

# Introduction

Date: Jan 31<sup>st</sup>, 2026

Author: Mostafa Allahmoradi

The following work has been done to set up the CI-CD pipeline and make sure the environment is ready for the development phase, and it is error-free:

- Set up “.git” file by running the following command:  
git init
- Created “.gitignore” file for QueueIQ project containing Python and Jupyter Notebook folders.
- Created three branches named “dev”, “staging”, and “main”.
  - dev: Used for development
  - staging: Used for User Acceptance and QA testing before releasing a new version
  - main: Used for releasing new versions of our app that passed User Acceptance and QA testing
  - More branches can be created for each feature but they need to be removed after they served their purposed and merged with the “dev” branch
- Set up branch protection rules for main and staging branches.
- Created and verified the “queueiq-ci.yml” file under the “.github\workflows” folder to finish off the CI-CD pipeline
- After the CI-CD pipeline is set up, I had some issues with having Jupiter Notebooks in the file structure; the test kept failing. I modified the “queueiq-ci.yml” file to fix this bug
- The CI-CD pipeline is all done and ready for the development phase 😊

## Before using the CI-CD pipeline

Make sure your local environment is set up and activated. Here are the commands needed to create and activate:

```
python -m venv .envv  
.envv\Scripts\Activate
```

## Working with the “dev” branch

1. Switch to Dev locally and pull the latest changes:  
git checkout dev  
git pull origin dev
2. Merge the feature branch to the “dev” branch:  
git merge feature-branch
3. Push the changes to the remote branch:  
git push origin dev

## Working with the “staging” branch

### **IMPORTANT:**

- Please do not push any changes into the “staging” without creating a pull request to merge the “dev” branch into the “staging”.
- Once the pull request is created, make sure the CI-CD test passes. Then one or more teammates should review the pull request and approve it before merging.

## Working with the “main” branch

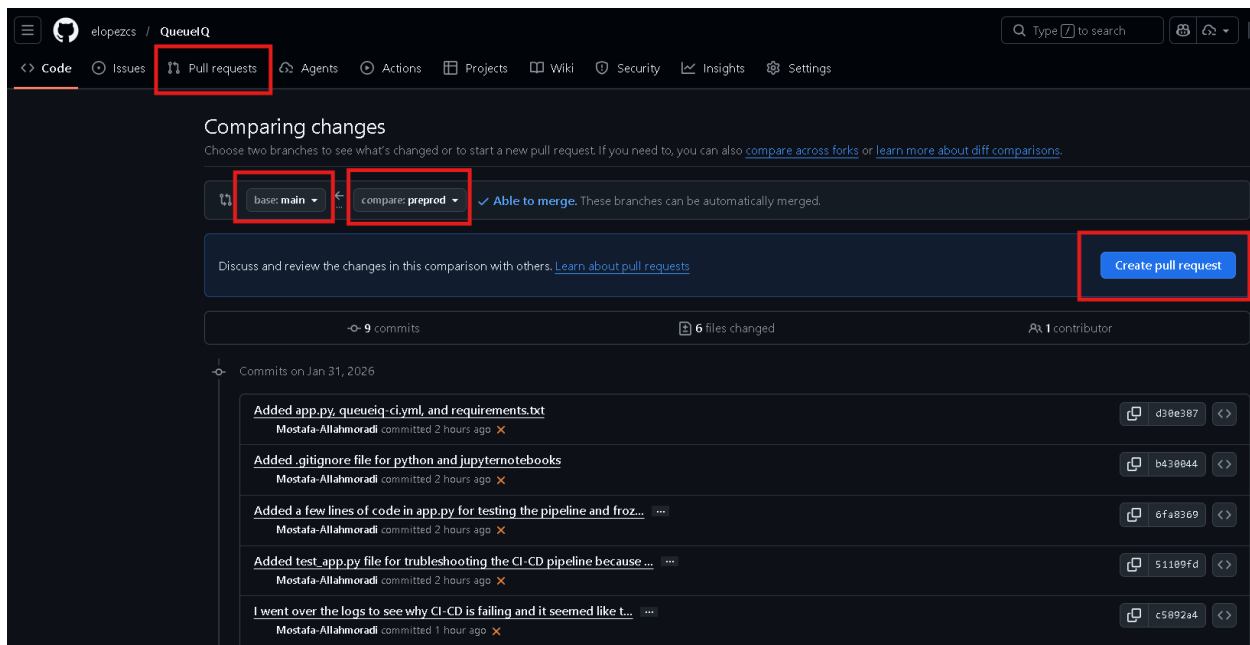
### **IMPORTANT:**

- Please do not push any changes into the “main” branch without creating a pull request to merge the “staging” branch into the “main”.
- Once the pull request is created, make sure the CI-CD test passes. Then one or more teammates should review the pull request and approve it before merging.

## Create a “Pull Request” to merge with the “main” branch

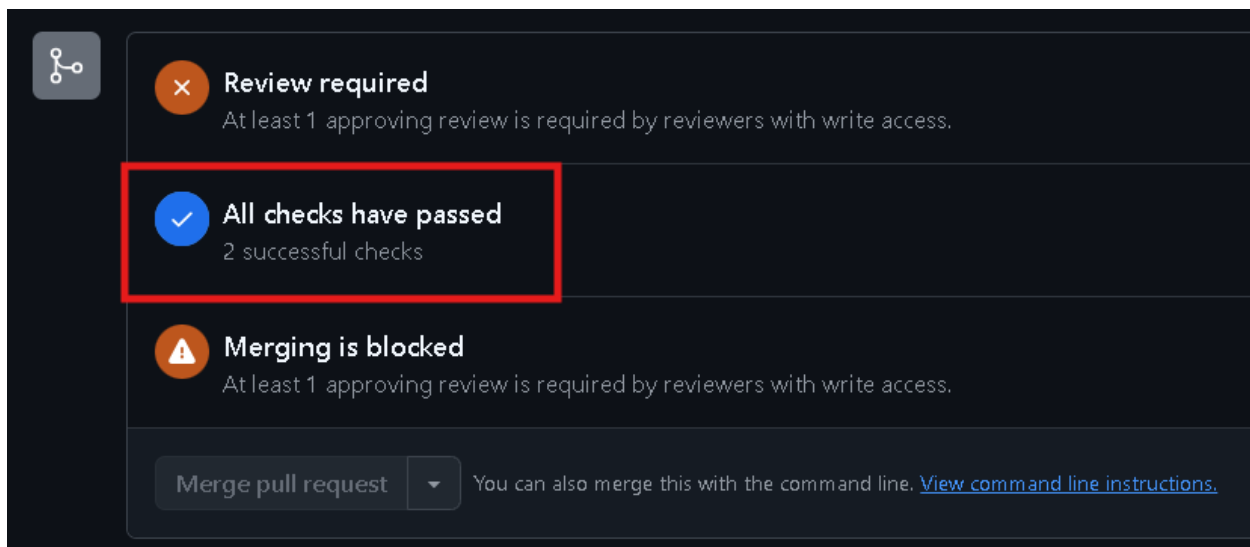
**IMPORTANT:** Before creating a pull request for the “main” branch, you have to merge the code to the “staging” branch and make sure the CI-CD pipeline passes the test. Also, the QA team should approve the feature's functionality before it's merged into the “main” branch.

1. Click on the “Pull Request” tab
2. Set “base” to be “main” branch
3. Set “compare” to be the “staging” branch
4. Click on “Create pull request”.

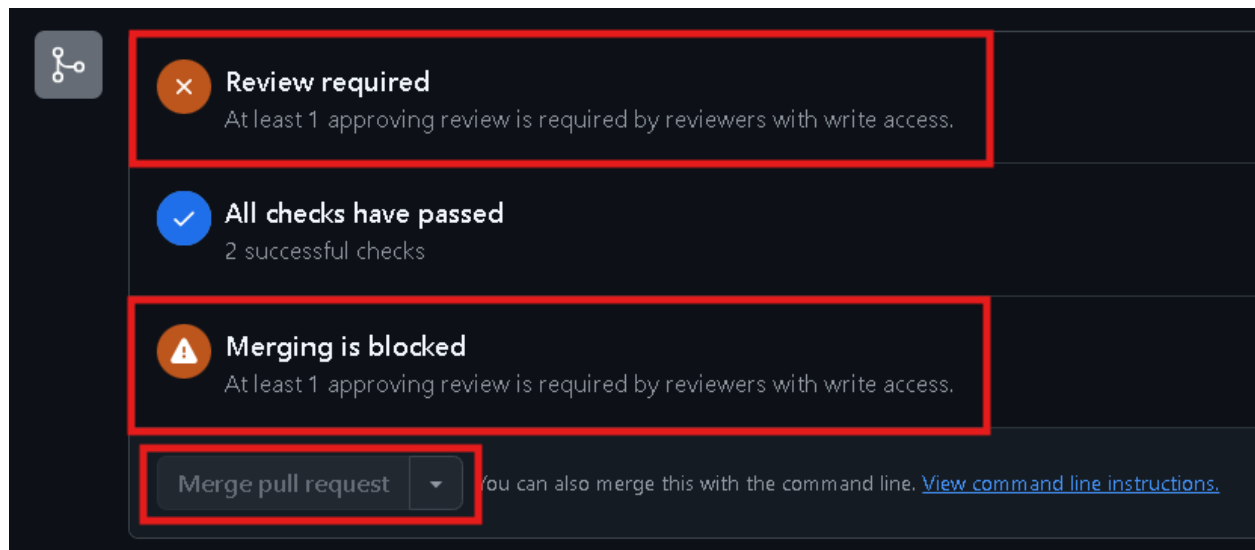


- Write a very descriptive title and description for the request. It can be the name and description of the feature you are trying to push to the “main” branch  
Sample of the title “Release: Working CI/CD Pipeline and Initial App Structure”

- Once the pull request is created, wait for the tests to pass as you can see below:



- One or more teammates should go over the changes and approve the pull request before the “Merge pull request” button goes blue and clickable. Sample below:



8. Once the “Merge pull request” is clickable, click on the button to complete the merge process.