



What's Git & GitHub

Alejandro G. Rodriguez Ramos

UIPR Aguadilla

Inter Compute Science Association



interAGUADILLA

What's Git?



- Git is a **free** “Distributed Version Control System” (DVCS).
 - It's fully open source.
- Its job is to keep track of the changes made to the files as you or your team collaborate on the repository.
 - Repositories or “repo” for short, is the name which Git uses to refer to its projects.
- Its DVCS will keep a detailed history of the changes made to the repository.
- Allows Trunk Based Development (TBD) for seamless creation and integration of features to the main branch or trunk code.

What's GitHub?



- GitHub is a **free** cloud hosting service for Git repositories.
 - Contains optional paid features.
- It provides developers with a robust tool-rich environment to take advantage of Git's features.
- Has an expansive community of over **100 million developers**, as well as over **420 million repositories** on their network.
- Created with team work at its core, it offers seamless integration of collaborative strategies into its development process.

Downloading and setting up Git:

- First, we need to verify that Git is not installed.
 - If on Windows, we use the search bar and type: cmd.
 - If on Mac, we will look for the terminal app.
 - Once on either the "cmd" or "terminal", we will type: ***git***. (Won't run if not installed).
- Second, we need to download git itself.
 - Make sure to download the version corresponding to your OS.
 - <https://git-scm.com/downloads> **(Git's Homepage).**
 - Leave the default options selected.
- Finally, once installed, open the terminal/cmd.
 - Once in the terminal/cmd type: ***git*** to verify if properly installed.

Creating a GitHub Account:

- First, we need to install GitHub.
 - We head on over to GitHub's website where we can go ahead and click "sign-up".
 - <https://github.com> (GitHub's homepage).
- After we create the account, we need verify it via email.
 - Make sure to not use your professional email to sign-up, since companies can look at your GitHub page.
- Once the email is verified, you've completed the set-up.
 - You can finally begin exploring all that GitHub has to offer.
 - Make your first repository, which can be either public or private.
 - Explore all the public repositories available to both learn from or even contribute if you so desire.

Installing Visual Studio Code:

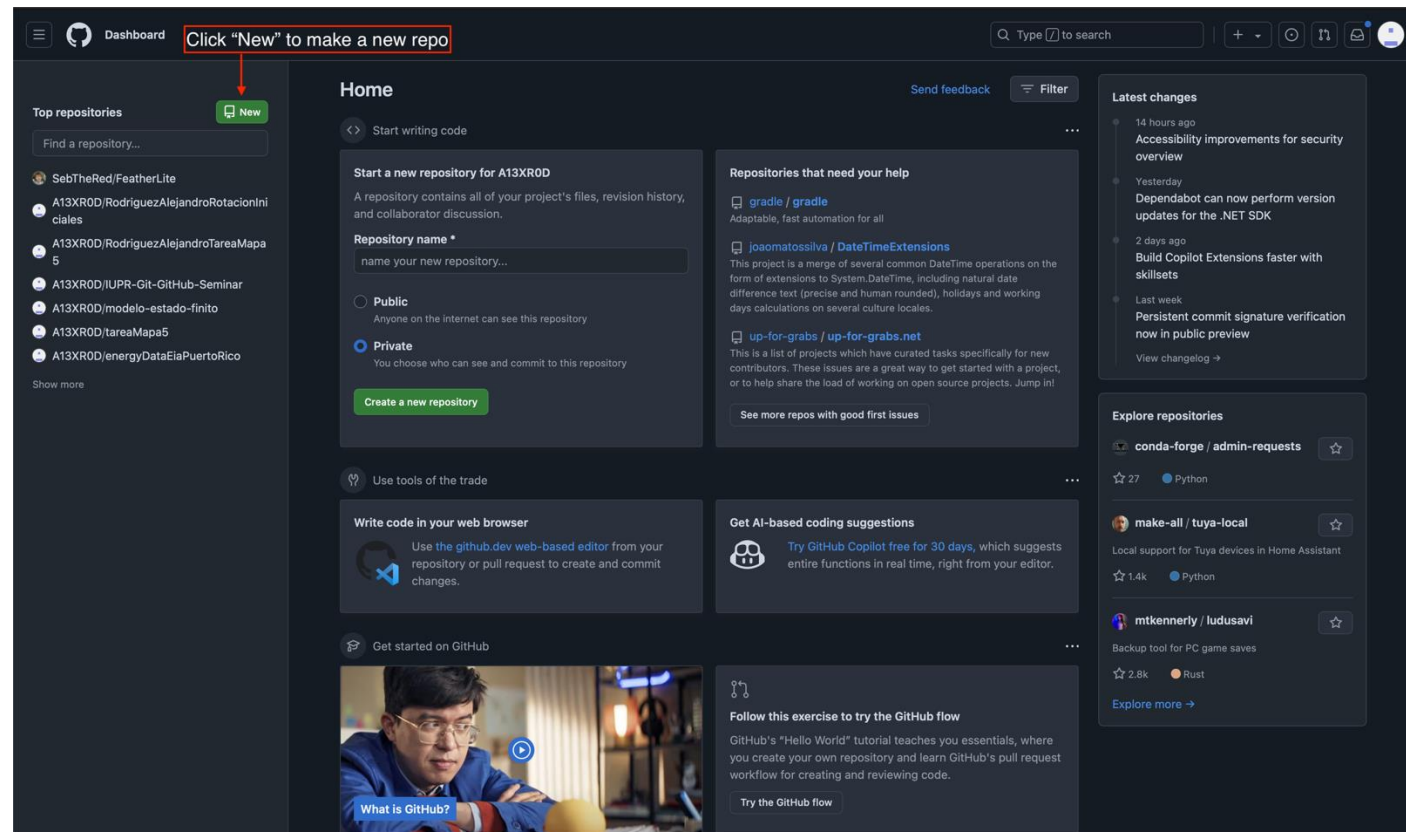
- First, we head on over to VS Code's website.
 - We head on over to GitHub's website where we can go ahead and click "sign-up".
 - <https://code.visualstudio.com> (**VS Code's homepage**).
 - VS Code is probably one if not the most versatile, customizable and powerful text editors (**NOT AN IDE**).
- We then proceed to download then we install onto our machine.
 - Follow the steps required for the install process, you can leave the default settings.
 - Make sure you download the version specific to your respective OS.
- Post installation, we can begin installing our extensions.
 - VS Code offers a huge library of both free or paid extensions for our text editor.

GitHub Extensions for VS Code:

- Once in VS Code, head on over to the extensions tab.
 - This is the icon with the three connected squares with one offset to the top right.
 - You can use the shortcuts “Shift+cmd+X on MacOS” and “Ctrl+Shift+X on Windows” to quickly access it.
- There are three extensions we’re going to install.
 - GitHub Actions: (This fully integrates most if not all the necessary tools to use GitHub via VS Code).
 - GitHub Codespaces: (This is a more advanced feature for team collaborations, allowing a customizable VE).
 - GitHub Pull Requests: (This lets you manage pull requests directly from GitHub).
- After this is all done, we can link our GitHub to VS Code.
 - This will provide us with a GUI, which allows us to get a visual representation of the actions taking place.
 - In my opinion, this method is better than using the *GitHub desktop app* and the *GitHub CLI*.
 - This build, provides us all the tools to seamlessly streamline the upload of the code itself to GitHub.

Creating a Personal Repo:

- Once on your GitHub dashboard, you'll be able to make a personal repo by clicking on the **"new" button**.



Creating a Personal Repo:

- After filling each of the respective blocks you can then hit the “**Create Repo**” button found on the bottom right corner.

New repository

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 * Repository name *

A13XR0D /

Great repository names are short and memorable. Need inspiration? How about [special-palm-tree](#) ?

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](#).

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

ⓘ You are creating a public repository in your personal account.

Create repository

Once you've filled in the respective block, just click create repo and you're done!

Seminar's practice GitHub Repo:

- After creating your GitHub account, you can use this QR Code to access the GitHub Repo.
- This repo contains useful resources to help you in the future.

Alternate Link:

<https://github.com/A13XR0D/UIPR-Git-GitHub-Seminar>

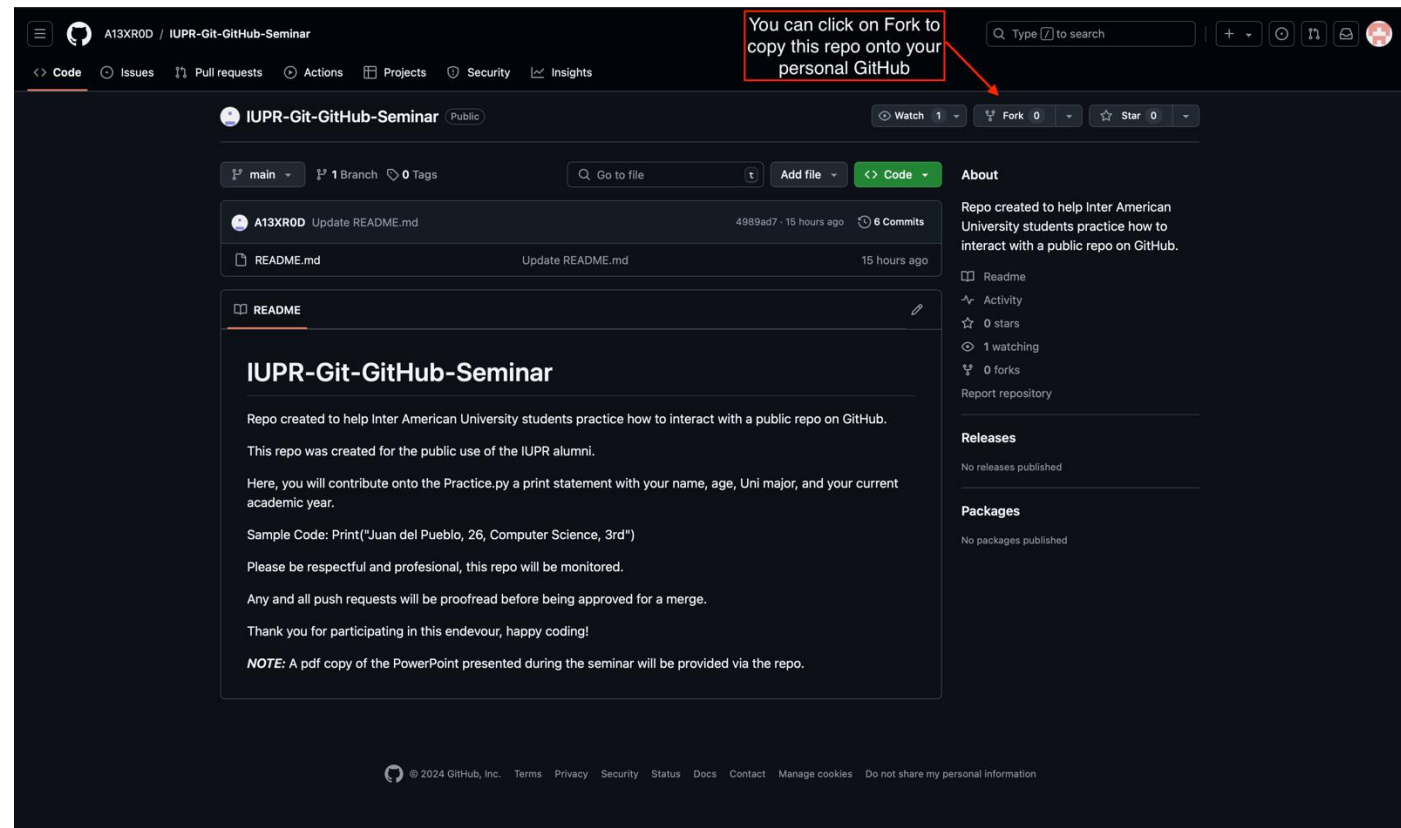


After the Accessing the Repo:

- Once in the GitHub repo, take a second to look around.
 - I recommend taking a quick second to look around the repo.
 - This way you know if this is a repo you'd be interested in contributing to.
- Read the description on the repo.
 - The description can be a quick way of getting the main purpose or functions of the current repo.
- Always make sure to look through the README.md.
 - This is a good way of getting a better understanding of the project you're trying to contribute to.
 - Sifting through the README.md, is vital.
 - Reading the README.md allows you to better understand both the rules, structure or even the base idea of the repo you're contributing to.

Contributing to a public Repo:

- Once you're in the public repo's dashboard, you can make a **fork** of the public repo to add a copy to your GitHub account.



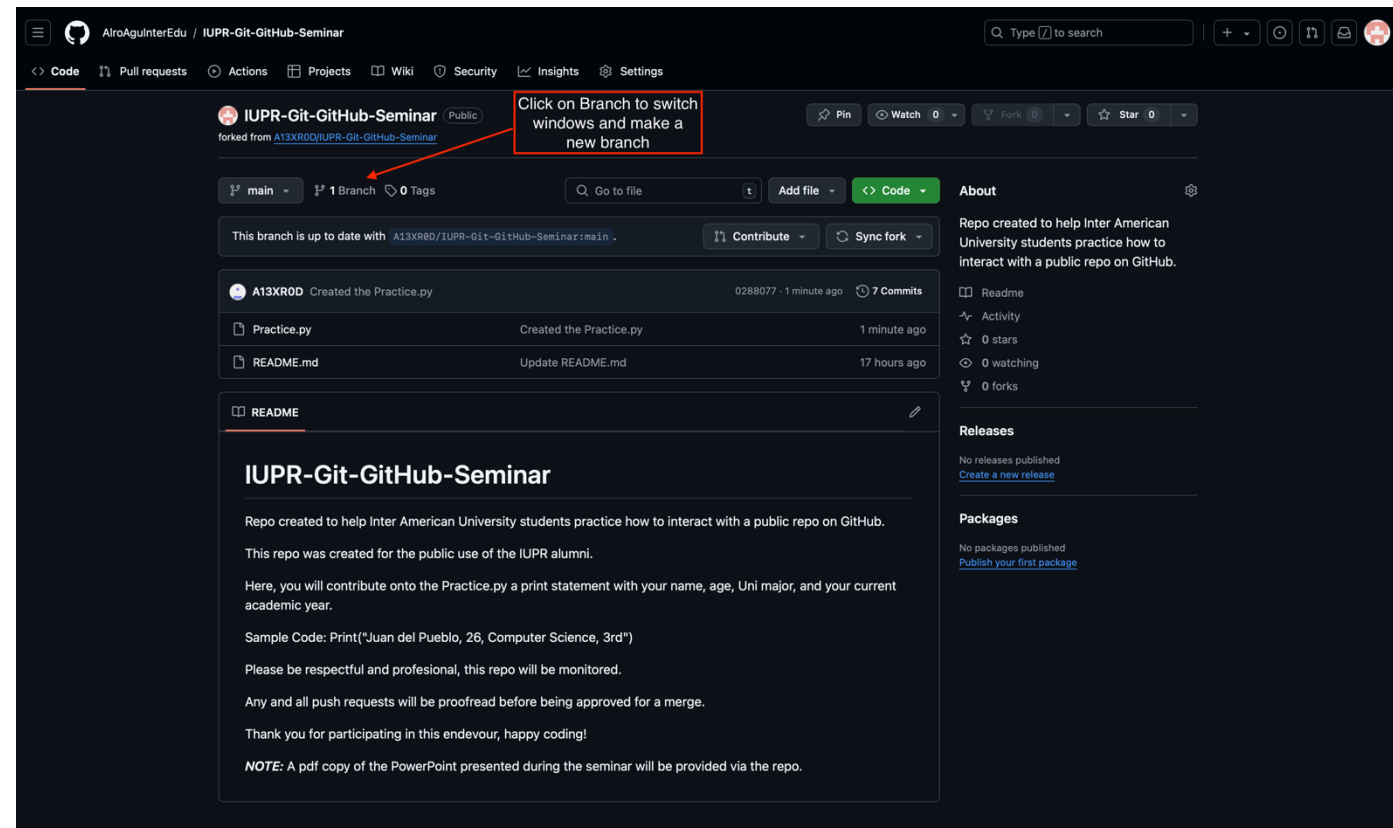
Contributing to a public Repo:

- Make sure to double check that the blocks are filled in as desired, if not. Make any changes you deem necessary and proceed to click on **“Create Fork”**.

The screenshot shows the GitHub interface for creating a new fork. The page title is 'Create a new fork'. Below the title, there is a brief explanation of forking. The form includes fields for 'Owner' (set to 'AlroAgulInterEdu') and 'Repository name' (set to 'IUPR-GitHub-Seminar'). A green checkmark indicates the repository name is available. There is a 'Description' field with the text 'Repo created to help Inter American University students practice how to interact with a public repo on Git'. A checkbox 'Copy the main branch only' is checked. At the bottom, there is a green 'Create fork' button. A red box with an arrow points to this button, containing the text: 'If everything looks good, you can click “Create fork”'.

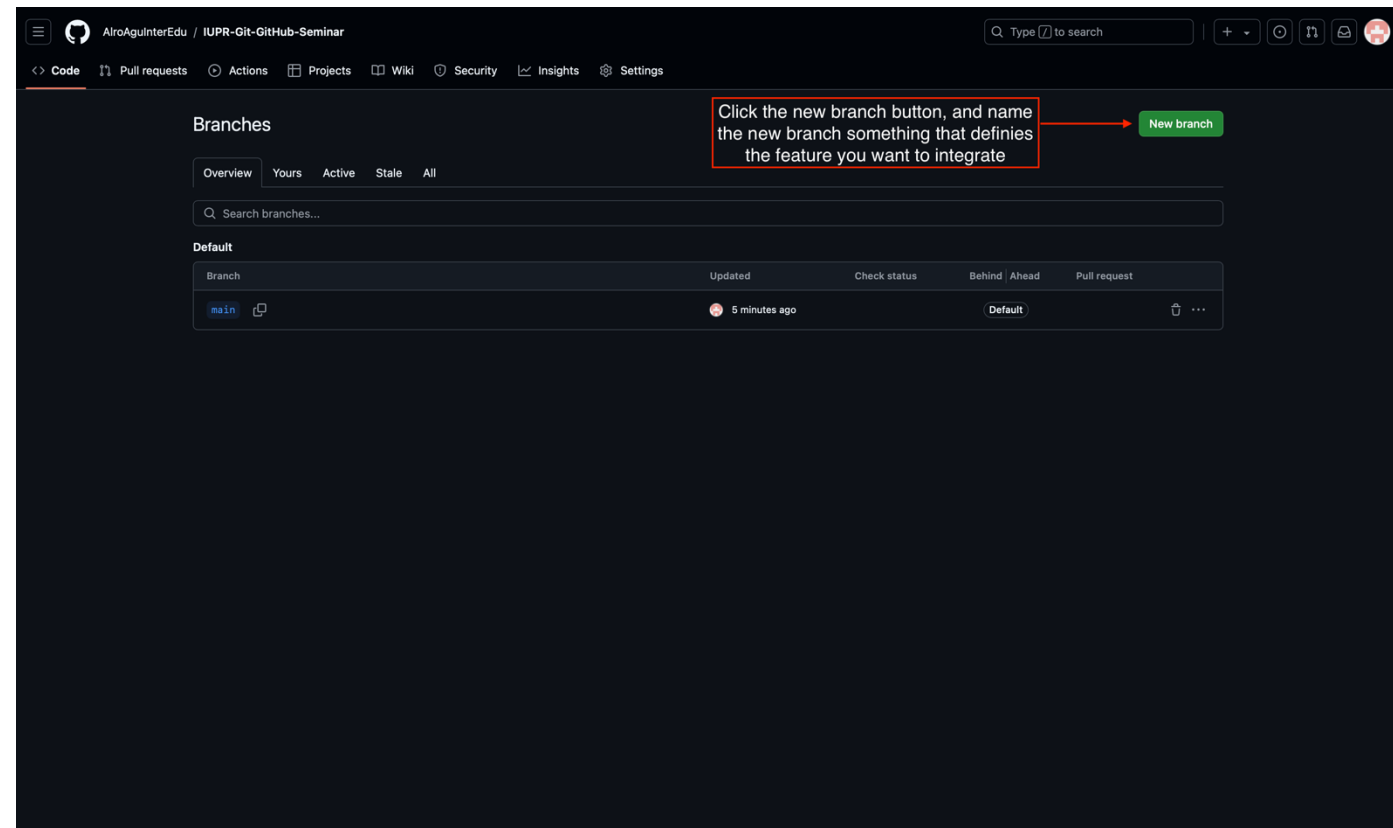
Contributing to a public Repo:

- After a quick little loading screen, you can access your fork and create a new branch. This way you can better track your contributions.



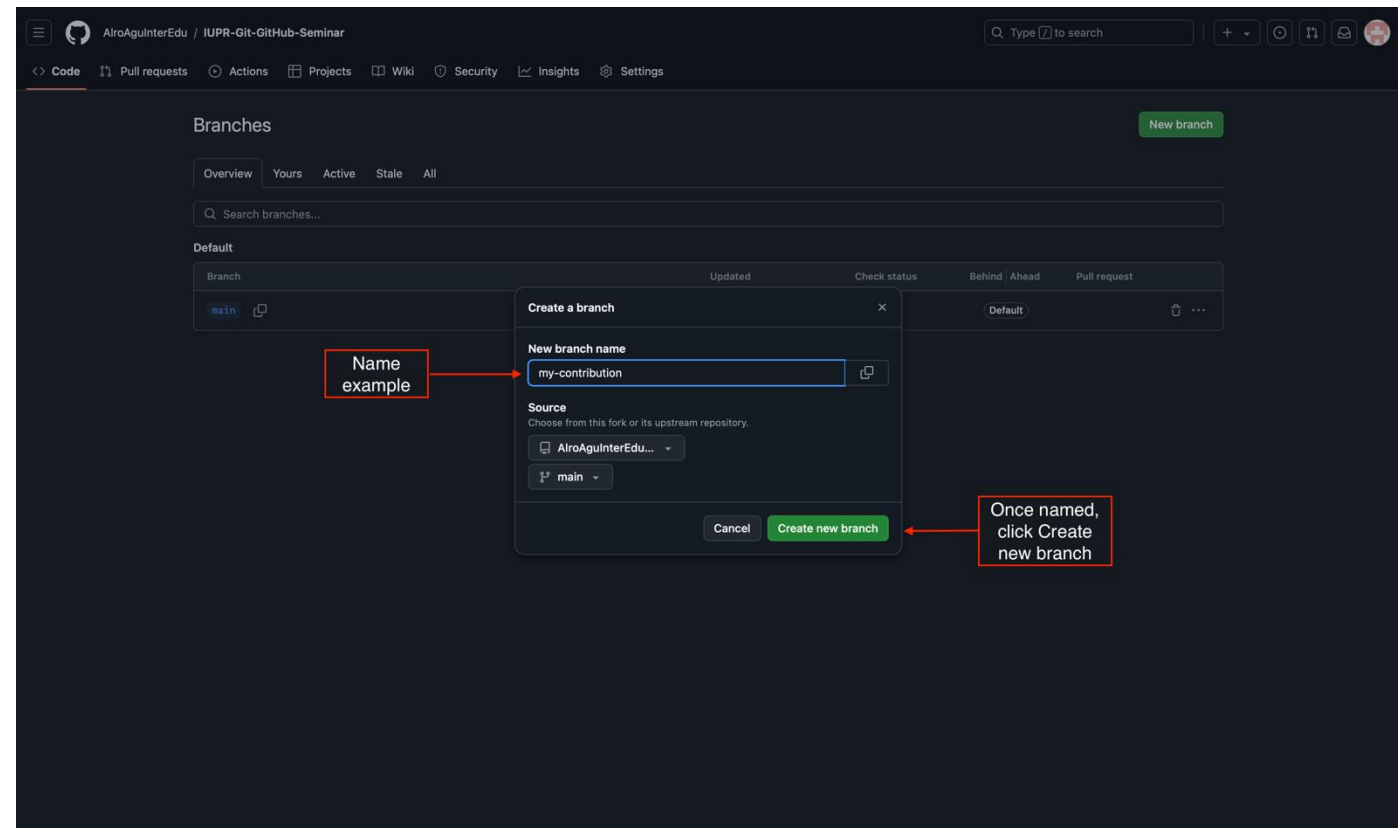
Contributing to a public Repo:

- After selecting the Branches tab. Proceed to click on the “**New Branch**” button on the top right corner of the screen.



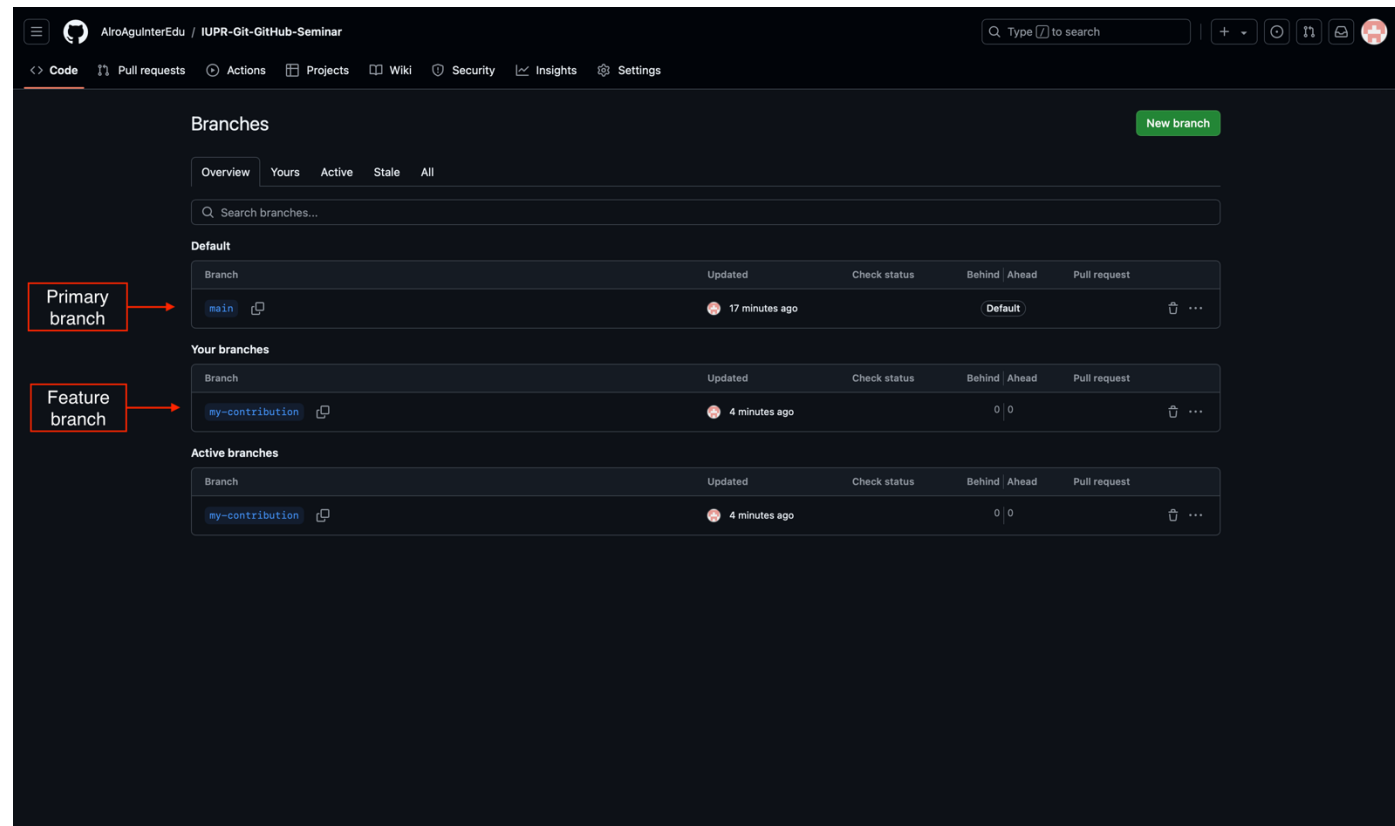
Contributing to a public Repo:

- Post clicking the button.
- You'll be prompted with a small window where you can name your branch.



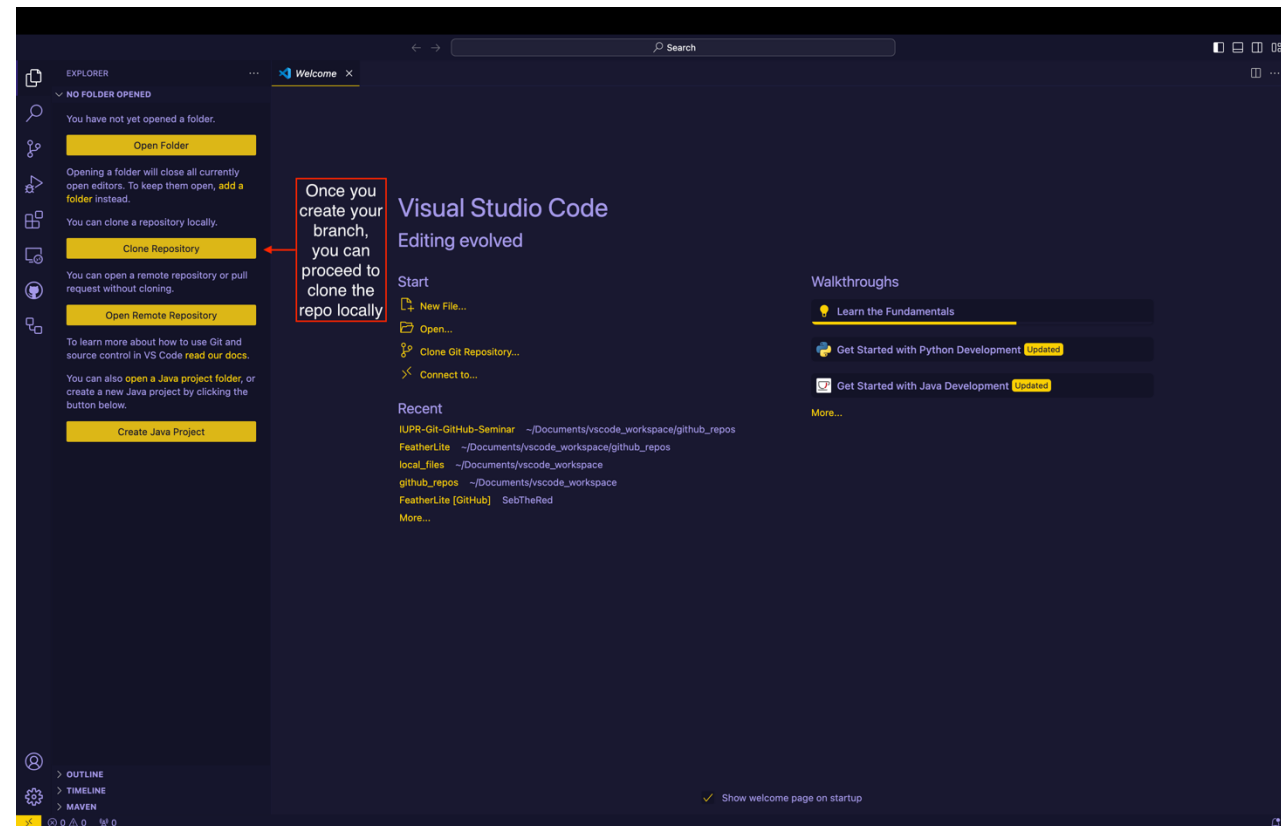
Contributing to a public Repo:

- As you can see, once the repo is created. It will appear in the branches tab.



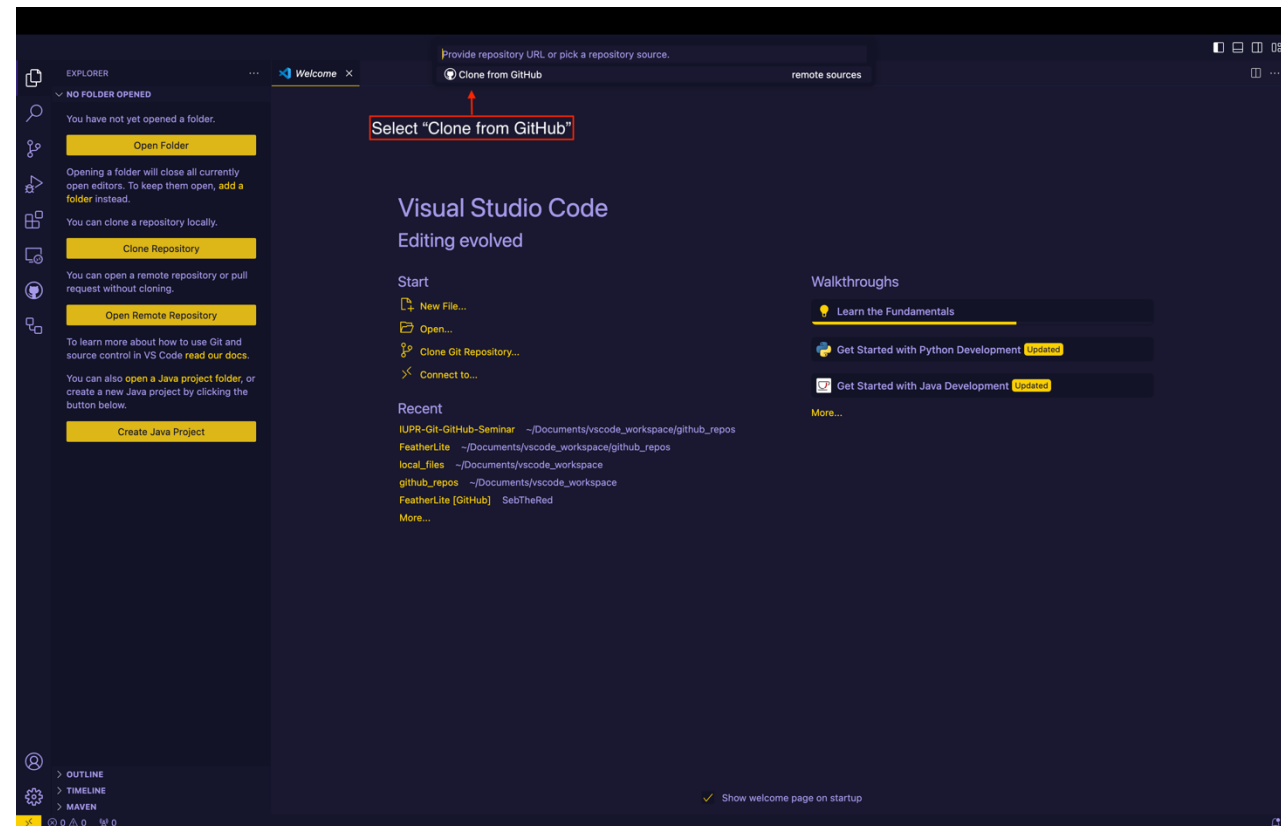
Contributing to a public Repo:

- Now we proceed onto VS Code.
- Where we can now create a clone of the repository onto our local file system in our machine.



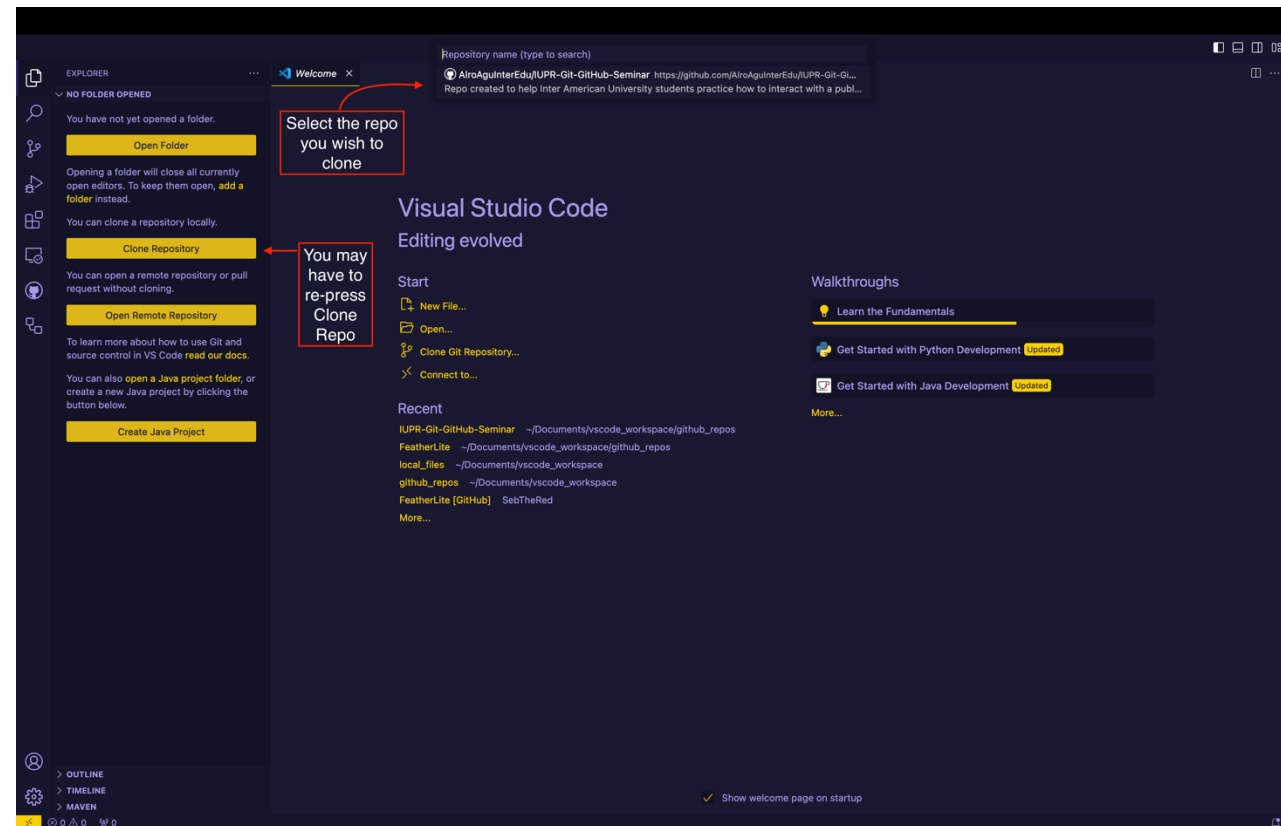
Contributing to a public Repo:

- After pressing the "**Clone Repo**" button, you'll be prompted with this window. Asking where you'd like to clone the repo from.



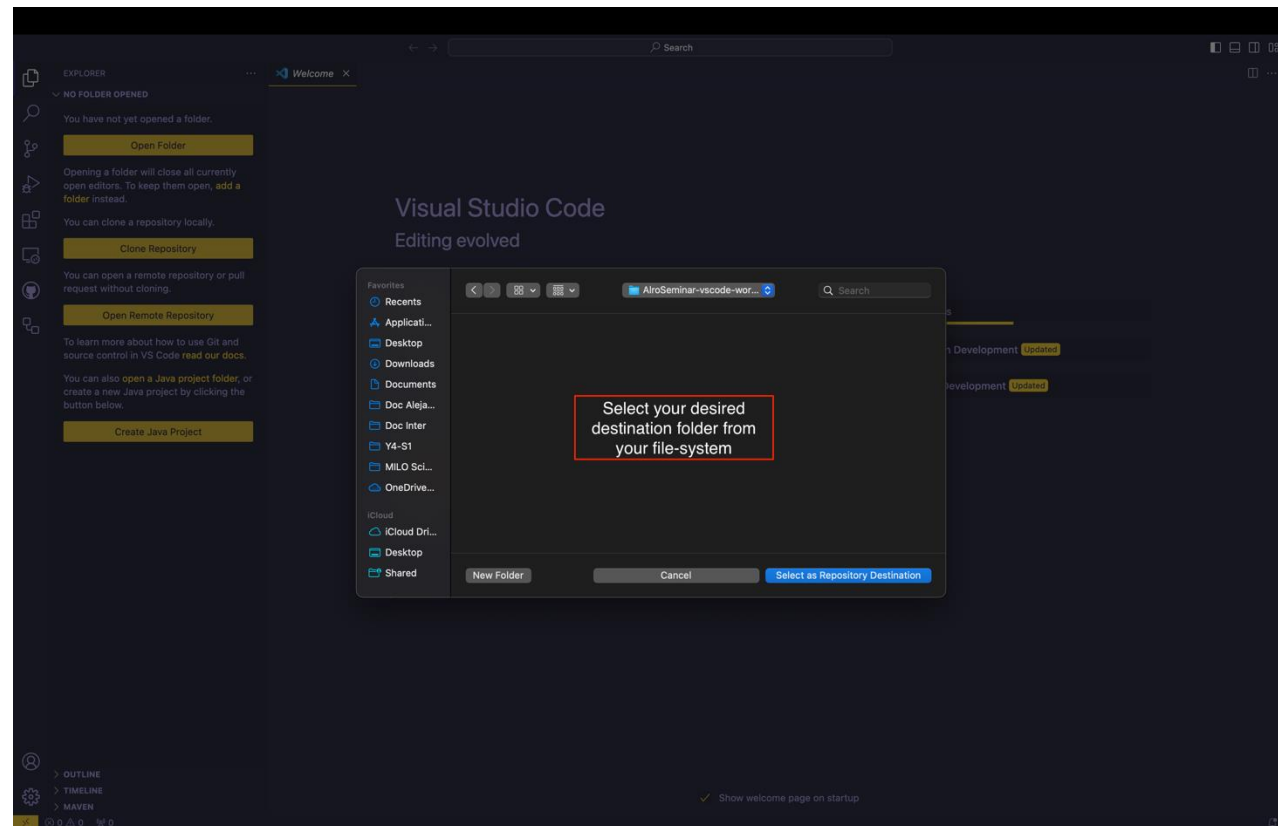
Contributing to a public Repo:

- Now you select the specific repo you want to clone onto your local file system.
- **NOTE** you may have to re-click the clone repo button.



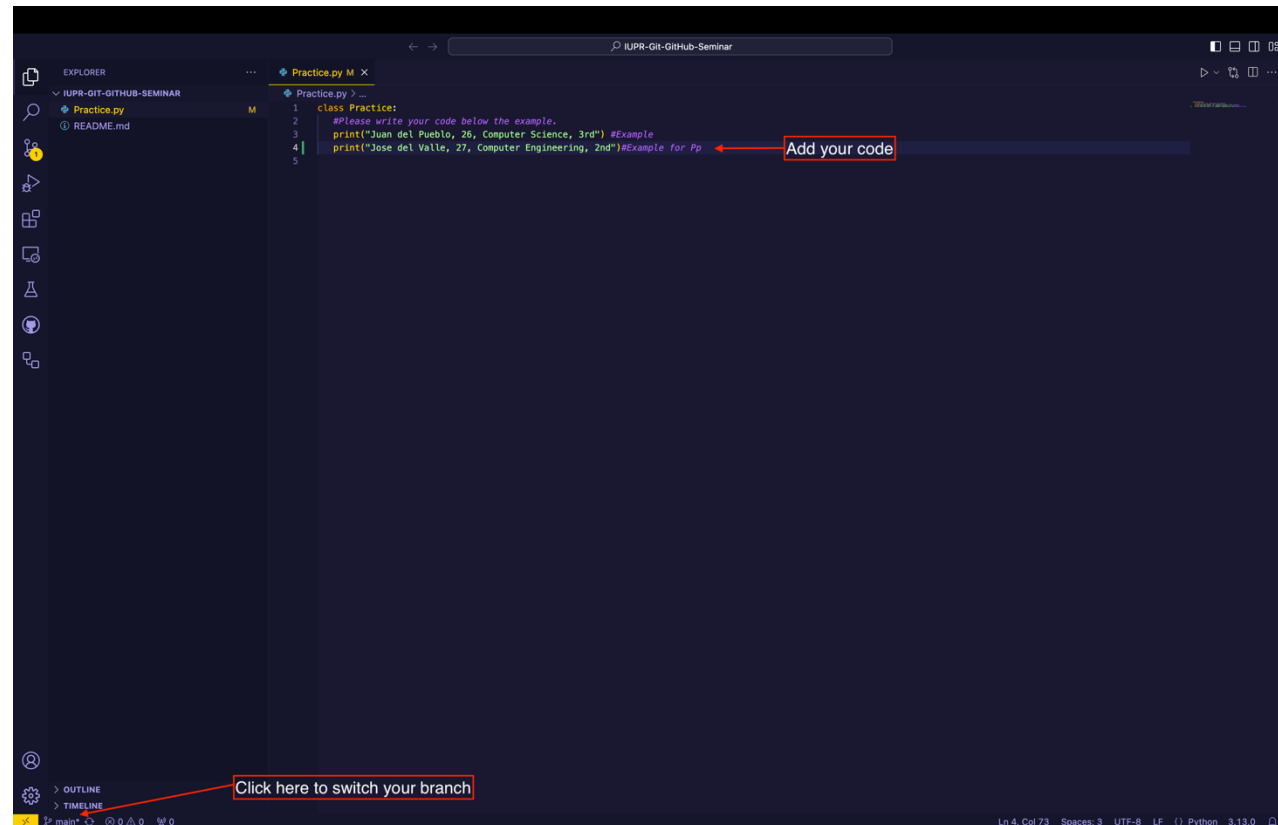
Contributing to a public Repo:

- After selecting the repo, you'll be prompted on where you'd like to store the specified repo we're cloning.



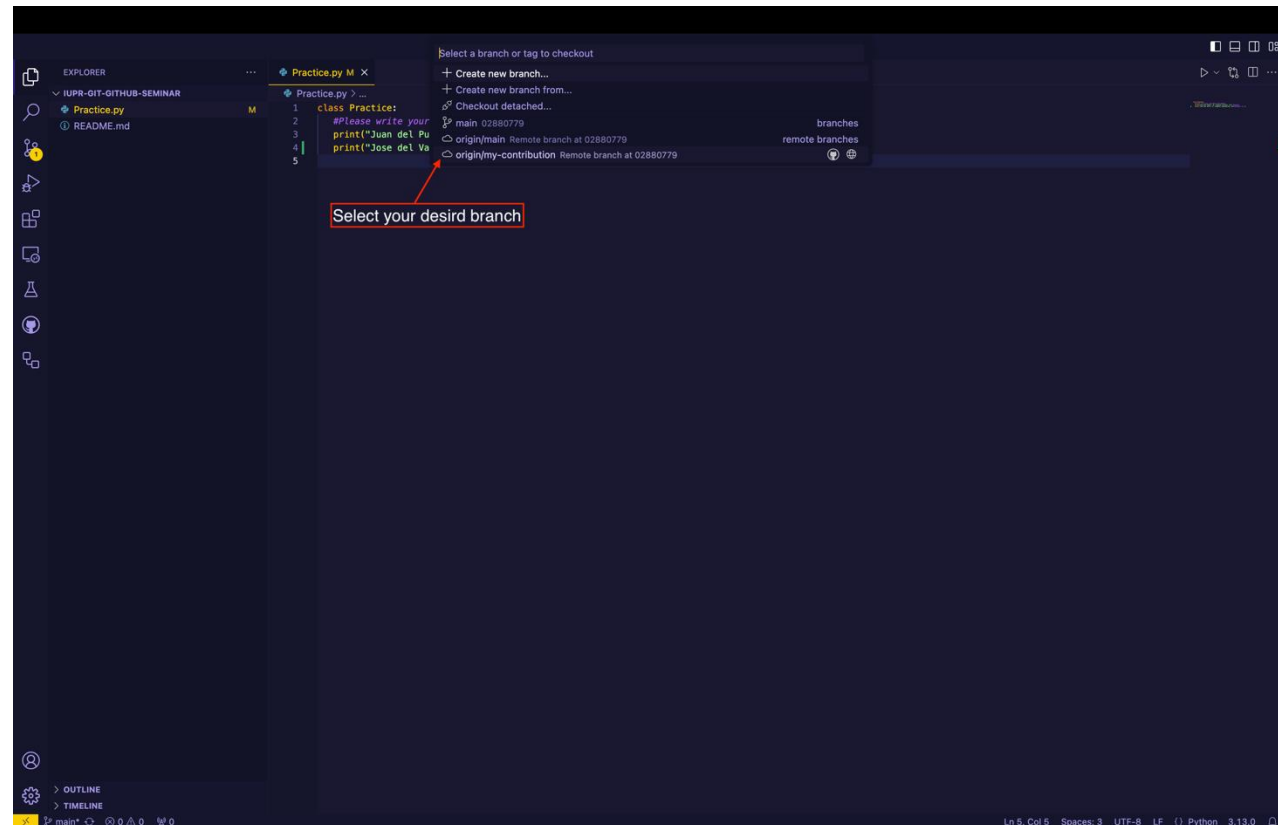
Contributing to a public Repo:

- You can now add your code and switch to the branch you'd like to make the contribution to.
- This can be done before adding the code.



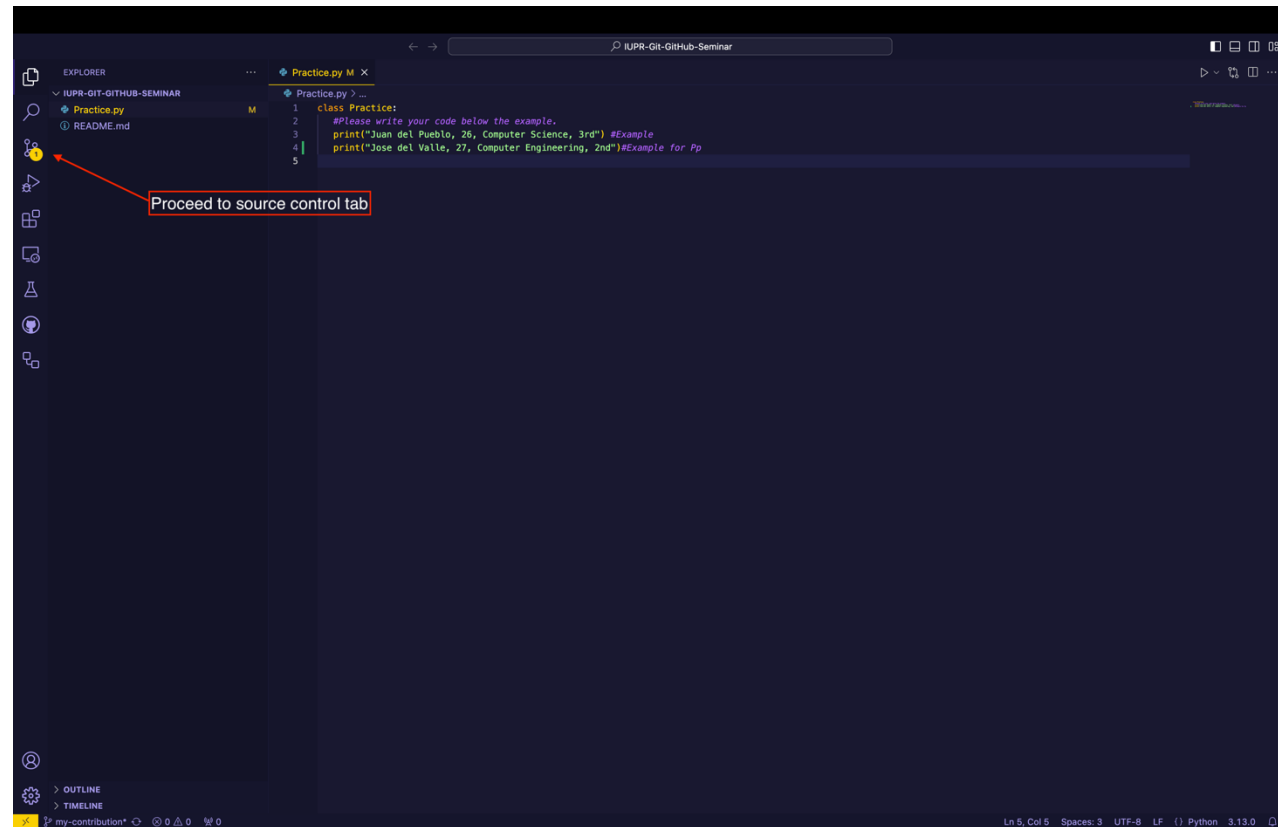
Contributing to a public Repo:

- Once you click the button on the bottom left corner to switch branches, you'll be prompted with this drop-down menu.



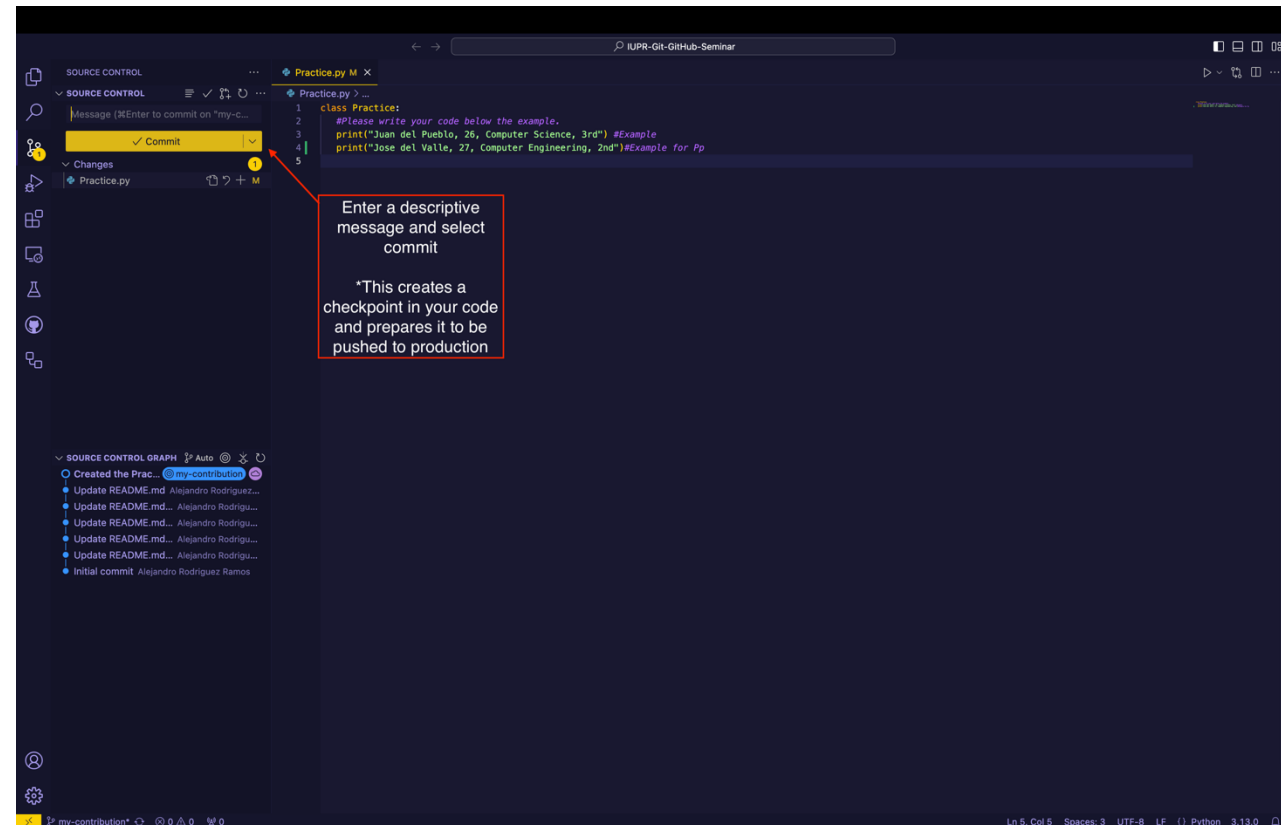
Contributing to a public Repo:

- Post adding your code and saving the file, you can proceed with switching to the source control tab.



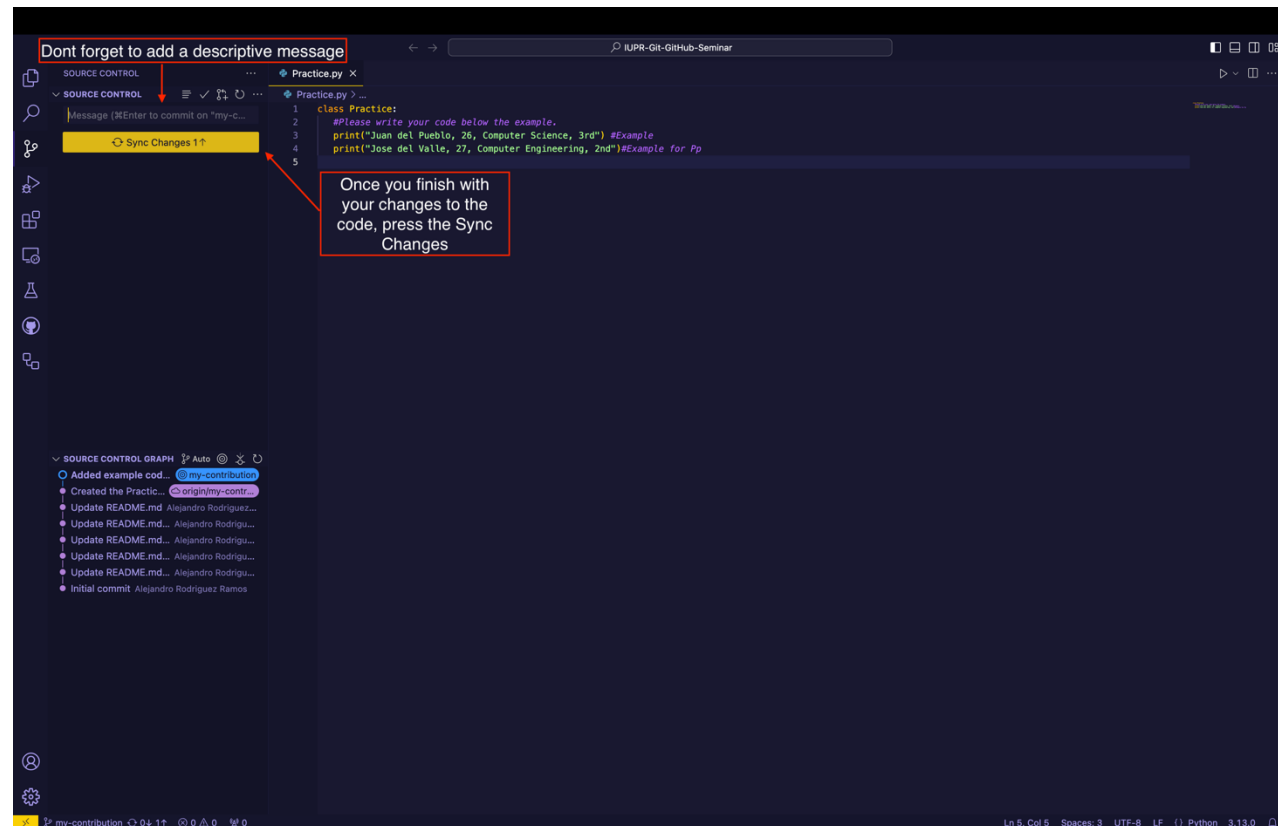
Contributing to a public Repo:

- After switching to the source control tab. You can Add a descriptive message and commit the code if you're pleased with your contribution.



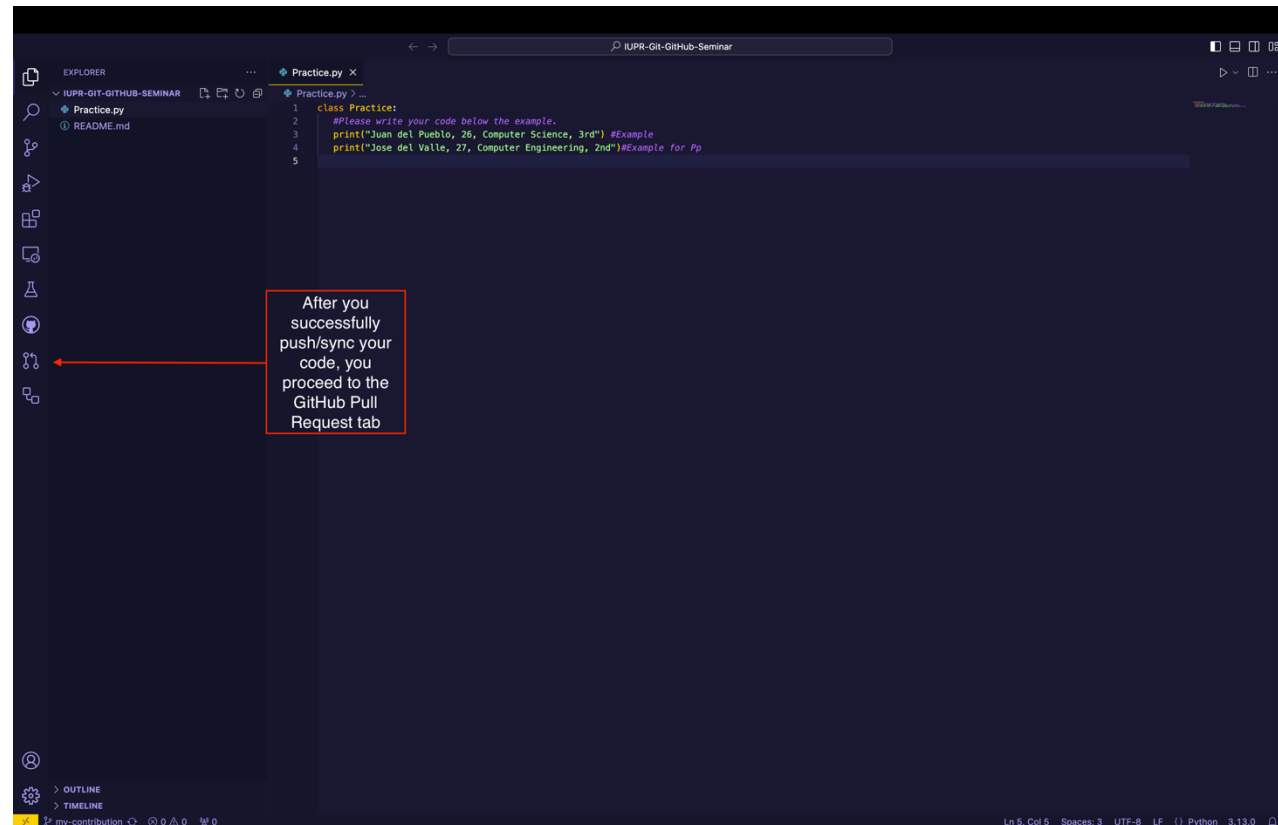
Contributing to a public Repo:

- If this is the only file you're committing. You can then add another descriptive message and push or sync the changes to the online repo.



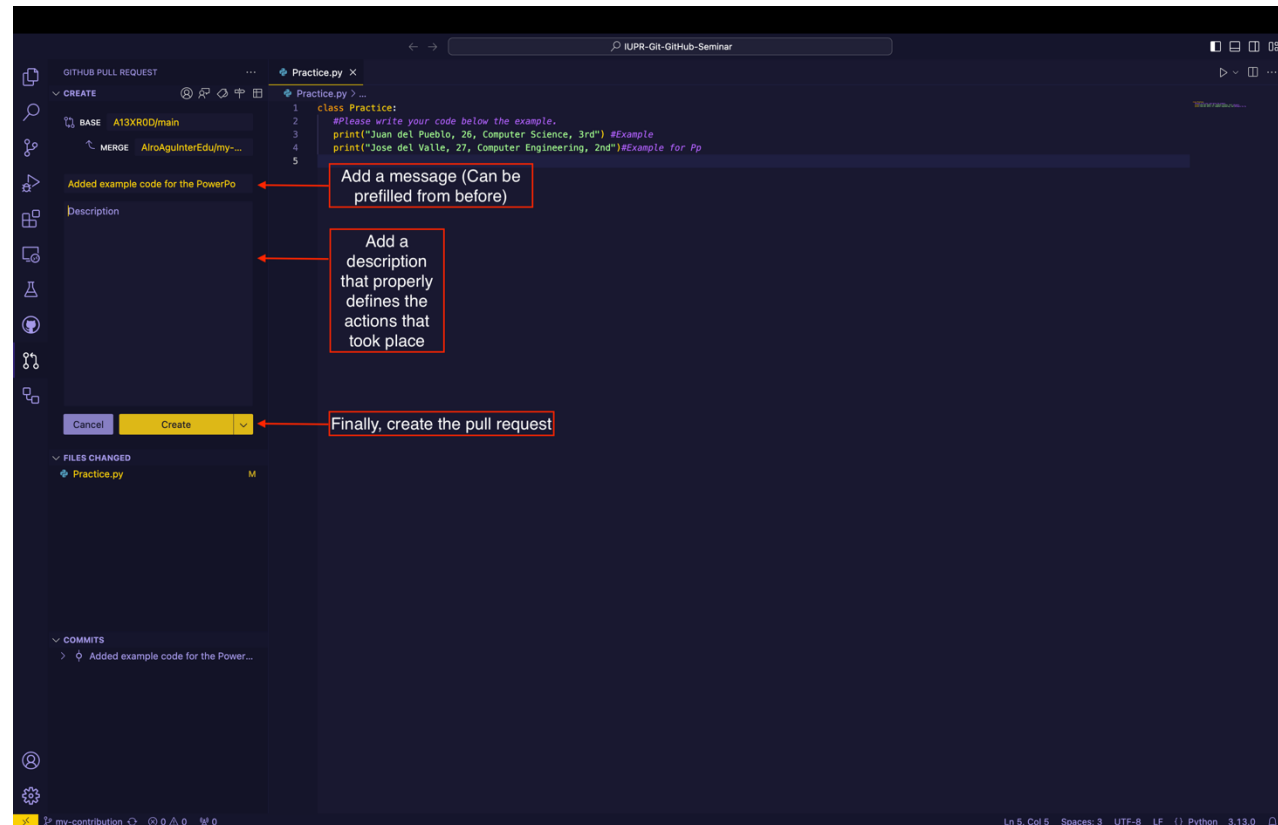
Contributing to a public Repo:

- After you've successfully pushed your code into your online repo.
- You now proceed to create a pull request to contribute to the primary repo.



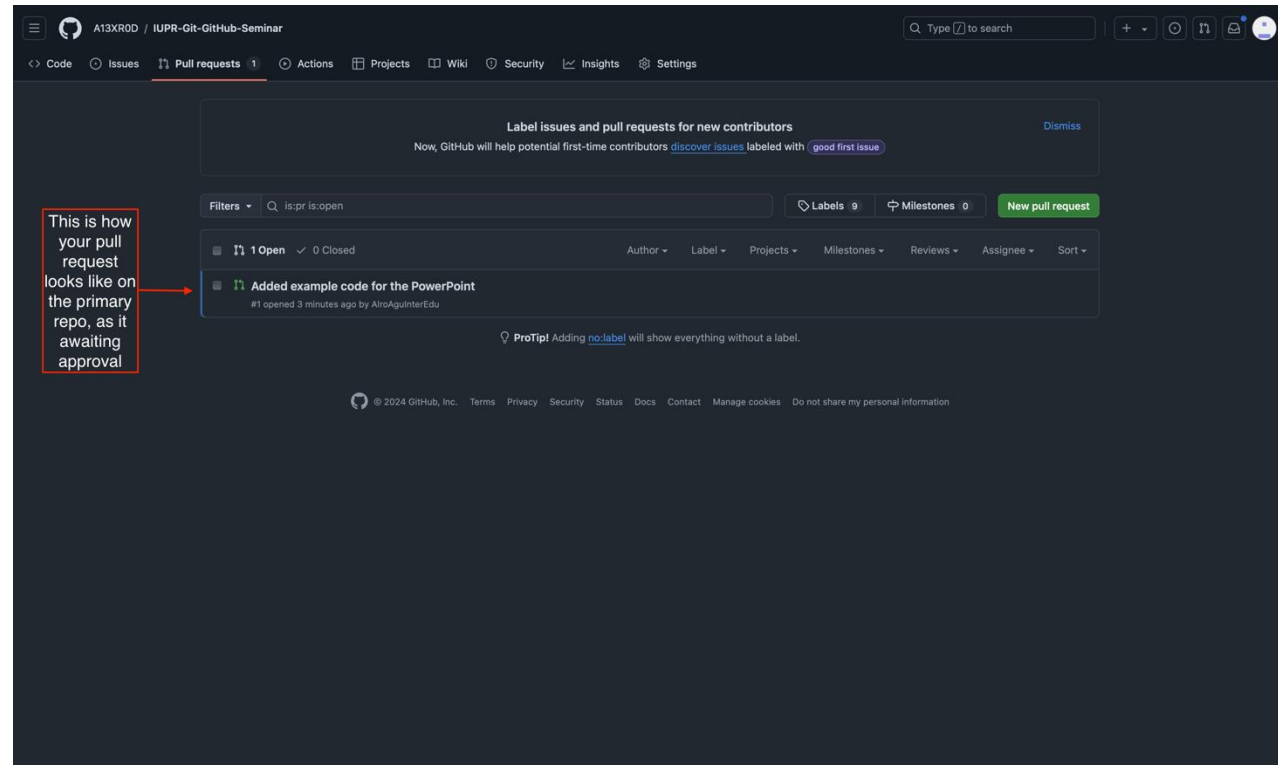
Contributing to a public Repo:

- Once you've switched to the Pull Request tab.
- Create a pull request.
- Essentially asking for permission to merge your code to the main repo.



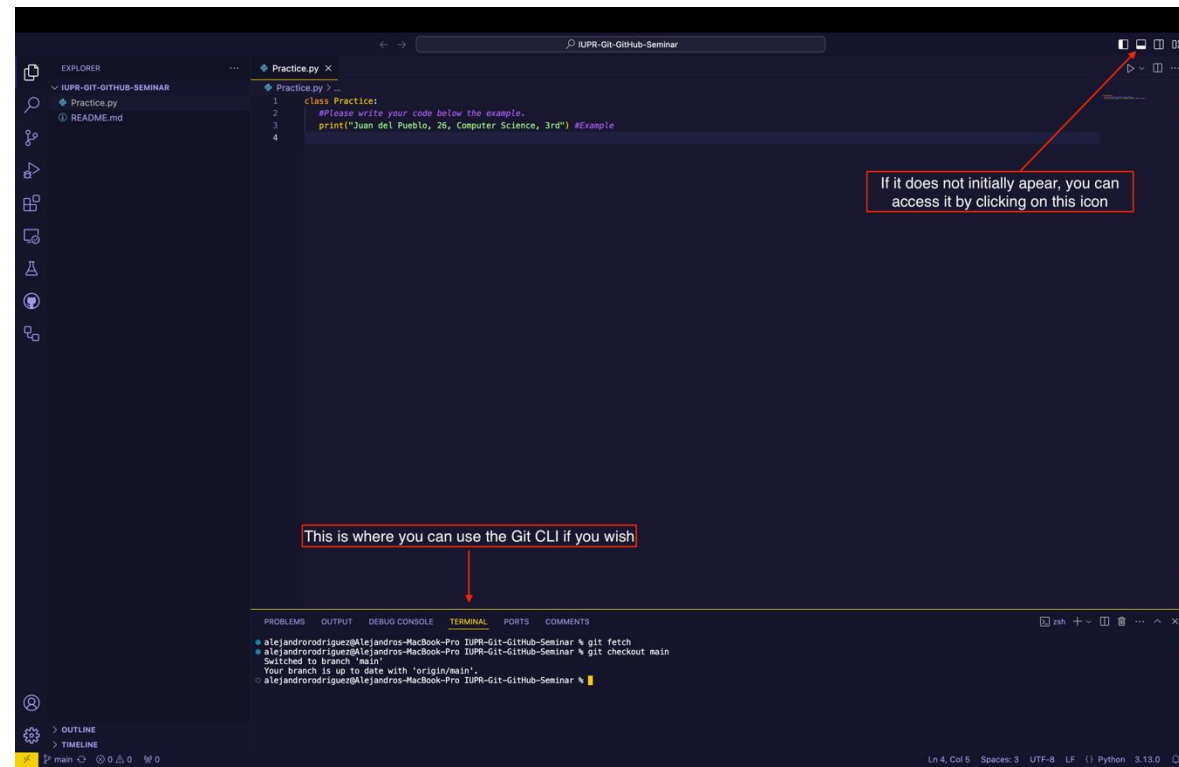
Contributing to a public Repo:

- *You've now successfully sent a request pull request.*
- All that's left do now is wait until the owner approves it and includes your feature in the main trunk of the code.



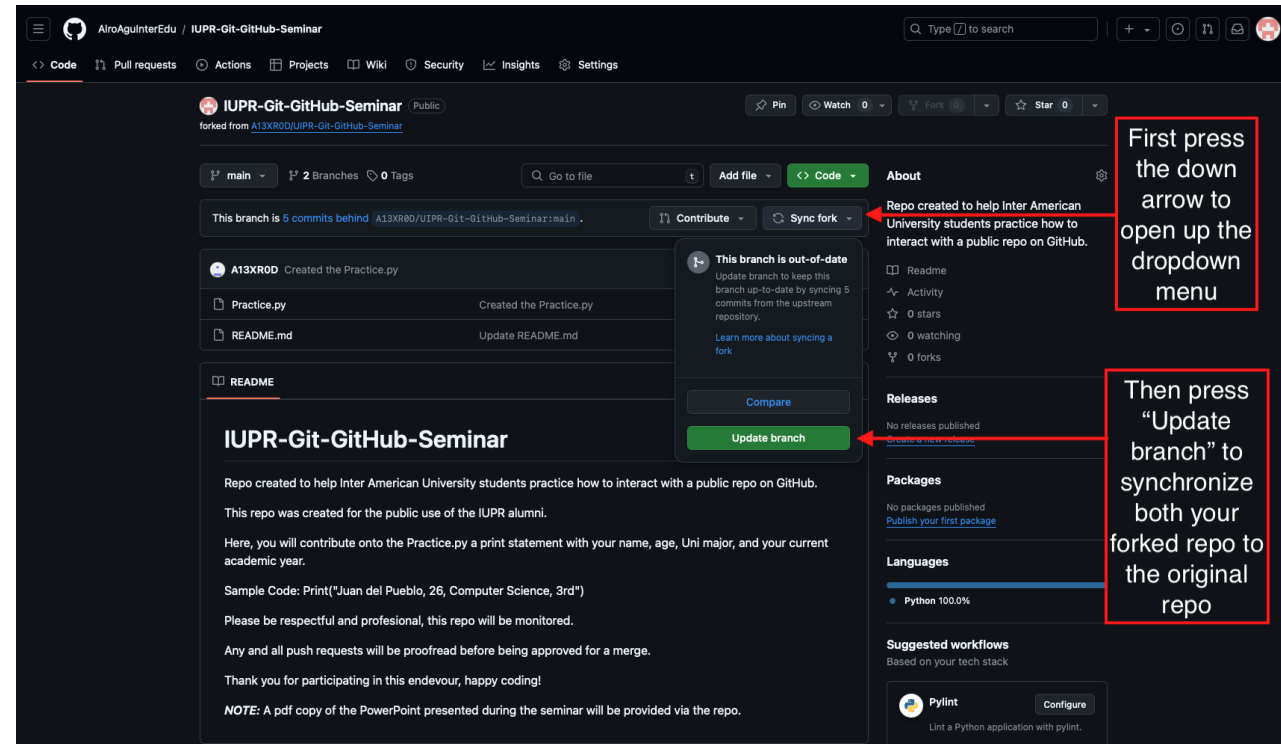
Using CLI in the terminal window:

- Using the built-in terminal provided by VS Code. You can use the git CLI instead of using the GUI.
- In case you don't see the terminal window; you can access it via the tab on the top right-hand side.



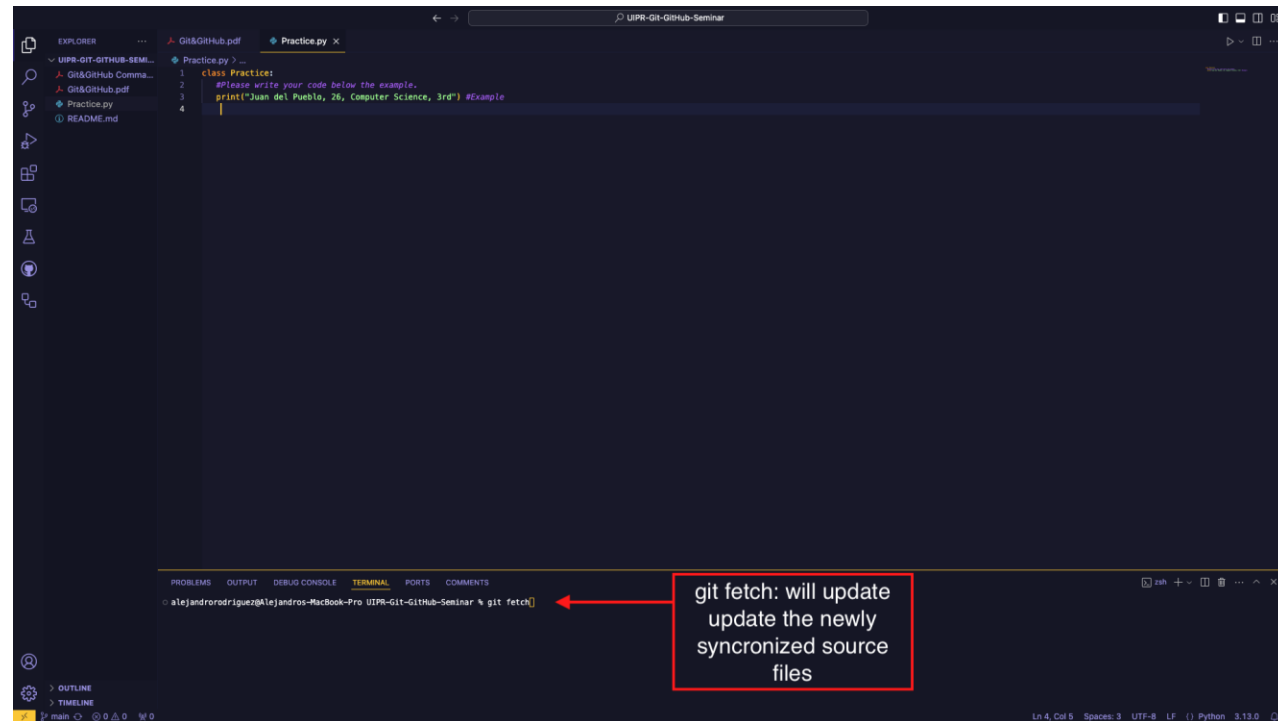
Updating forked repo:

- To update the forked repo, head on over to your repositories and click on the dropdown arrow and later click the “*Update branch*”.



Updating forked repo:

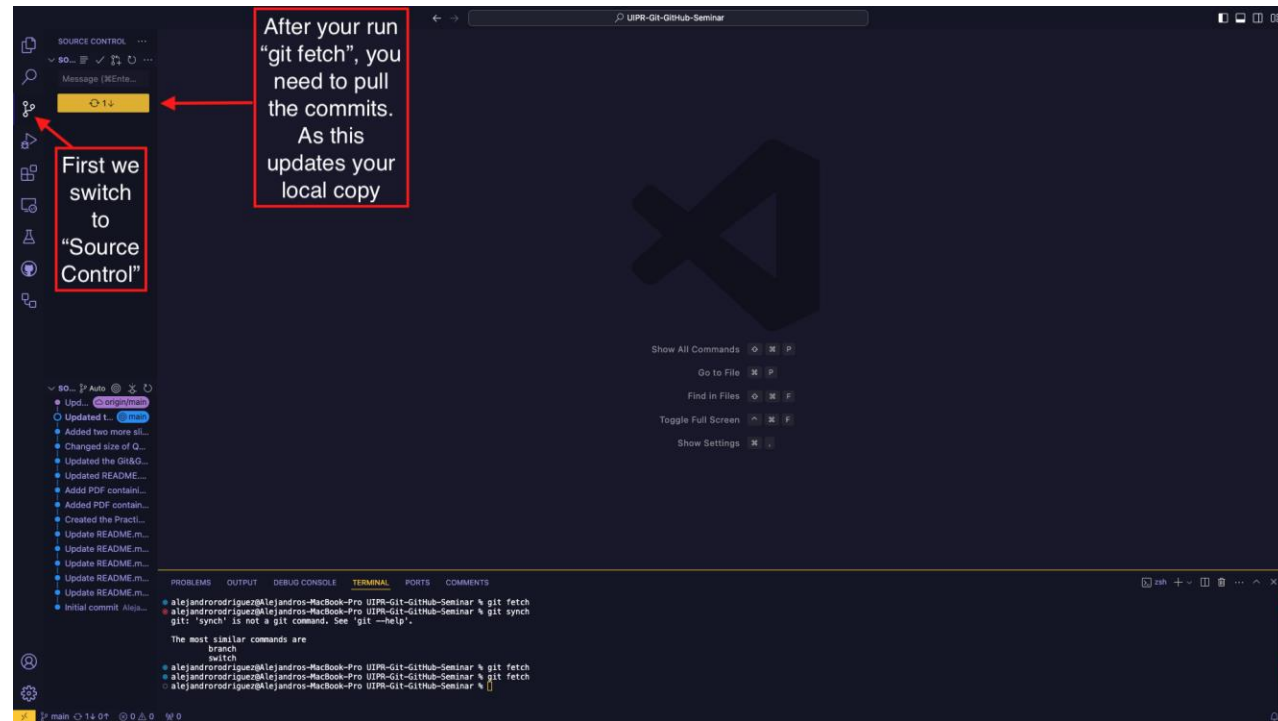
- *After updating your fork, you proceed to the clone's terminal and enter “**git fetch**”. This updates your local files to the latest version*



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project named 'UIPR-Git-GitHub-Seminar' with files like 'Practice.py' and 'README.md'. The main editor area shows the content of 'Practice.py', which includes a Python class definition and a print statement. At the bottom, the TERMINAL panel is active, displaying the command `git fetch` in a shell. A red arrow points from a text box to this command. The text box contains the text: `git fetch: will update update the newly synchronized source files`. The status bar at the bottom indicates the file is on the 'main' branch, using 'UTF-8' encoding, with 'LF' line endings, and is a Python 3.13.0 file.

Updating forked repo:

- Finally, after running “git fetch”, we switch to the source control tab and then we pull the commits from the online repo by pressing the yellow button.



Useful Links:

- GitHub's Quick-start for repositories:
 - <https://docs.github.com/en/repositories/creating-and-managing-repositories/quickstart-for-repositories?tool=webui>
- 10 Git Commands Every Developer Should Know:
 - <https://www.freecodecamp.org/news/10-important-git-commands-that-every-developer-should-know/>
- GitHub Student Developer Pack (Free stuff for students):
 - <https://education.github.com/pack>
- VS Code Homepage:
 - <https://code.visualstudio.com>

References:

- Cem Eygi, (2020, Jan 19). 10 Git Commands Every Developer Should Know. <https://www.freecodecamp.org/news/10-important-git-commands-that-every-developer-should-know/>
- GitHub, Inc, (2024). Let's build from here. <https://github.com/about>
- GitHub, Inc, (2024). About GitHub and Git. <https://docs.github.com/en/get-started/start-your-journey/about-github-and-git>
- GitHub, Inc, (2024). About Git. <https://docs.github.com/en/get-started/using-git/about-git>
- GitHub, Inc, (2024). GitHub Flow. <https://docs.github.com/en/get-started/using-github/github-flow>