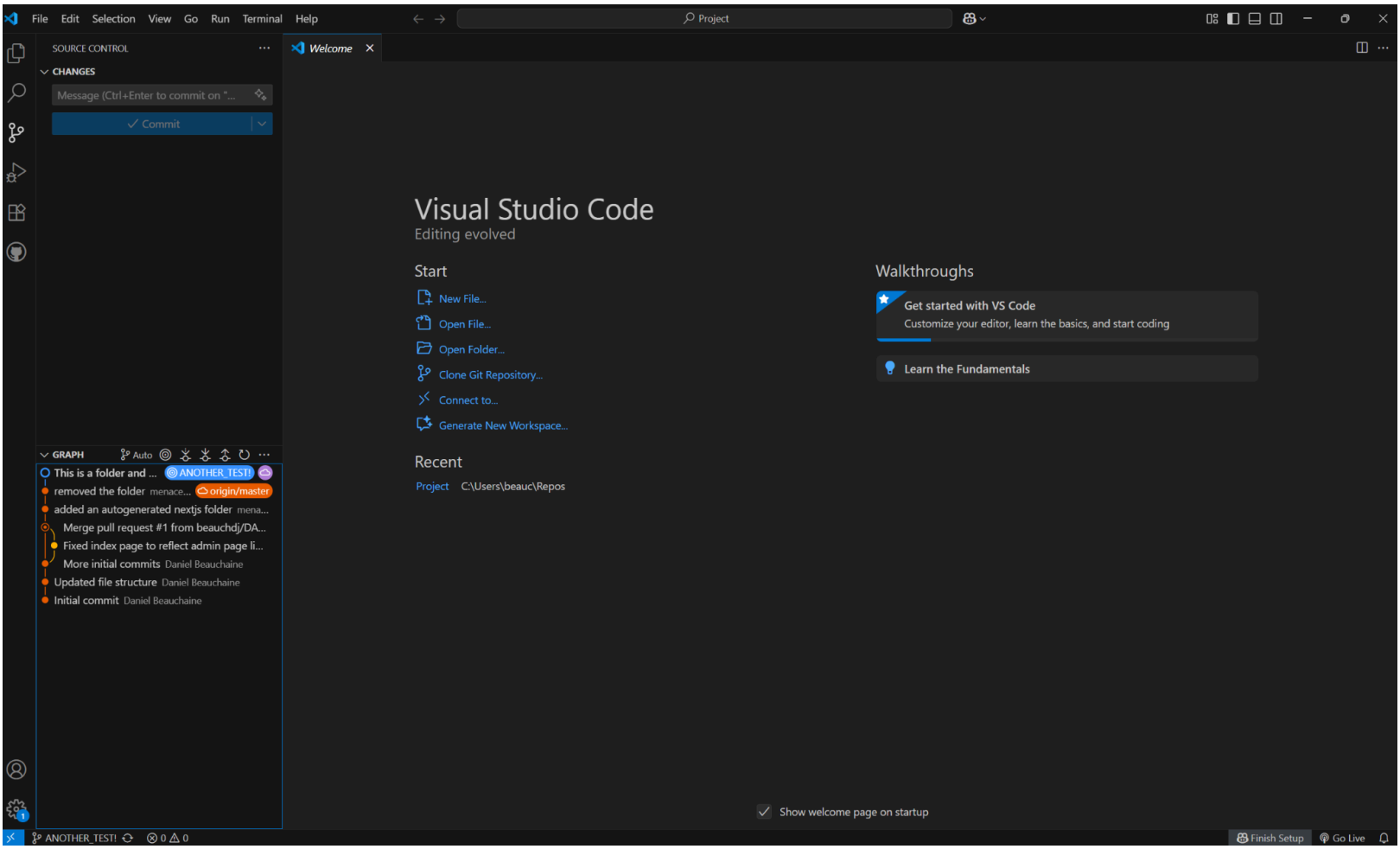
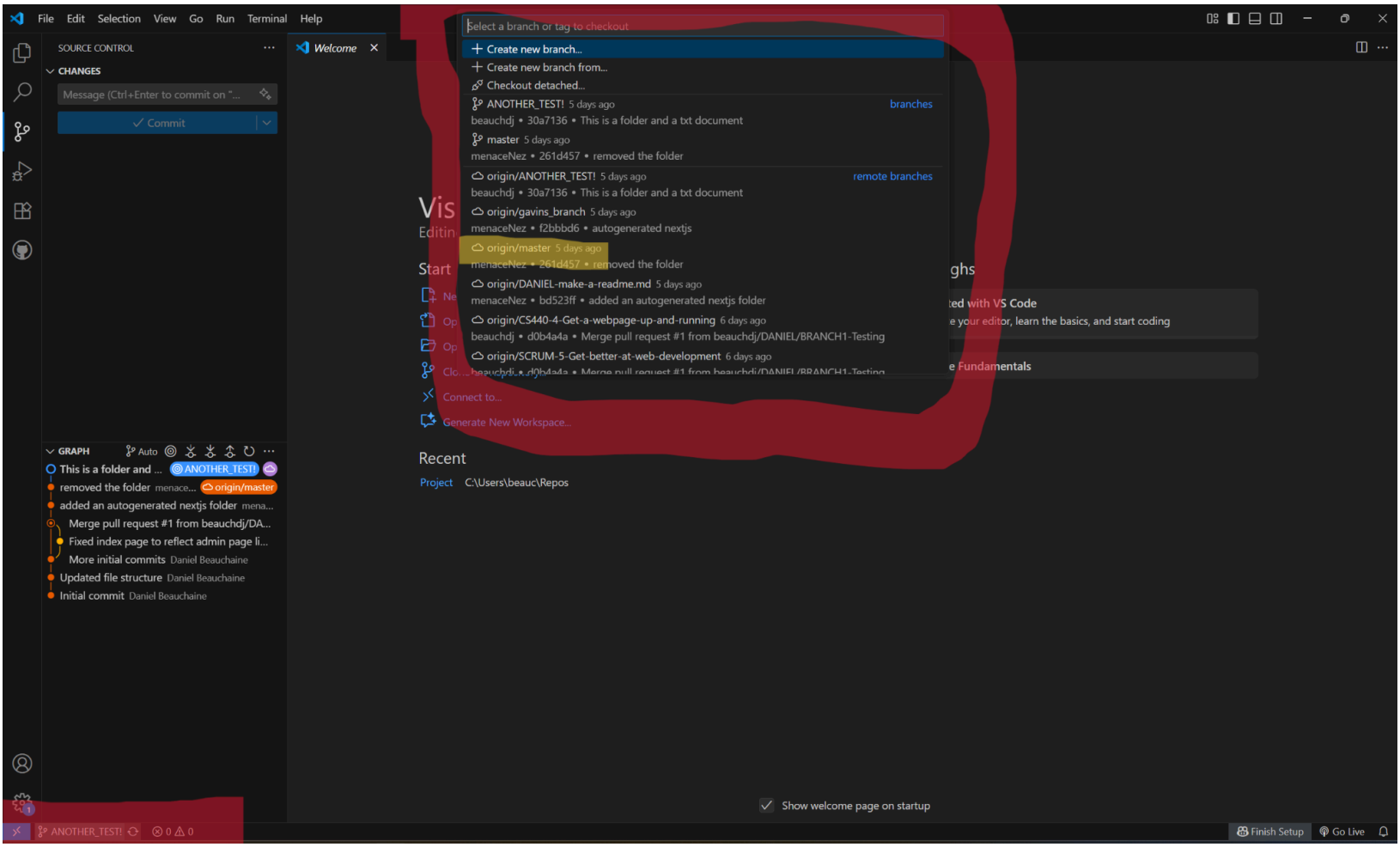
Starting from a blank slate.

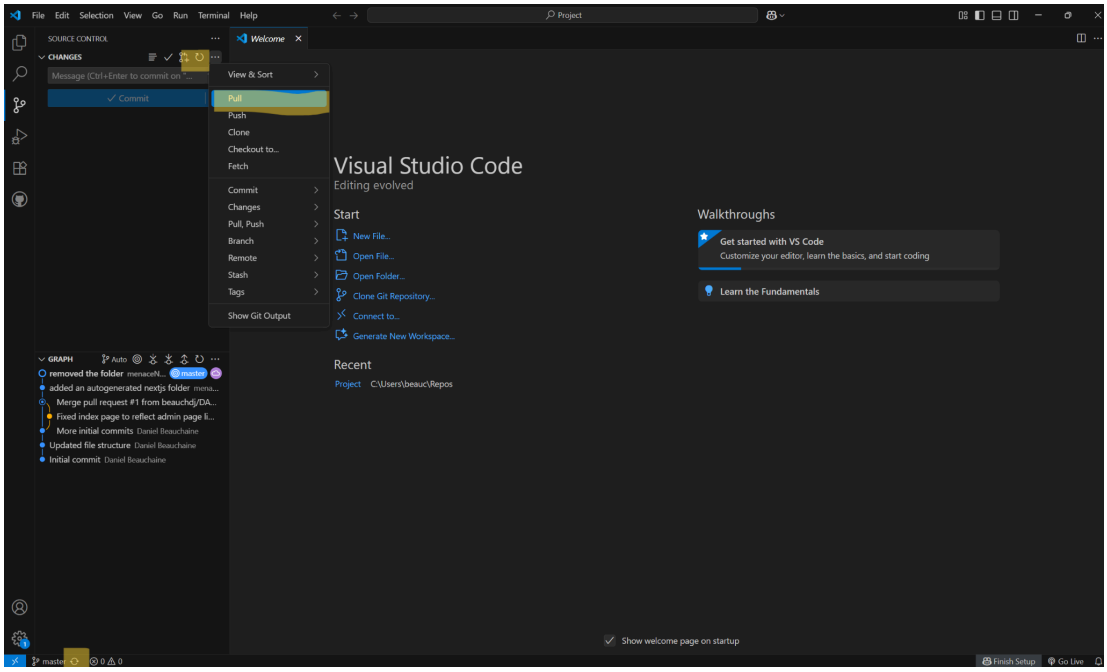
Git clone from github, and pick a repository for the project. I made a folder CS440 and put the project in there. If you are doing a git clone you should be starting fresh and you can skip these initial steps. If you are NOT starting fresh, the following steps will get YOUR local project sync'd with the master branch of the project. If you close whatever project you're working on, you are probably looking at a screen like this.



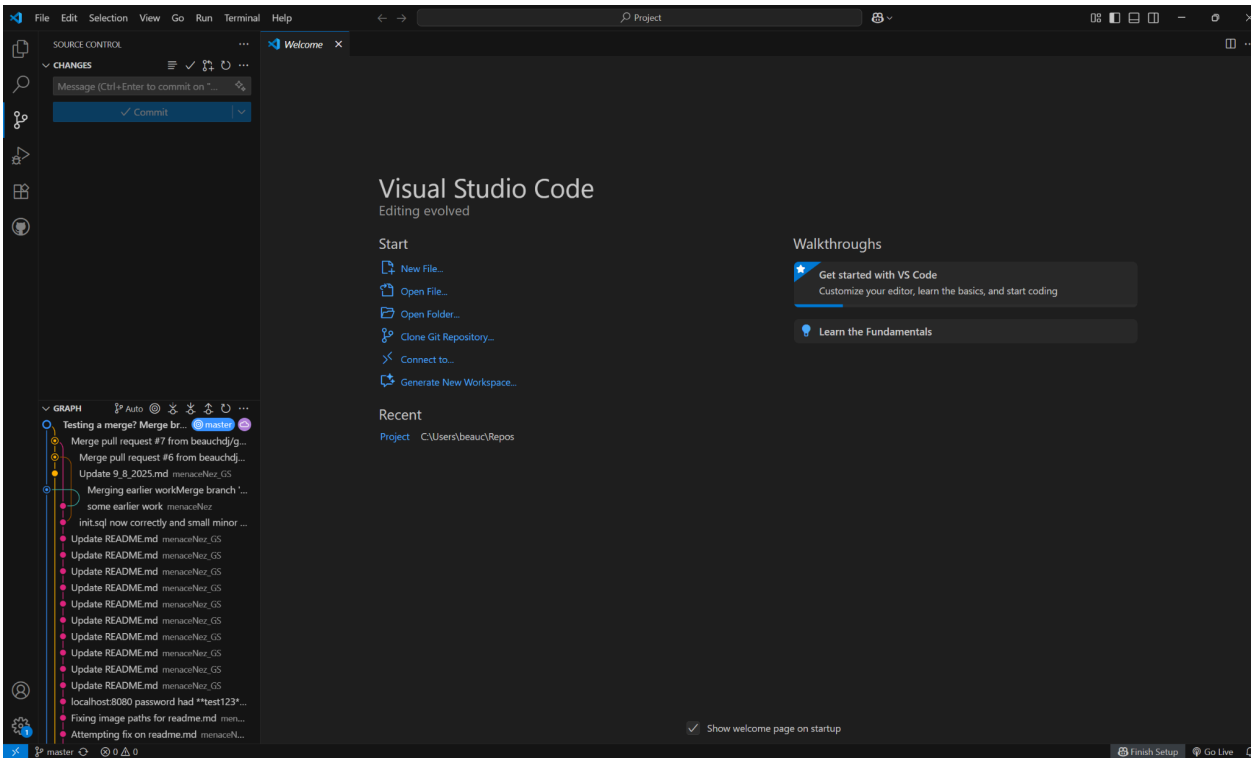
From here, if you look at the lower left corner and click, that will be your current branch, and it will show the other branches up top by the search bar. The one you want to select is origin/master from the “remote branches” section. The remote master branch is THE most up to date project branch with everyone’s changes. It’s the main project we want to keep updated and functional.



Now we want to synch the changes from whatever we have been working on to make sure we have the stuff from the master branch. In the version control tab you can pull, or in the lower left corner there's the refresh button, or in the version control tab there is the refresh button.

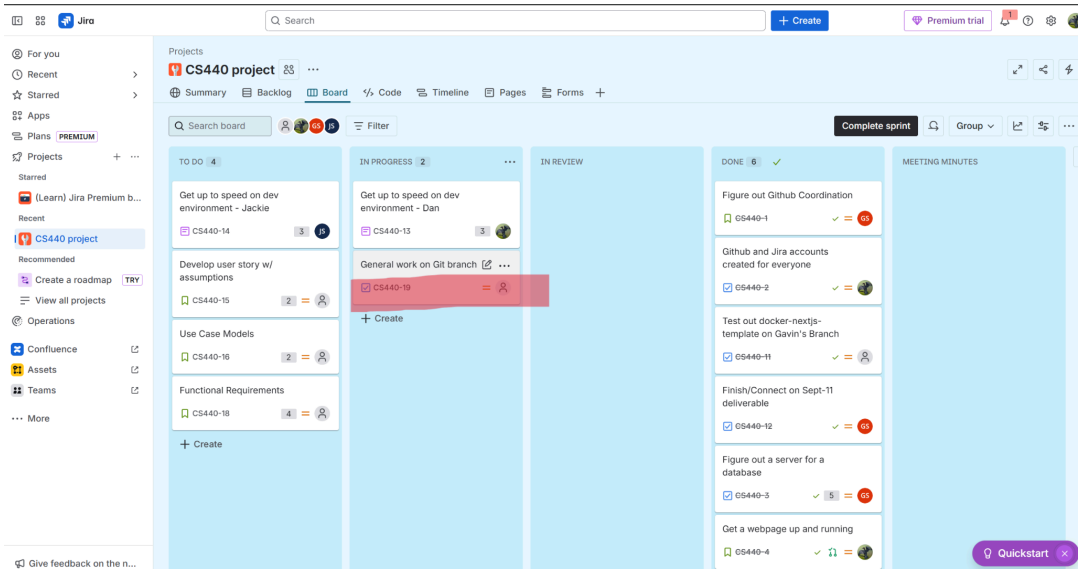


I haven't tried all combinations of this from working with different parts, but you get the idea, this should synch up the project.

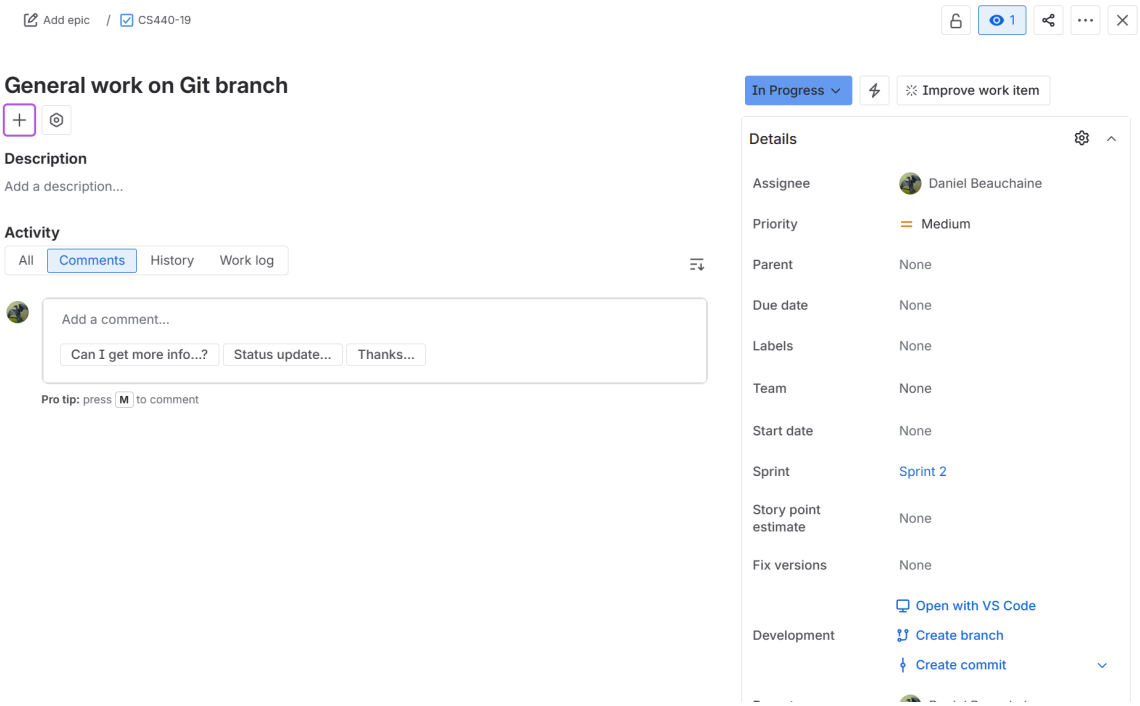


And you should see the master branch with changes updated in the lower left box.

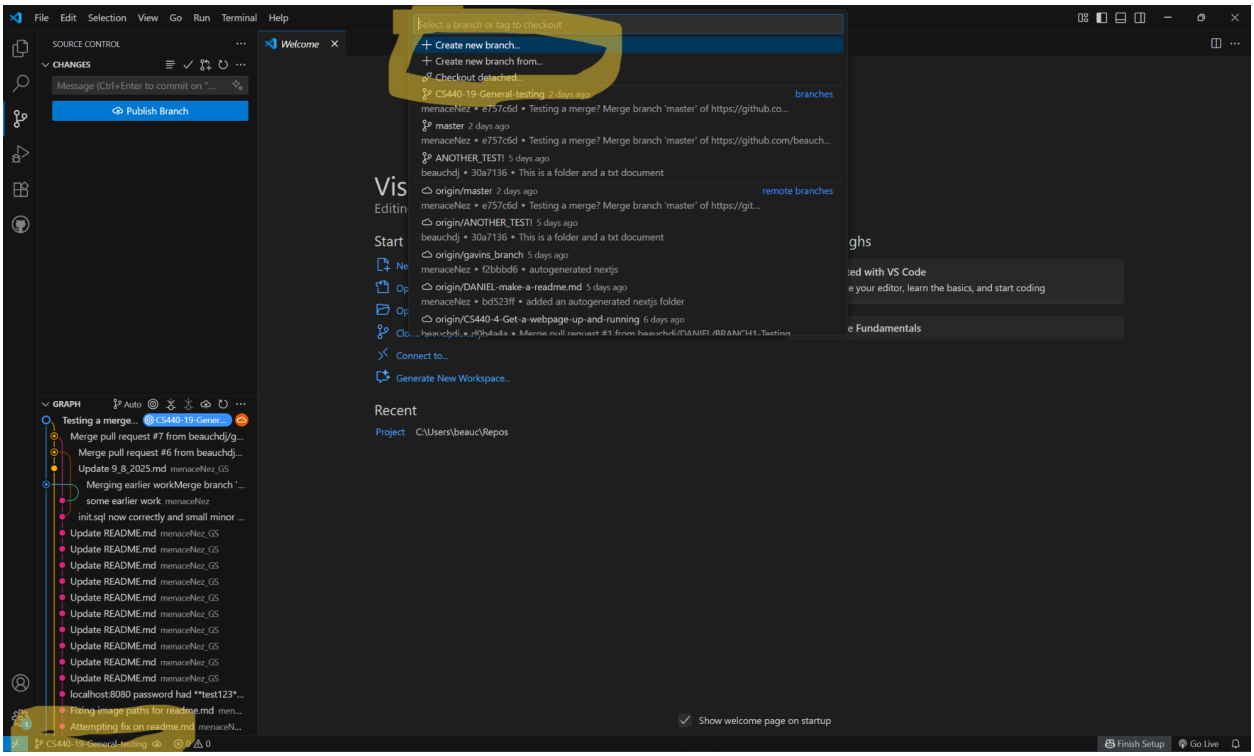
So I want to pick up a ticket from Jira and make a branch, how do I do that? Well thank you for asking that me, go to the Jira board and create/pick out a ticket you want to make a branch for. I made one and moved it to the “In Progress” tab already. Highlighted in red is what you should name your branch. It needs to START with CS440-XX (XX being the 19 in this example) in order to link to Jira, but you can name it CS440-XX-Work-on-xxxxxxx etc.



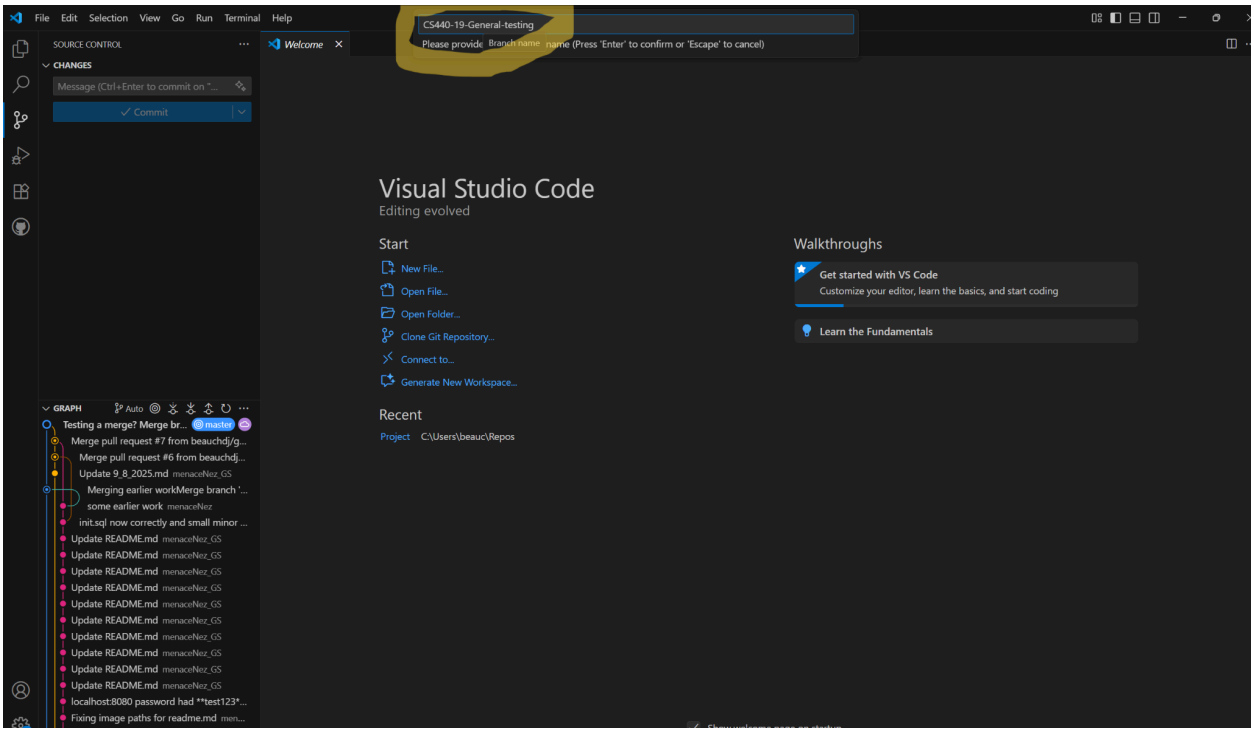
The ticket can be updated with description/etc but note how there is no Git branch associated with it…. yet.



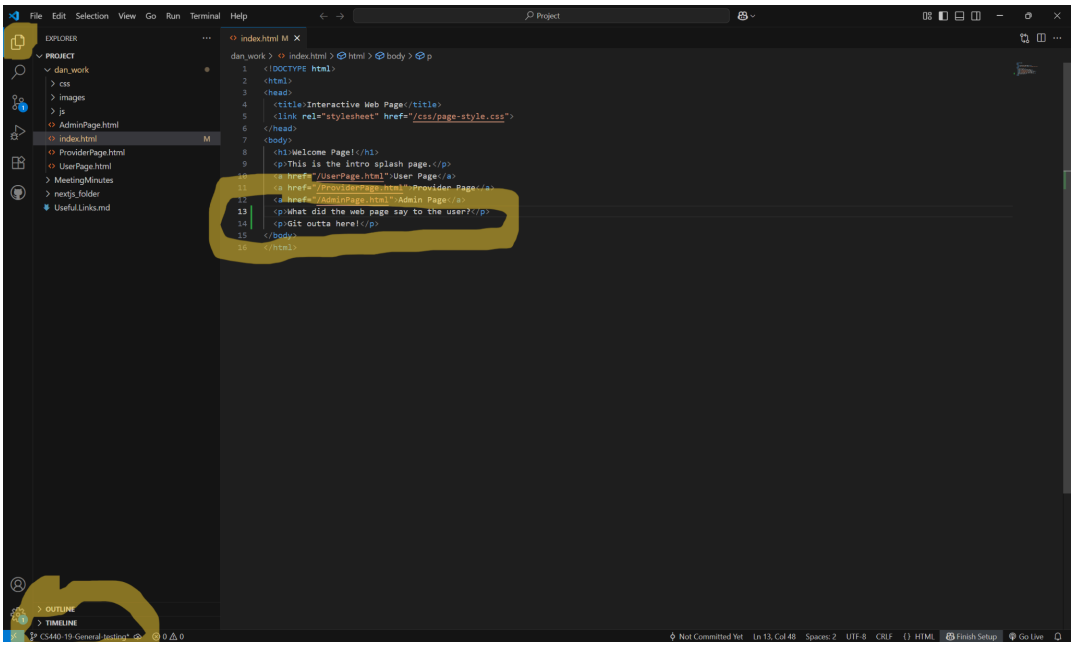
If you click on the lower left corner where it says “master” (this is just the name of the branch you’re currently holding) you will have some options.



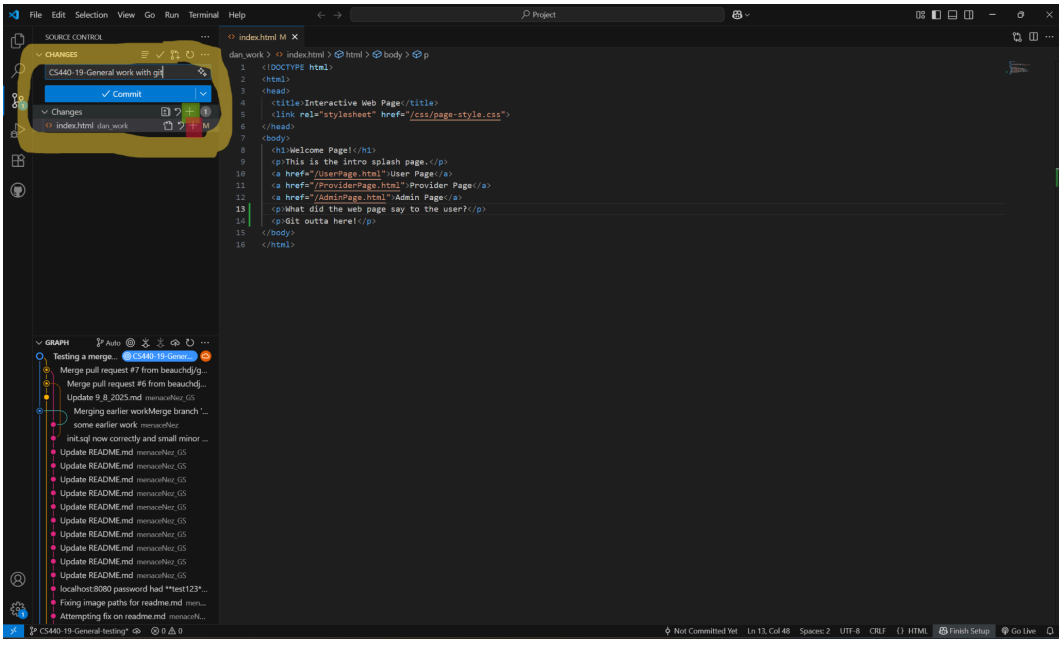
This is where you create a new branch, and name it according to the Jira ticket.



When you go to the files tab, and look at the lower left, you should notice you're working on your branch. When you change a file, in the text file at whatever line you change, it will show your updates with a green bar on the side of each line you change.

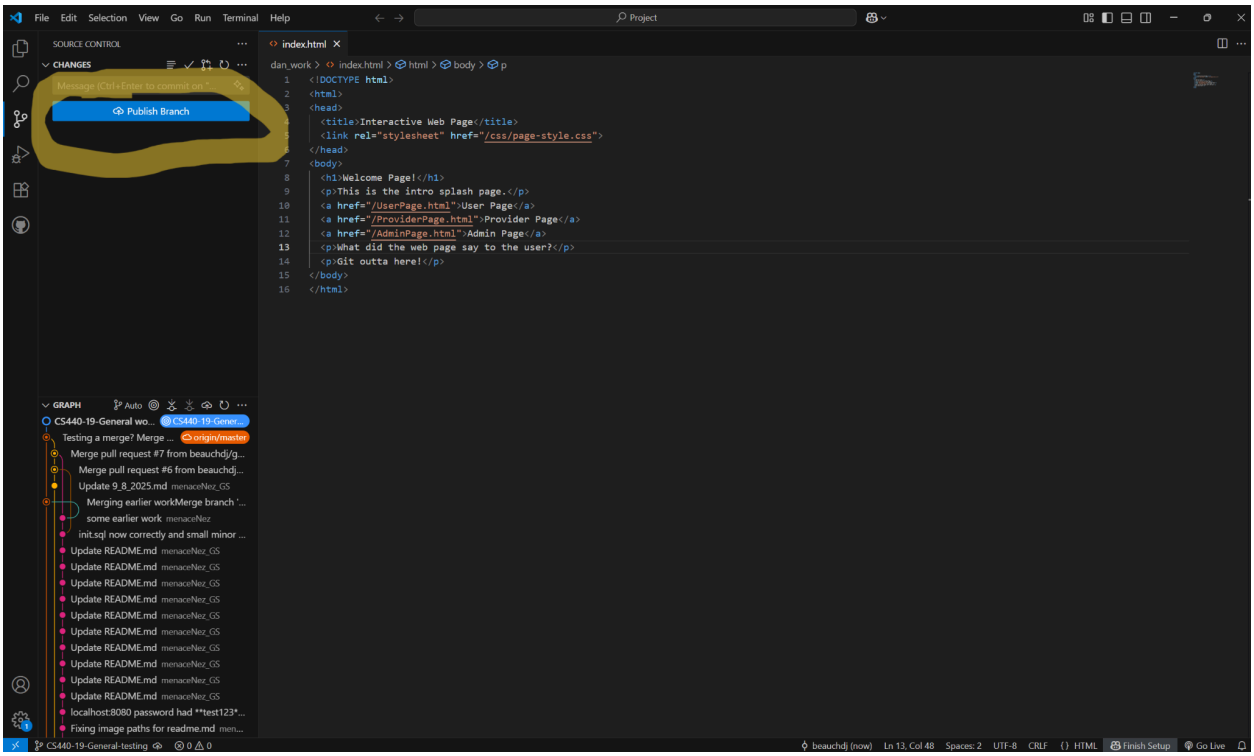


So once you're happy with the changes on your branch, or maybe just as a backup, you can commit, and publish your changes to the branch. This will take your "local" CS440-XX branch and make a remote copy, that you can work on whenever/wherever. THESE STEPS WILL NOT UPDATE ANYTHING ON THE MASTER BRANCH! So commit/publish whenever you want to make a backup or when you're ready to merge.

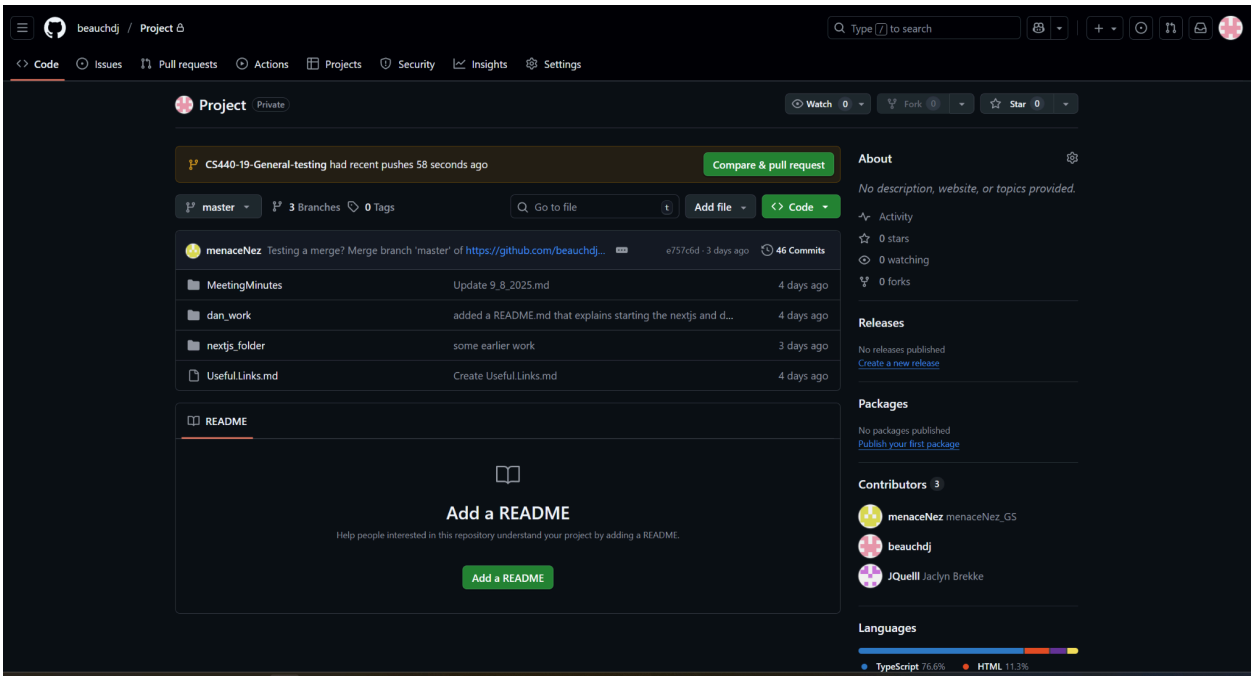


In the version control tab, (after all files in your projects are saved {ctrl+s} to local) it will show the changes. To "stage" the changes use the + highlighted in green, or for individual files use the + highlighted in red. When you click the file, it will show the file on the right side in the editing pane. If this is your first push, Name the commit "CS440-XX-(description)", in this example, the XX is 19 from the above Jira ticket. When ready, Commit!

Publish to push and update the remote branch. When you do this, on Github, you are updating the remote branch, and that's it. Any changes made will only apply to that branch. The "master/origin" branch will only be updated when you make a pull request. Also, I said above that the first push/publish should have a commit message with "CS440-XX-Description", but any following pushes can have whatever commit message you want. Like "fixed spelling error", "updated XXXXX" etc.

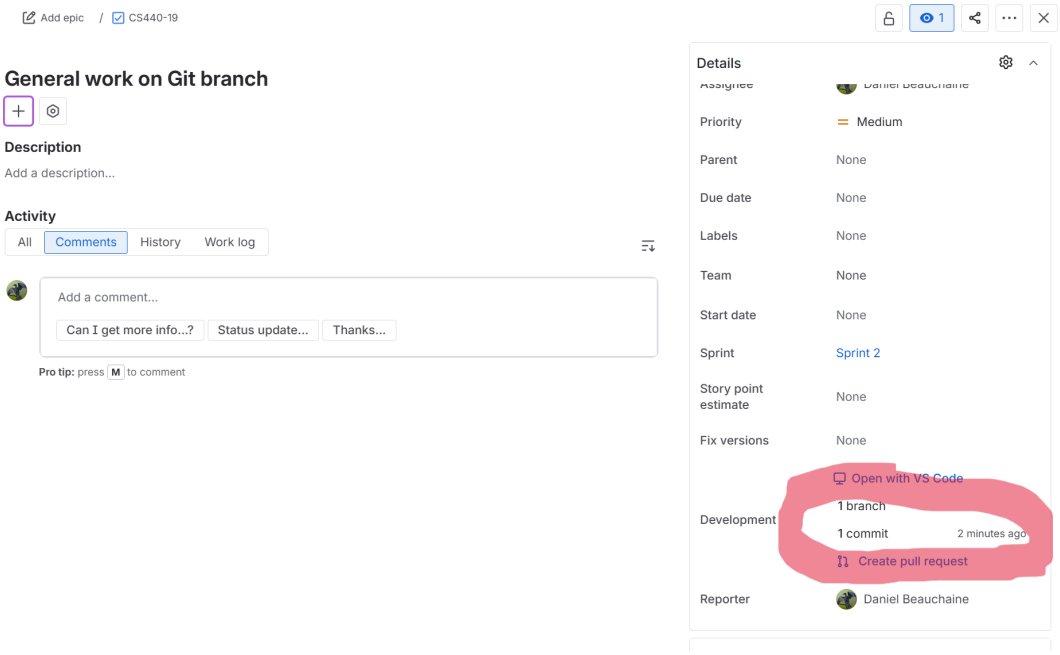


And in Github, this is how your branch page may look.

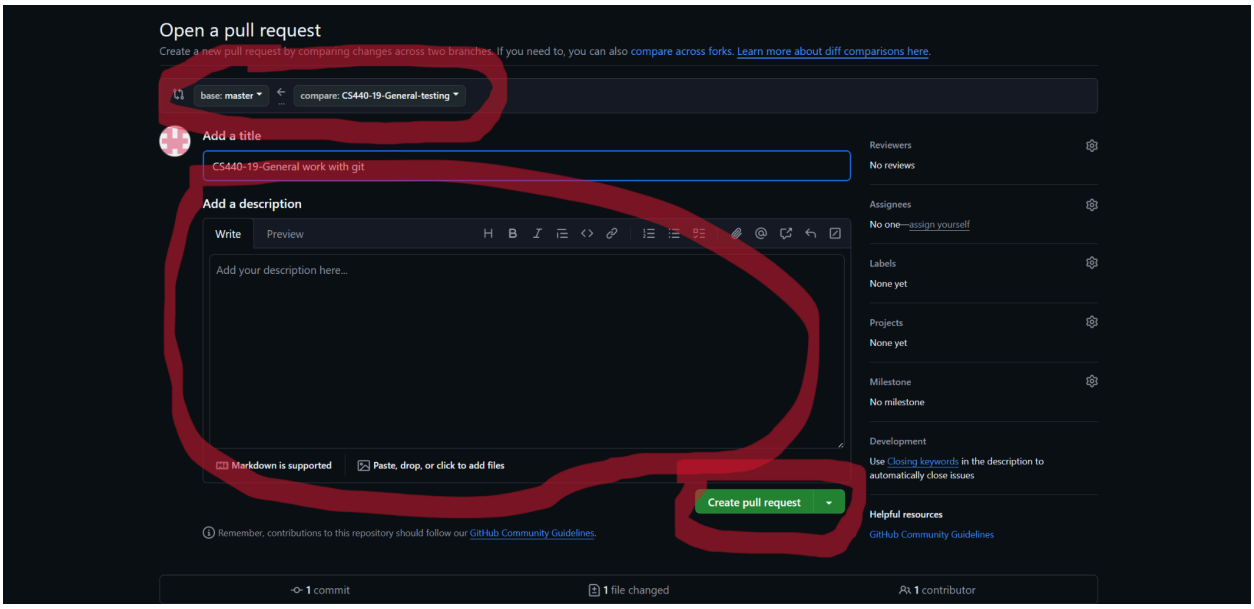


Before we do a proper merge, let's take a look at Jira!

When you make a push, Jira will see it (you may have to refresh the page after the git push). You can click the branch, or commit and it will take you to the github page, pretty nifty.

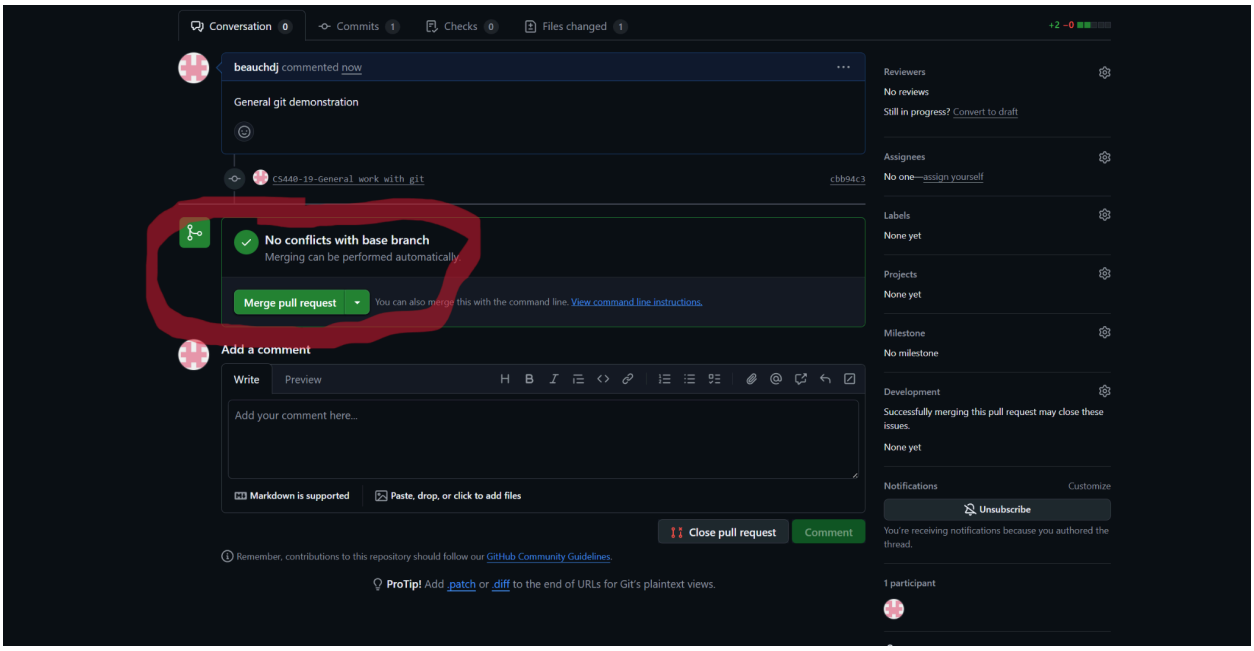


Now onto the pull request (PR). If you got to youre branch and create pull request, this is the point where you're trying to merge your branch into "master".

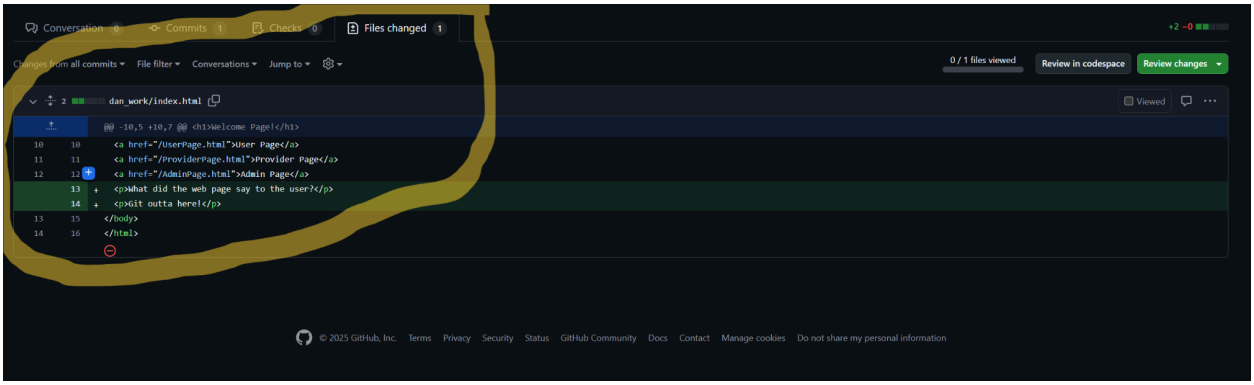


The top highlighted area is the branch interaction which is happening. So the idea is to merge the "base" as the "master", and then the arrow shows that the selected branch (your branch) is getting merged in. The middle is the description section, just highlight the changes/functionality that were made, and the "Create pull request" button will make the request to merge into master. If you scroll lower, it should show the files changed and the changes made.

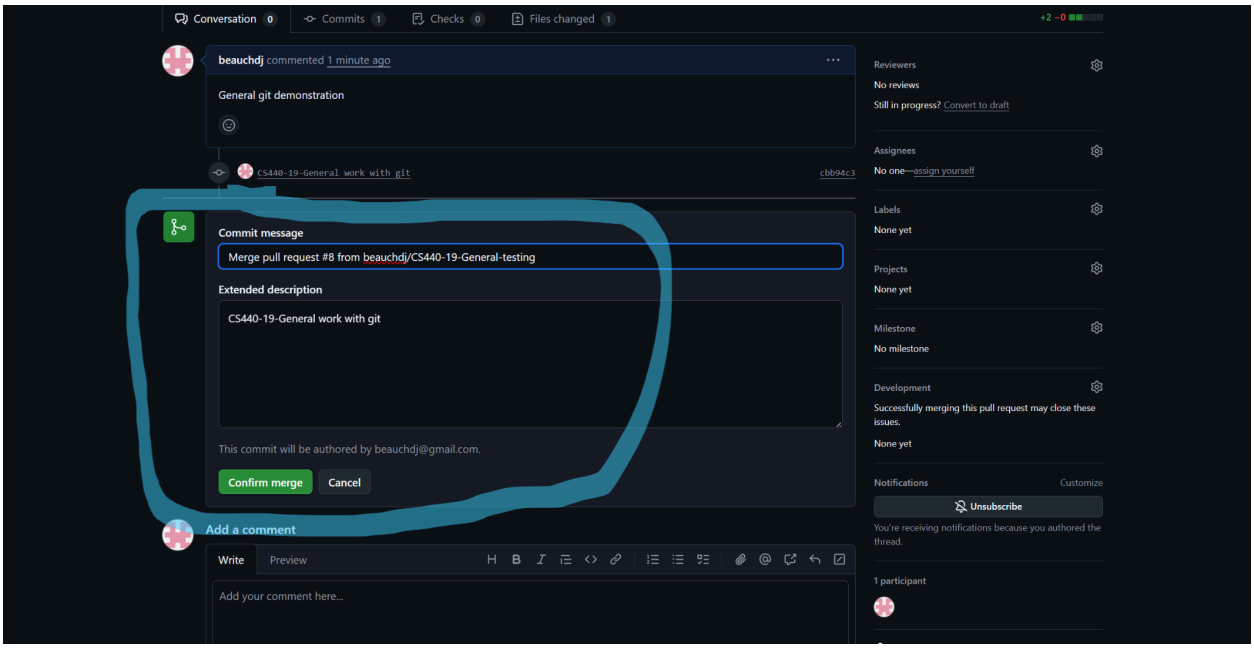
Peer review! Or self review. This pull request will trigger a PR action, and it will come up with any collisions/inconsistencies or just saw it's okay to merge.



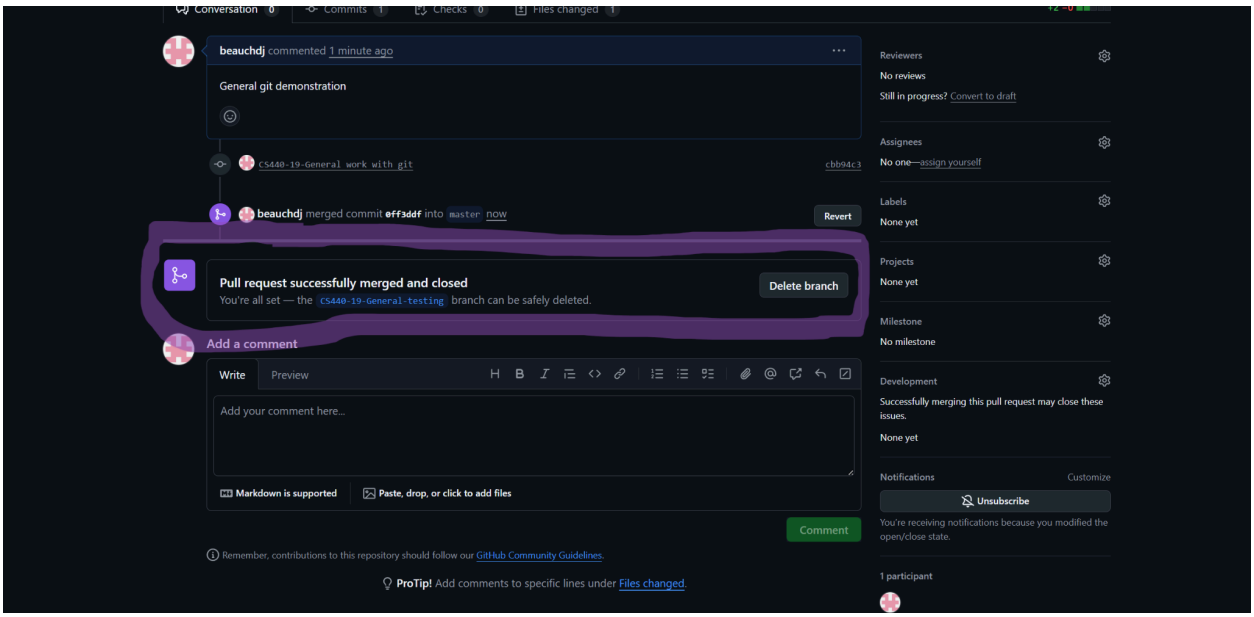
If you go to the files changed tab you can see the files changed.



If you're good with the changes, confirm merge.



And ALWAYS delete branch, so there's not a bunch of antiquated branches hanging out. When a ticket is complete, you're done.



It might take a bit for VS Code to refresh the changes that your branch is gone, but after your branch is done, you should repeat the steps to pull from remote->"origin/master", and you're ready to make branches and stuff again.