Pull Requests

Complete your research to learn about feature branch workflow, then attend or watch the first live session archive. Together we will examine how to practice using "Pull Requests". A Pull Request (also referred to as "PR") is like doing a git merge locally, but instead, it is done online so others can view publicly your ongoing development activity.

In this course, you are authorized to merge PRs from Feature Branches into the "dev" and "stage" branches following our course workflow. These branches are also sometimes referred to as "Development" or "Staging".

The "master" or "main" branch" is considered a locked-down branch. In other words, You must seek and receive approval before merging into the main or master branch.

Dev

Through the weekly release cycle, the **dev** branch will be in flux, having new feature branches added and PR'd together which may or may not cause a conflict. The dev branch is more of a holding area that contains the latest unreleased features that have been merged together via PRs. New feature branches should be created from this area so as to minimize conflicts and ensure that each new feature has the most up-to-date codebase.

A PR from "**dev" into "stage"**, should be made early in the weekly release cycle and titled to reflect the latest milestone. This PR will track all changes being made to the dev branch from feature branches. At the end of the milestone time period (also referred to as the next "Release Cycle", "Milestone", or "Sprint"), you can approve the open PR from dev to stage.

When merging feature branches to dev, merge conflicts may occur. If so, you can use the dev branch to resolve the conflict. If the conflict is extensive in nature you may need to majorly overhaul a feature being worked on. To do so, you can then checkout the individual feature branch causing the issue and make modifications there as needed. When modifications are done, follow the PR process to get that code back into dev before PR'ing the latest work to stage.

Stage

This branch represents all feature complete work, ready to release to the main or master branch. This is primarily why the term "Staging" is often used to describe this branch. You are *staging work* to be released. In this course, you are authorized to merge to this staging branch from the dev branch.

Merging to the main or master branch ***does*** require approval, and is generally not required unless this is requested by the instructor or course activities.

Before merging to main or the master branch from the stage branch, a final check of the work is ALWAYS needed, This is the developer's last chance to test and identify any problems. Releasing bugs or broken code to users, or your instructors may result in negative consequences. So... utilize this branch/environment to check and verify that your work, works!

**Review Approval** is required to PR code from staging into Main or Master.

*Only instructors can provide this approval. The developer's workflow will not be interrupted while awaiting approvals as they are able to continue with the Github Workflow merging Feature Branches into dev and then stage.*

Once features are released into the main or master branch it is a common practice to delete the feature branch on GitHub and locally**.** This reduces clutter as those branches will likely not see further direct development. This deletion can be handled using the GitHub.com interface and git commands locally. Refer to your assigned research to learn how to delete a branch.

**Note:** Deleting feature branches is not required in this course. If you wish to clean up your list of feature branches, this is fine, just be sure the code was merged before deleting the branch. Otherwise, it's possible you may lose some critical work in progress.

Master or Main

Anything merged to master or main is considered *live to the world*. Only well-tested code should be PR'd to this final location, and only after developers have fully tested the work in staging. Final approvals, also known as the "green light" or ok to "Go Live!" is always needed.