programming\git_bootcamp> git mv repo1.txt repo.txt
```

Making ignore files

- If there are files in your local repo which you don't want to upload them you can use .gitignore file you need to make a file with this name.



- To ignore any file you need to write its name in the .gitignore file and after that it won't show on the status as untracked.



- To ignore files with a particular extention write *.<extention> eg.

Command	Description
git init	Initializes a new Git repository
git clone <repo_url>	Clones a repository from a remote URL
git status	Shows the working directory and staging area status
git add <file>	Stages a specific file for commit
git add .	Stages all changes for commit
git commit -m "message"	Commits staged changes with a message
git commit --amend -m "new message"	Amends the last commit message
git log	Shows the commit history
git log --oneline	Shows a condensed commit history
git diff	Shows unstaged changes
git diff --staged	Shows changes staged for commit
git branch	Lists branches in the repository
git branch <branch_name>	Creates a new branch
git checkout <branch_name>	Switches to a different branch
git switch <branch_name>	Also switches to a different branch (modern alternative)
git checkout -b <branch_name>	Creates and switches to a new branch
git switch -c <branch_name>	Creates and switches to a new branch (modern alternative)
git merge <branch_name>	Merges a branch into the current branch
git rebase <branch_name>	Reapplies commits on top of another branch
git stash	Temporarily saves changes without committing
git stash pop	Restores stashed changes
git stash list	Shows saved stashes
git remote -v	Lists remote repositories
git remote add <name> <url>	Adds a remote repository
git fetch	Fetches changes from a remote repository
git pull	Fetches and merges changes from a remote repository
git push	Pushes local changes to a remote repository
git push -u origin <branch_name>	Pushes a branch and sets upstream tracking
git reset --soft HEAD~1	Moves the HEAD back one commit, keeping changes staged
git reset --mixed HEAD~1	Moves the HEAD back one commit, unstaging changes
git reset --hard HEAD~1	Moves the HEAD back one commit, discarding changes
git rm <file>	Removes a file from the repository
git tag <tag_name>	Creates a tag at the current commit
git show <commit/tag>	Displays details of a commit or tag
git cherry-pick <commit_hash>	Applies a specific commit to the current branch
git revert <commit_hash>	Creates a new commit that undoes a specific commit
git bisect start	Starts binary search debugging for faulty commits
git blame <file>	Shows who modified each line of a file