**GitHub Desktop**

FOR DUMMIES

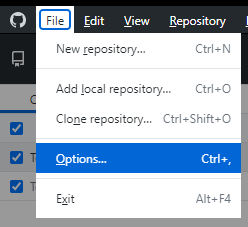
**Download GitHub Desktop**

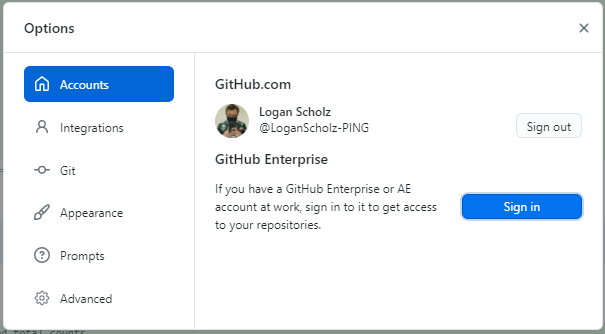
Download Link: <https://desktop.github.com>



**Linking GitHub Desktop to your GitHub account**

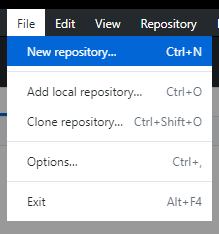
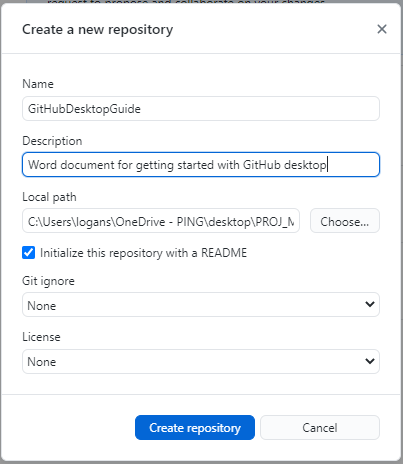
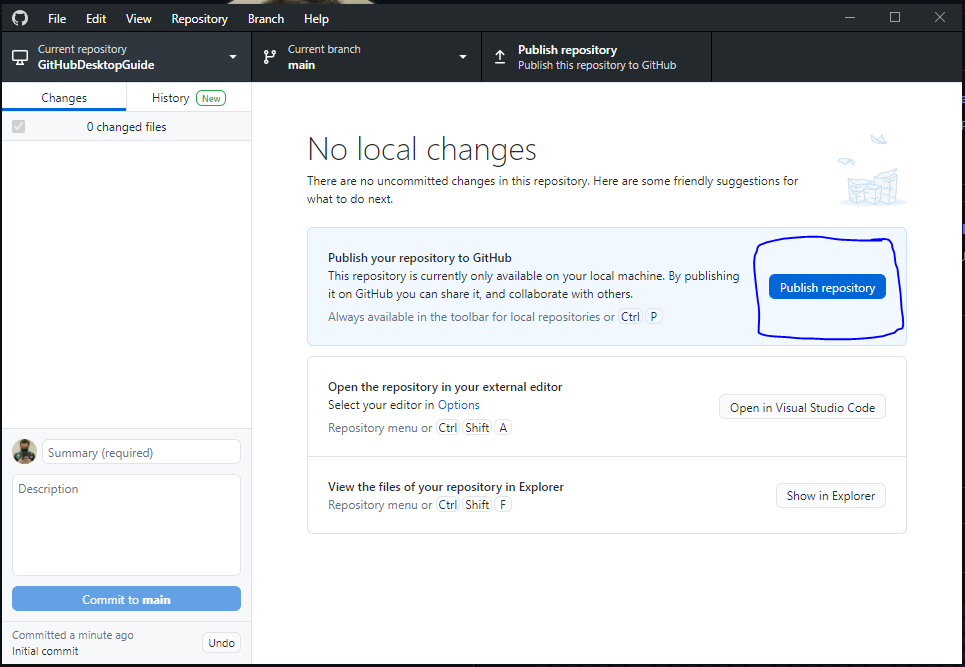
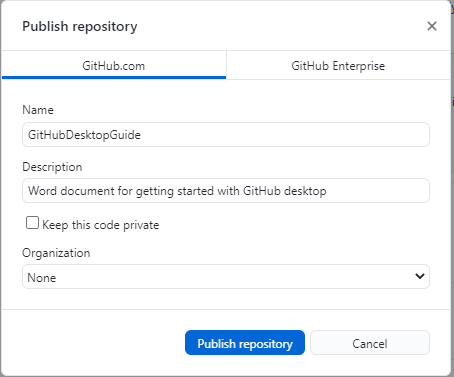
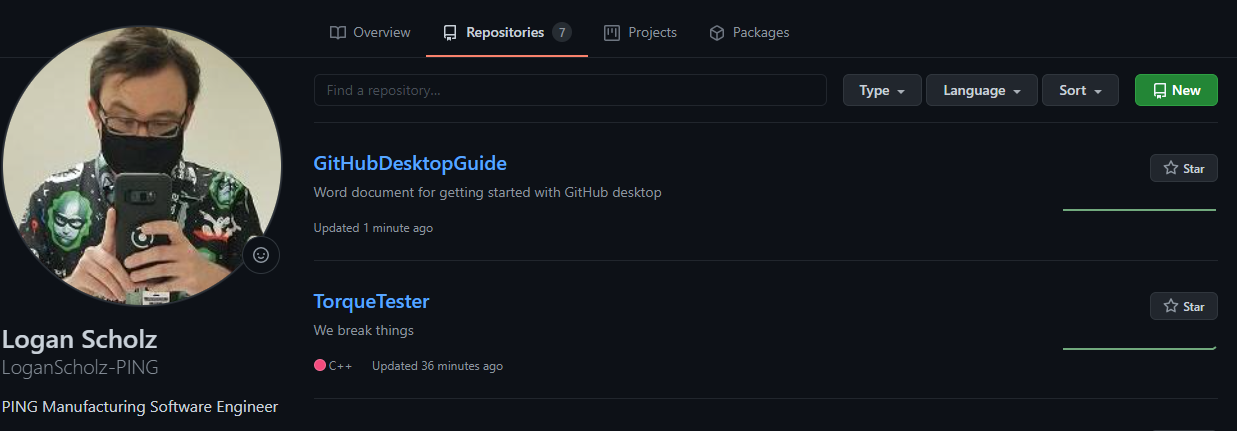
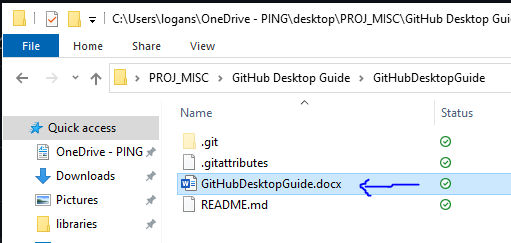
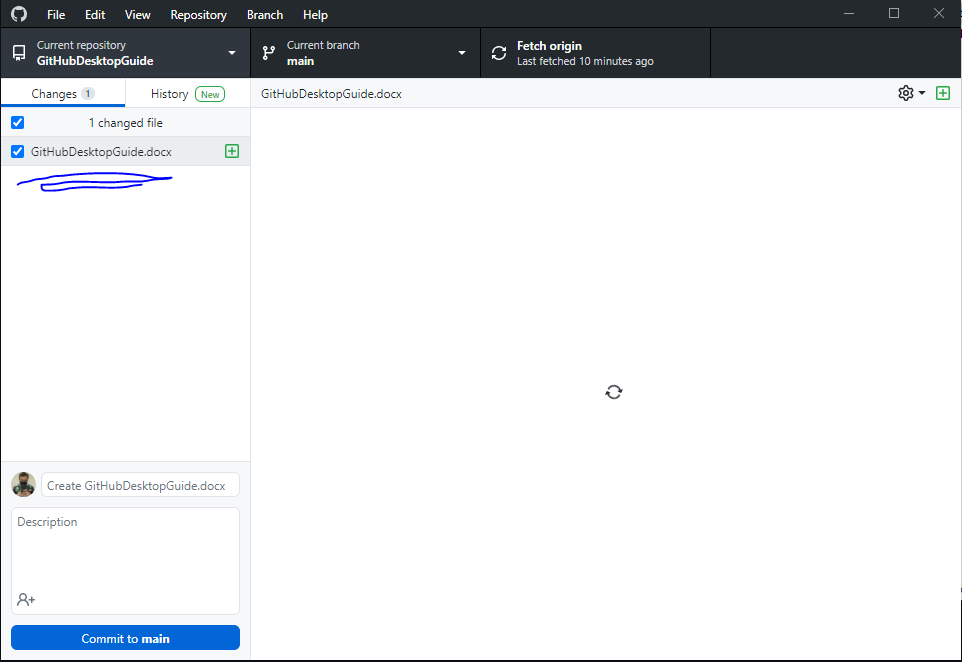
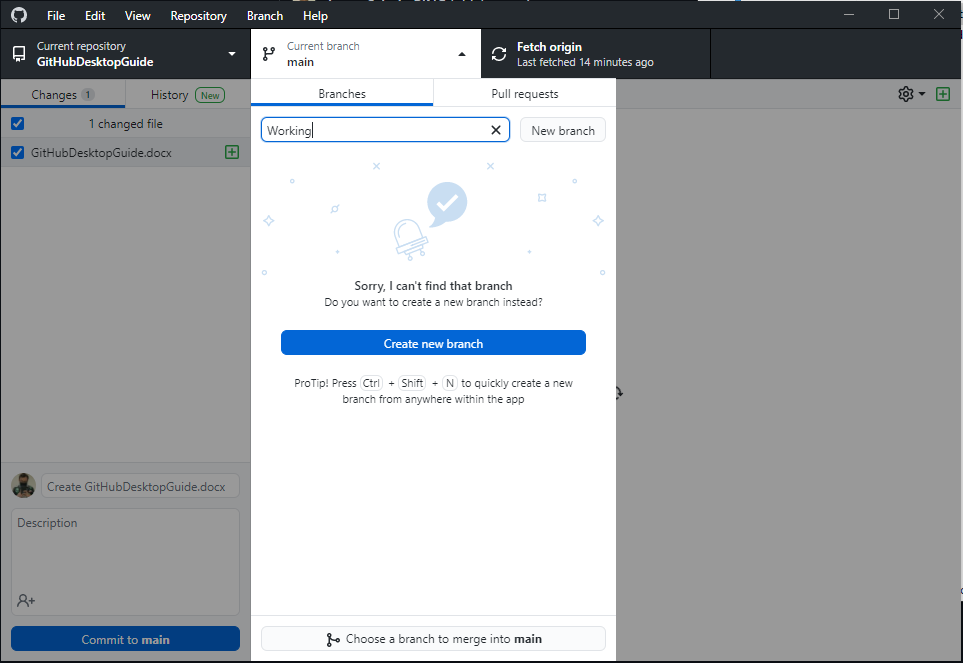
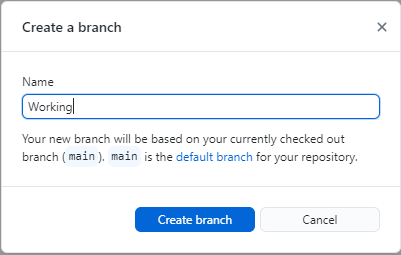
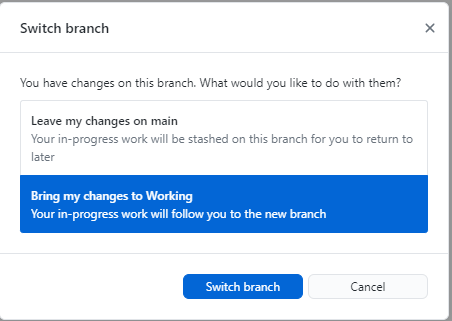
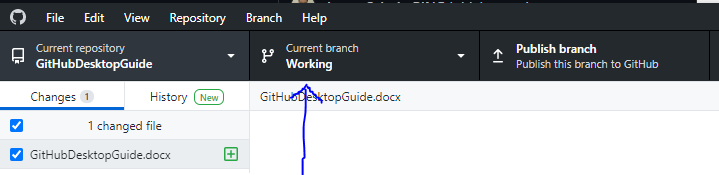
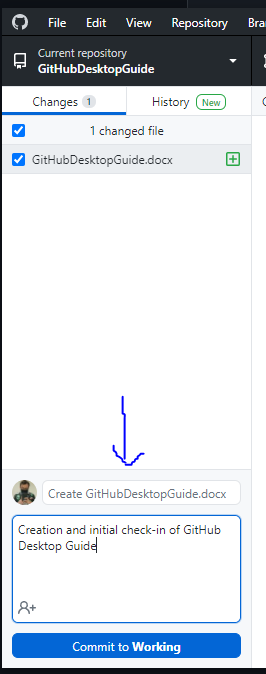
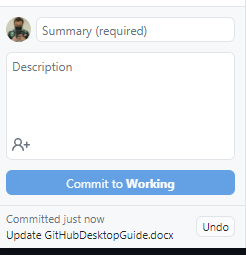
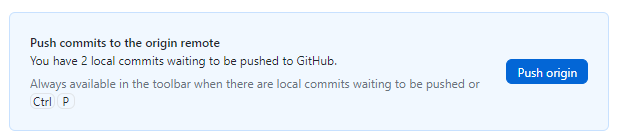
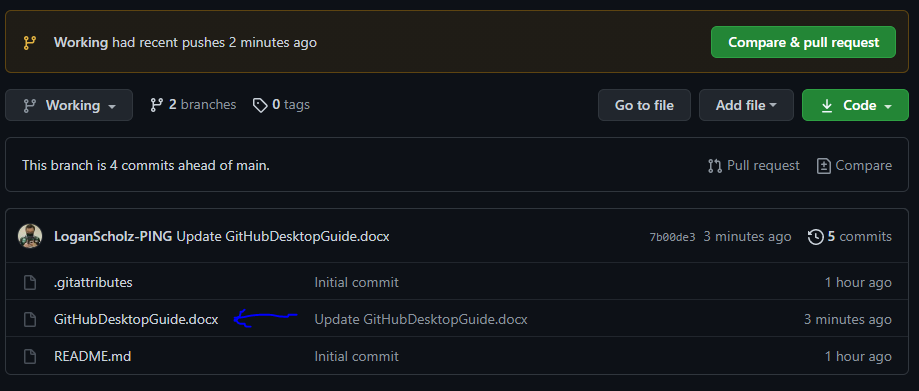
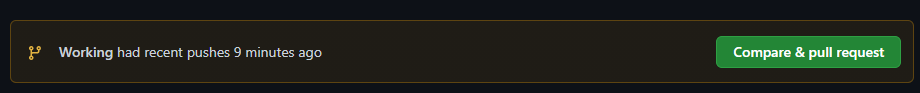
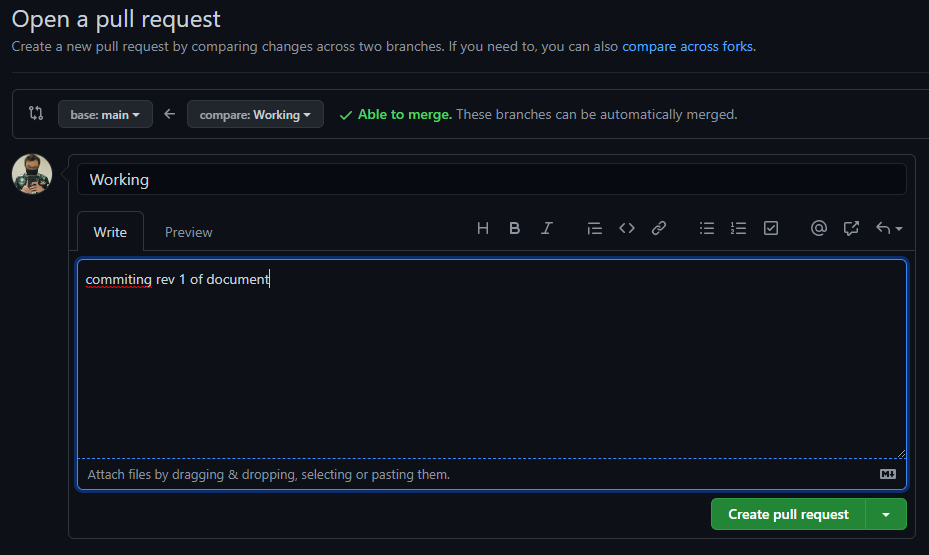
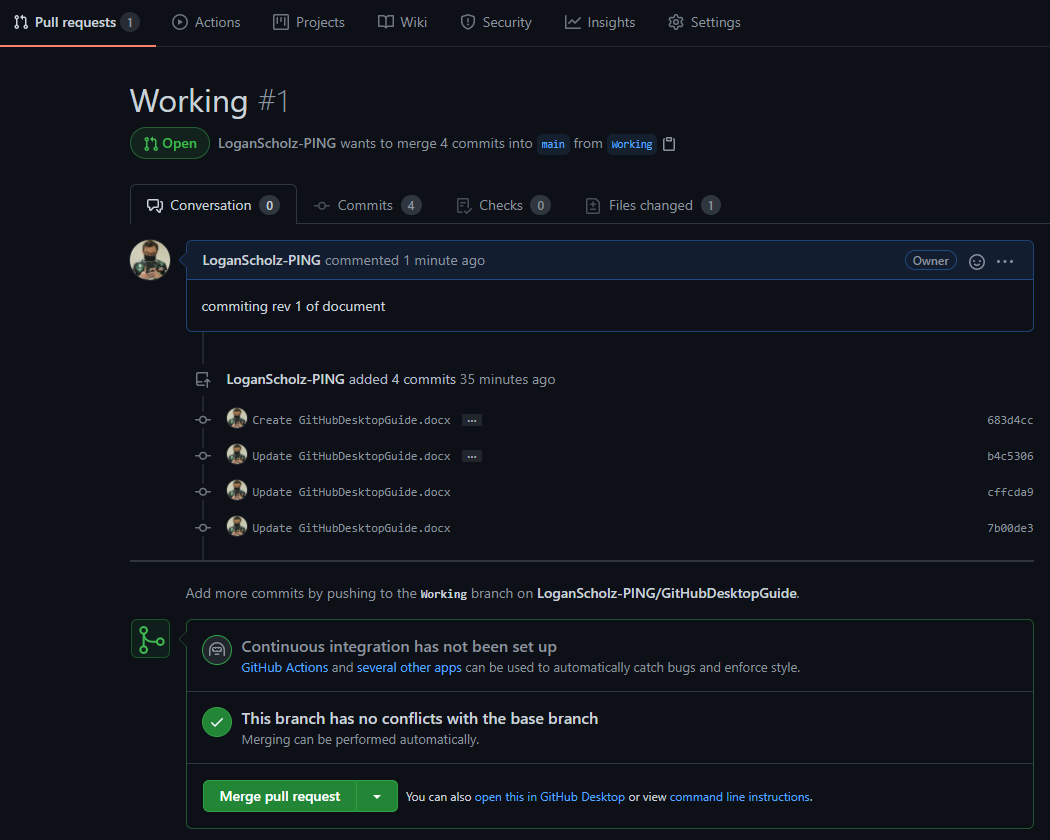
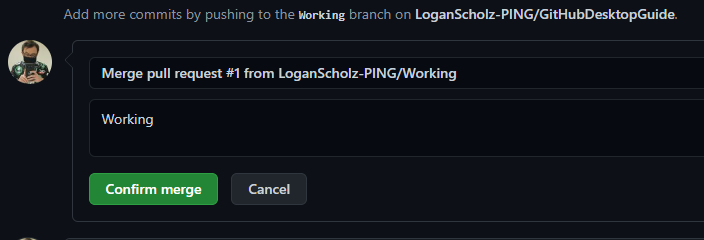
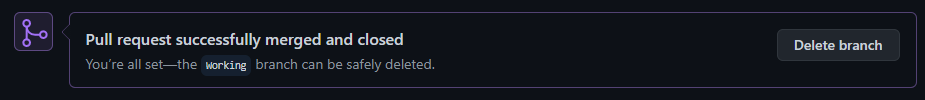
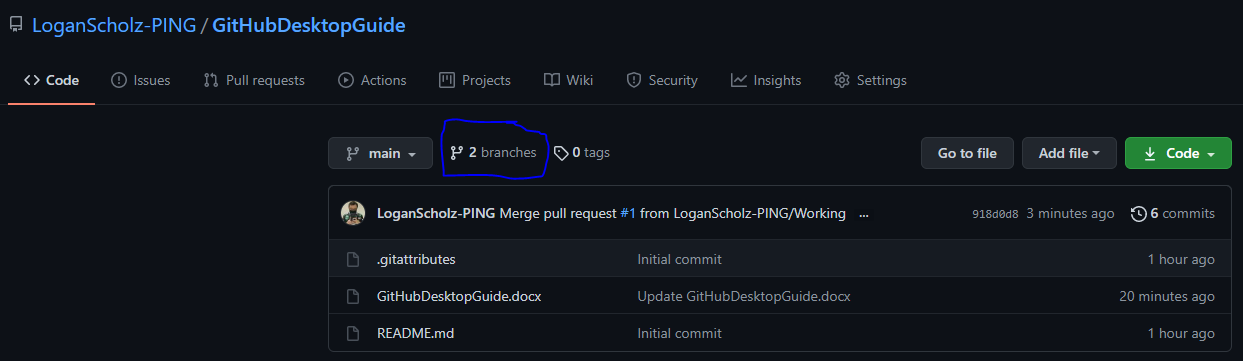
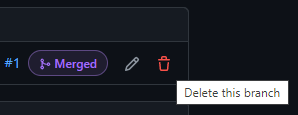
1. Once GitHub desktop is installed, go to File->Options:



1. Then “Accounts” and follow the prompts to link your GitHub web account to the GitHub desktop app:  
   

**Creating a NEW GitHub repository with GitHub Desktop**

As our first test of GitHub desktop, we are going to create a repository for tracking modifications to this document, as well as allowing you all to have access to the latest instructions

1. In GitHub Desktop, go to File->New Repository  
   
2. Fill out the information for the new repository:  
   
3. Click “Create repository” and allow the process to finish
4. Your repository has been created locally but has not been officially added to your GitHub repositories online yet. To officially check this repository in, click “Publish repository”  
   
5. The “Publish repository” window will appear, everything will be pre-filled out for you, but un-check the “Keep this code private” checkbox (unless you want to be the only person to see this repository along with anyone else you specifically invite to collaborate)  
     
   Click “Publish repository”
6. Proceed to the repositories list on your GitHub.com account and confirm that the new repository has been added:  
   
7. If we click on our new repository (“GitHubDesktopGuide” in this case), we find that the repository contains no particularly useful files. We will add our files in the next step
8. In step 2 you added a local path to contain this repository on your computer. Navigate to that folder and add the file or start a project:  
   
9. Return to the GitHub desktop application and notice that GitHub desktop understands that you have modifications in the local folder:  
   
10. Let’s commit these changes to a “Working “ branch (NOT MAIN!!! DON’T COMMIT WORKING FILES TO MAIN!!). To do this, first create a “Working” branch by left clicking “Current branch” in the GitHub Desktop application and typing “Working” into the text box:  
    
11. Click “Create new branch”, GitHub desktop will double-confirm the name of this new branch:  
      
      
    Click “Create branch”
12. In the “Switch branch” window, I recommend choosing “Bring my changes to Working”:  
      
      
    Click “Switch branch”
13. On GitHub Desktop, confirm that your Current branch is now the “Working” branch:  
    
14. We’re ready to commit our document now! To commit your changes on GitHub desktop, you need to make a title for the commit and fill out a description:  
    
15. Click “Commit to Working” to commit our current changes:  
      
    Notice that a message appears confirming our action
16. These changes are now checked in to our local machine, but they are not present inside our online GitHub repository. To publish these changes, click the “Publish to origin” or “Push to origin” button that appears after you’ve clicked the “Commit to Working” button:  
    
17. If we proceed back to our online GitHub repositories, we see that our changes have been committed to the “Working” branch, and our file is now available online!  
    
18. Repeat this document update and then “Commit to working” process until you have a document you are satisfied to commit to the “Main” branch of your repository.   
    Note that checking files in to the “Main” branch is a big deal, you need to be confident that what you are checking into the “Main” branch is solid and ready for the bigtime.
19. Once you are confident in the state of your working files you are ready to commit them to the “Main” branch. To do this you need to create a “Pull request” in order to “pull” your working changes back up and into the “Main” branch.  
    I recommend you do this “Pull request” process inside the online GitHub.com:  
    
20. Click “Compare & pull request” and fill out the information relating to this pull request:  
    
21. Click “Create pull request”
22. Once you double check the differences between the “Main” branch and the current “Working” branch and confirm that your files are ready to commit to “Main”, click “Merge pull request”:  
      
    Then click “Confirm merge”:  
    
23. Once the pull request is complete and the changes are merged back into the “Main” branch you will see this message:  
      
    If you want to clean up the working branch so you can re-use the name, proceed to the root level of the current online GitHub repository and click the “branches” button:  
    
24. We see that “Working” has the purple “Merged” symbol next to it:   
      
    This “Merged” symbol means the changes have been successfully merged to the “Main” branch and can be deleted. Click the red trash-can icon next to the “Working” branch to remove the “Working” branch so we can re-use the name:  
    
25. Branches are never really “gone” in GitHub. This is part of the appeal of a document configuration utility. Notice that the deleted “Working” branch still has a “Restore” button next to it. In an absolute worst-case scenario, you can re-open this “Working” branch to view the files “frozen” in the state they were in when they were committed back to the “Main” branch:  
    