Git

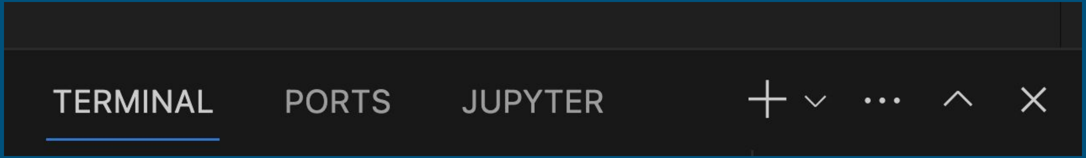
Manas Jyoti Das, PhD
Computer Science

An important message

- Working with the terminal may be a little bit intimidating initially once you get the hang of it it will become your friend
- My philosophy: The add-ons/extensions and GUI interface will keep on changing. But the basic commands will always be there
- Install conda and update all the environment variables as stated in previous class slides, open your terminal and type conda if no error good
- How to check vscode and conda integration

An important message contd…

- Open your terminal in vscode and in the terminal type conda. If no error great!!
- If error: There may be some problem with your environment variable. Also try to change the terminal type
- Click on the little down arrow near the + sign in the terminal window of vscode. There should be two options 1) powershell 2) cmd
- Change between them and type conda in each terminal type
- If it is working great!



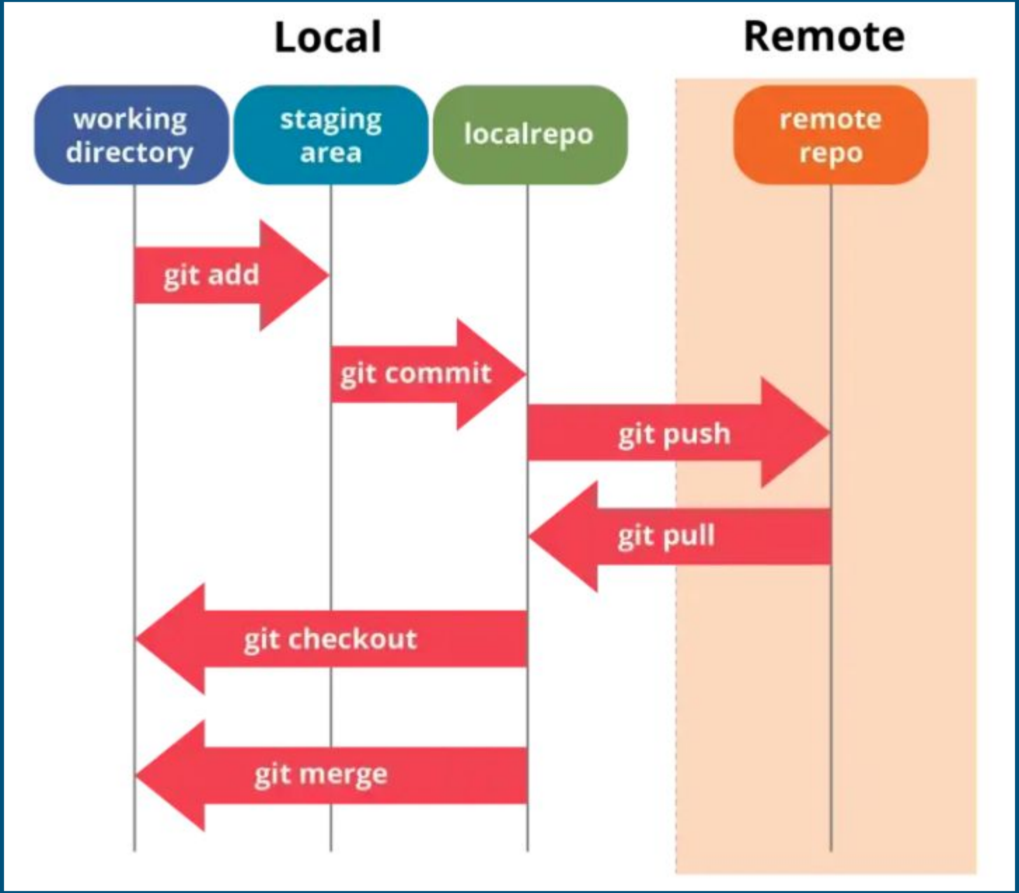
Yet another important msg…

- If you are able to access conda from your terminal in vscode, create a new environment, instead of the (base) environment. `E.g. conda create --name xyz python=3.9`
- To run a program do not hit the run button of the vscode. It is not going to use your conda environment where you will be installing your packages
- To run a program please use the terminal. Navigate to the folder where the code is being stored. Then use the command: `python3 filename.py`
- Remember to see in which conda environment you are in right now before running the code
- It may happen you are running the code in a different environment and you installed the packages in a different environment (most common mistake among students)

Git

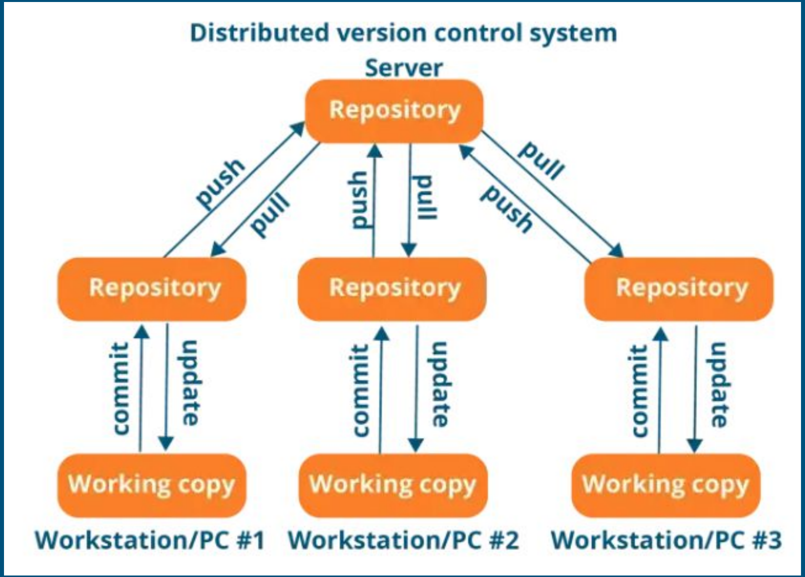
- Git is a distributed version control system, that is used to track changes in computer files
- It will take snapshots of the files and save it
- It is decentralized: That is code is stored with all the collaborators of the project

Git workflow



Git

- Operations (except push & pull) are fast because you are usually accessing your local hard drive.
- Committing new changes can be done locally without changing the main repository
- Since every contributor has a full copy of the project repository, they can share changes with one another if they want to get some feedback before affecting changes in the main repository.
- If the central server gets crashed at any point of time, the lost data can be easily recovered from any one of the contributor’s local repositories.



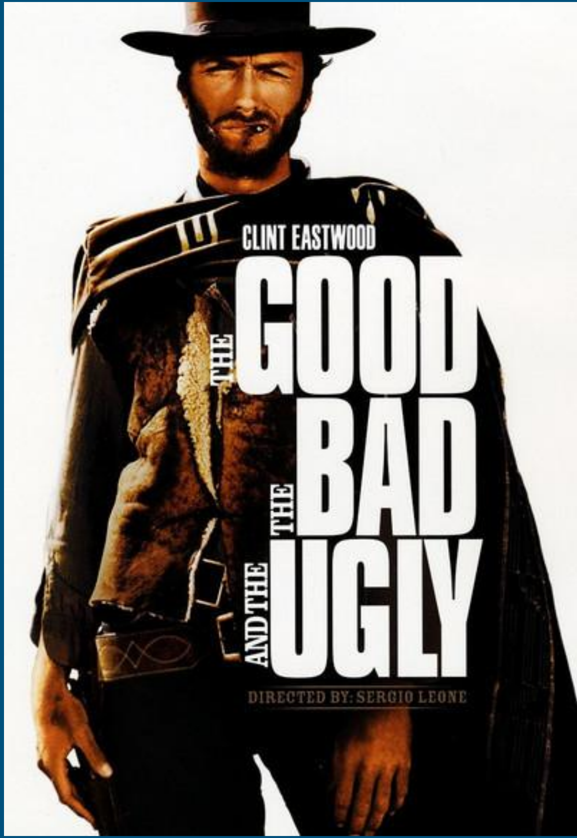
Git setup

- It may be setup in your local machine
- It may be setup locally in your company
- It may be setup to reside at a central repository
- Examples of central repositories: Github, Gitlab, Xetlab, etc.

Installing Git

- Linux users: good (out of the box)
- Mac users: bad (out of the box)
- Windows users: ugly

Use this link only: https://git-scm.com/download/win



How git is tracking the state of the code

- Need to initialize the folder with git, to track its content
- `git add <filename>`
- A subset of files can be also be added `git add abc.py xyz.py`
- `git add .` (add all the files in the folder)
- Git only track the files that you added. If you do not add a file it will not get tracked. In may be in the same folder

To save the code

- Save it where? Locally (add and commit store info locally)
- Command: git commit
- Commit means git creates a snapshot of your project at that point in time
- The git commit command takes two arguments:
  - the first argument is the message for the commit
  - the second argument is the list of files that you want to commit
- If you want to commit all of the changes in your working tree, you can use the -a flag

```
git commit -m "message" myfile.txt

git commit -a -m "message"
```

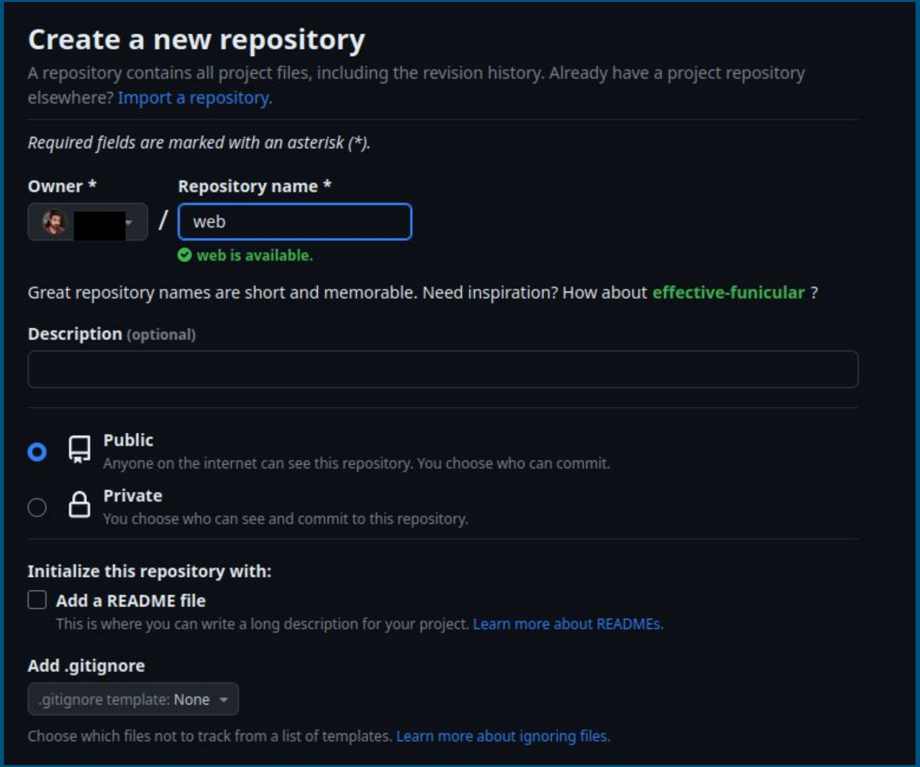
To save the code remotely*

- The git push command is used to push changes from your local repository to a remote repository
- (*)The remote repository can be hosted on a server, such as GitHub or GitLab, or it can be a local repository on another computer/server that you own
- The git push command takes two arguments:
    - the name of the remote repository and
    - the name of the branch that you want to push.
- git status

ENOUGH TALK. LET'S CODE

First step

- Create a github account and create a new repository
- Make it private, you can always change it to public after the completion of the project
- Public means everyone can see the progress of your project
- After the creation of the repository, a page with lots of commands will be shown. It will be useful later on



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * [redacted] / Repository name * web

web is available.

Great repository names are short and memorable. Need inspiration? How about effective-funicular ?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

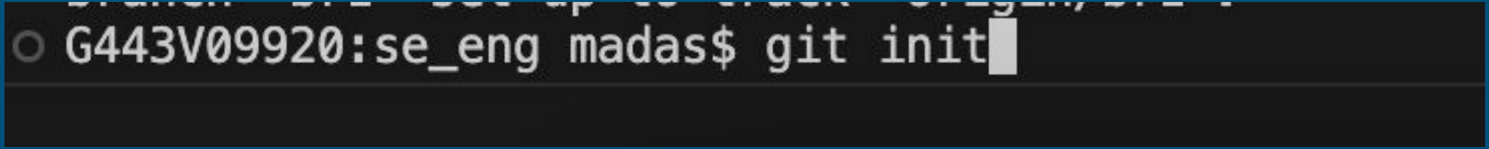
Initialize this repository with:

Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore
.gitignore template: None
Choose which files not to track from a list of templates. Learn more about ignoring files.

Second step

- Open your vscode and create a folder
- Inside of this folder create a file called README.md
- Please learn if you do not know how to navigate folder to folder in windows
- Open your terminal and navigate to the folder that you just created
- Assuming you are inside of the folder where README.md is residing type the command : `git init`
- Example: In the image below I am inside a folder called se_eng.



```
G443V09920:se_eng madas$ git init
```

Third step

- Run the following commands now:
    - git branch -M main
    - git remote add origin https://github.com/your_user_name/soft_eng.git
    - The link shown above can be copied from the output page as mentioned in slide-14
- What happened till now?
    - We created a repo (repository) in github and wanted to link this repo with the personal machine, so that you can do collaborative work and version control
    - You created a folder in you computer and initialized the folder with git (command: git init)
    - Then you said to the git that please use the branch main (command git branch -M main)
    - Then you said to use the link as a remote repo to store all the information (command: git remote add origin https://github.com/your_user_name/soft_eng.git)

Fourth step

- Adding files to the repo
  - Since the README.md file is already created lets edit it write # CS235
  - # means heading-1
  - ## means heading-2 (smaller)
  - ### means you are smart!!
  - Try to learn markup language (max 10 minutes!! All the basics)
- Adding this file to the repo
  - `git add README.md`
- Committing
  - `git commit -m “adding a readme file”`
- Push
  - `git push -u origin main`

Fifth step

- Refresh your repo page the file README.md will be there and a readme will be written on the repo page
- Create another file called file_name.py in the same directory and repeat everything
- Now while(true) ;-)

Git branch

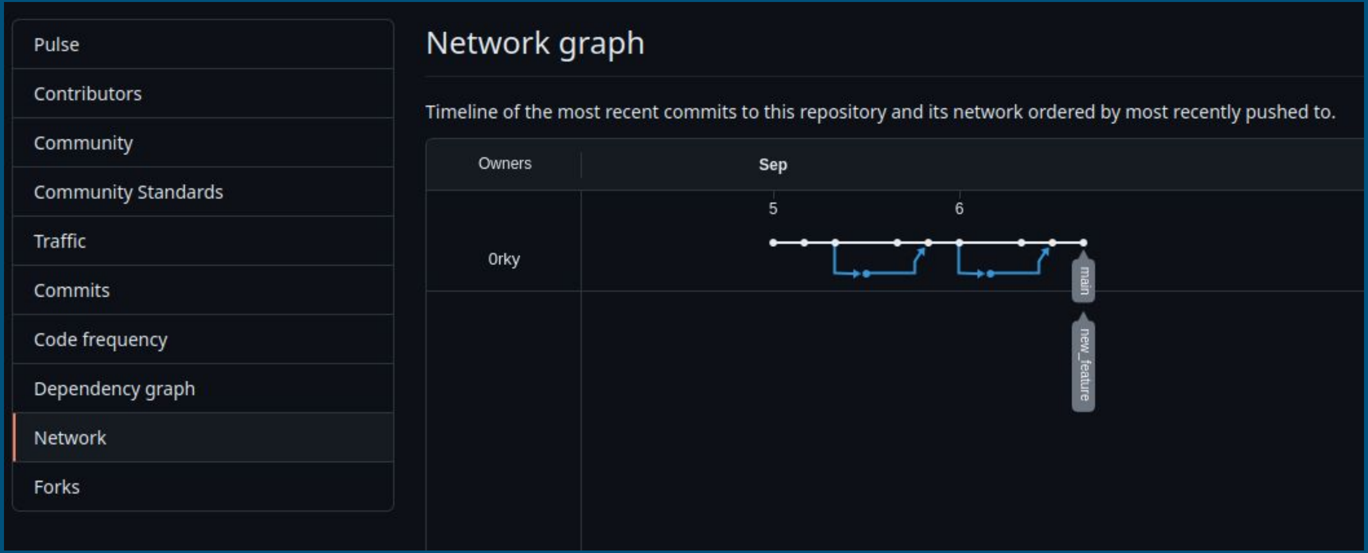
- Working with a software is not always linear
- To test new ideas we should be able to do it without disturbing the existing code base
- Branching is helpful in this scenario
- Each branch will have a name: main/master (the main branch)
- To switch from one branch to another you have to move the head to the appropriate branch

Git branch

- To create a branch
  - `git branch <branch_name>`
- To move your head to the new branch
  - `git checkout <branch_name>`
- To check your head!!!
  - `git branch`
- Pushing to a specific branch
  - `git push –set-upstream origin <branch_name>`
  - `git push origin <branch_name>`

Git branch

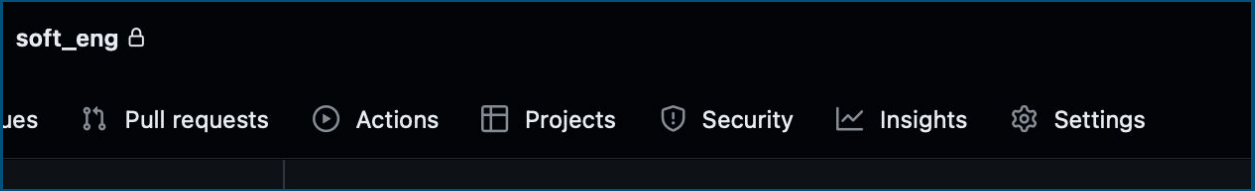
- If your repo is private you will not be able to view the branch graph as shown below



Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.

Git branch

- To see the branch graph and other statistics go to Insights



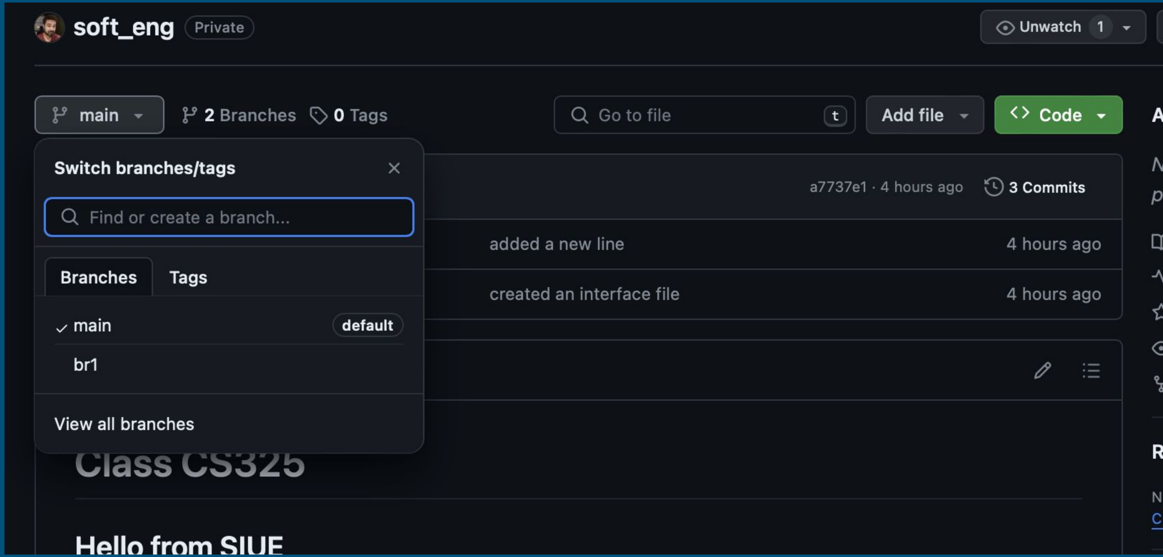
- Inside of **Insights** there is an option called **Network**, it will show the branches
- It will work for public repo, you can create an example repo in public mode and play with it
- It will work with private repo also if you pay for the github pro account

To create a branch, First step

- Create a branch with the following command
- `git branch br1` (please choose a meaningful name)
- Run the command : `git branch` (you will see two branches **main** and **br1**)
- In front of main there should be a *
- Now to change the branch head to br1, the command is:
  - `git checkout br1`
- Run `git branch` again the * should be in front of br1
- Now change the content of file_name.py file created in slide-18
- add it commit it and push it (not so soon, go to next slide)

To push from a new branch

- Command to push from a new branch is
  - `git push --set-upstream origin br1`
  - Change the branch name appropriately in all the above commands
- Go to github repo page
- Look under main button



added a new line 4 hours ago

created an interface file 4 hours ago

Class CS325

Hello from SIUE

Thank you and see you all in the next class