Working with Git.

Loïc Roldán Waals
Manager Data & AI
+31 6 43 92 62 84
loic.roldanwaals@sia-partners.com

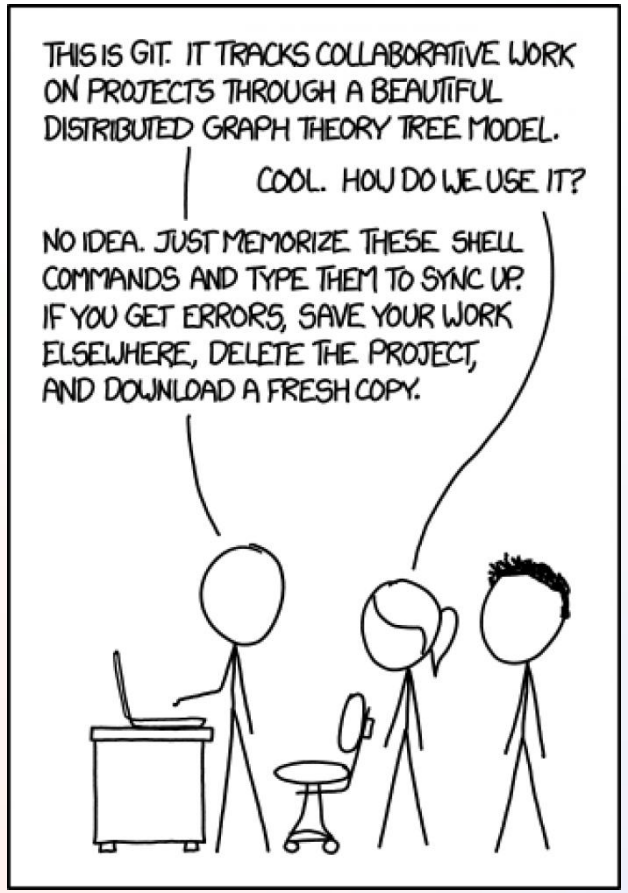
Agenda

1. Context
2. Setup
3. Commands
4. PRs
5. Merge Conflicts
6. Practice

Git was created for version control while working on Linux

- We use Git to version control our code in a scalable way.
- Of the three most common version control systems, Git is (by far) the most popular

Popularity of version control systems in 2023 (link)



Git was created for version control while working on Linux

[master] 6c6faa5 My first commit - John Doe

[develop] 3e89ec8 Develop a feature - part 1 - John Doe

[develop] e188fa9 Develop a feature - part 2 - John Doe

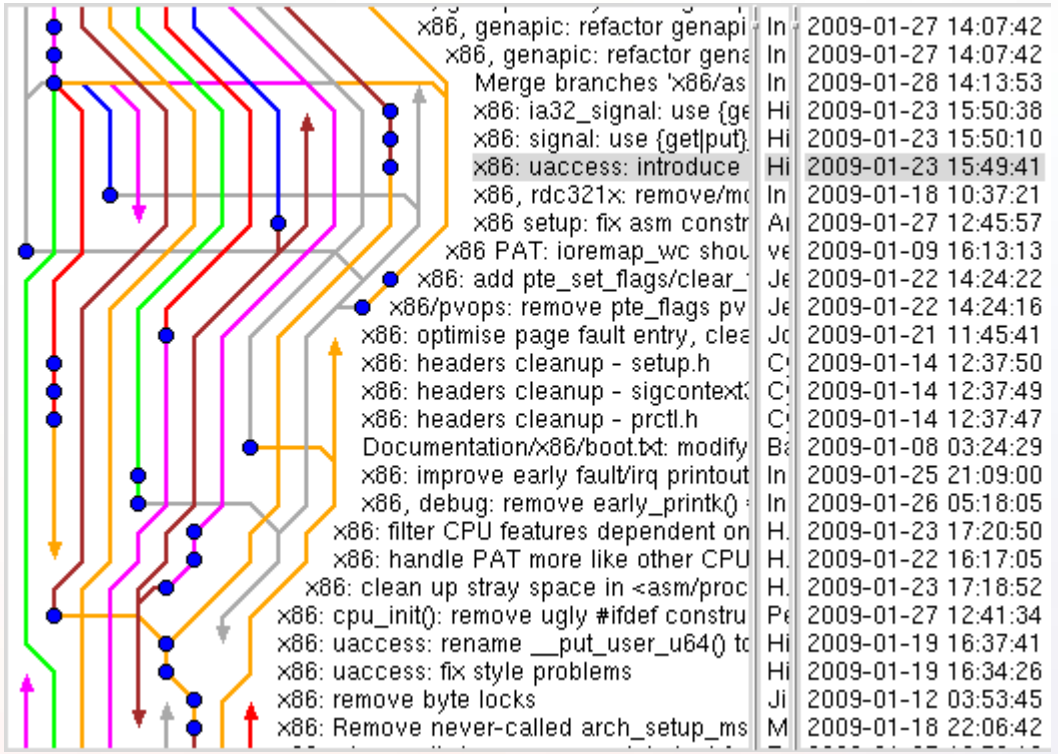
[master] 665003d Fast bugfix - John Fixer

[myfeature] eaf618c New cool feature - John Feature

[master] 8f1e0e7 Merge branch `develop` into `master` - John Doe

[master] 6a3dacc Merge branch `myfeature` into `master` - John Doe

0.1 [master] abcdef0 Release of version 0.1 - John Releaser



To use Git, a hosting tool and an access tool is required

Hosting tools

- Central place to host the repository
- Can be accessed by all collaborators
- Typical examples include:

GitHub

Azure DevOps

GitLab

Bitbucket

Access tools

- Way to interact with the host repository and perform local version control
- Often included with an IDE, but can be stand-alone
- Typical examples include:

Command Line

Github Desktop

VS Code

PyCharm

Changing some configurations will make it easier to work with git.

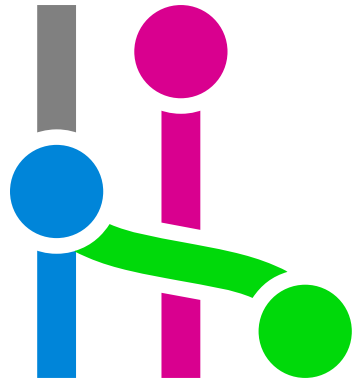
VS Code

- For VSCode settings, use Ctrl + Shift + P and write "open default settings (JSON)".
- Choose the JSON option and paste the provided settings.
- Install Git Graph and GitLens for nicer GUI

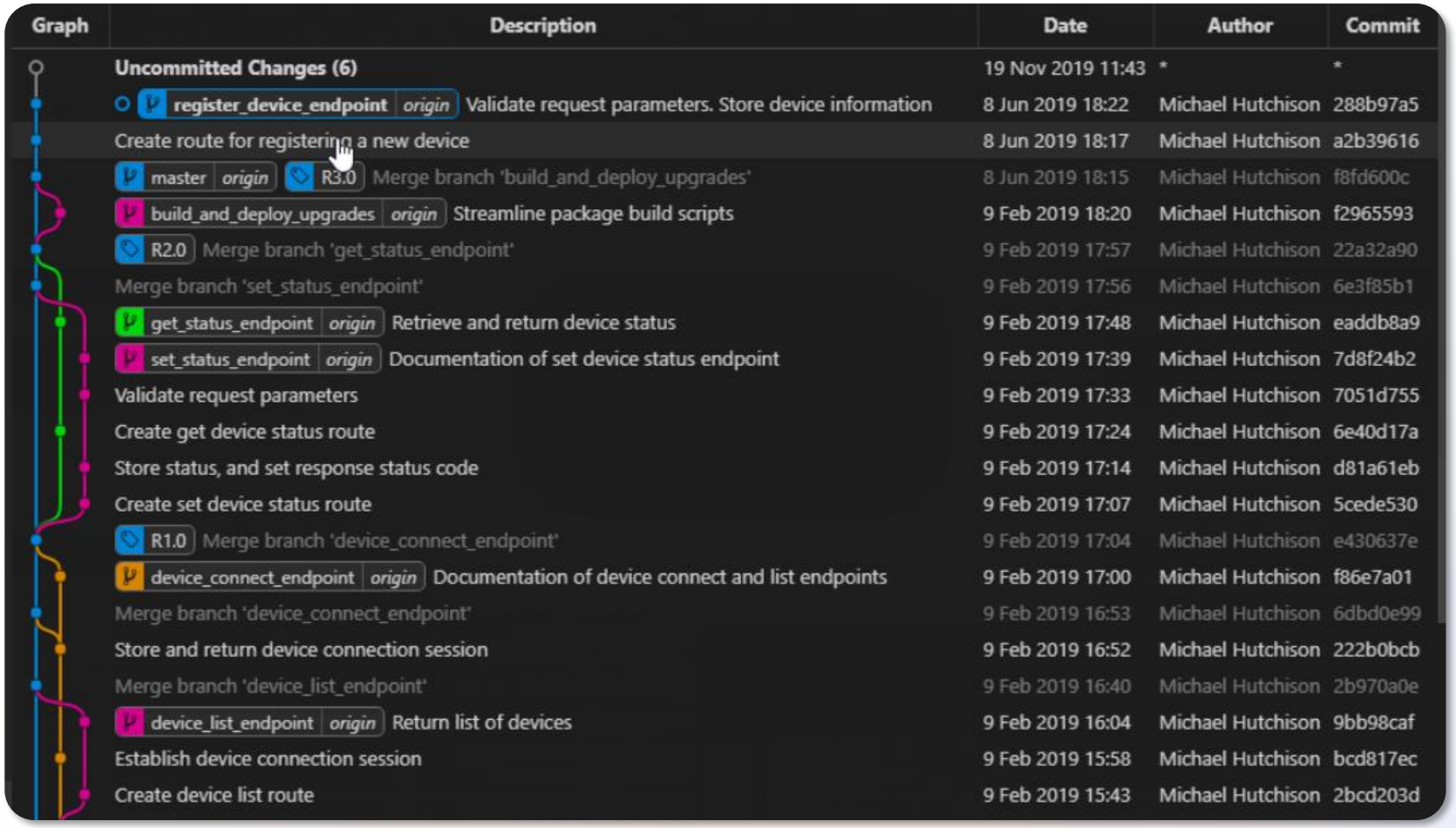
Git

- Open a terminal
- Enter the commands:
  - git config --global core.editor "code –wait"
  - git config --global –e
- Then paste the provided example settings.
- Close the file to save it.

VSCode has nice extensions that allow for a GUI interface.



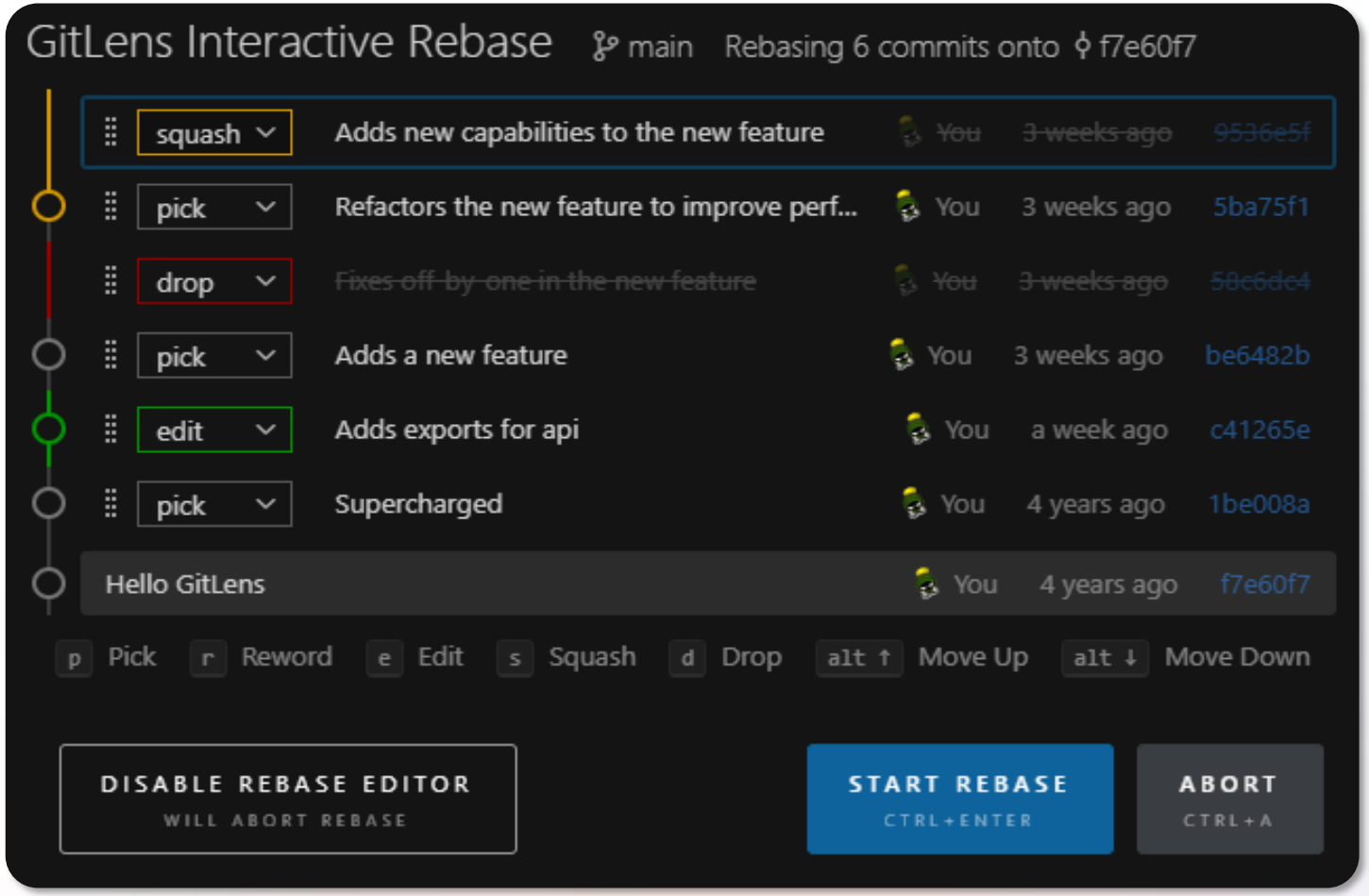
Git graph



Graph	Description	Date	Author	Commit
	Uncommitted Changes (6)	19 Nov 2019 11:43	*	*
	register_device_endpoint origin Validate request parameters. Store device information	8 Jun 2019 18:22	Michael Hutchison	288b97a5
	Create route for registering a new device	8 Jun 2019 18:17	Michael Hutchison	a2b39616
	master origin R3.0 Merge branch 'build_and_deploy_upgrades'	8 Jun 2019 18:15	Michael Hutchison	f8fd600c
	build_and_deploy_upgrades origin Streamline package build scripts	9 Feb 2019 18:20	Michael Hutchison	f2965593
	R2.0 Merge branch 'get_status_endpoint'	9 Feb 2019 17:57	Michael Hutchison	22a32a90
	Merge branch 'set_status_endpoint'	9 Feb 2019 17:56	Michael Hutchison	6e3f85b1
	get_status_endpoint origin Retrieve and return device status	9 Feb 2019 17:48	Michael Hutchison	eaddb8a9
	set_status_endpoint origin Documentation of set device status endpoint	9 Feb 2019 17:39	Michael Hutchison	7d8f24b2
	Validate request parameters	9 Feb 2019 17:33	Michael Hutchison	7051d755
	Create get device status route	9 Feb 2019 17:24	Michael Hutchison	6e40d17a
	Store status, and set response status code	9 Feb 2019 17:14	Michael Hutchison	d81a61eb
	Create set device status route	9 Feb 2019 17:07	Michael Hutchison	5cede530
	R1.0 Merge branch 'device_connect_endpoint'	9 Feb 2019 17:04	Michael Hutchison	e430637e
	device_connect_endpoint origin Documentation of device connect and list endpoints	9 Feb 2019 17:00	Michael Hutchison	f86e7a01
	Merge branch 'device_connect_endpoint'	9 Feb 2019 16:53	Michael Hutchison	6dbd0e99
	Store and return device connection session	9 Feb 2019 16:52	Michael Hutchison	222b0bcb
	Merge branch 'device_list_endpoint'	9 Feb 2019 16:40	Michael Hutchison	2b970a0e
	device_list_endpoint origin Return list of devices	9 Feb 2019 16:04	Michael Hutchison	9bb98caf
	Establish device connection session	9 Feb 2019 15:58	Michael Hutchison	bcd817ec
	Create device list route	9 Feb 2019 15:43	Michael Hutchison	2bcd203d

VSCode has nice extensions that allow for a GUI interface.

GitLens



Most of your Git workflow can be achieved with 6 commands

pull or **fetch**

Get changes from the remote. Pull also applies your changes locally.

add and **commit**

Choose what parts you want to add to the “timeline” of the repo.

(force) **push**

Update the remote “timeline” with your local “timeline” for a specific branch.

branch and **checkout**

Create a new branch in the “timeline”. Checkout switches you to your chosen branch.

stash

Save your changes locally without adding them to the “timeline”.

rebase and **reset**

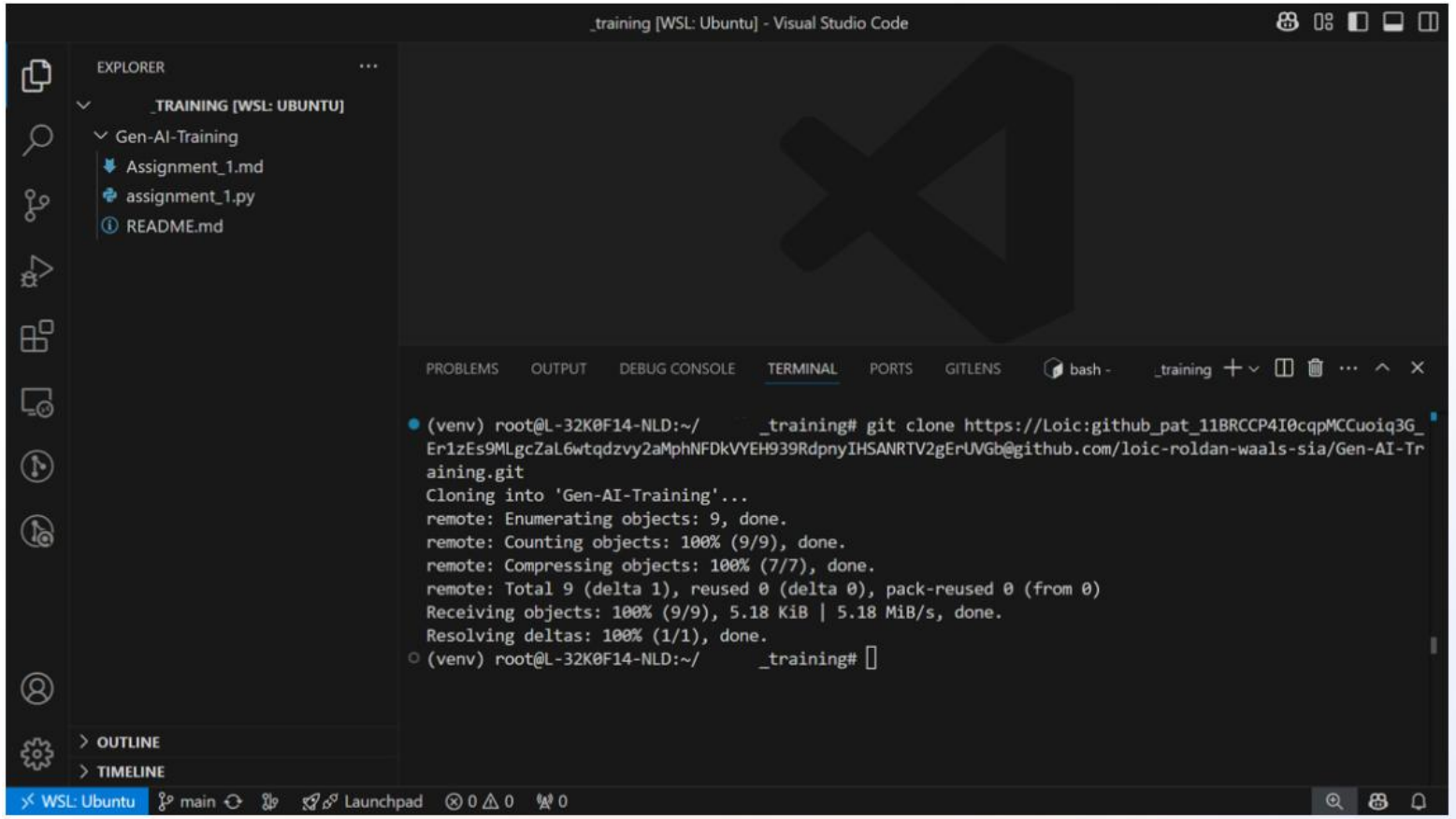
Allows you to edit the history and which branch your branch is based on.

Let’s try to use these commands! Try to complete them all.

1. **Clone** repo with PAT
2. Create new **branch** called *feature/loics-feature*, based on *fake-main*.
3. Make some simple code changes.
4. **Stash** them and switch to some other branch (*feature/someones-feature*).
5. Make a new **commit** and **push** changes.
6. Make some more simple changes.
7. Come back to the original branch (*feature/loics-feature*) and **un-stash** them.
8. Wait for some other changes on *fake-main*.
9. **Rebase** *feature/loics-feature* on the latest changes from fake-main.
10. Make multiple **commits** and “accidentally” add some with some unwanted changes.
11. **Reset** (--mixed) changes and create one new clean **commit**.
12. **Force push** changes to fix the remote.

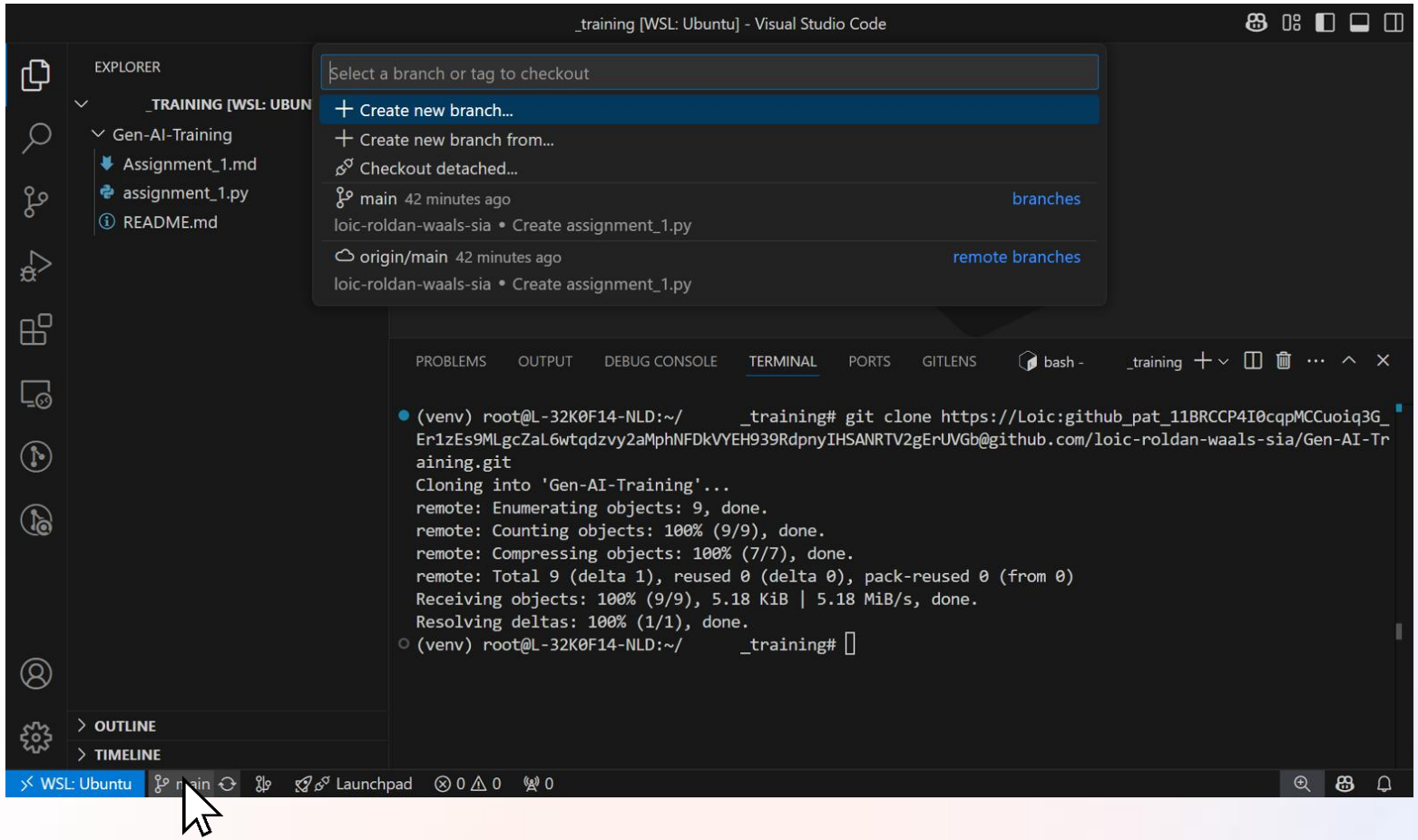
Let’s try to use these commands!

- **Clone** repo with PAT
- Other methods are also available, this is just the simplest to mange for us.



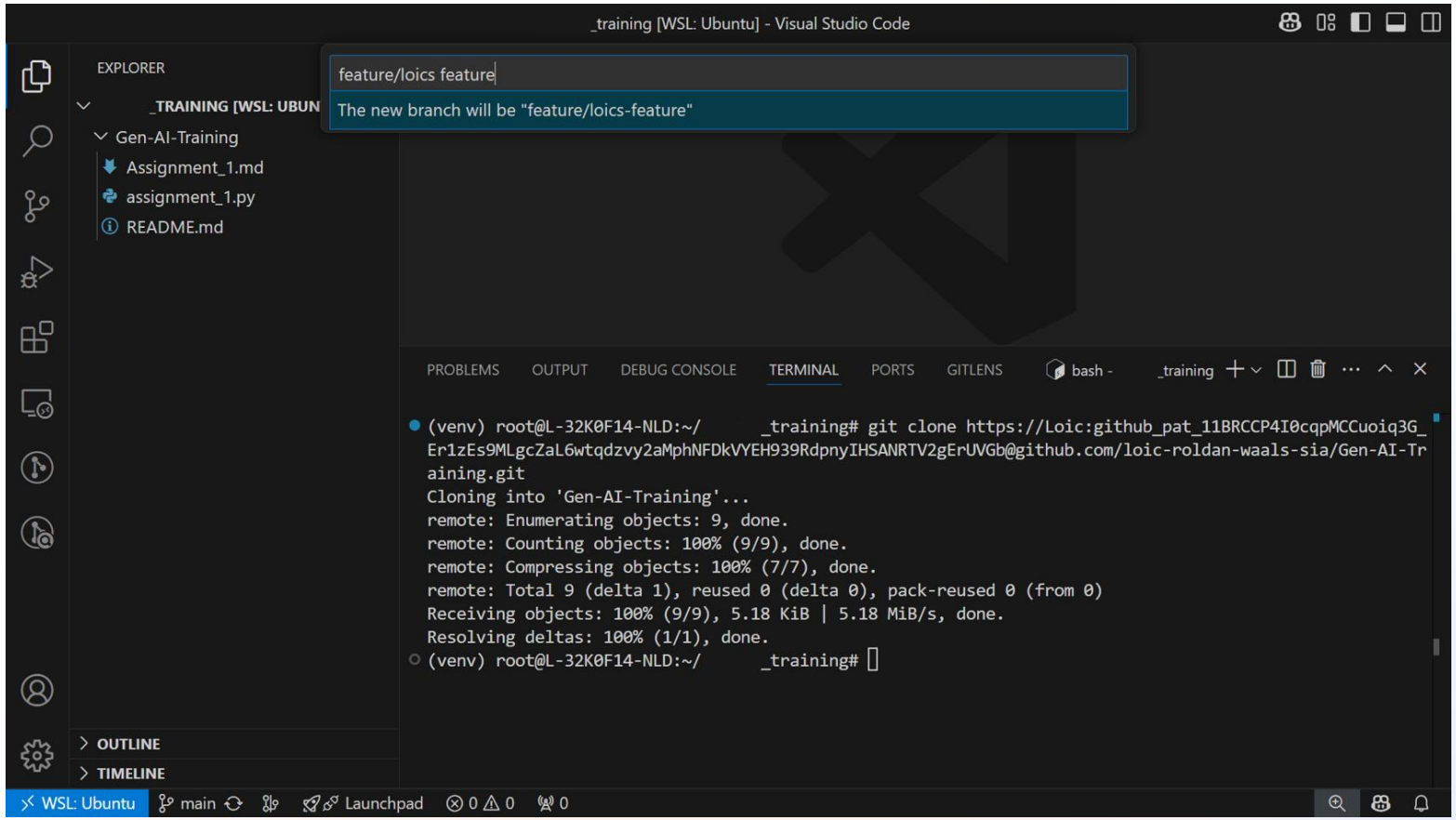
Let’s try to use these commands!

- Create new **branch** called *feature/loics-feature*, based on *fake-main*.



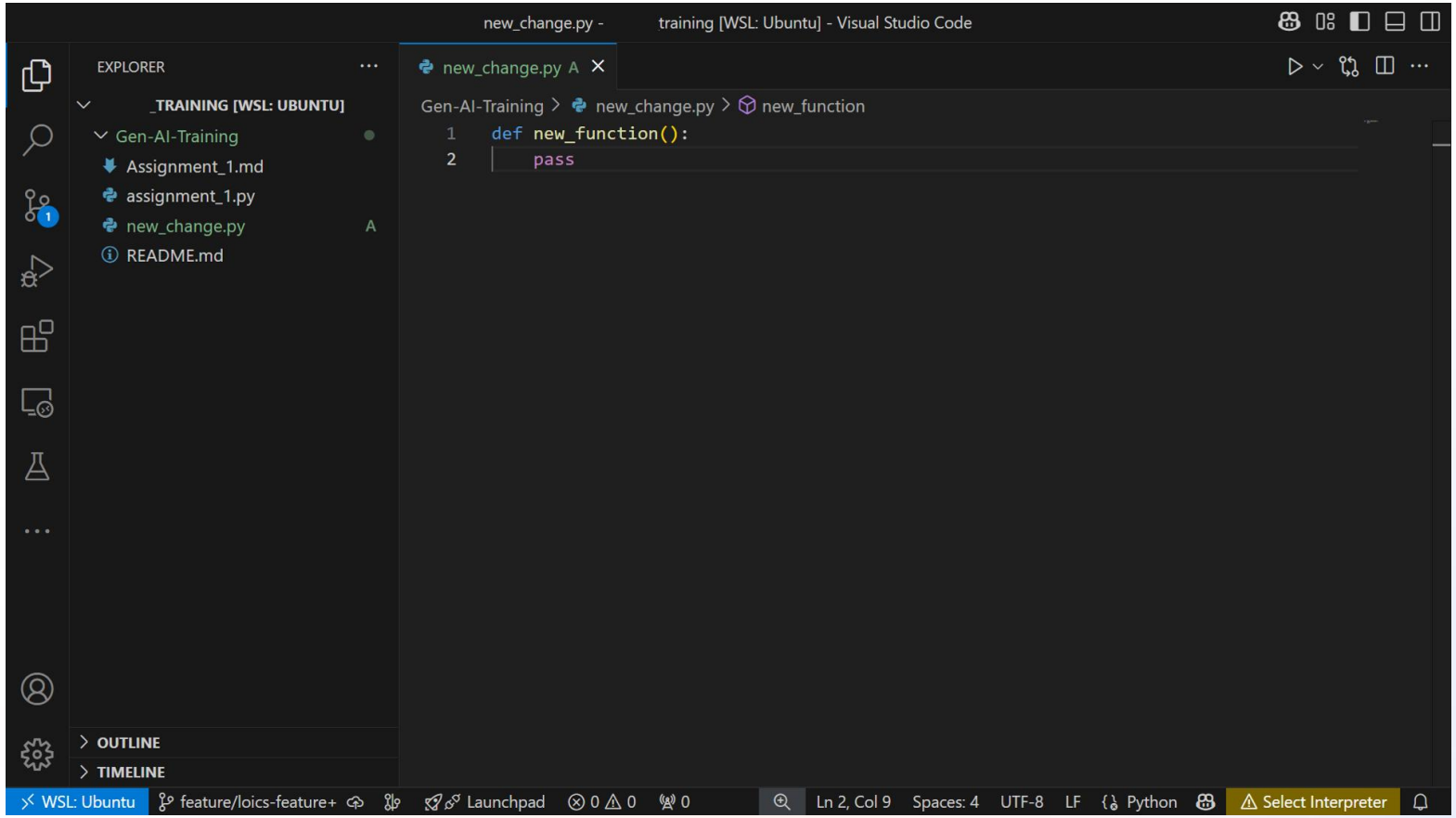
Let’s try to use these commands!

- Create new **branch** called *feature/loics-feature*, based on *fake-main*.



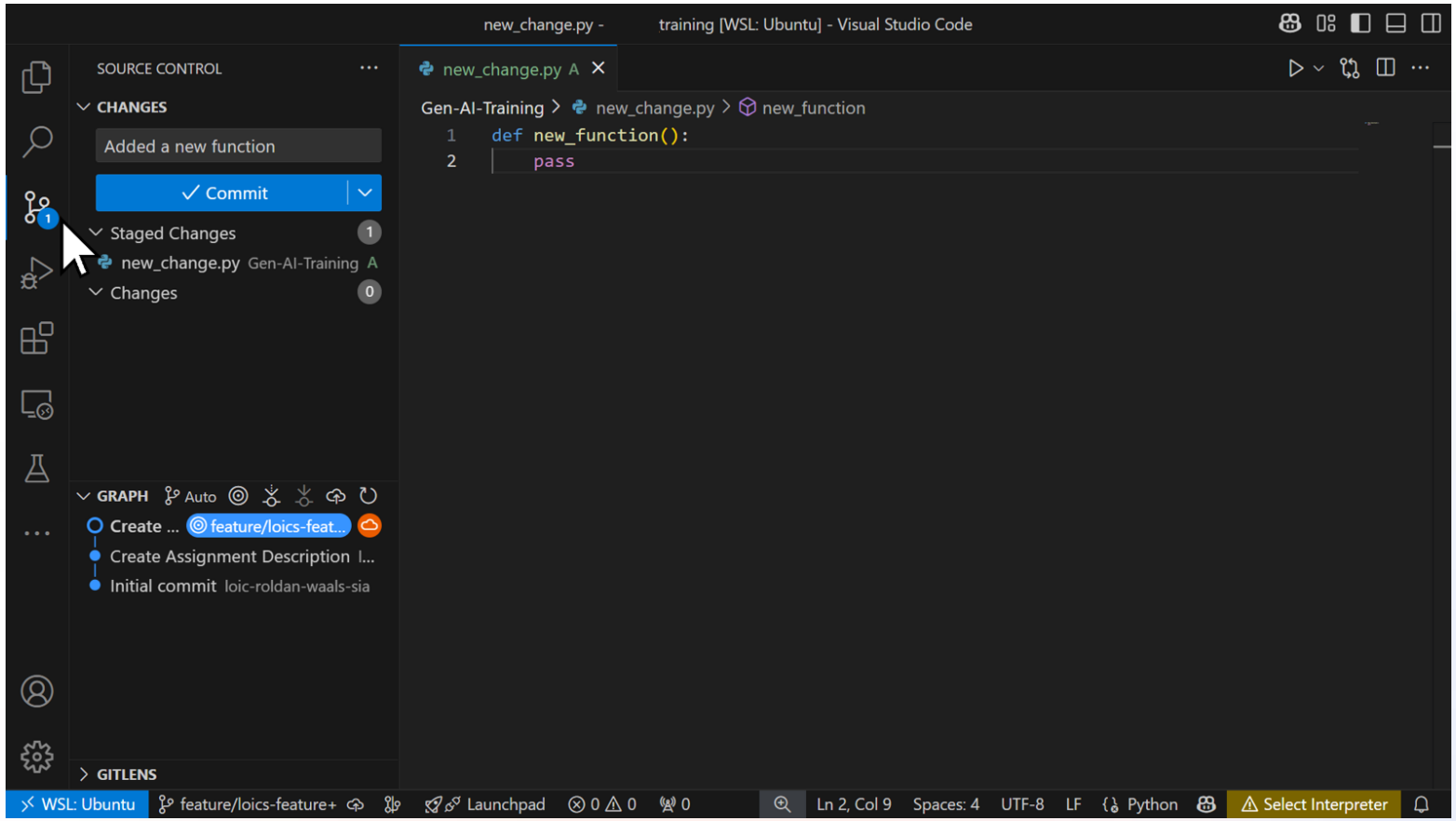
Let’s try to use these commands!

- Make some simple code changes.



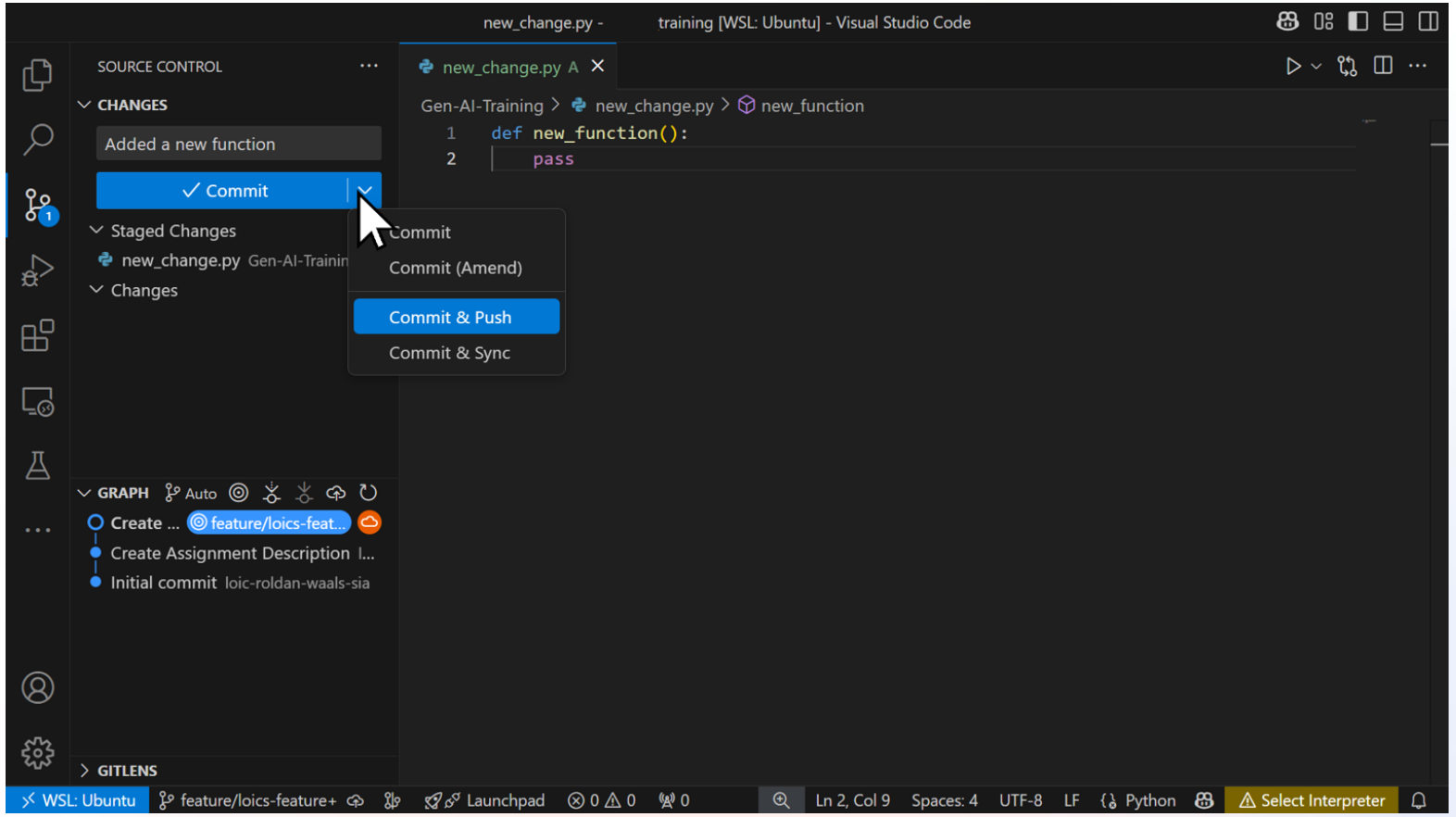
Let’s try to use these commands!

- Make a new **commit** and **push** changes.



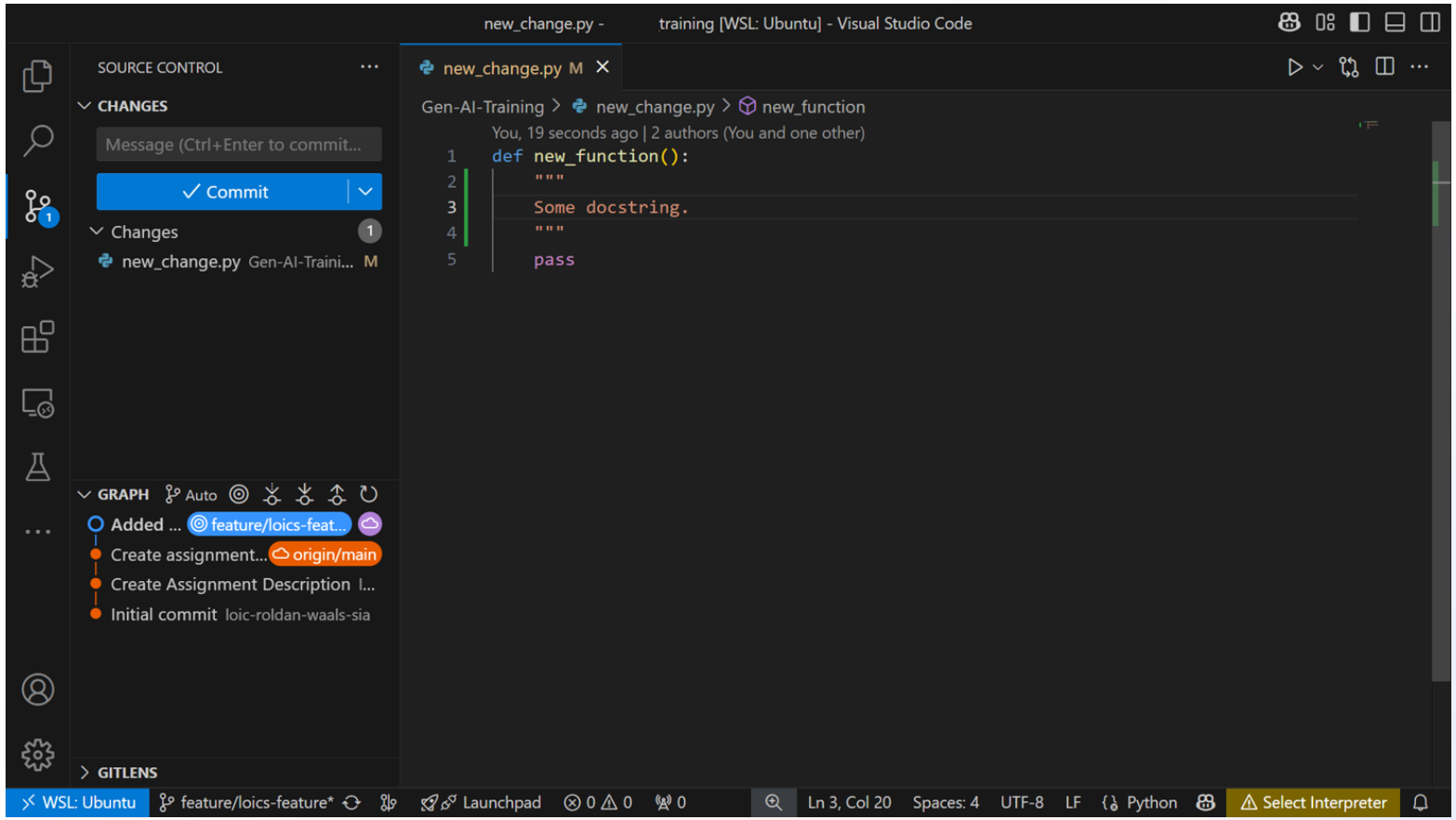
Let’s try to use these commands!

- Make a new **commit** and **push** changes.



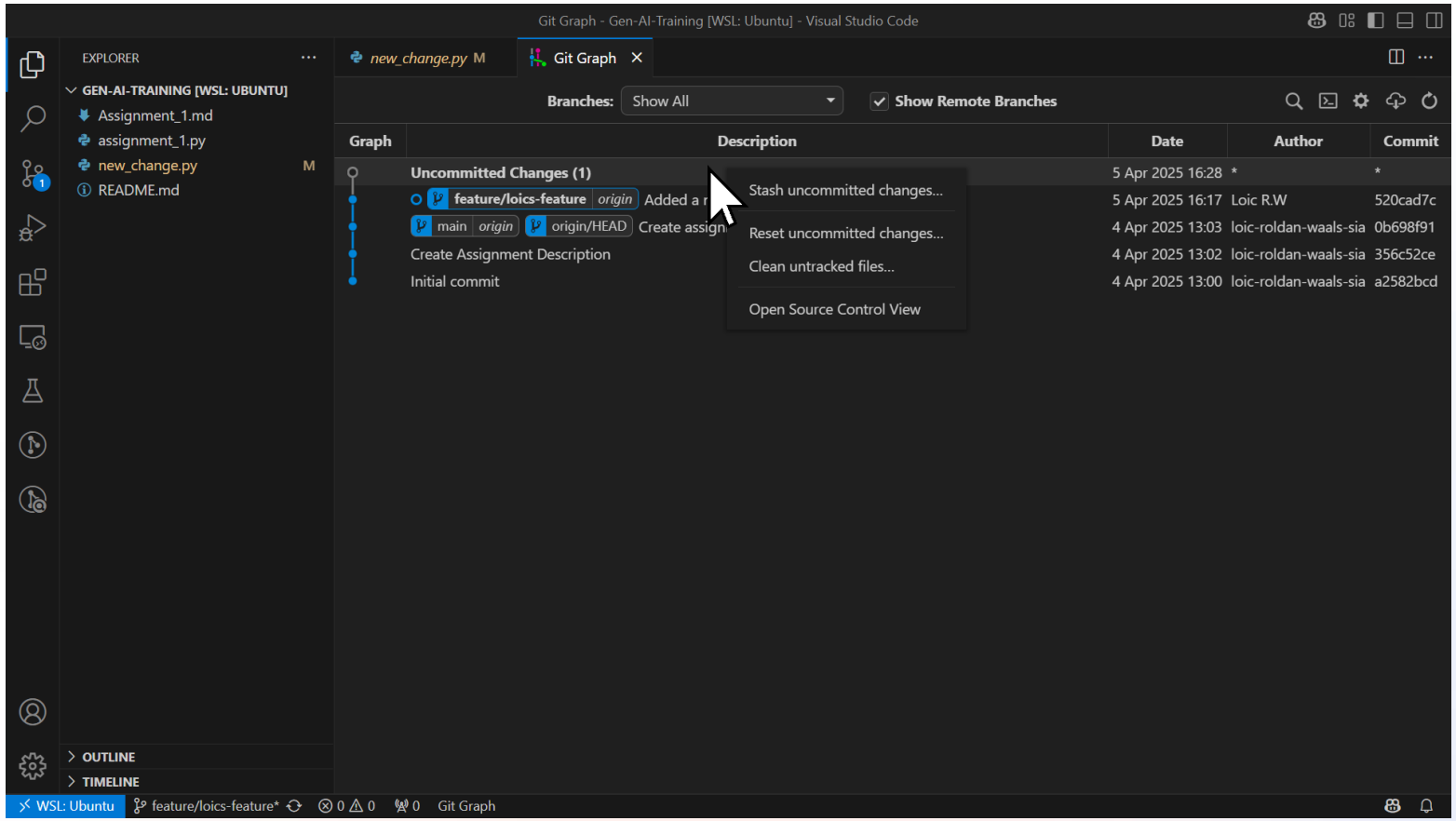
Let’s try to use these commands!

- Make some more simple changes.



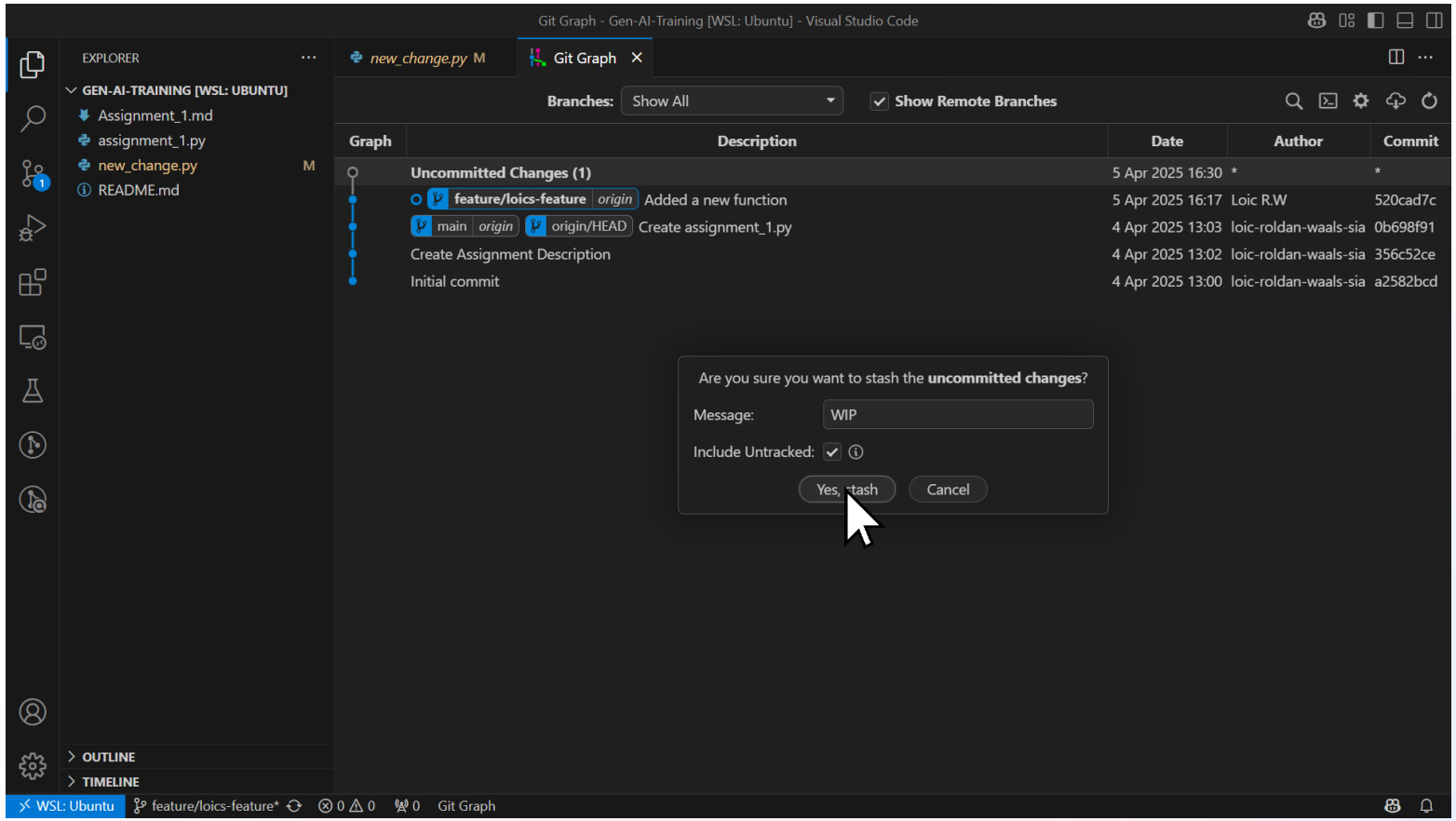
Let’s try to use these commands!

- **Stash** them and switch to some other branch (*feature/someones-feature*).
- This is common when you are reviewing someone else’s work on their branch, but you are not yet finished with your own work.



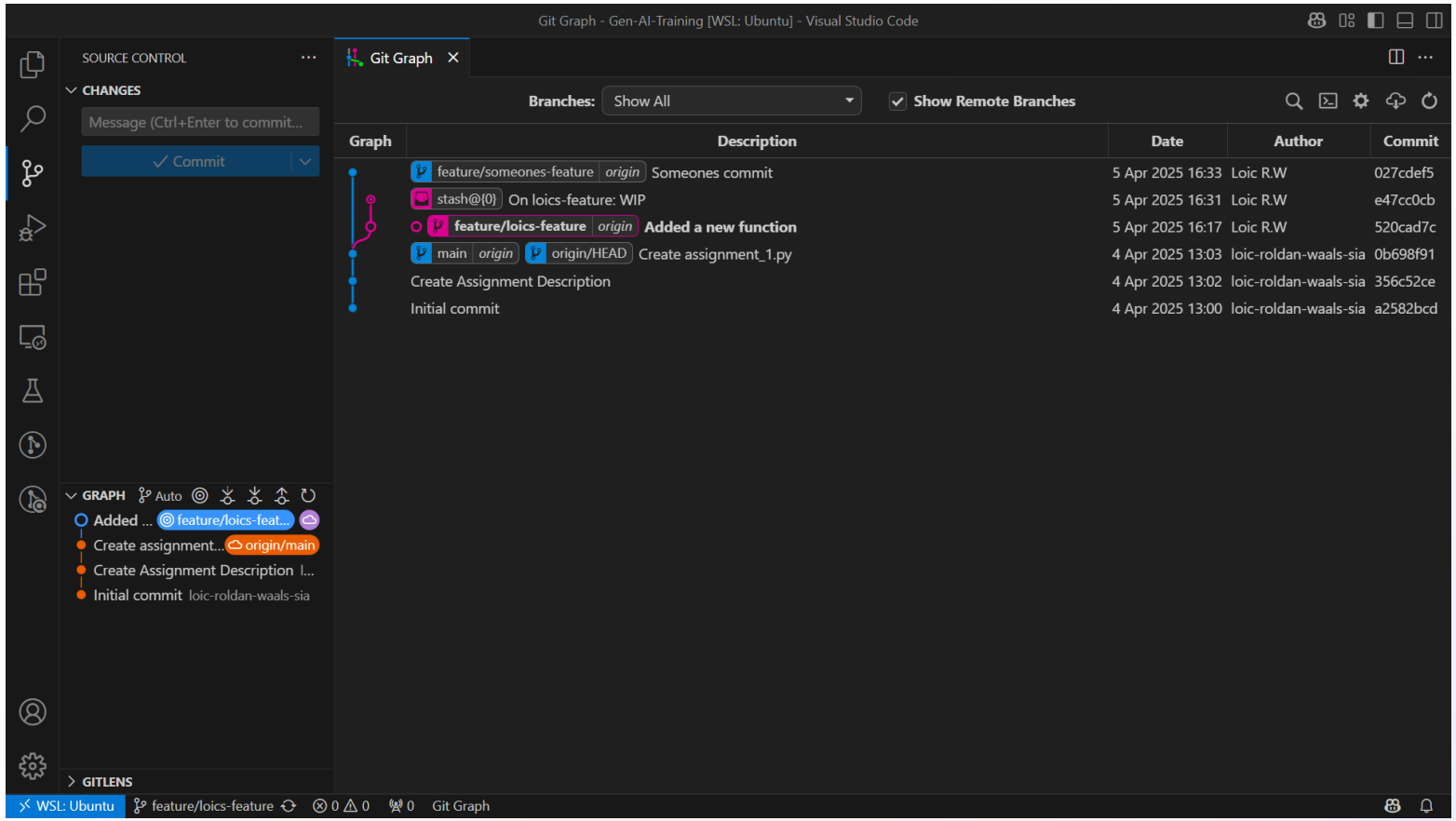
Let’s try to use these commands!

- **Stash** them and switch to some other branch (*feature/someones-feature*).
- This is common when you are reviewing someone else’s work on their branch, but you are not yet finished with your own work.



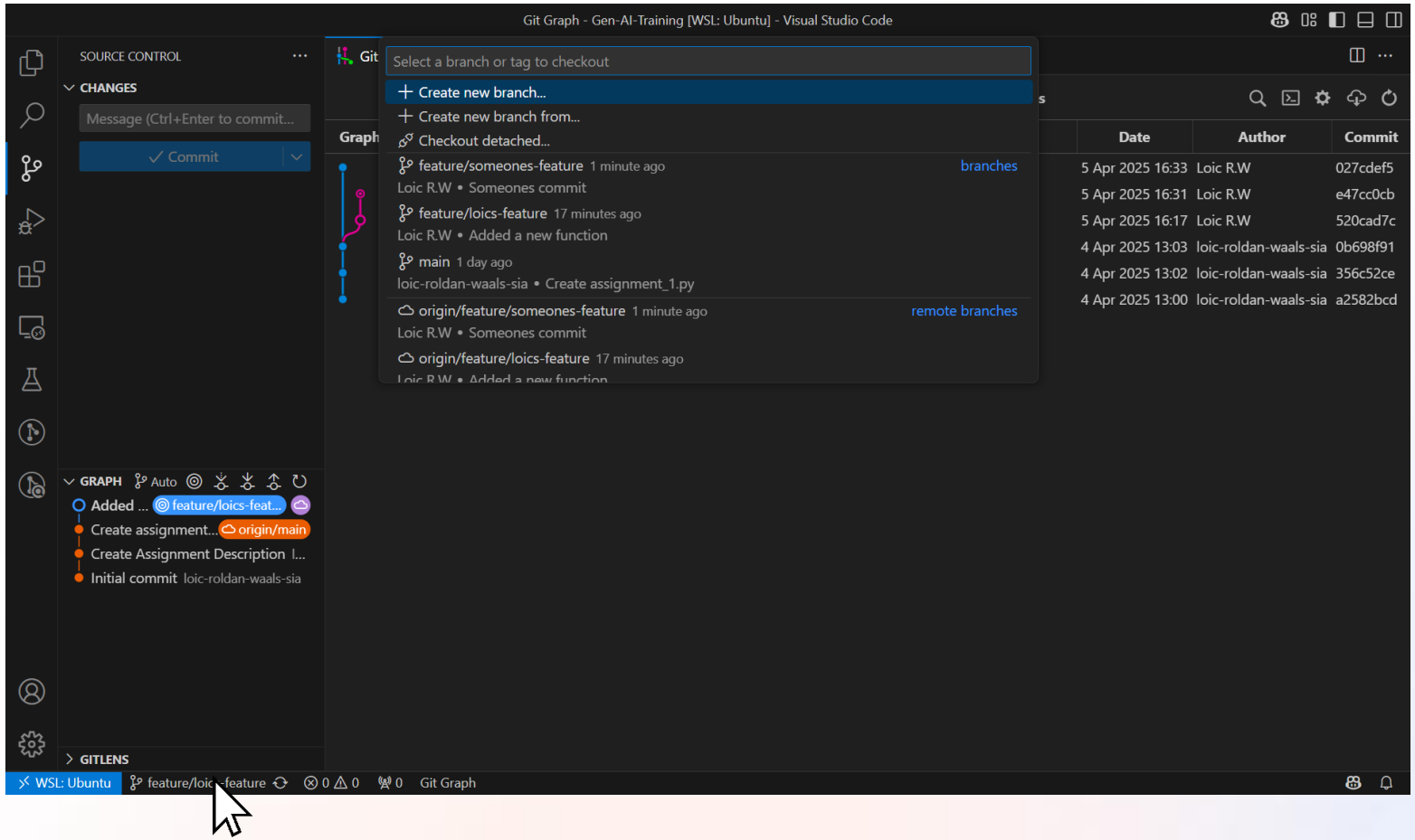
Let’s try to use these commands!

- **Stash** them and switch to some other branch (*feature/someones-feature*).
- This is common when you are reviewing someone else’s work on their branch, but you are not yet finished with your own work.



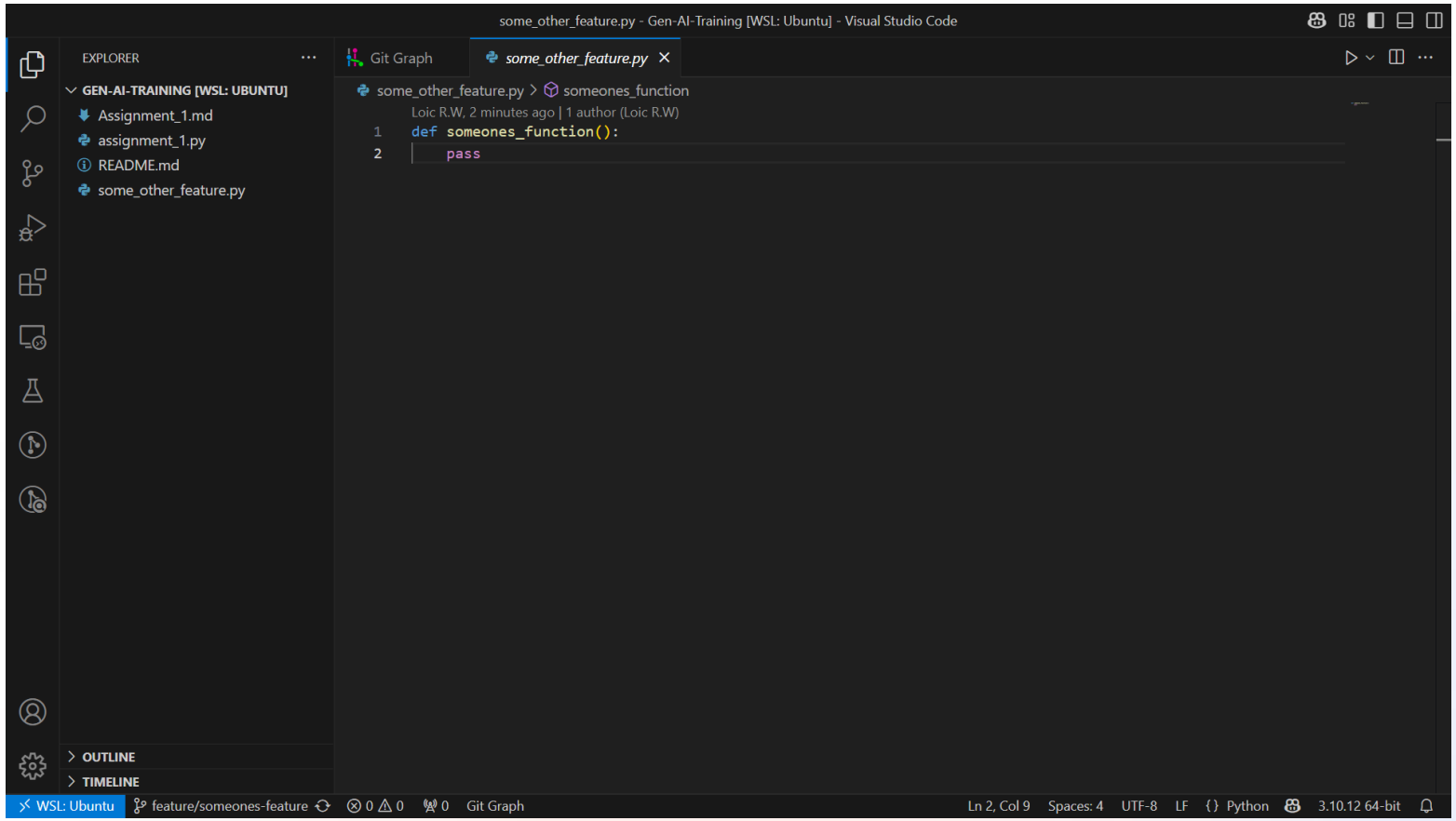
Let’s try to use these commands!

- **Stash** them and switch to some other branch (*feature/someones-feature*).
- This is common when you are reviewing someone else’s work on their branch, but you are not yet finished with your own work.



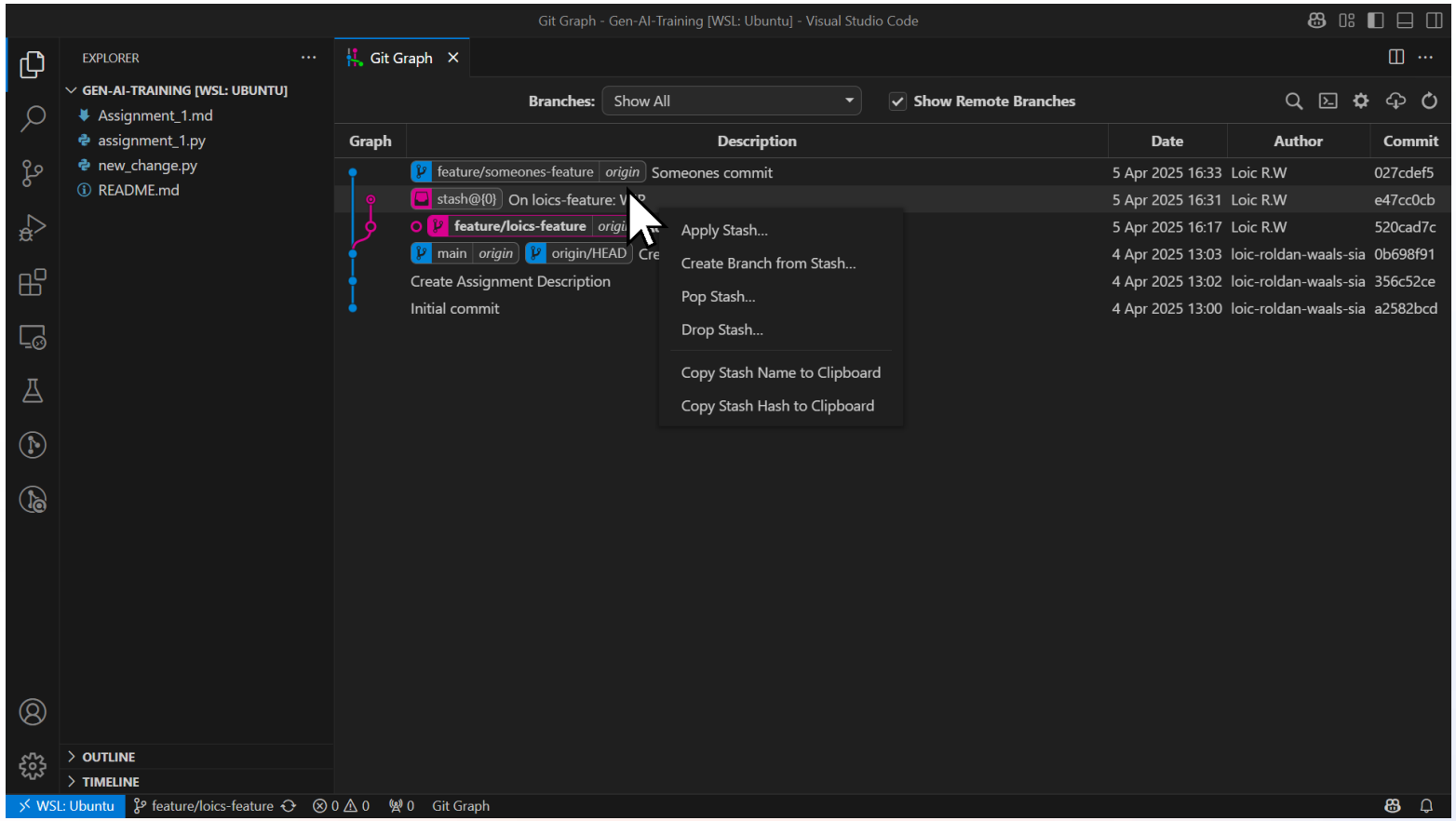
Let’s try to use these commands!

- **Stash** them and switch to some other branch (*feature/someones-feature*).
- This is common when you are reviewing someone else’s work on their branch, but you are not yet finished with your own work.



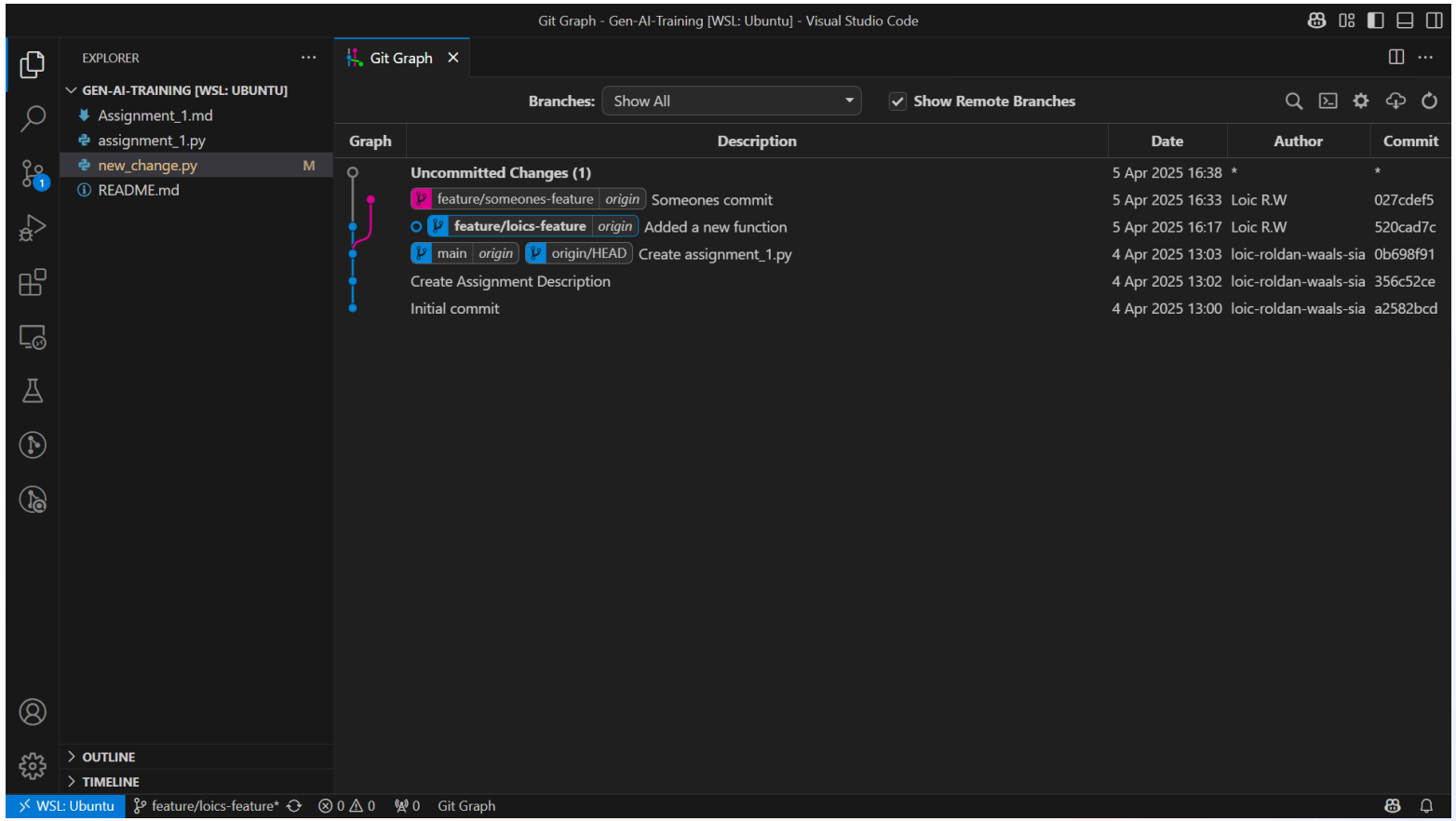
Let’s try to use these commands!

- Come back to the original branch (*feature/loics-feature*) and **un-stash** them.



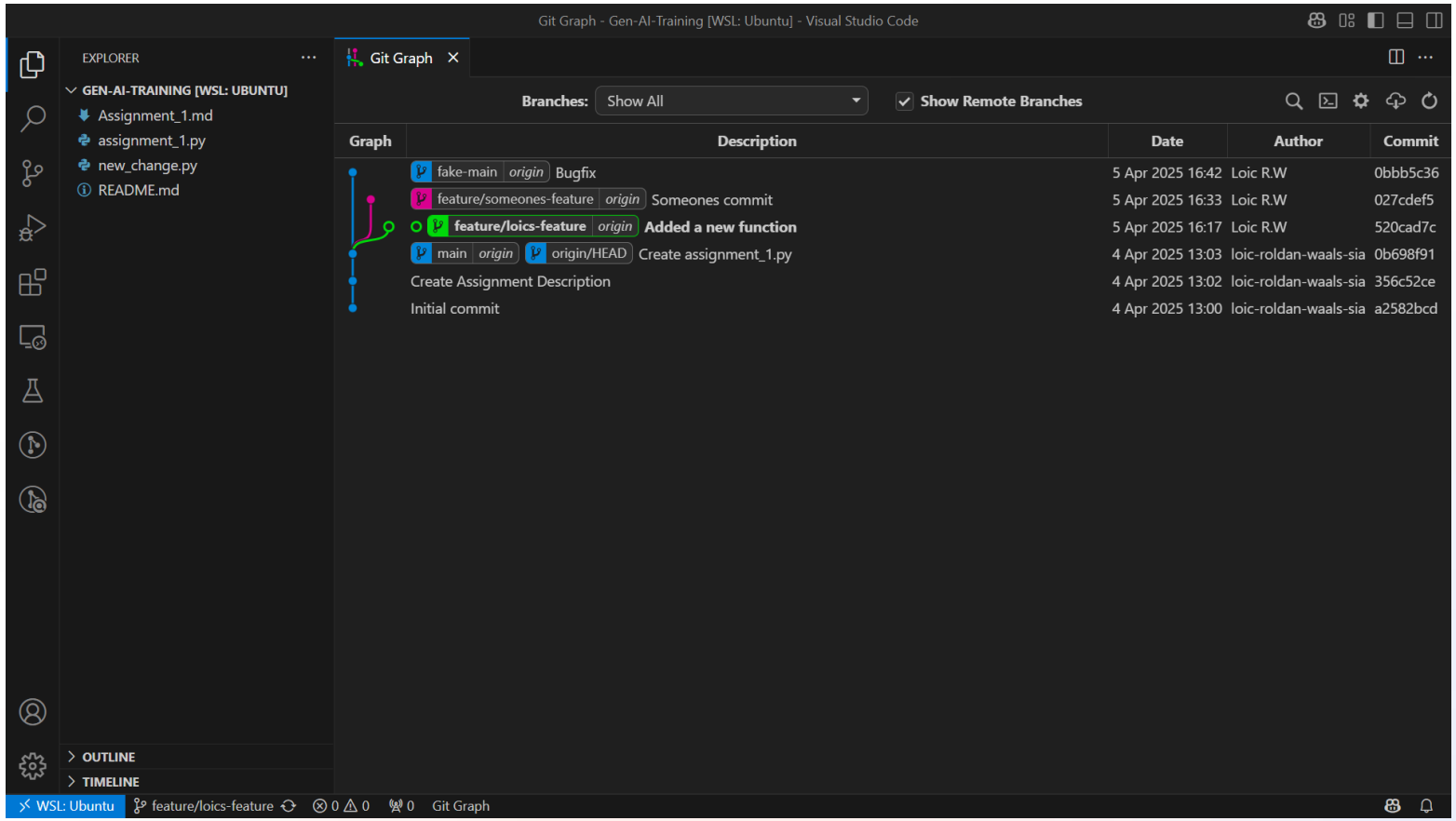
Let’s try to use these commands!

- Wait for some other changes on *fake-main*.



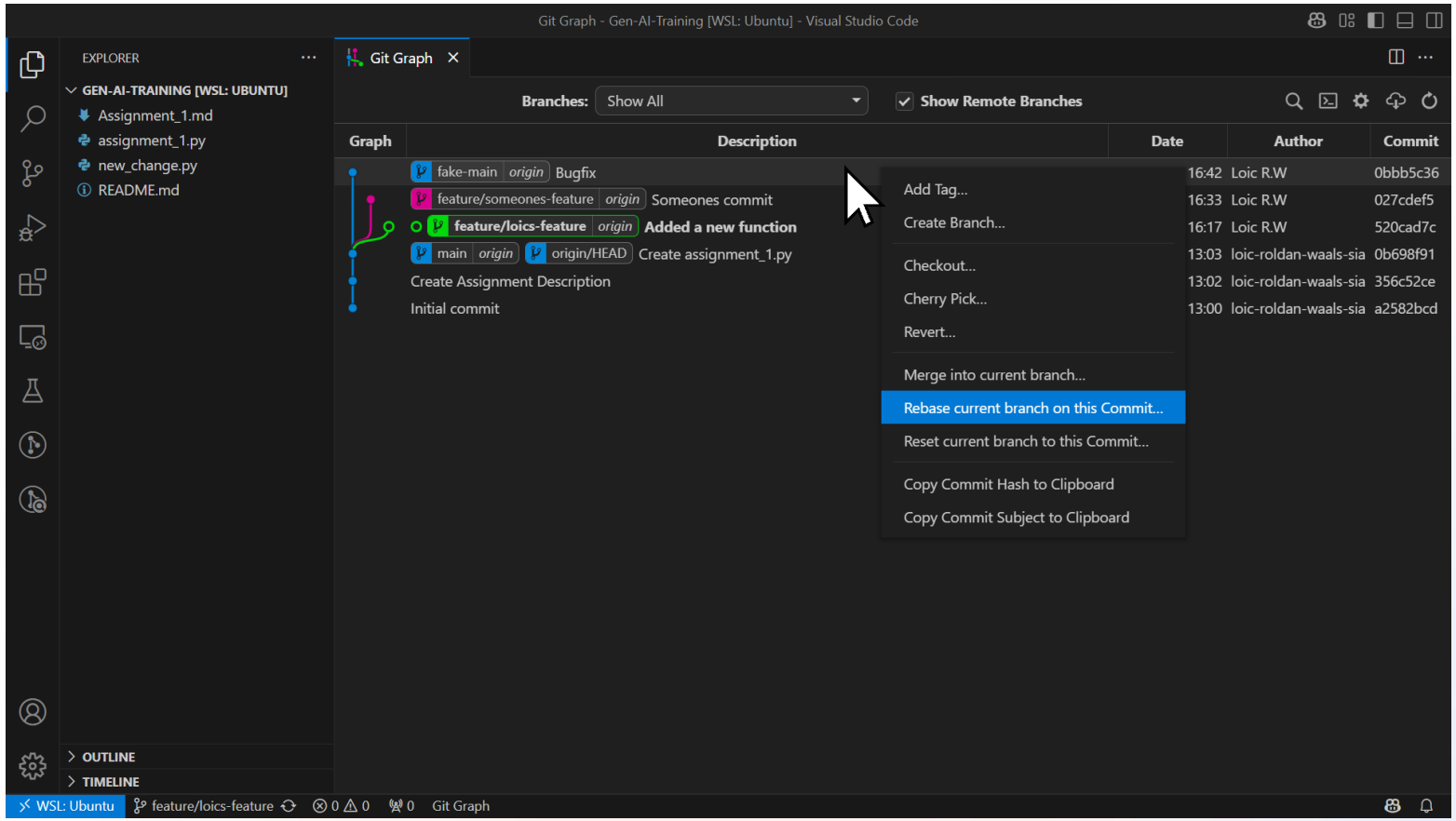
Let’s try to use these commands!

- Wait for some other changes on *fake-main*.



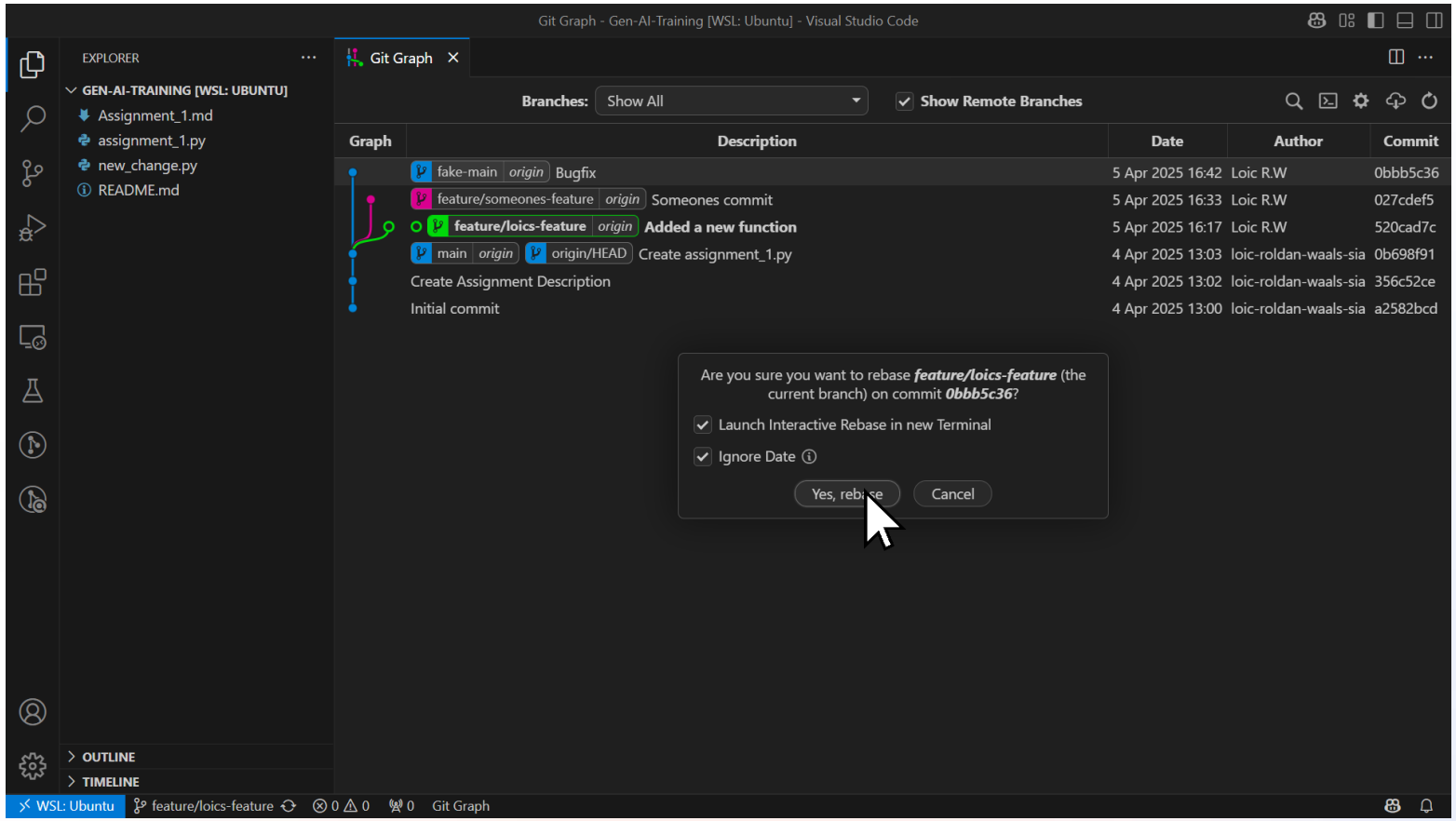
Let’s try to use these commands!

- **Rebase** *feature/loics-feature* on the latest changes from fake-main.



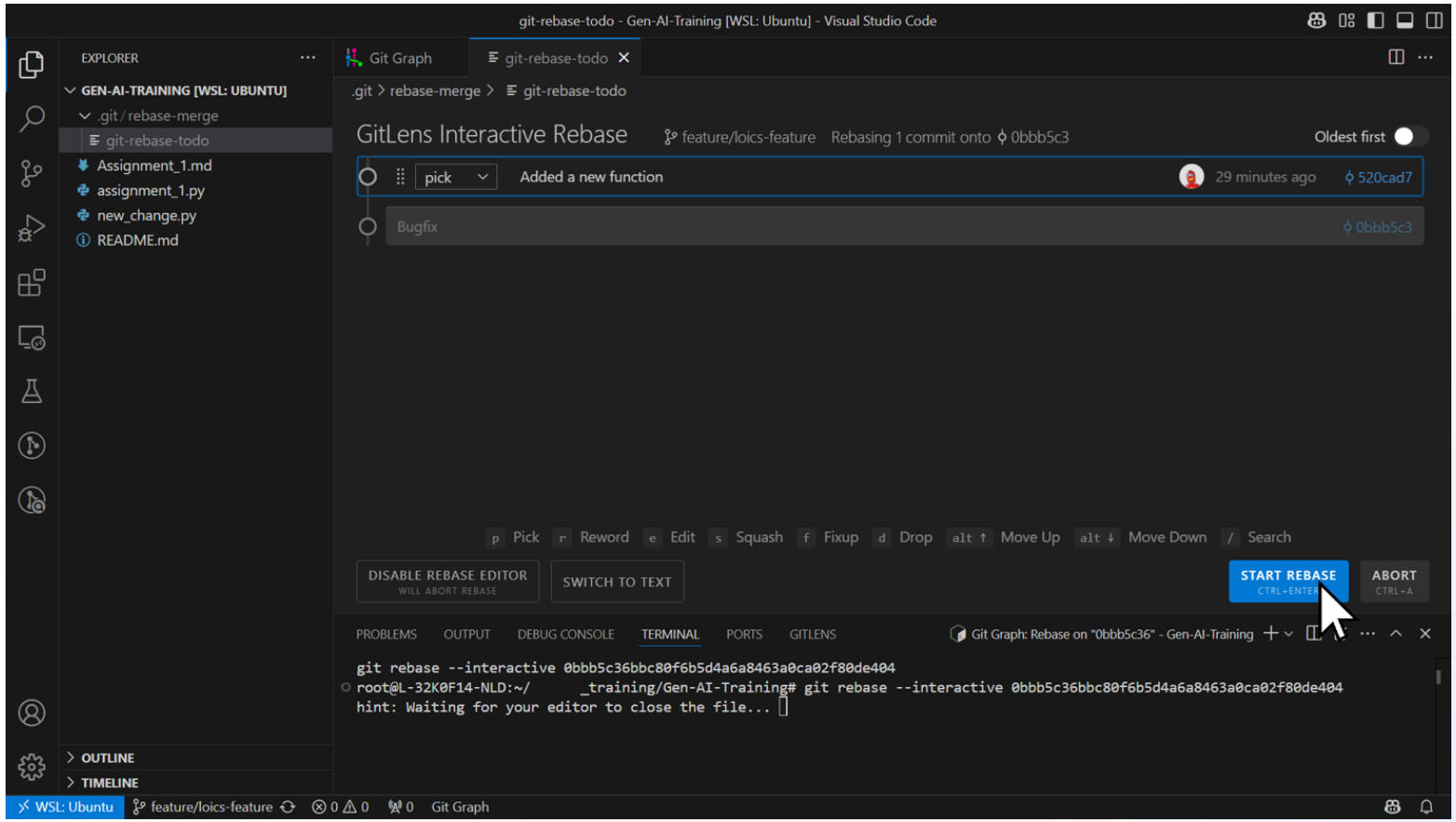
Let’s try to use these commands!

- **Rebase** *feature/loics-feature* on the latest changes from fake-main.



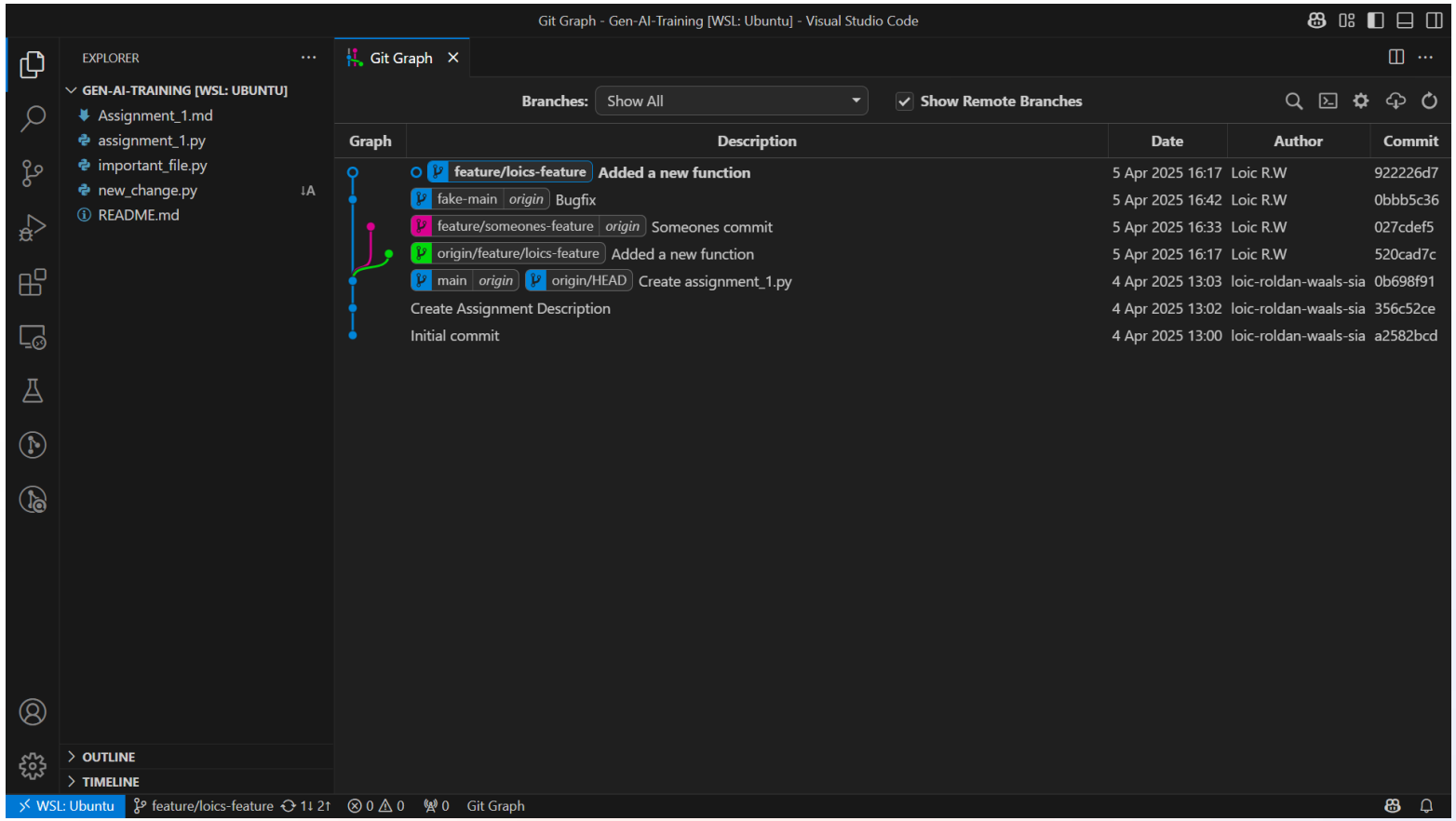
Let’s try to use these commands!

- This is the interactive rebase editor. Here you can:
  - Reword commits
  - Reorder commits
  - Squash (combine) commits (if you want to turn 10 commits into 1)
  - Drop commits (if you want to delete them from the history)
- For now, let’s just start the rebase.



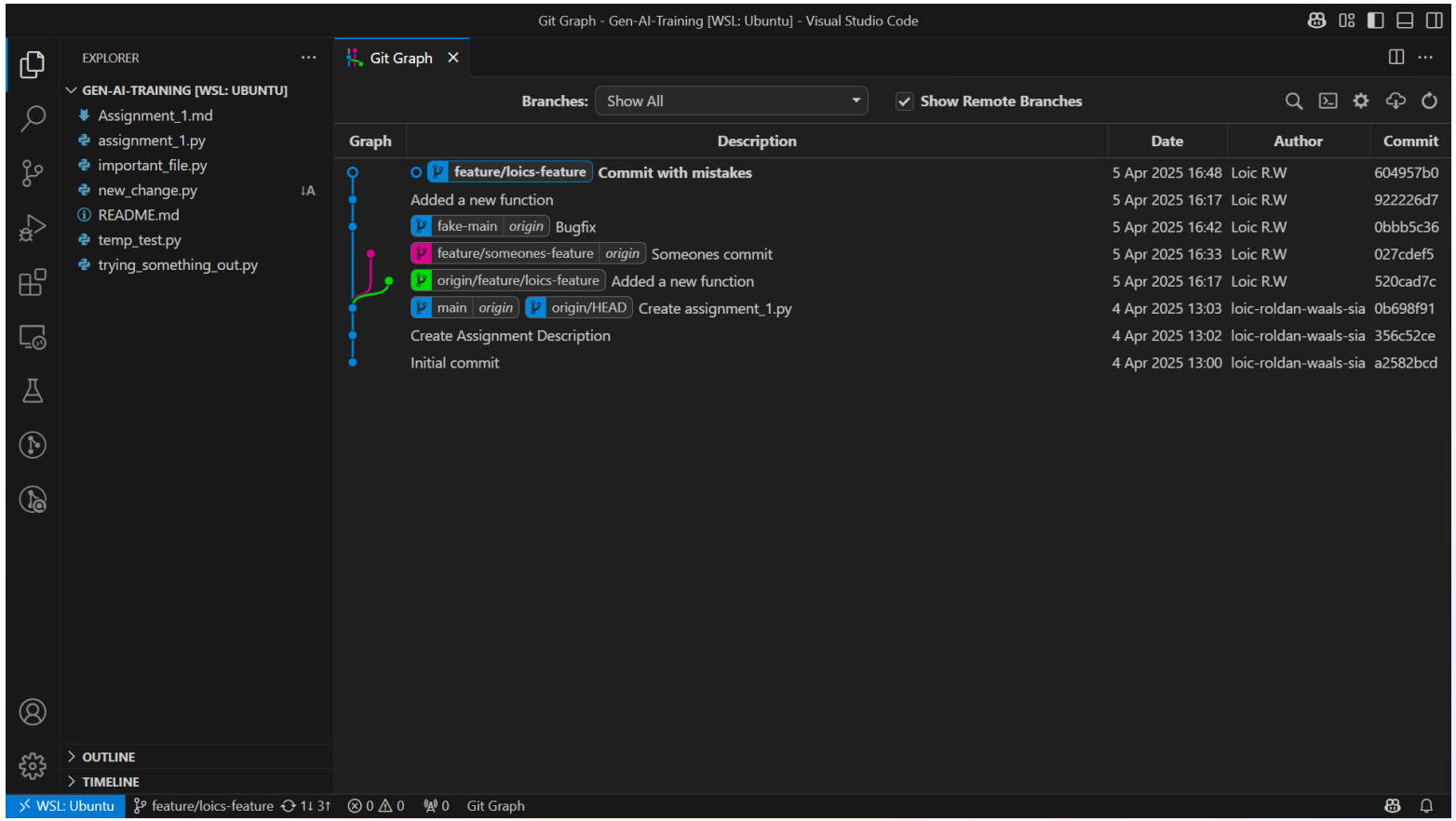
Let’s try to use these commands!

- Now our local is rebased.
- We can continue working on this branch now.



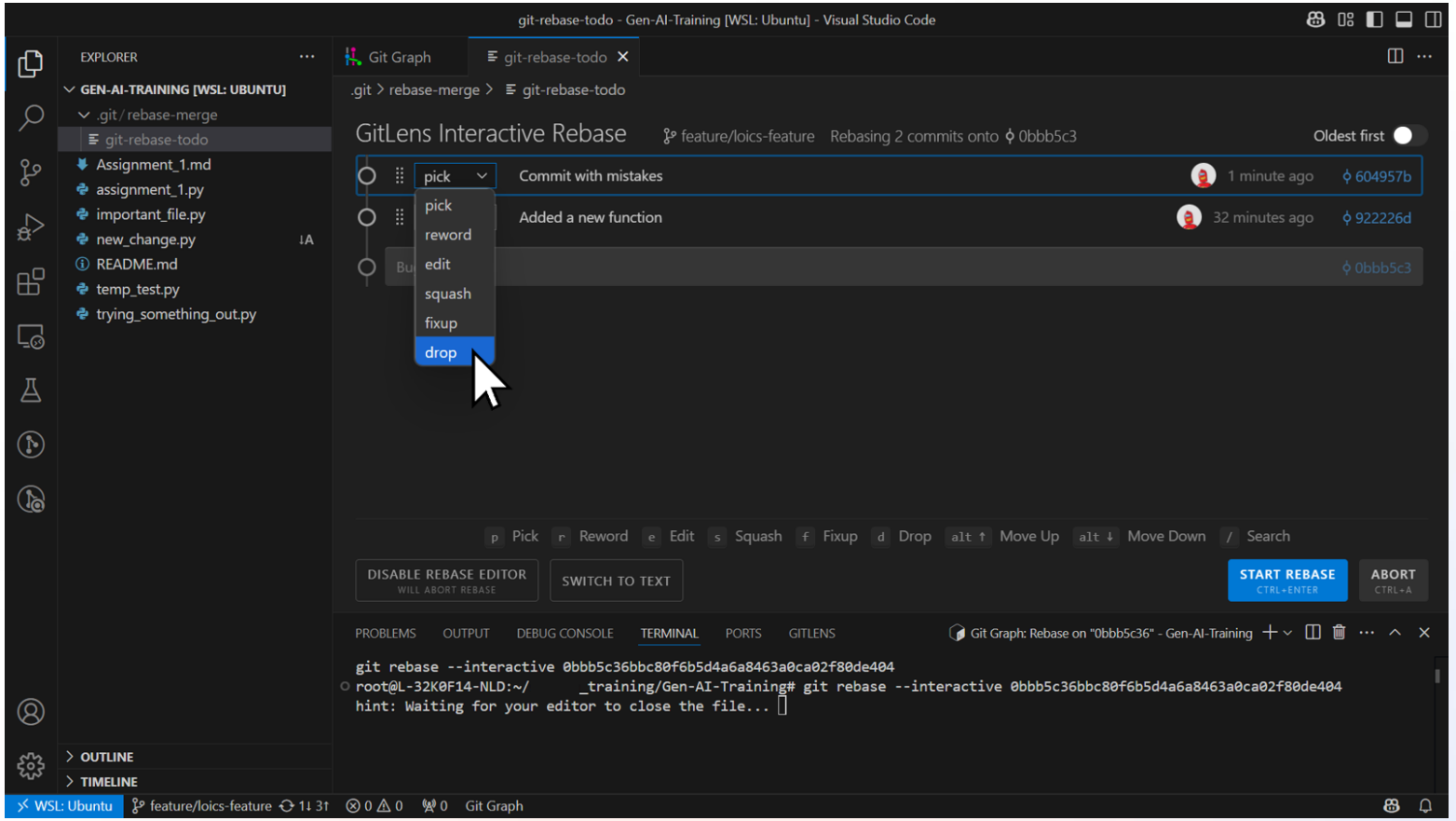
Let’s try to use these commands!

- Oh no! We make multiple **commits** and “accidentally” add some with some unwanted changes.
- **Reset** (--mixed) or **Rebase** changes to create one new clean **commit**.



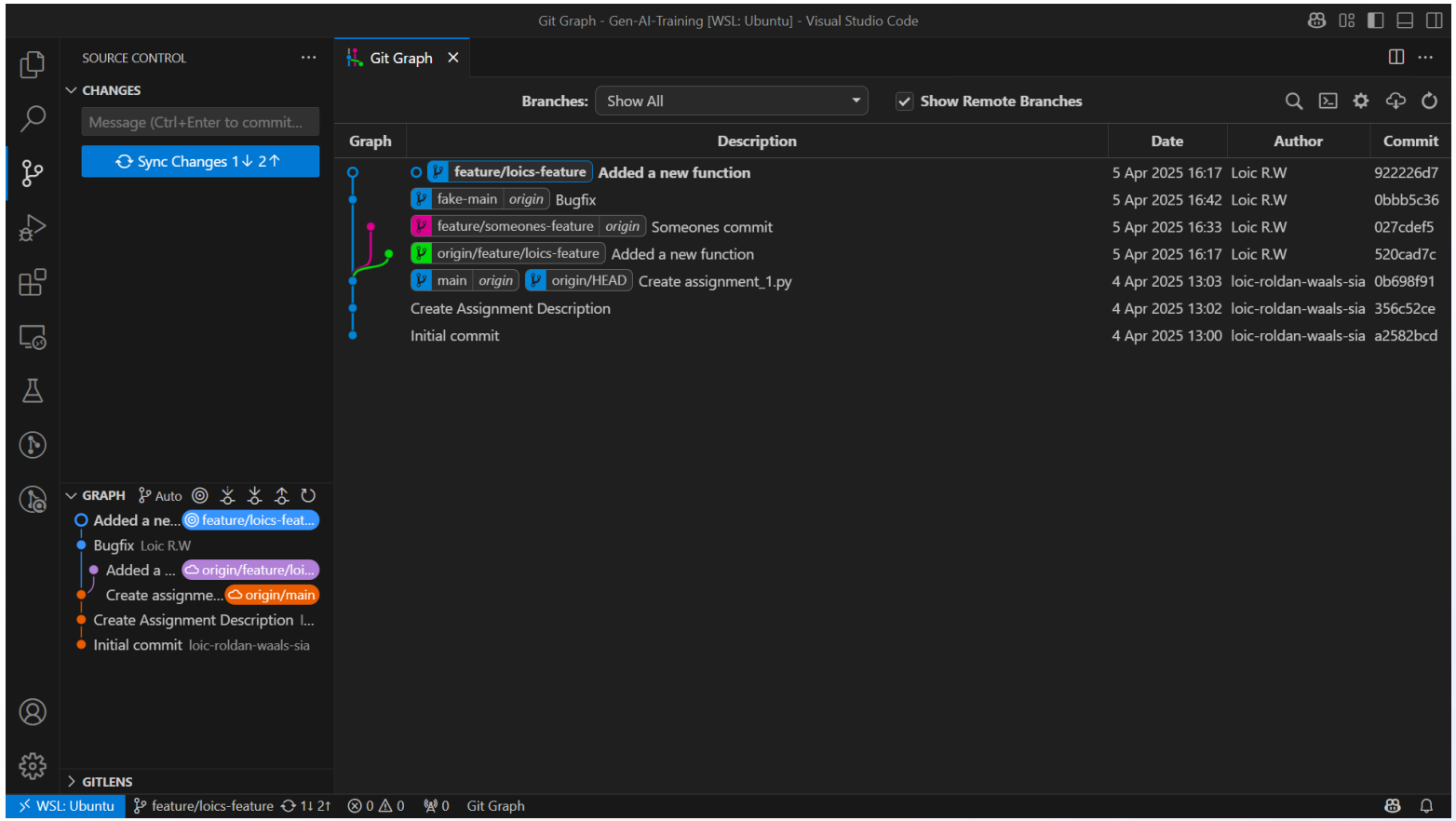
Let’s try to use these commands!

- Oh no! We make multiple **commits** and “accidentally” add some with some unwanted changes.
- **Reset** (--mixed) or **Rebase** changes to create one new clean **commit**.



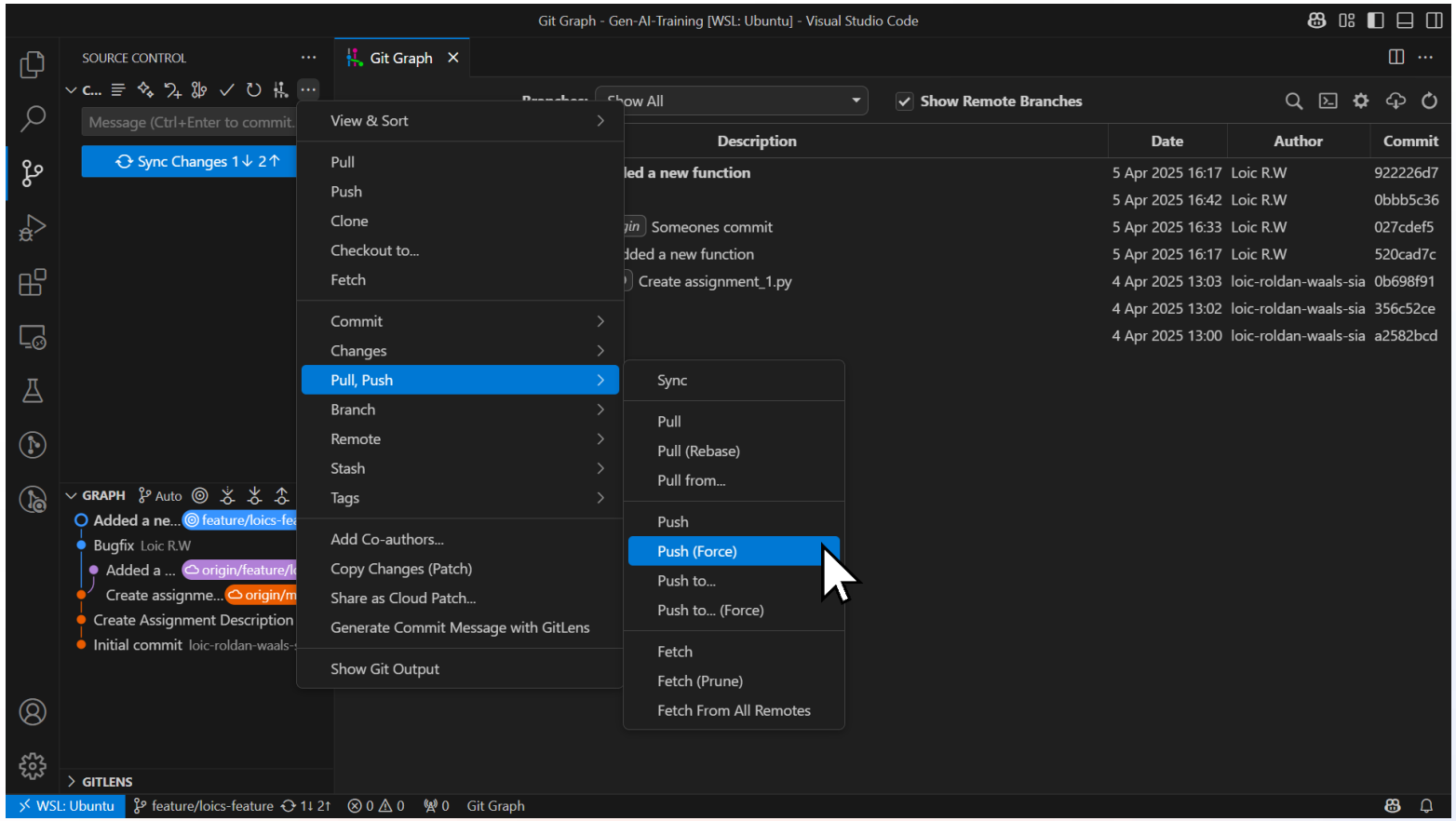
Let’s try to use these commands!

- **Force push** changes to fix the remote.



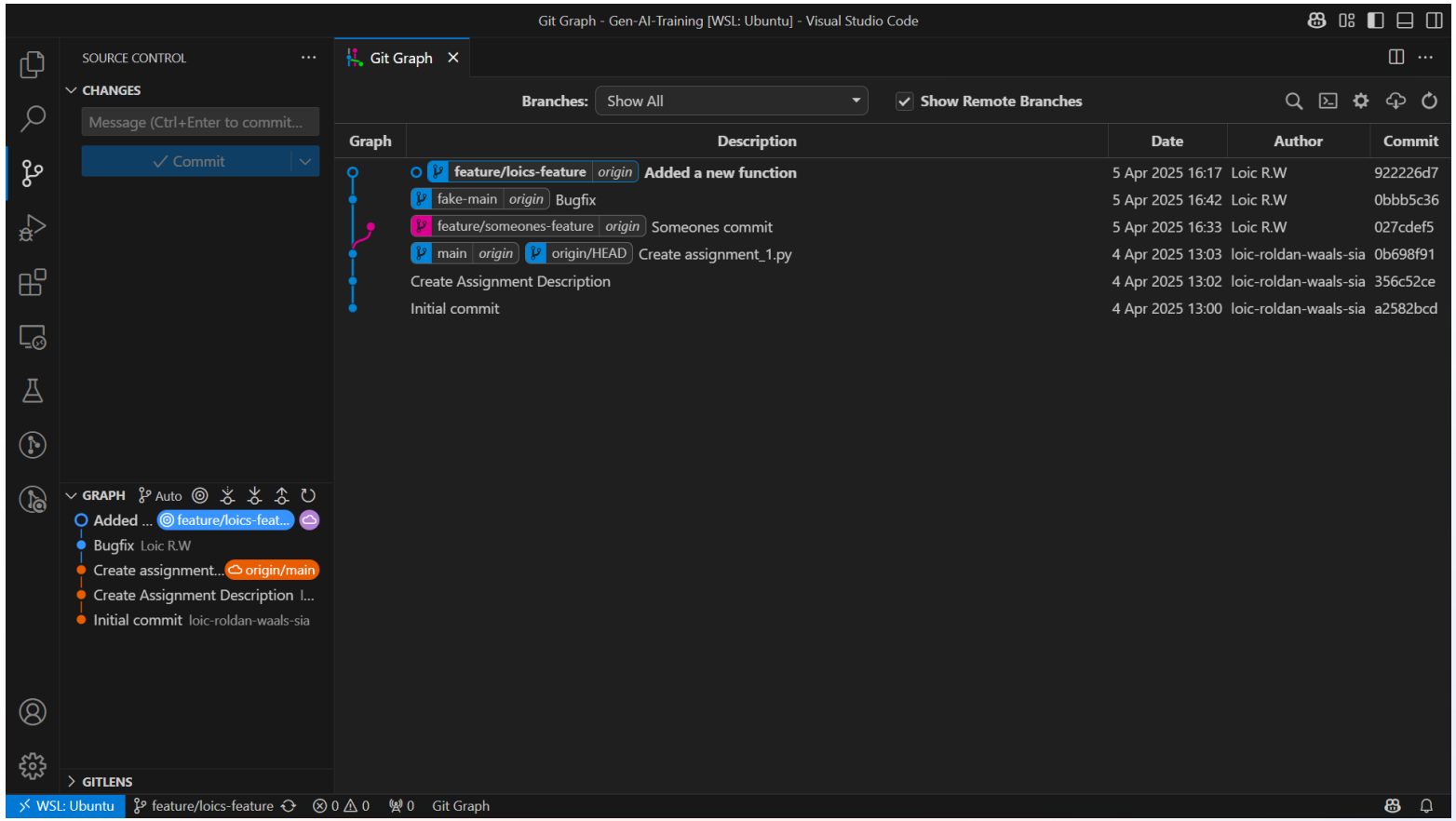
Let’s try to use these commands!

- **Force push** changes to fix the remote.



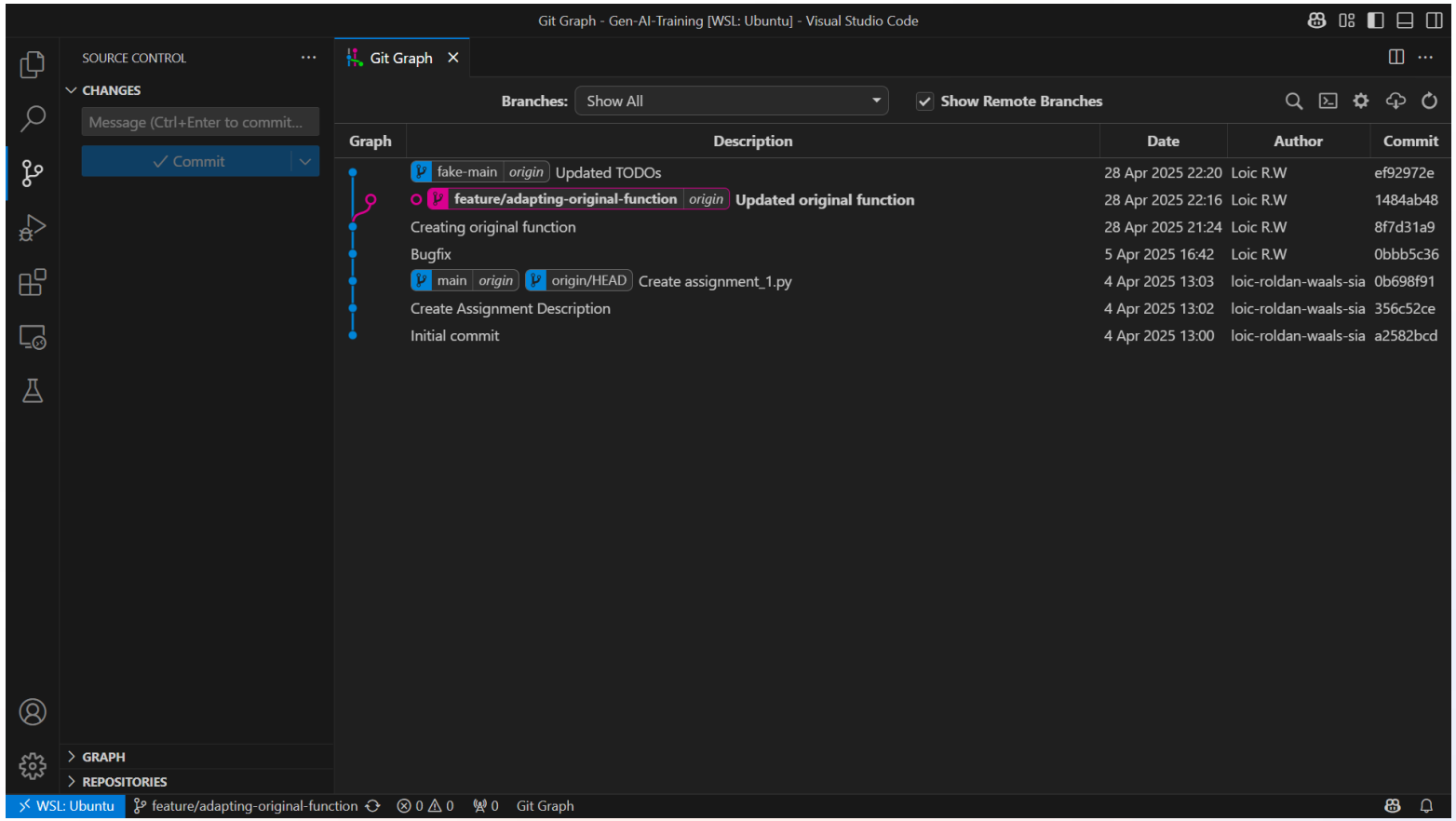
Let’s try to use these commands!

- All done!



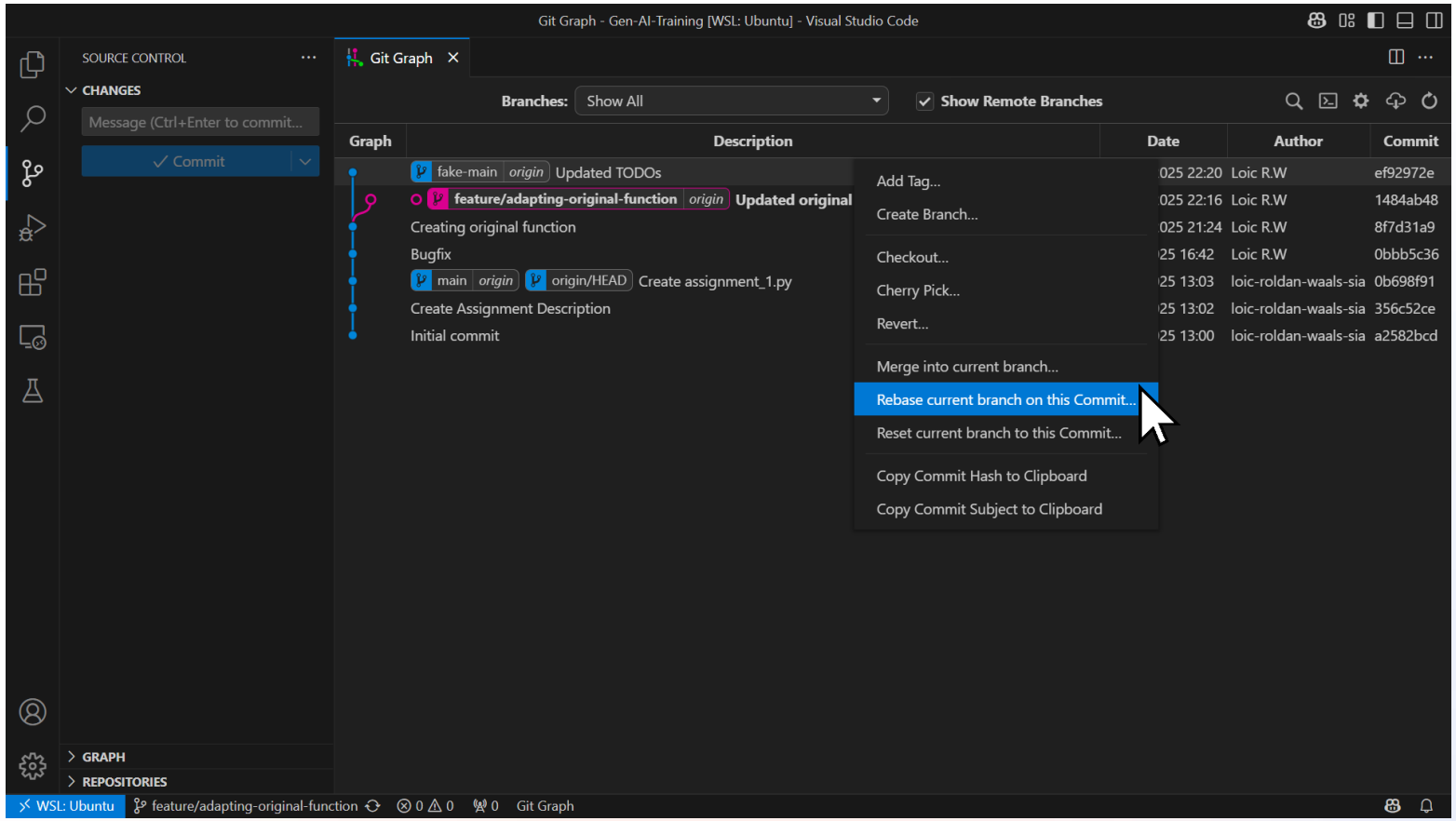
Let’s try to resolve merge conflicts!

- *feature/adapting-original-function* is based off *fake-main.*
- The same file is changed in both branches.
- When we try to rebase onto *fake-main* we trigger the conflict.



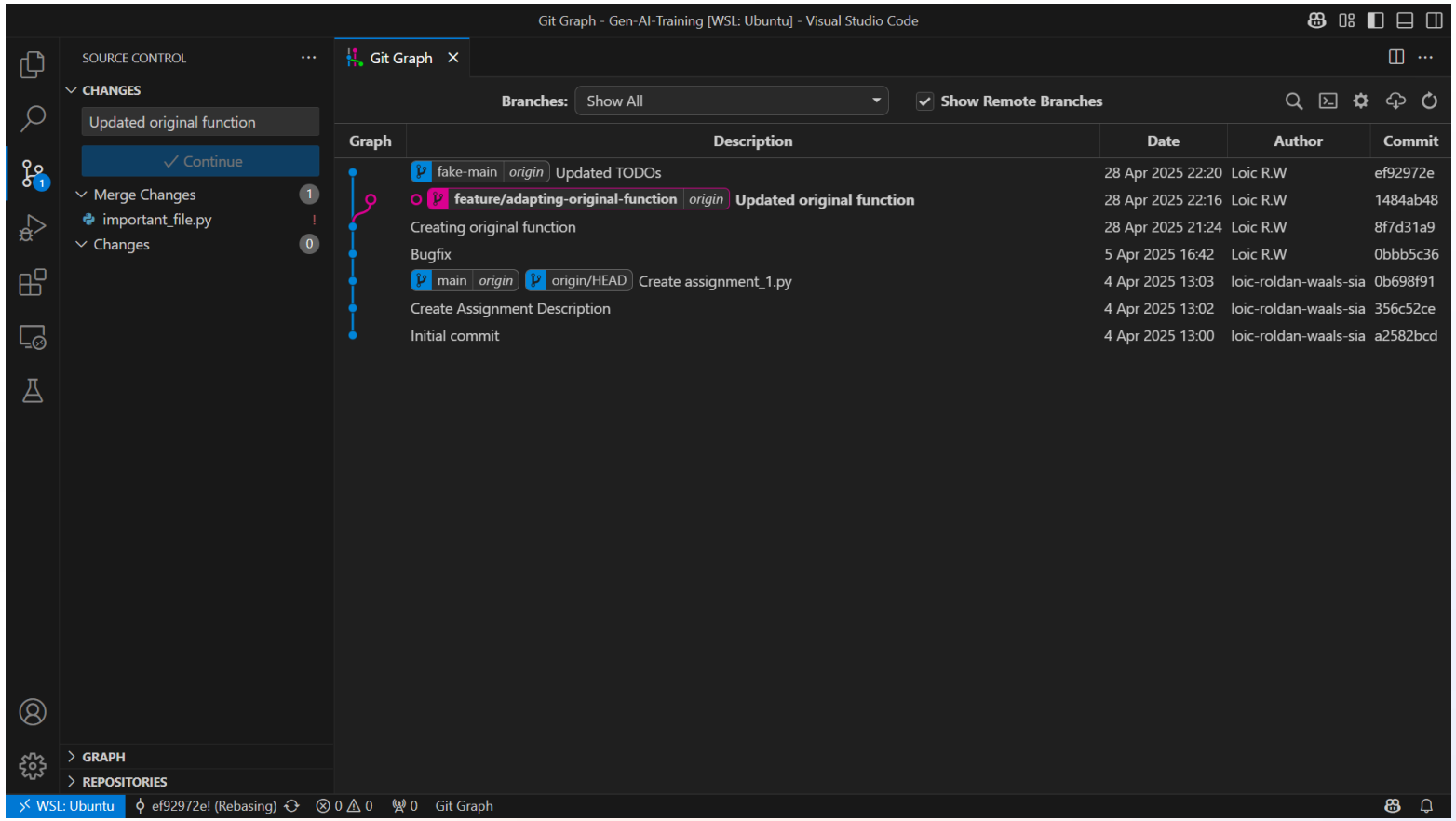
Let’s try to resolve merge conflicts!

- *feature/adapting-original-function* is based off *fake-main.*
- The same file is changed in both branches.
- When we try to rebase onto *fake-main* we trigger the conflict.



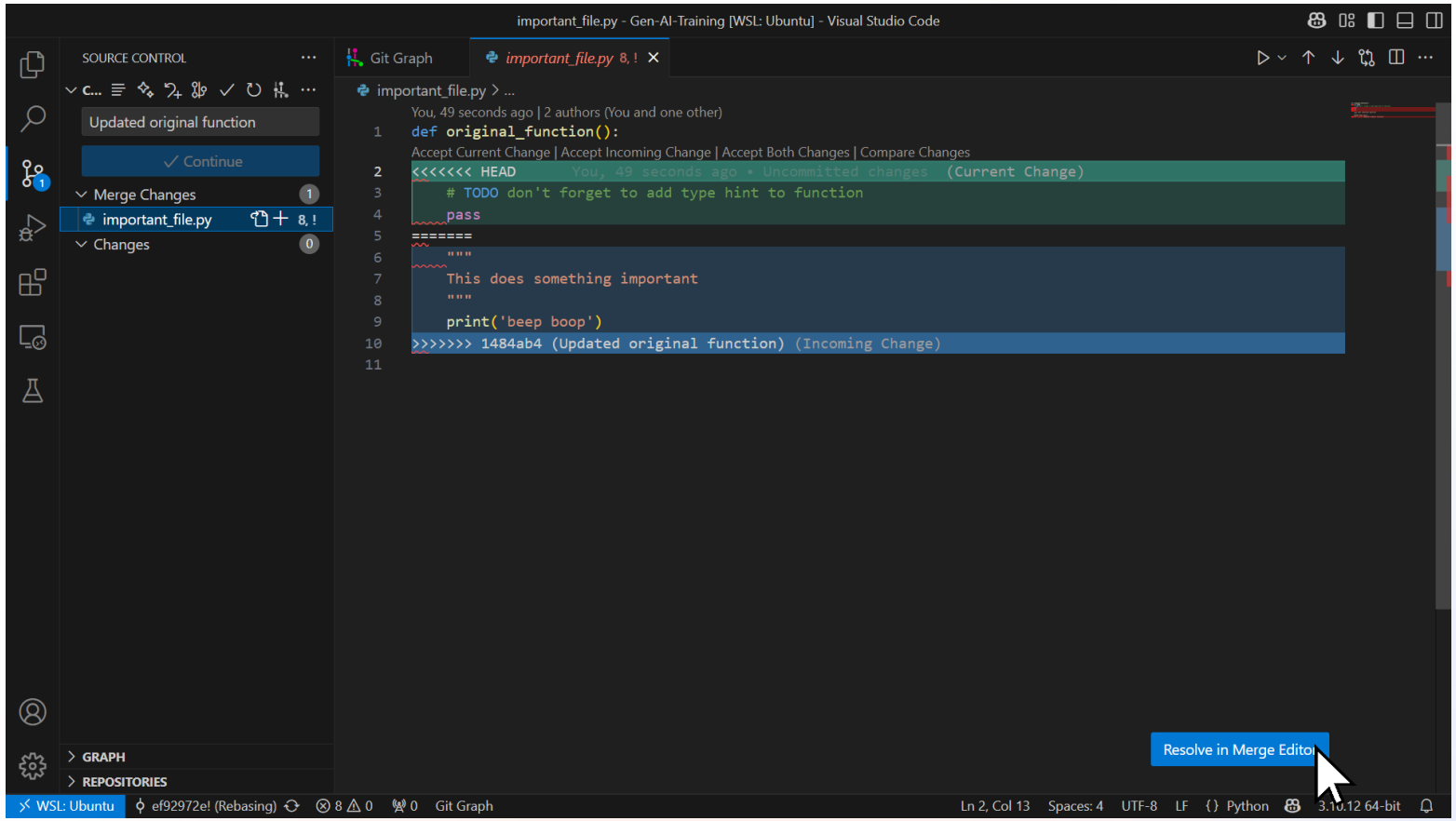
Let’s try to resolve merge conflicts!

- *feature/adapting-original-function* is based off *fake-main.*
- The same file is changed in both branches.
- When we try to rebase onto *fake-main* we trigger the conflict.



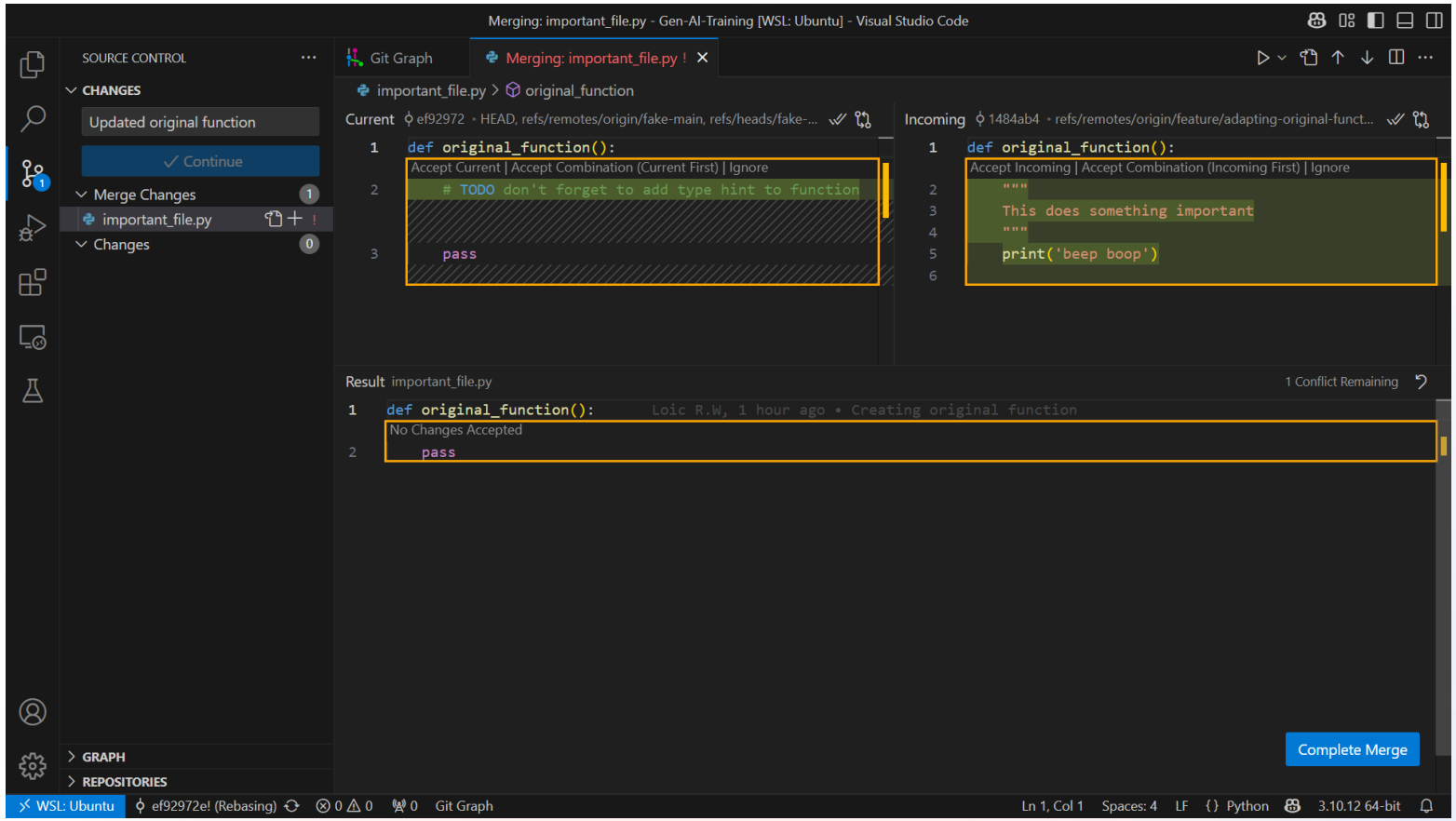
Let’s try to resolve merge conflicts!

- We can fix the changes (file by file) by clicking them in the source control tab and pressing the button on the bottom right that says “Resolve in Merge Editor”



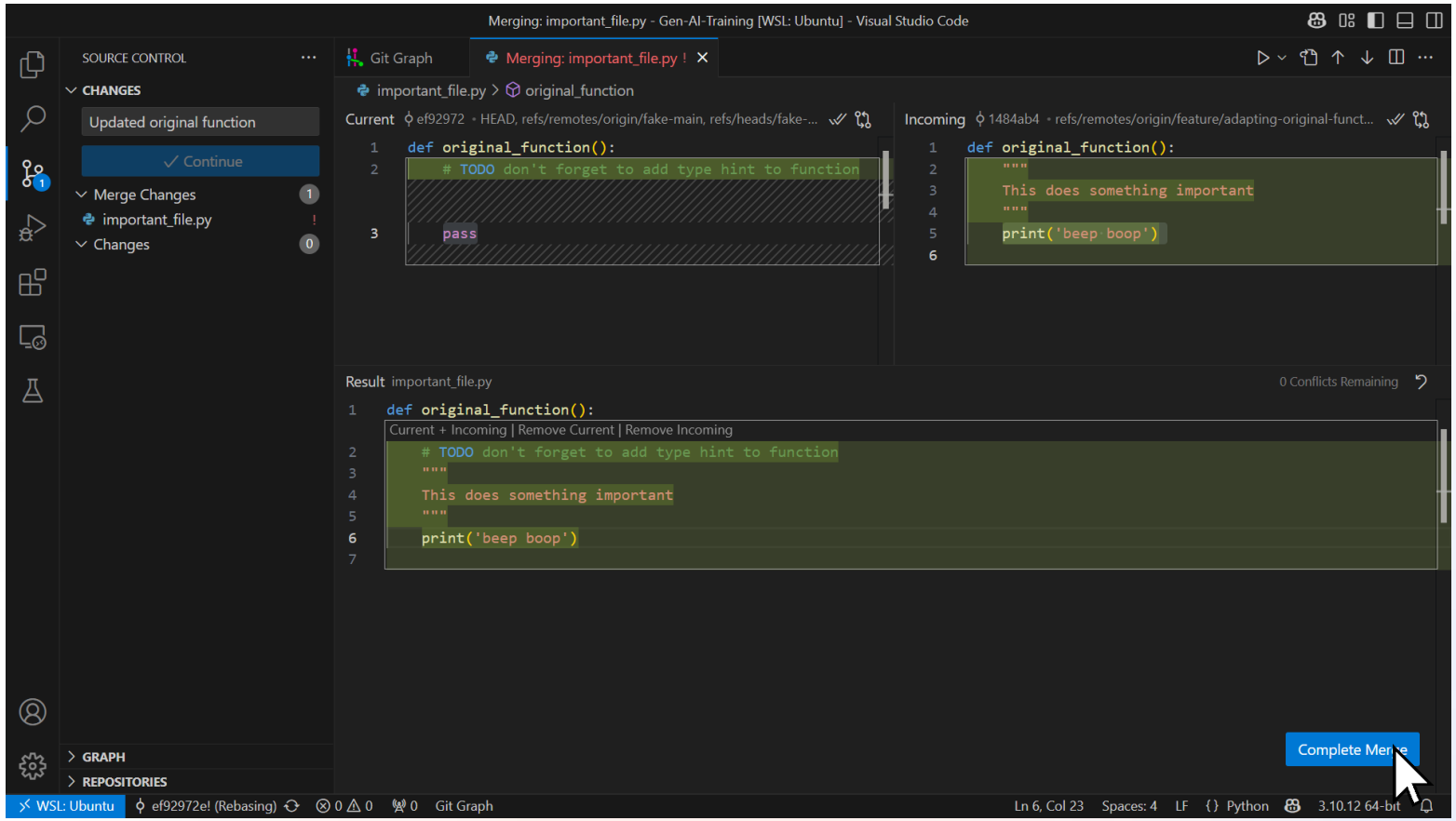
Let’s try to resolve merge conflicts!

- Ensure that the code in the “Result” tab on the bottom is what you would like to see.
- You can use the buttons like “Accept Current” to speed this up



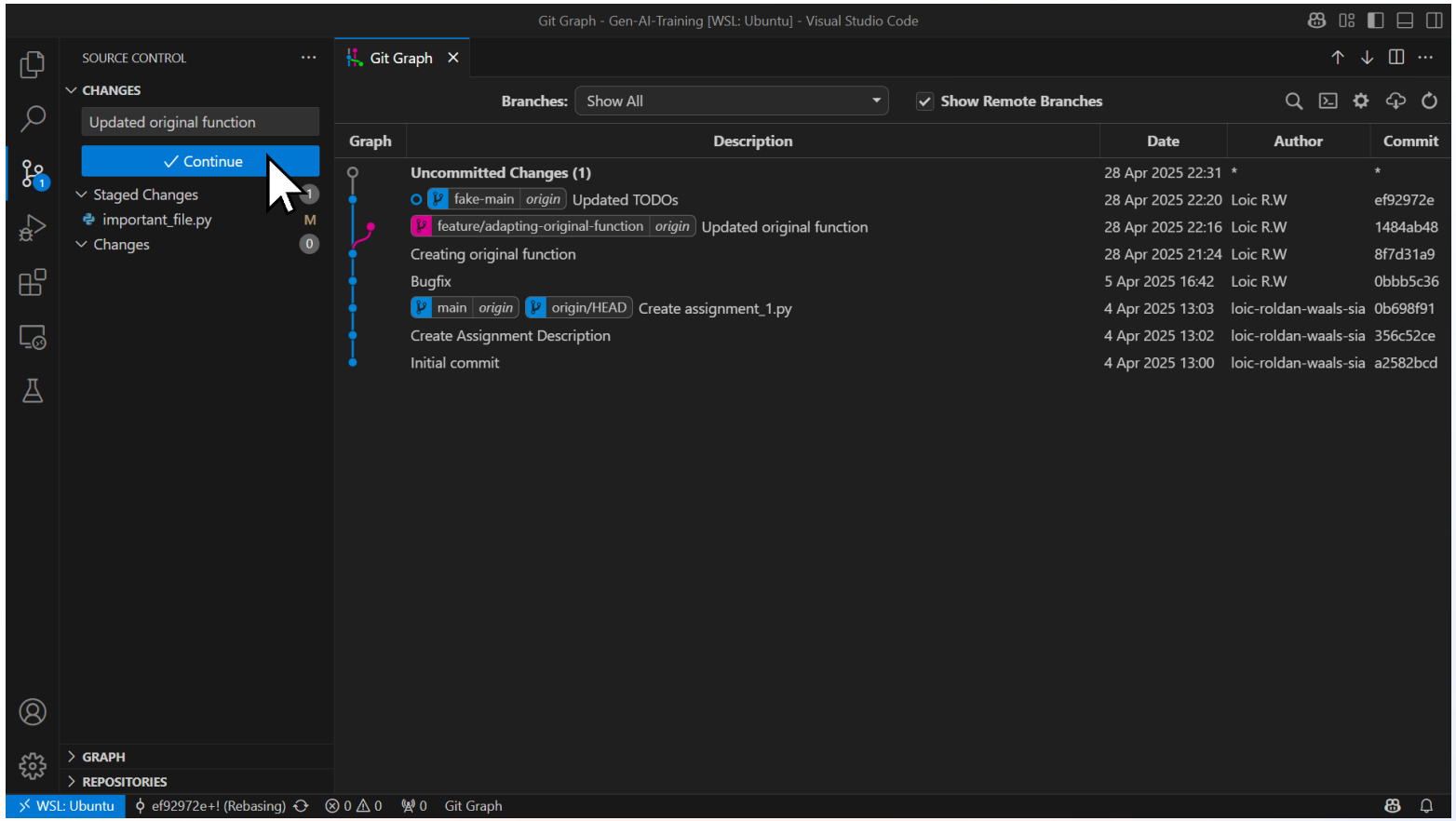
Let’s try to resolve merge conflicts!

- Press “Complete Merge” to finish.



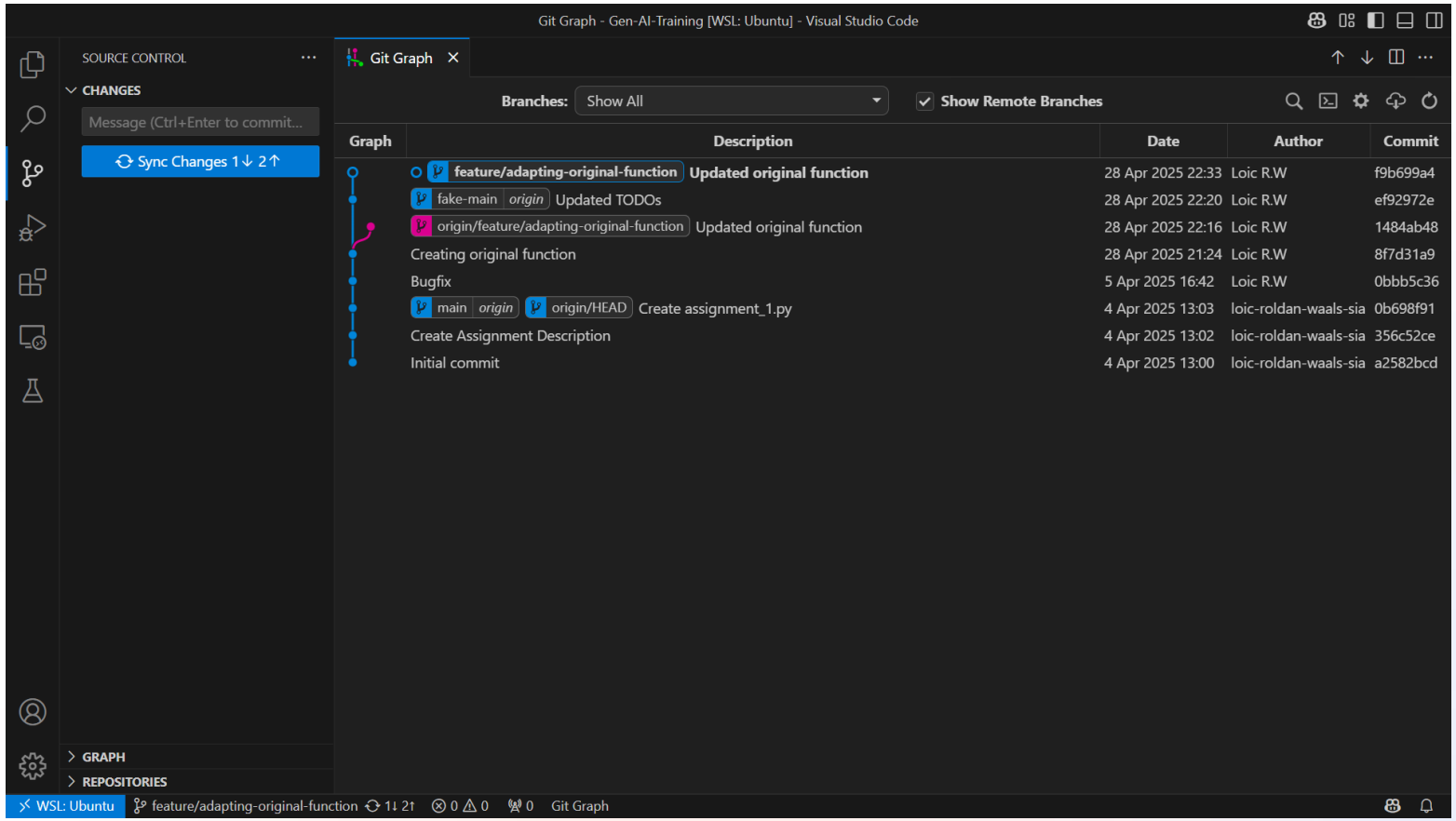
Let’s try to resolve merge conflicts!

- At this point you can add additional changes to the commit.
- Press “Continue” to confirm your merges.



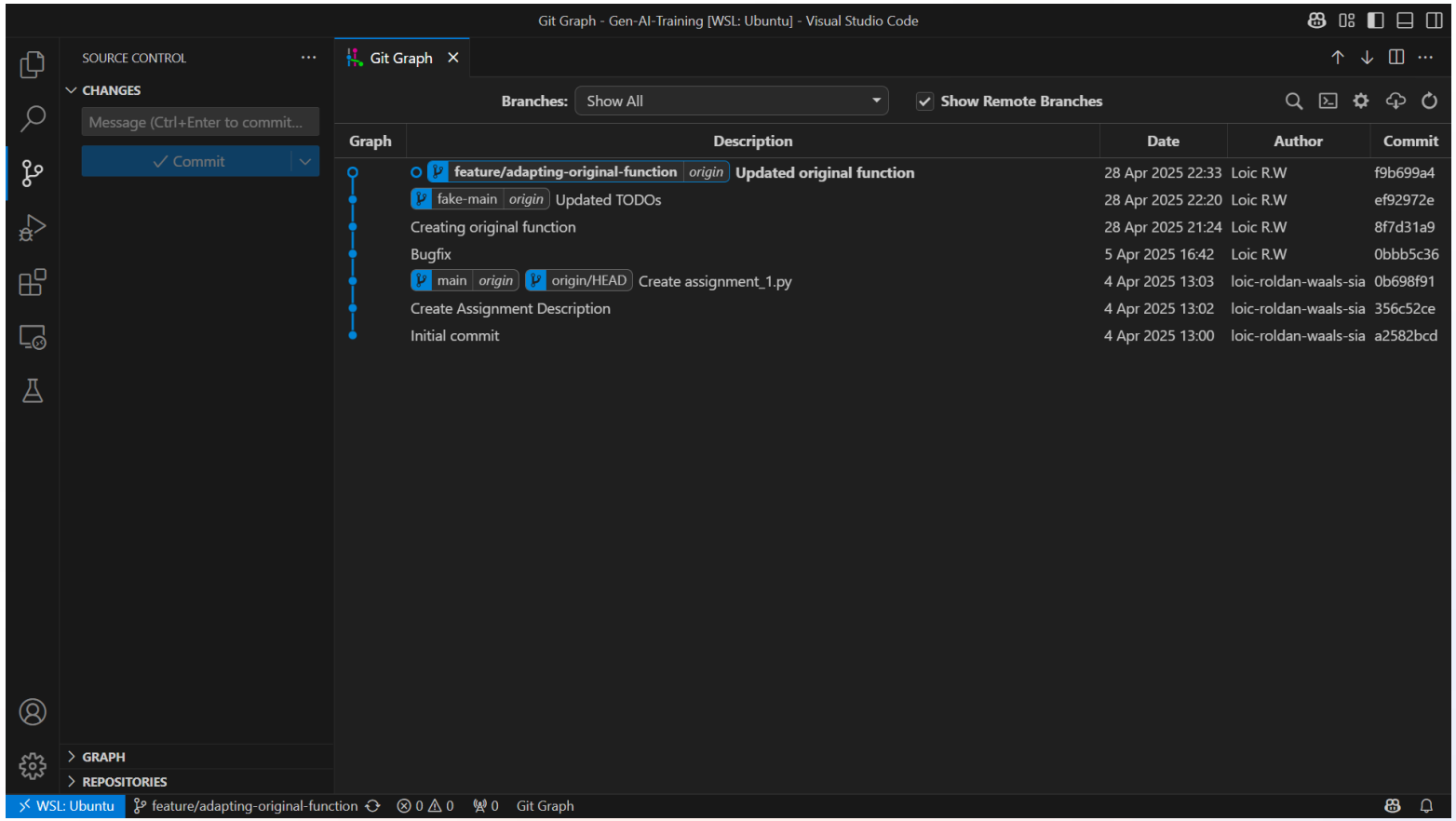
Let’s try to resolve merge conflicts!

- Lastly, force push your changes to overwrite the (incorrect) commit on the remote.



Let’s try to resolve merge conflicts!

- Lastly, force push your changes to overwrite the (incorrect) commit on the remote.



To ensure only correct code goes to production we review each other’s work

Common PR Challenges

- Large Pull Requests -> Large pull requests are difficult to review. Break them down into smaller, more manageable chunks. Only include code that actually contributes to the change.
- Unresolved Conflicts -> Regularly sync your branch with the main branch to avoid complex conflicts later.
- Stale Pull Requests -> If a pull request becomes outdated, communicate with the team to update or close it.

Good reference guide from google: Link

PR Best Practices

- Keep Pull Requests Small.
- Write Descriptive Titles and Descriptions.
- Address Feedback Promptly.
- Communicate Clearly.
- Branch hygiene.

Tips for Merge Conflicts

- Ensure you have as few commits as possible.
- Always re-test your code completely.
- They are always a headache, preventing them (with frequent rebases) is better than fixing them.

Let’s practice!

Changing some configurations will make it easier to work with git.

VS Code

- For VSCode settings, use Ctrl + Shift + P and write "open default settings (JSON)".
- Choose the JSON option and paste the provided settings.
- Install Git Graph and GitLens for nicer GUI

Git

- Open a terminal
- Enter the commands:
  - git config --global core.editor "code –-wait"
  - git config --global –e
- Then paste the provided example settings.
- Close the file to save it.

SIA

We're a next-gen consulting group for a new generation of change.

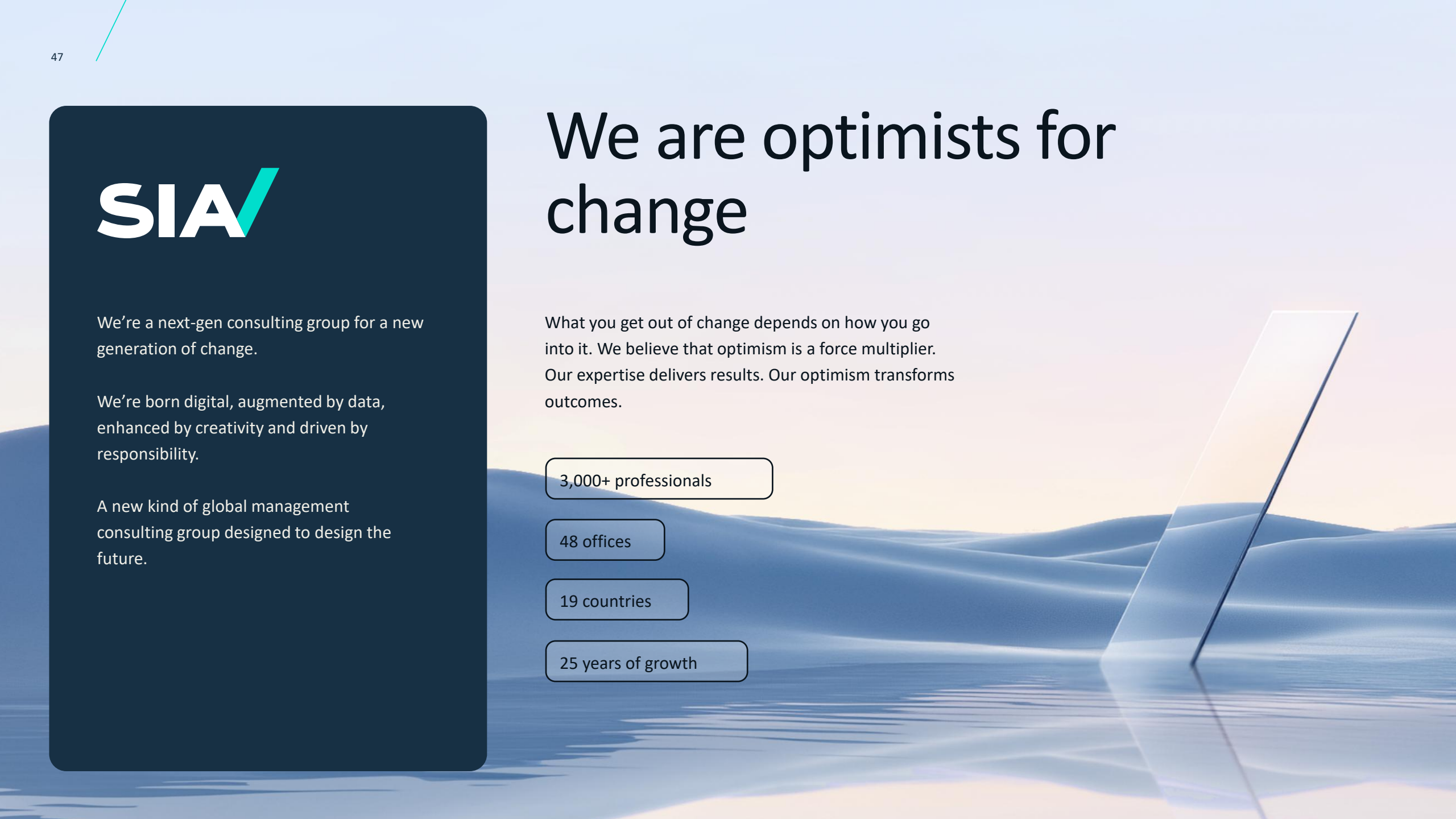
We're born digital, augmented by data, enhanced by creativity and driven by responsibility.

A new kind of global management consulting group designed to design the future.

We are optimists for change

What you get out of change depends on how you go into it. We believe that optimism is a force multiplier. Our expertise delivers results. Our optimism transforms outcomes.

- 3,000+ professionals
- 48 offices
- 19 countries
- 25 years of growth



Positioned as a leader in AI adoption

SIA AI

AI and data science is the powerhouse of Sia, augmenting consulting services and accelerating productivity.

- 350+ AI & data experts
- 12 R&D labs
- Advanced GenAI assistants

Our AI x consulting model delivers scalable business solutions

SIA AI

AI and data science is the powerhouse of Sia, augmenting consulting services and accelerating productivity.

- 350+ AI & data experts
- 12 R&D labs
- Advanced GenAI assistants

AI consulting services

Building data driven, AI driven organization and agentic AI architectures and covering
end-to-end AI value chain, from ideation to industrialization (data science, data marketing, data engineering & AI technologies)

In-house AI platform and AI-fueled datasets built

Thanks to unique and distinctive data scraping capabilities and data science expertise, available for consultants and clients

Augmented consulting services

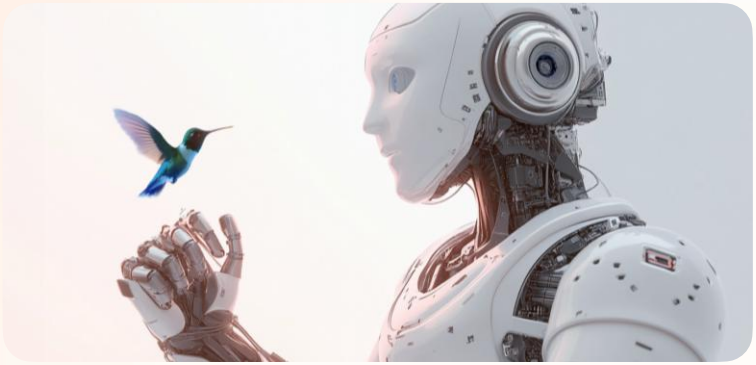
Combining strategy/industry expertise
and AI in the same engagements

SiaGPT

Set of advanced GenAI assistants, based on evolving multi-LLMs technologies. Available as white label GPT infrastructure and AI/GenAI use cases industrialization platform

12 R&D labs coverings all AI techniques

NLP & GenAI, computer vision, time series, operational research, data capture, data duality, engineering, marketing analytics, advanced analytics, climate, asset & liability management, blockchain & cryptocurrencies



Merci

Sia is a next-generation, global management consulting group—born digital, augmented by data, enhanced by creativity, and driven by responsibility. Our 3,000+ professionals partner with clients to resolve challenges and capitalize on opportunities. We believe that in today's world of change and disruption, optimism is a force multiplier.

sia-partners.com | follow us: @SiaPartners

SIA

Optimists for change