

# Conda, Git & GitHub

---

COGS 108 Fall 2025

Jason Chen

Week 2

[xic007@ucsd.edu](mailto:xic007@ucsd.edu)

OH: Tue 3-5 pm

# Reminder

- Practice Assignment: Due on TODAY (10/08)
- D1: Due on Friday (10/09)

# Datahub/JupyterLab



Do not forget to submit(time-stamps)

Submitted assignments		
Practice_COGS108_FA24	COGS108_FA24_A00	<a href="#">Fetch Feedback</a>
2024-09-28 07:25:28.062723 UTC		
2024-10-02 23:16:02.164214 UTC		
2024-10-04 05:57:15.640169 UTC		
PythonReview_COGS108_FA24	COGS108_FA24_A00	<a href="#">Fetch Feedback</a>
2024-10-01 02:44:25.028562 UTC		

## Validate

- Only pass public tests
- Validate != Submit
- might need to optimize your code if runs too long

# Datahub/JupyterLab



Do not re-submit after due dates

- Please double check before submitting
- JupyterLab won't store your code

Do not delete the original cell

- autograde based on cell ID

Note: Make sure you provide your GitHub username, **not the email address** you used to create the account, follow instructions, and avoid typos. You will not get credit for this section if typos are made.

In [1]: ID: cell-784114344a572182 Autograded answer

```
### BEGIN SOLUTION
PID = 'A1234567'
github_username = 'ShanEllis'
### END SOLUTION
```

In [2]: -

```
# if you navigate to this link
# it should open your GitHub page
# if it doesn't, fix your username above
'www.github.com/' + github_username
```

Out[2]: 'www.github.com/ShanEllis'

In [3]: Points: 0 ID: cell-216ded1c5ab2c280 Autograder tests

```
assert PID
assert github_username
```

ID: part2-intr Read-only

Part 2: Python (5.5 points)

# Conda



Conda is a package manager and environment manager for data-science tools.

It helps you:

- Install software libraries (like NumPy, Pandas, PyTorch)
- Manage versions of Python and packages
- Keep projects isolated from each other

Works across platforms (Mac / Windows / Linux) and supports multiple languages (Python, R, C, etc.).

# Conda virtual environment



A virtual environment is a self-contained workspace with its own Python version and packages.

Each project can have different dependencies without conflicts.

Example:

- Project A uses Python 3.9 and TensorFlow 2.6
- Project B uses Python 3.11 and PyTorch 2.2
- → Both can coexist safely.



# Conda virtual environment



Why do we need virtual environments?

- Prevents the “dependency hell” problem.
- Lets you reproduce results easily — crucial in data science.
- You can share environments using a .yaml file.

If you're using windows powershell and it cannot recognize `conda`

1. Make sure conda is properly installed

2. Run PowerShell as Administrator  
Right-click PowerShell → "Run as Administrator".

3. Find where Anaconda actually is  
`Get-Command conda | Select-Object Source`

4. Add the Anaconda paths to the system PATH variable (replace `\path\to\conda` with the actual path you found in step 3)

```
setx PATH  
"$($env:PATH);C:\path\to\conda;C:\path\to\conda\Scripts;C:\path\to\conda\Library\bin"
```

5. Restart PowerShell so the new PATH takes effect

6. Verify  
`conda --version`



# If the code can't find a package called unixodbc

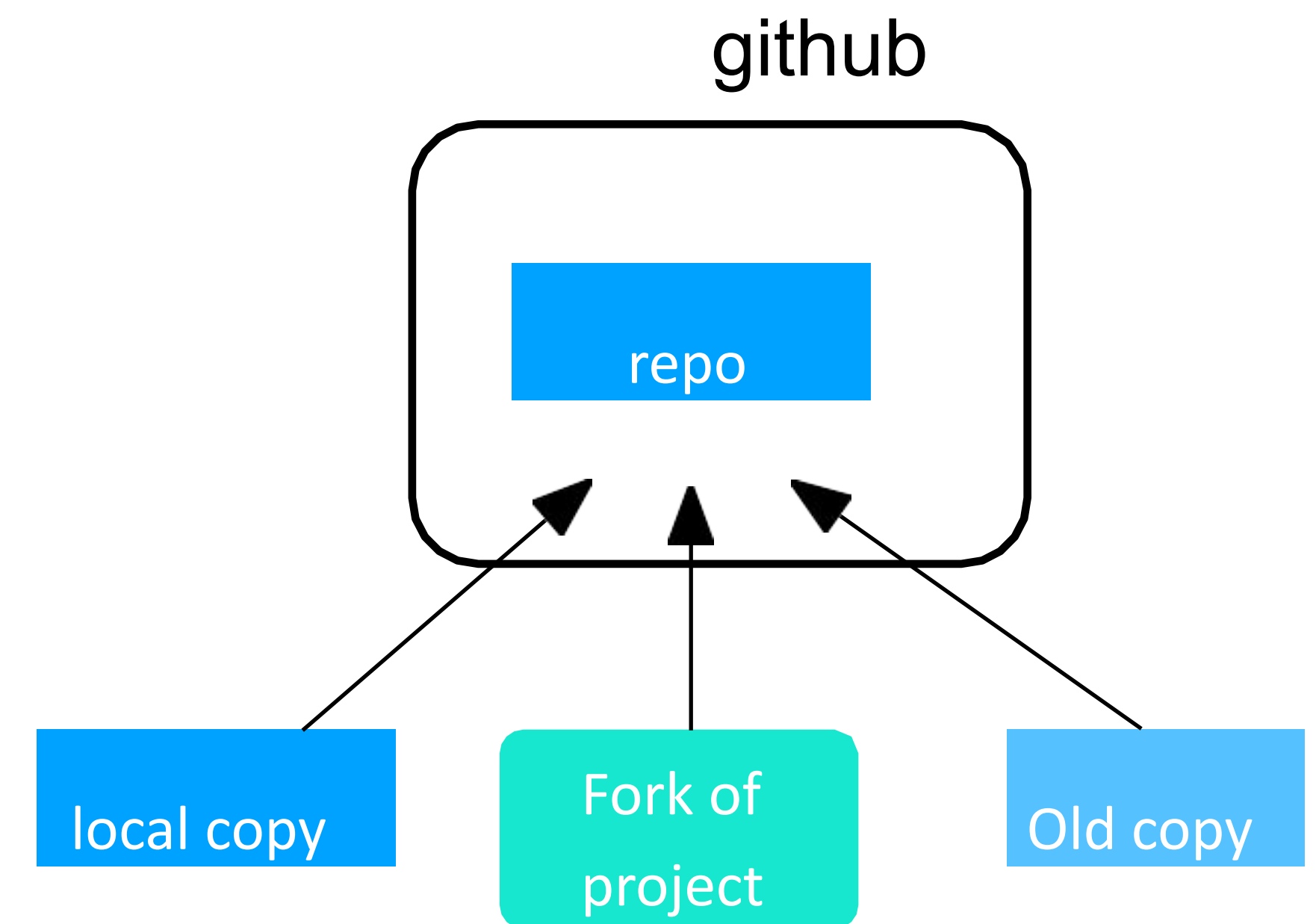
1. Open `conda_env_COGS108_FA25.yml`
2. Comment out the line containing `unixodbc`  
# - unixodbc

# Git

- Version control system!
- Go to <https://git-scm.com/downloads>
  - Choose your Operating System (Windows/OS X/Linux)
  - Follow the steps specific to your OS
  - Verify installation: In terminal type "git --version"

# What is git + GitHub?

- Somewhere online to store a copy of a project (Github)
- Plus a tool to interact with this copy (Git)
  - Command line and desktop versions
- A way of keeping track of changes you make to a project
- Does everyone have a GitHub account?



# Why use git + GitHub?

- Collaboration: Git allows you to work on code projects with other people. It's the preferred tool for many projects, like...
  - Python: <https://github.com/python/cpython>
  - Jupyter: <https://github.com/jupyter/>
  - COGS 108: <https://github.com/COGS108/>
- Backup
- Version control (*undo* on a large scale)
- Code reuse

# GitHub



## 1. Fork

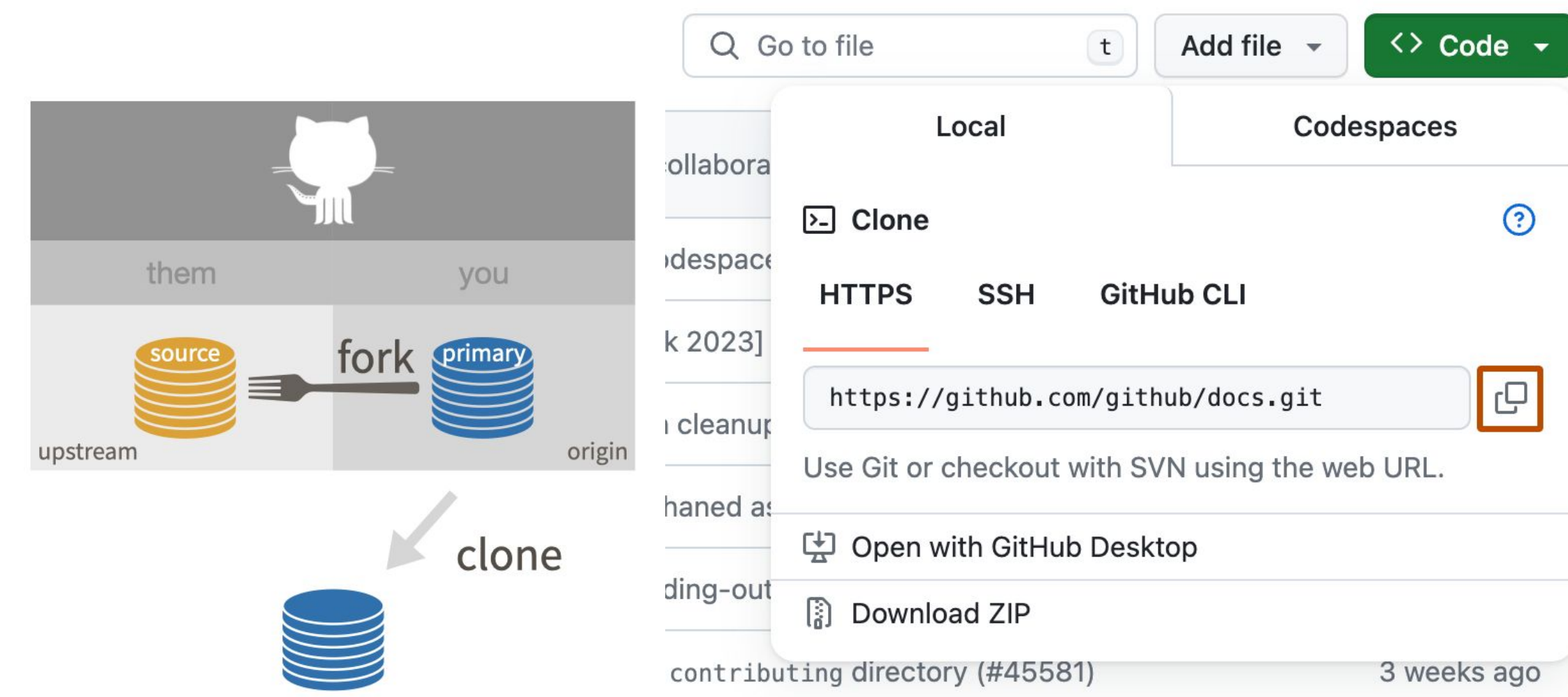
A “fork” is just an independent copy of a repository that you can develop on without affecting the original.

## 2. Clone

Creates a local copy of a repository on your machine.

## 3. You can “push” changes back to the remote repository if you have permissions.

```
$ git clone https://github.com/user/repo.git
```

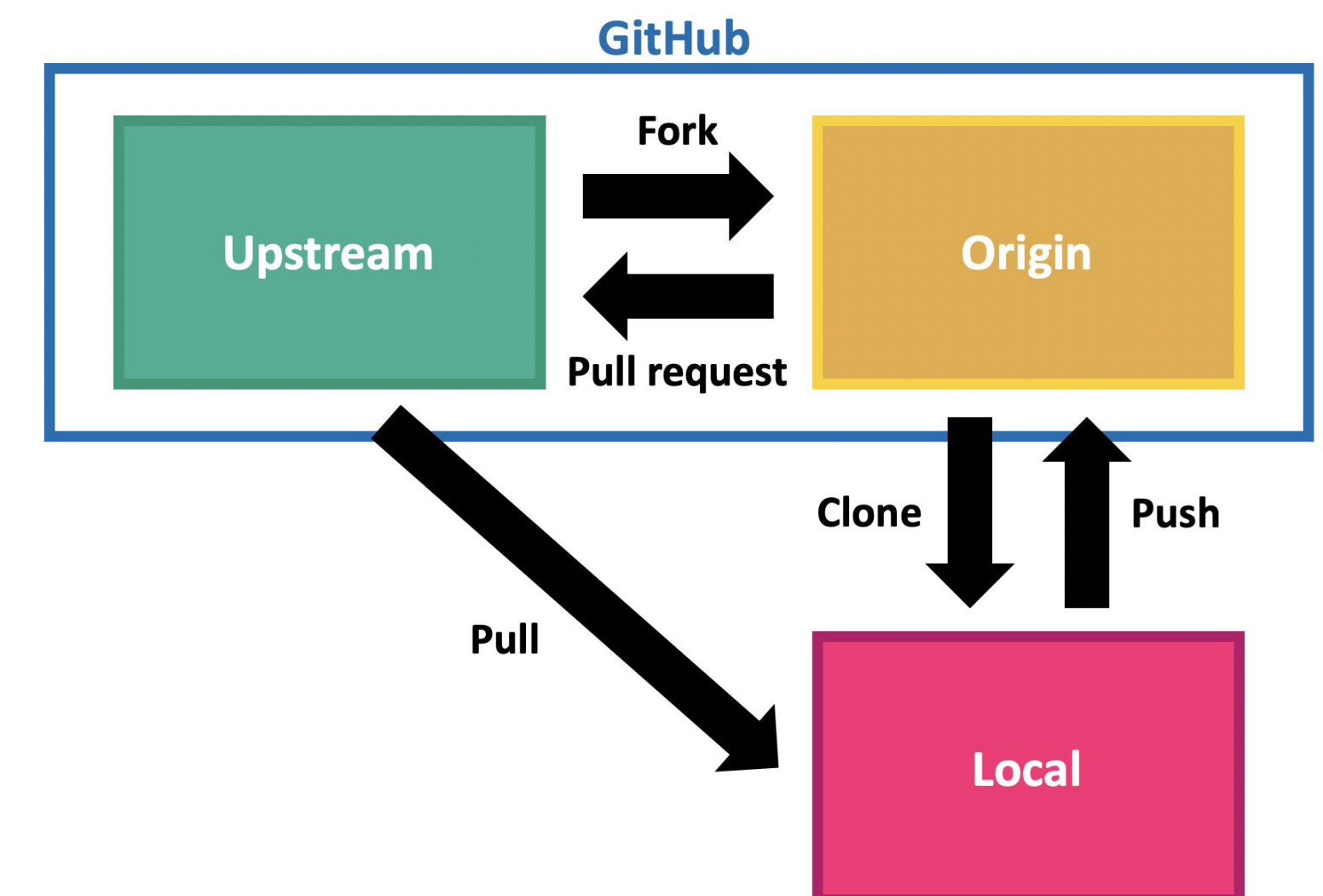
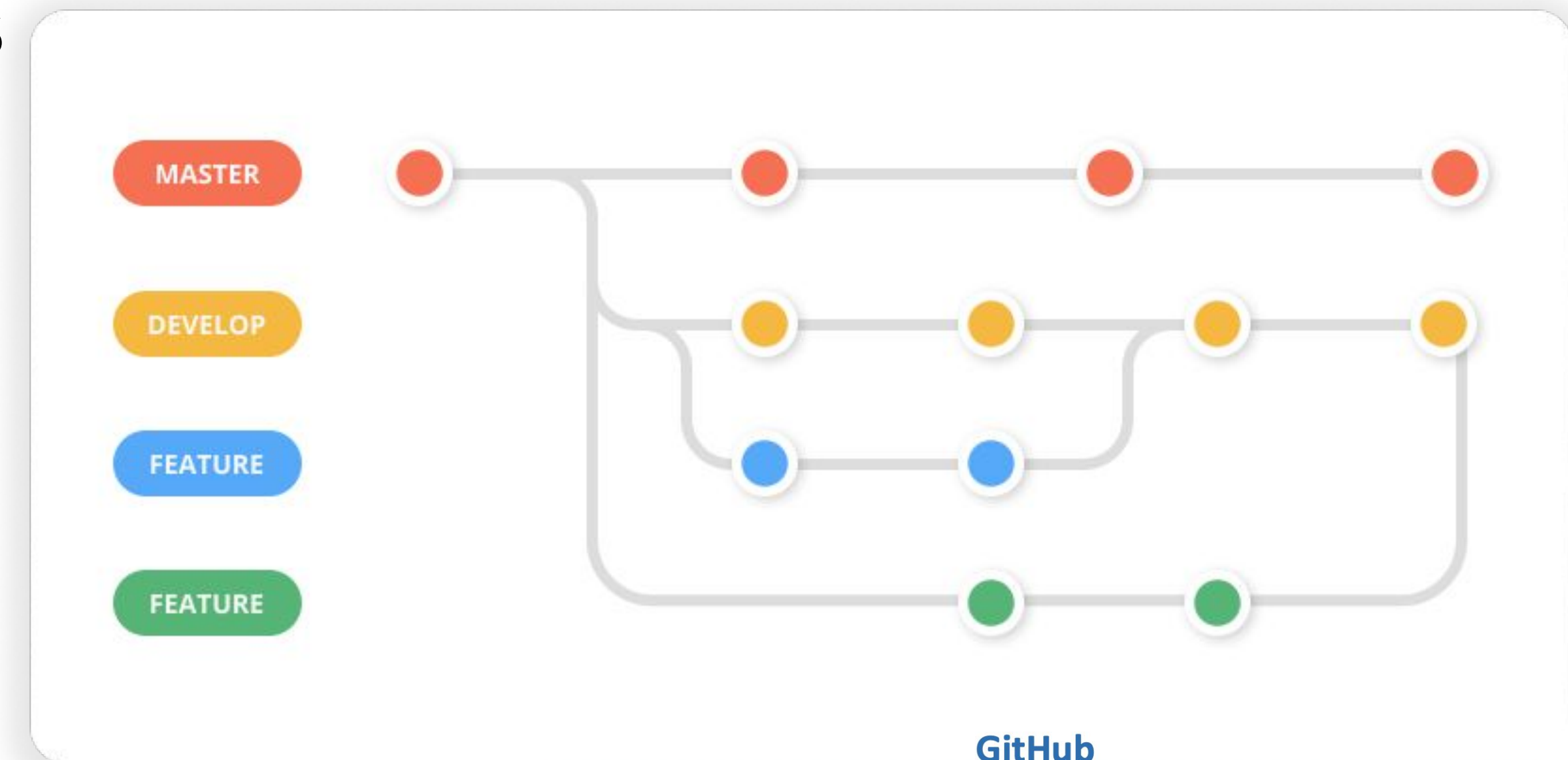


# Branching

Branch will be independent of the clean, functioning “master” code and is a safe place for you to delete, modify and add code.

## Pull Request

Asks the maintainers of the original repository to take a look at and hopefully integrate your code changes into their repository.





GitHub Issues are used to track tasks, enhancements, and bugs for a project, allowing team members to discuss and manage work collaboratively.

GitHub Projects is a tool to organize and track the progress of issues and pull requests in a board view, helping teams visualize workflows and prioritize tasks.

🔒 OctoArcade Invaders

📅 Planning

📅 Sprint Board

📅 Alpha

📅 Roadmap

📅 My work

📅 Features

📅 Priority

📅 By person

📅 Status Board

📅 By status

📅 By Sprint

📅 Done

Filter by keyword or by field

🕒 Not Started 19 Estimate: 37

🕒 planning-tracking-demo #810

Beta go-no-go meeting

🕒 planning-tracking-demo #800

Save score across levels

🕒 planning-tracking-demo #784

Interviews with media outlets

epic

🕒 Draft

Enable for teams

🕒 planning-tracking-demo #1161

tweak difficulty

🕒 planning-tracking-demo #1167

Update README.md

🕒 Draft

Prevent the Konami code from bringing down all of GitHub

+ Add item

🕒 Planning 19 Estimate: 109

🕒 planning-tracking-demo #823

Updates and bug fixes to engine from Beta

bug demo

🕒 planning-tracking-demo #824

Beta signup page

need help

🕒 planning-tracking-demo #806

[Tracking] Upsell / Growth experience

backlog feature

🕒 planning-tracking-demo #818

Account subscription design

🕒 planning-tracking-demo #828

Acquire domain for launch

🕒 planning-tracking-demo #832

Final creative shots from game

🕒 planning-tracking-demo #829

+ Add item

🕒 Building 8 Estimate: 40

🕒 planning-tracking-demo #1160

Update documentation

🕒 planning-tracking-demo #814

Updates to collision logic

enhancement

🕒 planning-tracking-demo #816

Free and paid levels

need help

🕒 planning-tracking-demo #831

Documentation and Support

need help

🕒 planning-tracking-demo #821

Updates to alien, beam, bomb and cannon sprites

#370

🕒 planning-tracking-demo #802

Updates to velocity of the ship and alien movements

+ Add item

🕒 Review 5 Estimate: 17

🕒 planning-tracking-demo #822

Hero site - Development

#12 #1160 in-review task urgent

🕒 planning-tracking-demo #808

General bug fixes from Alpha feedback

#992

🕒 planning-tracking-demo #1151

Design new launch screen

#374 web

🕒 planning-tracking-demo #793

Polished alien, beam, and cannon sprite files

🕒 planning-tracking-demo #1101

[Tracking] Integrate payments system

backlog feature

+ Add item

🔒 OctoArcade Invaders

📅 Planning

📅 Sprint Board

📅 Alpha

📅 Roadmap

📅 My work

📅 Features

📅 Priority

📅 By person

📅 Status Board

📅 By status

📅 By Sprint

Title	Status	Assignees	Priority	Labels
👤 Squad 1 6				
1 [Tracking] Allow users to perform a manual resetting #1100	Done ✓	👤 shiftkey	🌟🌟🌟 High	duplicate enhancement epic f
2 Integrate with Commerce Service #3	Done ✓	👤 rileybroughten	🌟🌟🌟 High	
3 Hero site - Development #822	Review ▶	👤 azenMatt	! Urgent	in-review task urgent
4 Integrate with backend systems #809	Done ✓	👤 keisaacson	! Urgent	demo prototype urgent
5 Prevent the Konami code from bringing down all of GitHub	Not Started ⌚			
6 investigate latency	Not Started ⌚			
+ Add item				
👤 Squad 2 9				
7 Integrate with Leaderboard Service #811	Done ✓	👤 dusave and jolem	🌟🌟🌟 High	enhancement
8 [Tracking] Upsell / Growth experience #806	Planning 📅	👤 mariorod	🌟🌟 Medium	backlog feature
9 Account subscription design #818	Planning 📅	👤 azenMatt	🌟 Low	
10 Free and paid levels #816	Building 🛠️	👤 JannesPeters	🌟 Low	need help



You might run into this error when using git:

```
remote: Invalid username or token. Password authentication is not supported for Git operations.
```

# We need to create a ssh key in the terminal and add it to github

On your terminal, create an SSH key

```
ssh-keygen -t ed25519 -C "your\_email@example.com"
```

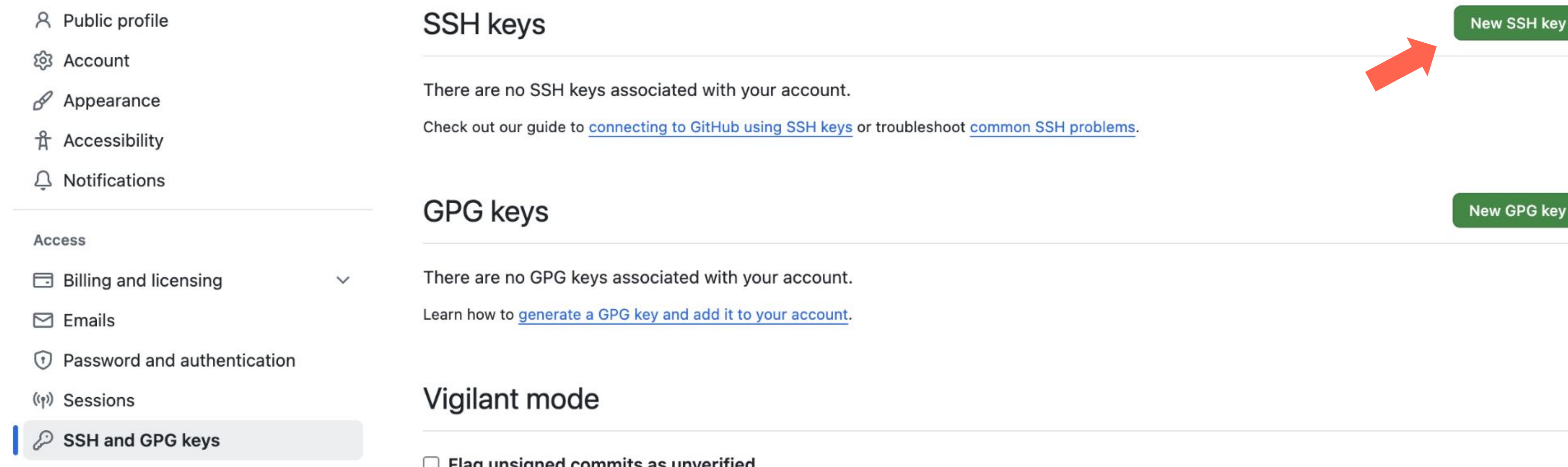
When prompted 'Enter file in which to save the key', simply press Enter

When prompted to set passcode, simply press Enter

Print out the key and copy it. Enter the following in your terminal:

```
cat ~/.ssh/id_ed25519.pub
```

Go to Github Settings -> SSH and GPG keys and add your key



The screenshot shows the GitHub Settings interface. On the left is a sidebar with navigation links: Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and licensing, Emails, Password and authentication, Sessions, and SSH and GPG keys (which is highlighted with a blue bar). The main content area is titled 'SSH keys' and contains the text 'There are no SSH keys associated with your account.' followed by a link to a guide. Below this is a 'New SSH key' button. A red arrow points from the text 'add your key' in the instructions to this button. Further down, there is a 'GPG keys' section with similar text and a 'New GPG key' button. At the bottom, there is a 'Vigilant mode' section with a checkbox labeled 'Flag unsigned commits as unverified'.

## Test the connection

```
ssh -T git@github.com
```

You might see this message, just type yes

```
The authenticity of host 'github.com (140.82.116.3)' can't be established.  
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.  
This key is not known by any other names  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

You have succeeded if you see this message:

```
Hi z5yǐng! You've successfully authenticated, but GitHub does not provide shell access.
```

# Demo of GitHub Repo and Commands

- Cloning a repo on your local machine
- Working on your local repo (making changes to files)
- Stage, commit and push these changes to your Github repo
- Commands you should know:
  - Git clone -This will download the latest version of the repo to your local PC
  - Git status (not really needed – but really helpful)
  - Git add - This adds the changes in the staging area
  - Git commit -m "" - This will save a "Snapshot" of your most recent changes
  - Git push - This will upload your local changes to your GitHub Repo
  - Git pull - This will update your local repo to the latest version which is on GitHub

# Your time to ...

- Talk to your classmates to find potential teammates!
- Work on D1

# Section Materials

Section materials can be accessed at:

[https://github.com/JasonC1217/COGS108\\_FA25\\_B07-B08](https://github.com/JasonC1217/COGS108_FA25_B07-B08)



**SCAN ME**

Questions?