How to Submit to GitHub using Pull Requests (PRs)

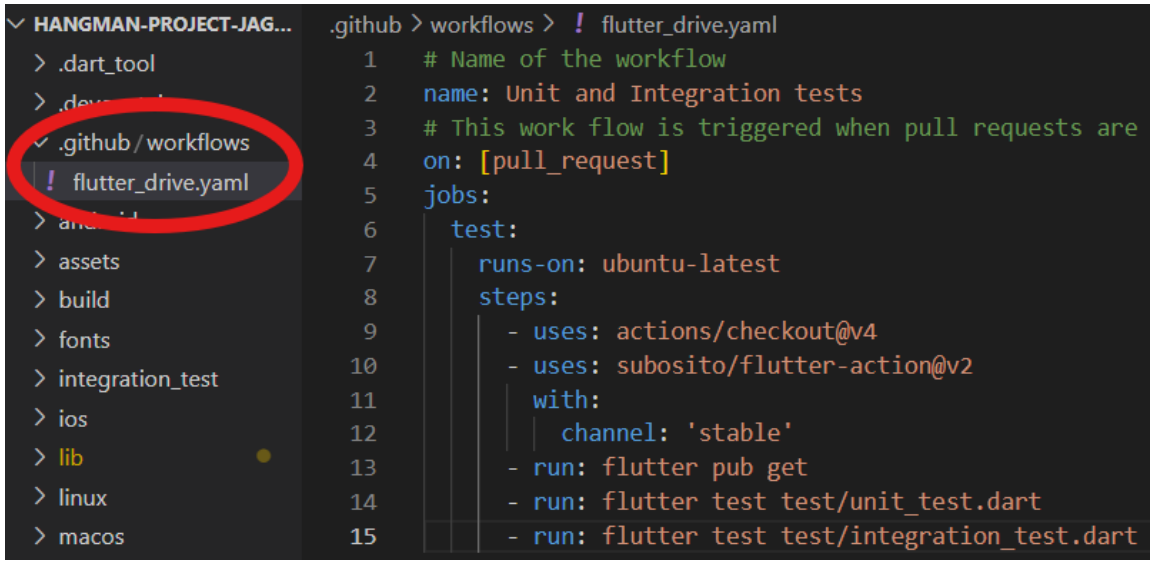
CSCI 3341 SOFTWARE ENGINEERING II

How to Submit to GitHub using PRs

1. Configuration of Code (CoC)
2. Create a new branch
3. Create a PR
4. Undo a PR (if necessary)

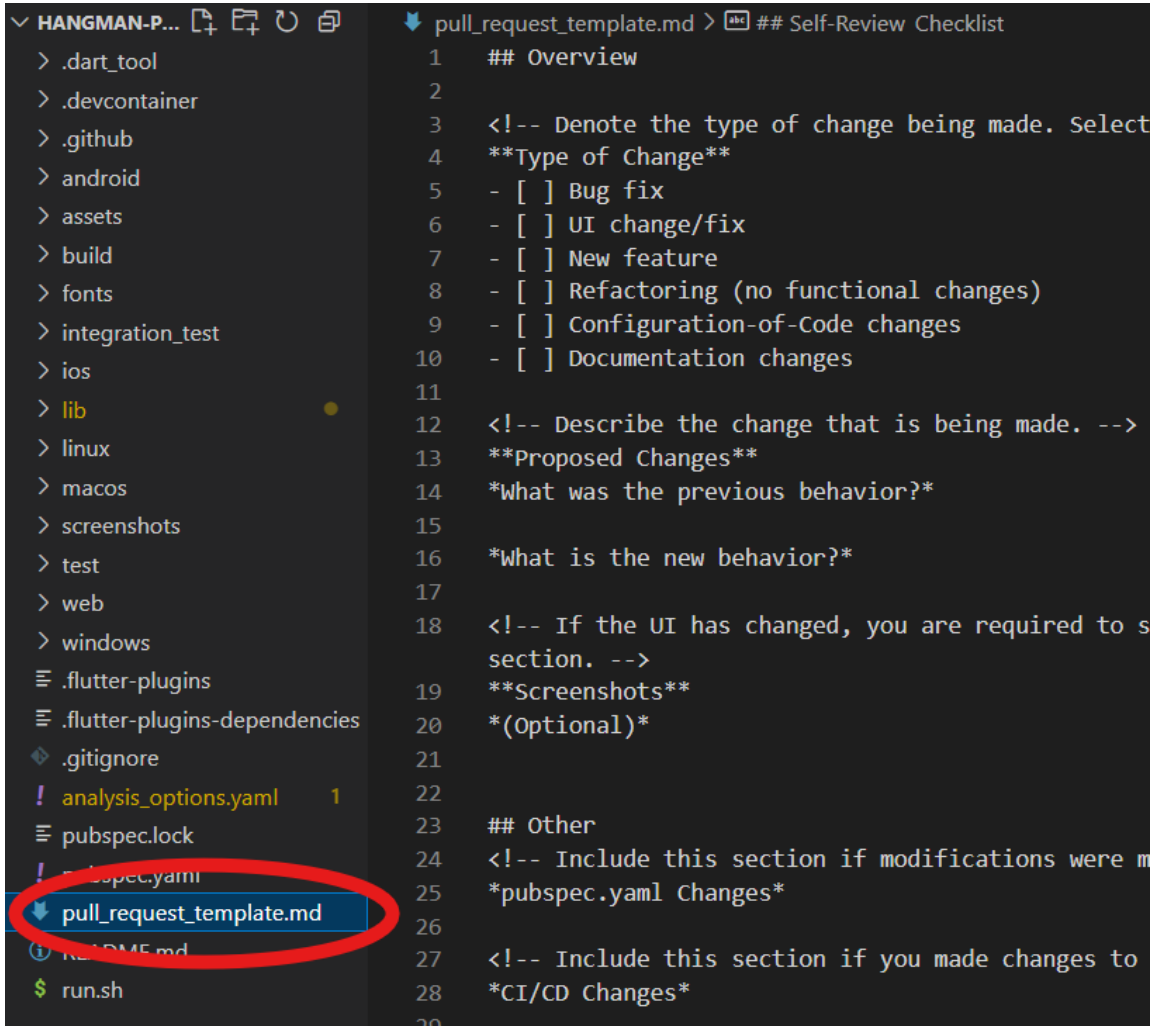
Configuration of Code (CoC)

1. Create a YAML file



Most of your project assignments will already have a YAML file. However, if you're starting from scratch, you can follow these steps.

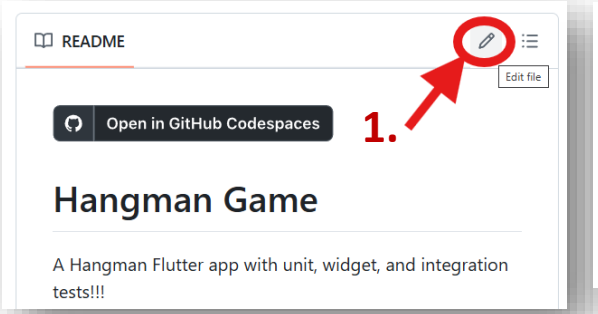
1. Create a folder/directory **.github**
2. Create a folder/directory **workflows** inside **.github**
3. Create a file **tests.yaml**
4. Add the following code inside **tests.yaml**

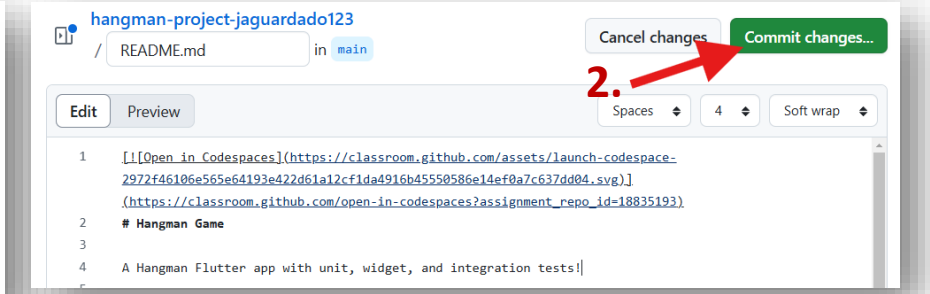


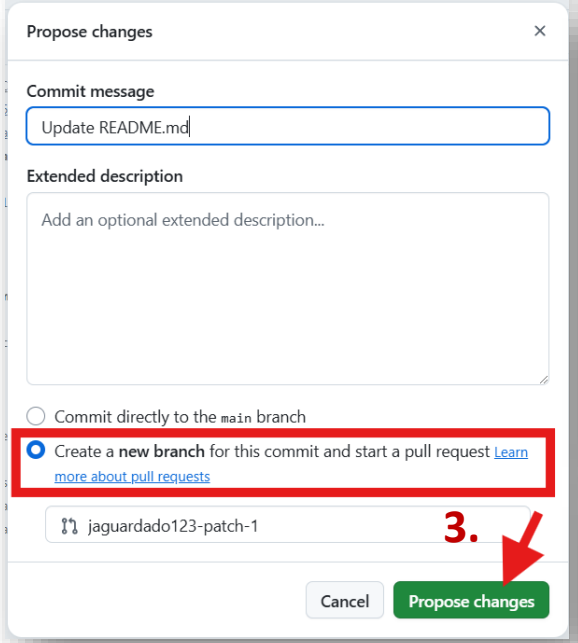
2. Create a pull request (PR) template

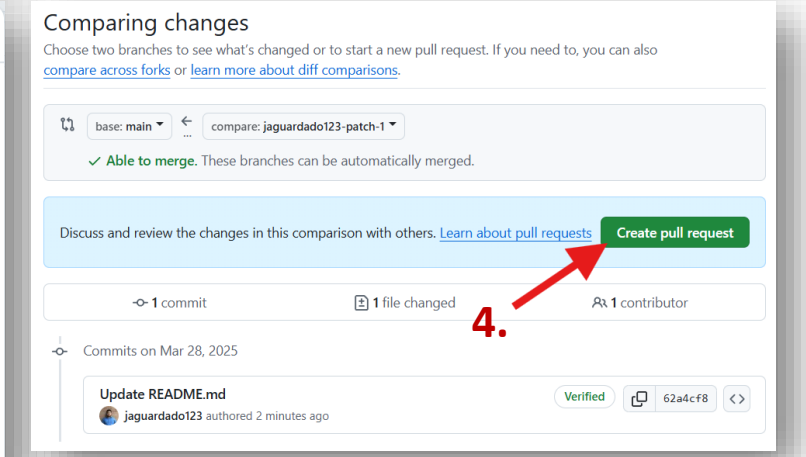
Most of your project assignments will also already have a PR template. However, if a PR template is missing, you can follow these steps.

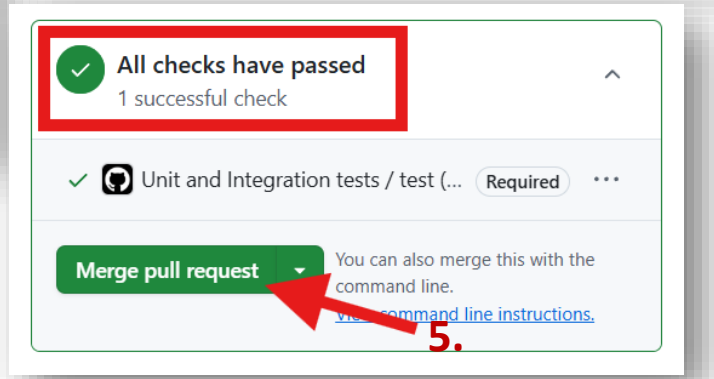
1. Create a file **pull_request_template.md** in the project's root directory.
2. Add the following code inside your **pull_request_template.md**.









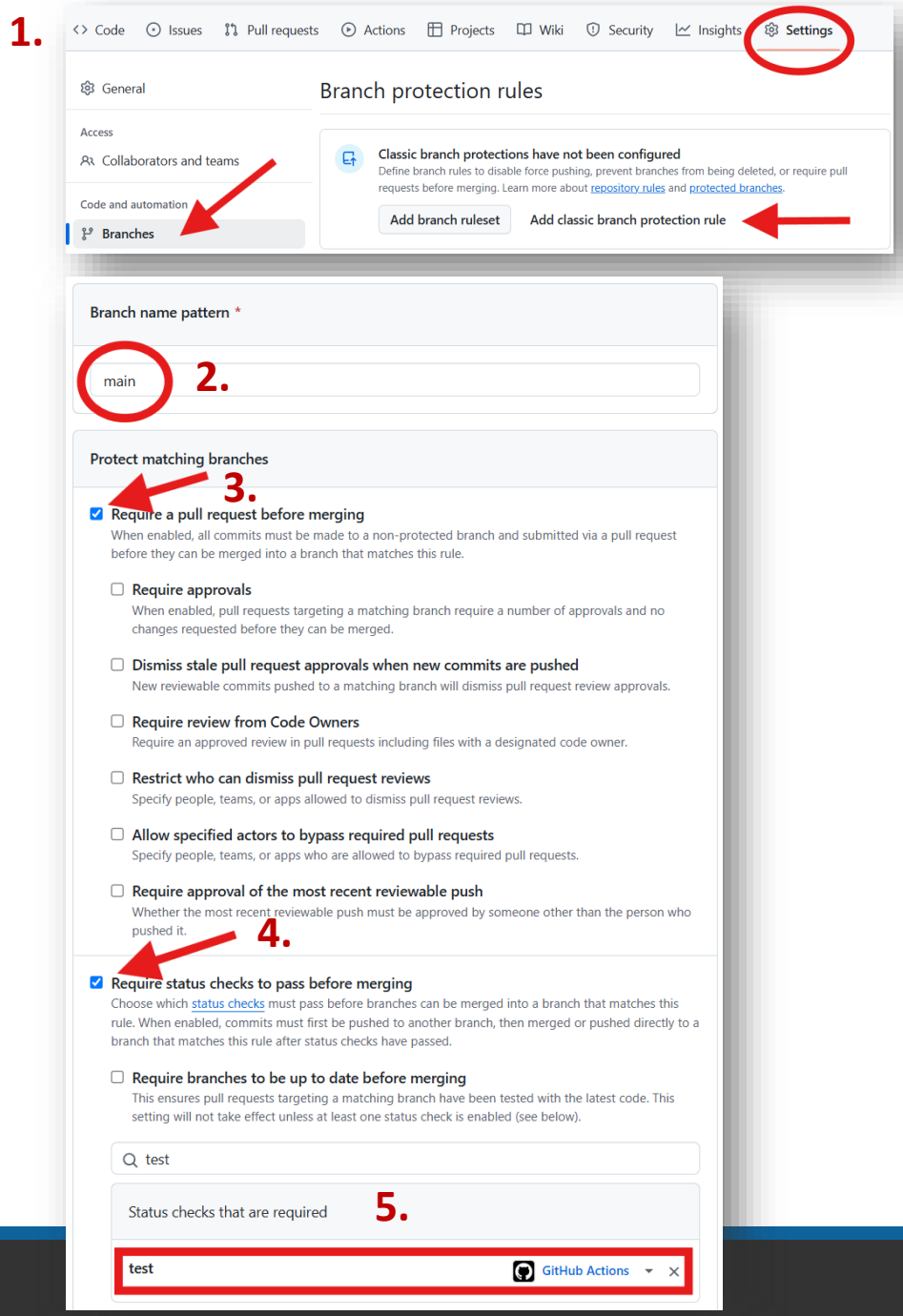


3. Make GitHub recognize your tests

Next, you want to configure GitHub to only allow PRs to the main/master branch. But first, you need to make a PR for GitHub to recognize your tests.

Do a simple README change to create a PR. Go to your project's repo and follow these steps.

1. **Click on the edit button** on the README.
2. **Add an extra decimal somewhere** in your README and **click Commit changes…**.
3. Next, **select the "Create a new branch"** option and **click Propose changes**.
4. **Click Create pull request** (*do this twice*).
5. Wait for the tests to run and pass (*may take a minute*), then **click Merge pull request**.



4. Create a Branch protection rule

Now, configure GitHub to only allow changes to the main/master branch through a PR. To do this you need to setup a Branch protection rule.

1. Go to **Settings**, select the **Branches** tab, and click on **Add classic branch protection rule**.
2. For the **Branch name pattern**, give the branch rule the name of branch you want to protect.
3. Select **Require a pull request before merging** and <u>unselect</u> **Require approvals**.
4. Select **Require status checks before merging**.
5. Look for the test we ran in the previous slide and select it.
6. Lastly, click on **Create** to create the branch rule.

Create a new branch

1. Create, Checkout, & Publish a new branch to work in

Open a new terminal and create a new branch with a name relative to its purpose. For example, “conf_setup”, “pass_unit_tests” or “ui_refactoring”.

- **git branch pass_unit_tests**

Next, checkout the new branch.

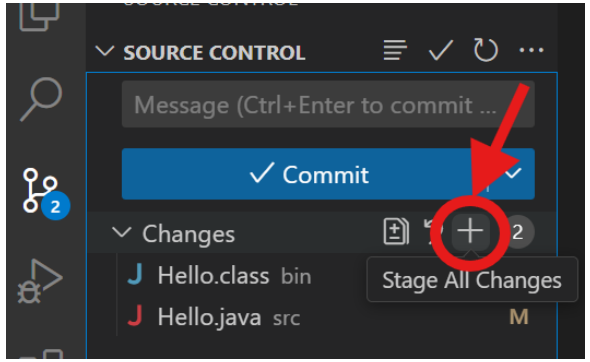
- **git checkout pass_unit_tests**

Lastly, publish the branch to GitHub since it currently only exists in our local repo.

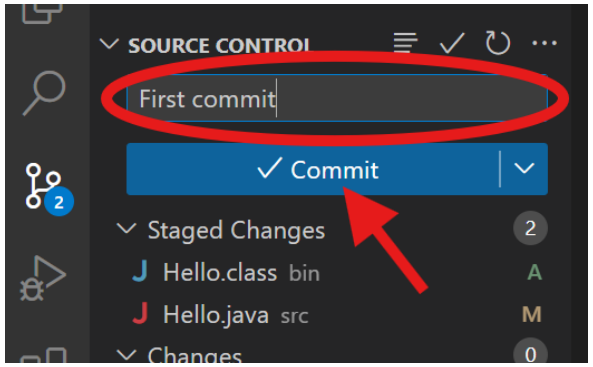
- **git push –u origin pass_unit_tests**

```
root@codespaces-8c437e:/workspaces/hangman-project-jaguardado123# git branch pass_unit_tests
root@codespaces-8c437e:/workspaces/hangman-project-jaguardado123# git checkout pass_unit_tests
Switched to branch 'pass_unit_tests'
root@codespaces-8c437e:/workspaces/hangman-project-jaguardado123# git push -u origin pass_unit_tests
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
```

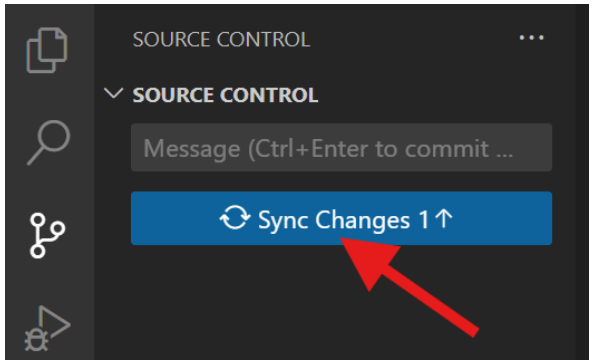
1.



2.



3.



2. Push your changes (Using VS Code)

Once you are in your new branch you can start editing/adding files in your project. Once you are confident in your changes you should stage, commit & push them.

1. **Click on the + button** next to Changes to stage your changes.
2. **Add a message in the text box** and **click on the Commit button**. The commit will save the new state of your code.
3. To push your changes to GitHub, **click on the Sync Changes button**.

2. Push your changes (Using terminal)

An alternative to using Visual Studio Code's UI is to use the terminal to push your changes to GitHub. This commands can be useful for cases where you don't have access to a UI like VS Code or GitHub Desktop.

Stage your changes.

- **git add .**

Commit your changes with a message.

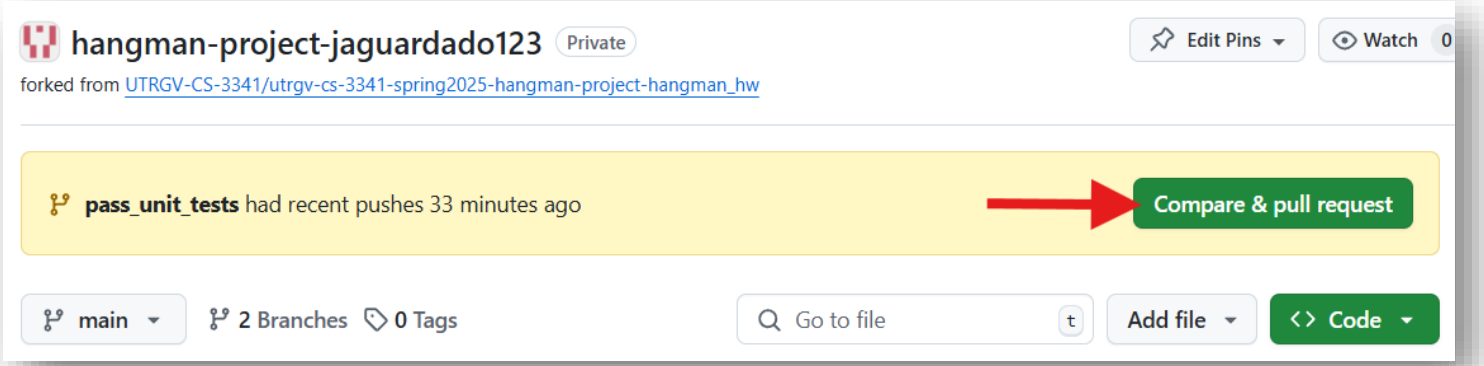
- **git commit –m "Passed some tests"**

Push your changes to GitHub.

- **git push**

```
root@codespaces-8c437e:/workspaces/hangman-project-jaguardado123# git add .
root@codespaces-8c437e:/workspaces/hangman-project-jaguardado123# git commit -m "Passed some tests"
[pass_unit_tests a9fed50] Passed some tests
 1 file changed, 1 insertion(+), 1 deletion(-)
root@codespaces-8c437e:/workspaces/hangman-project-jaguardado123# git push
Enumerating objects: 9, done.
```

Create a pull request (PR)



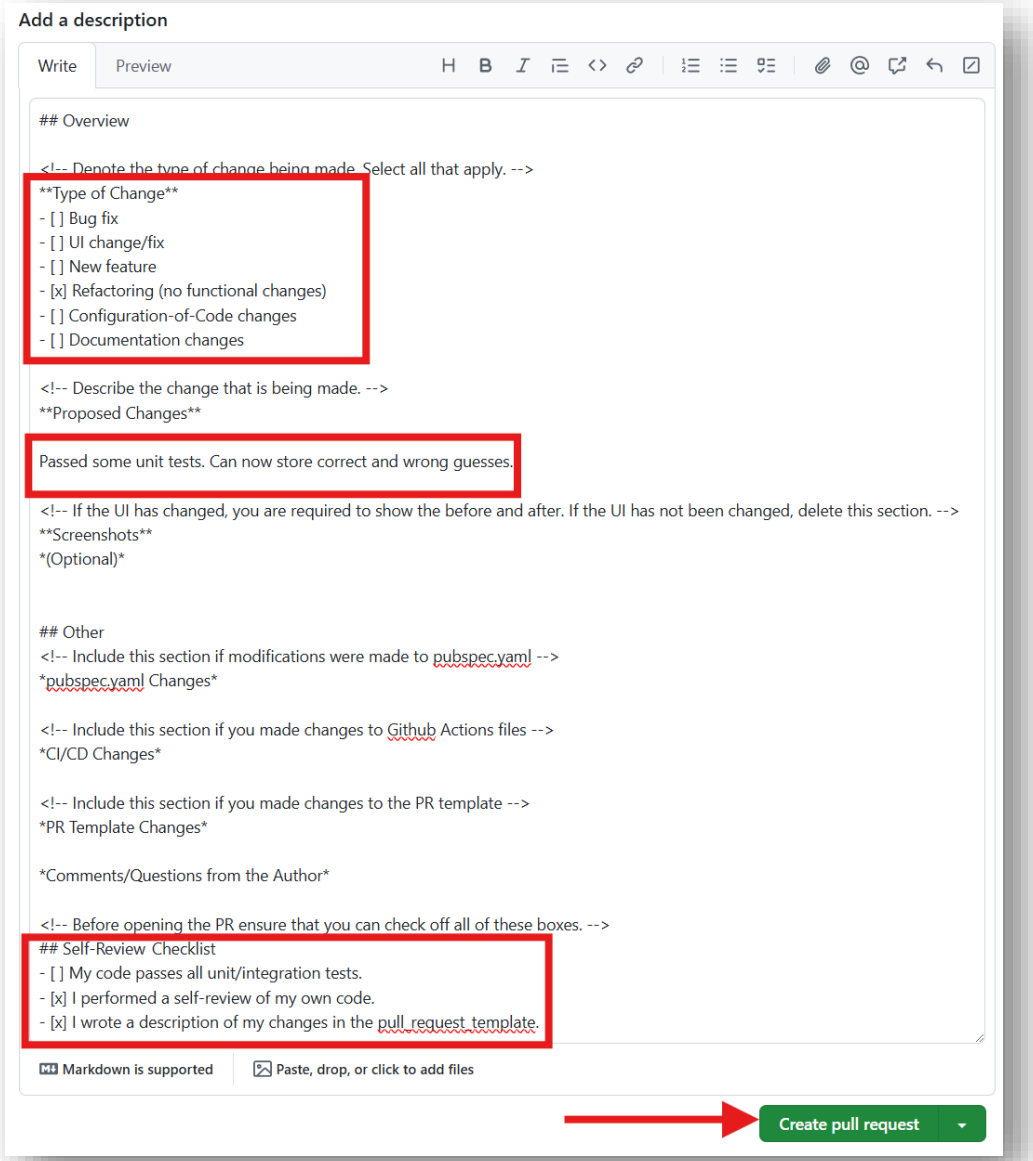
1. Create a PR

Now that you've committed & pushed your changes it's time to merge them with main through a PR.

Go to the project's GitHub repository.

The first thing you should see is a message to create a PR.

Click on the Compare & pull requests button.



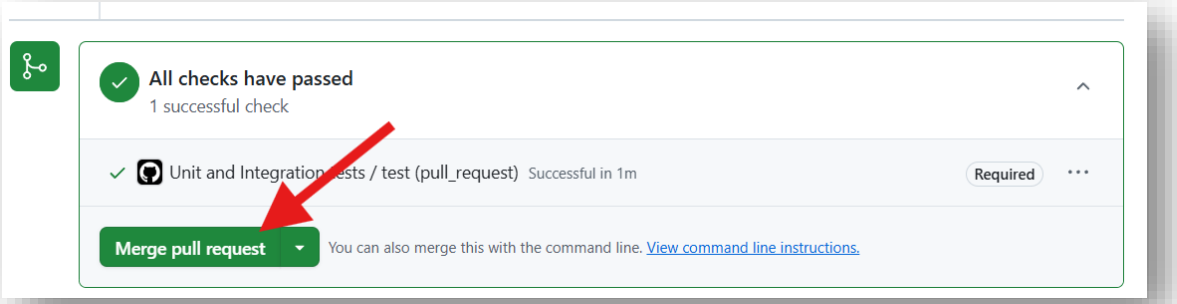
2. Fill out your PR template

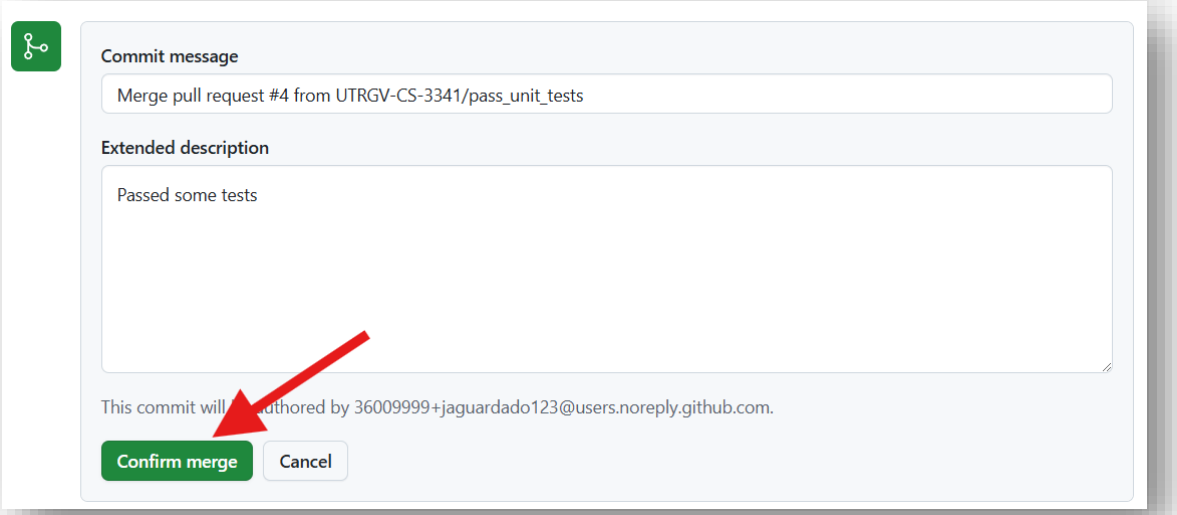
Next, **fill out your PR template**. Select the type of change you made and include a short description of the change.

Make sure to review the changes in your commit and check them off your self-review checklist.

Once you have reviewed your code & filled out the PR template, go ahead and **click on the Create pull request button**.

3. Verify and merge your commit

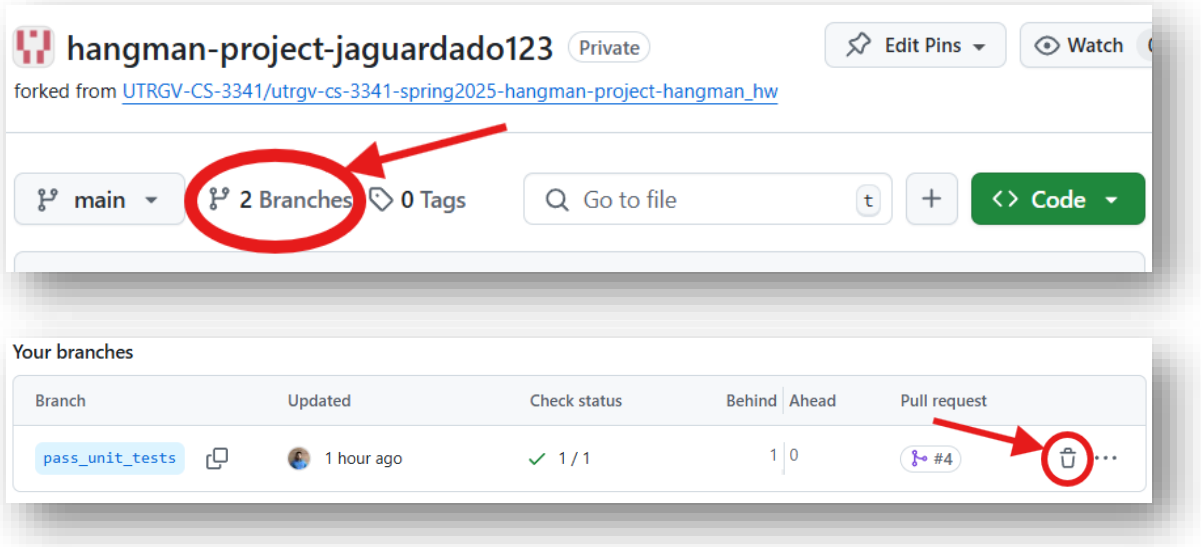




After creating a pull requests, your tests will begin to run. Thanks to the Branch protection rule, GitHub won't let you merge your changes until all tests pass.

Once all your tests pass **click on Merge pull request** and then **click on Confirm merge**.

Repeat steps 1-3 until you have completed your sprint or feature.



4. Delete branch (only when finished)

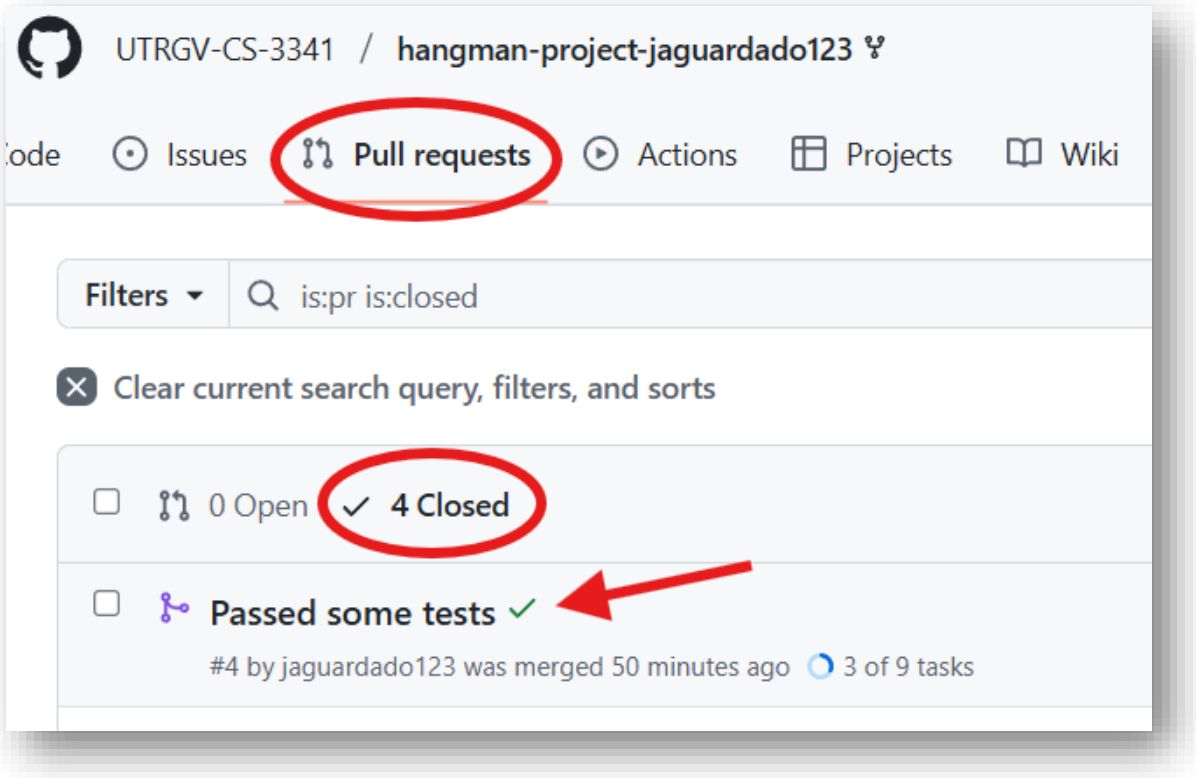
Once you are finished adding a feature or completing a sprint (*ex: passing all unit tests*), you are ready to create a new branch to move on to the next feature/sprint.

Once you have created a new branch, you should delete your old branch.

Go to your project's repo and **click Branches**.

Next, find your branch and **click the trash icon to delete it**.

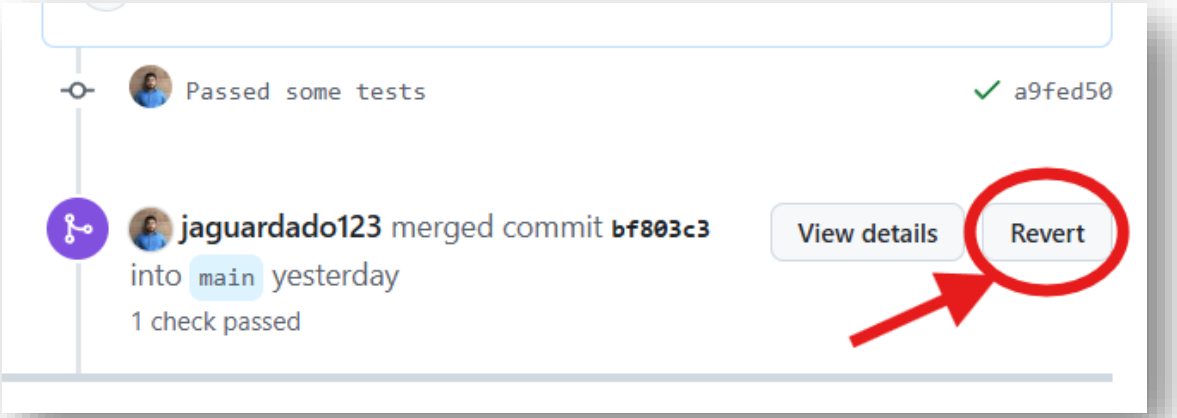
Undo a PR (if necessary)

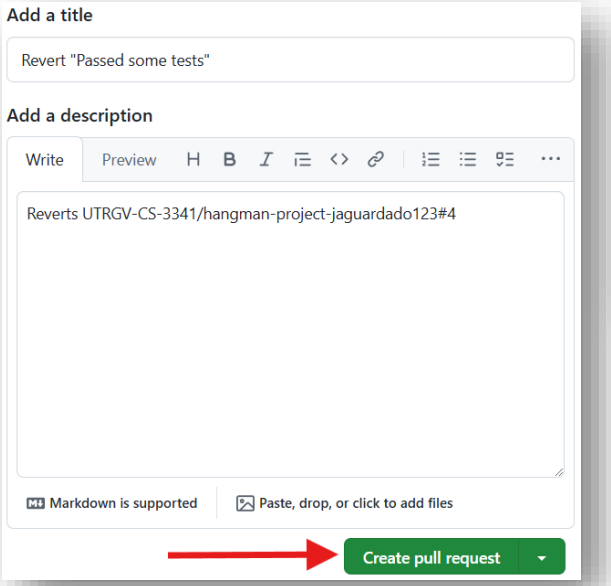


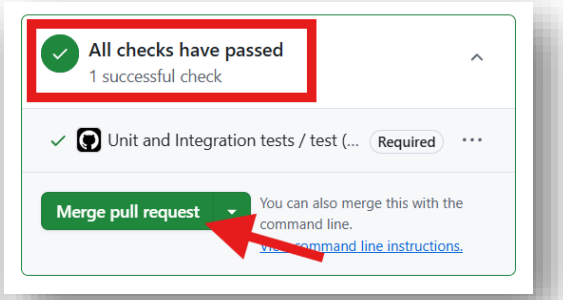
1. Got to GitHub

We all make mistakes, so it's safe to assume at some point we will make a PR that shouldn't have been made. When that happens, undoing a PR is very simple.

Got to GitHub, **select Pull requests**, then **select Closed** to view past PRs. Look for the PR you wish to undo and select it.







2. Undo a PR through a new PR

Once you have selected the undesired PR you wish to undo, **click on the Revert button**.

This will create a new PR to undo the undesired PR, **click on Create pull request**, wait for all the tests to pass and **click on Merge pull request**. This will undo the changes from the undesired PR.