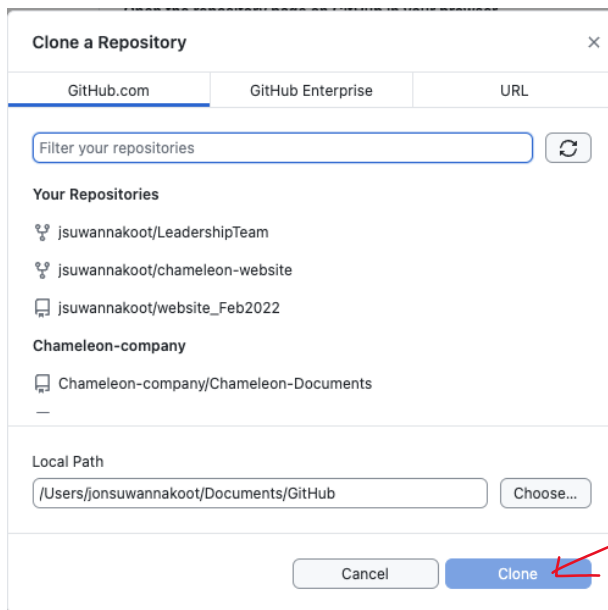


GitHub Desktop

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

First, clone the repository from the main branch

Go to file → Clone repository.

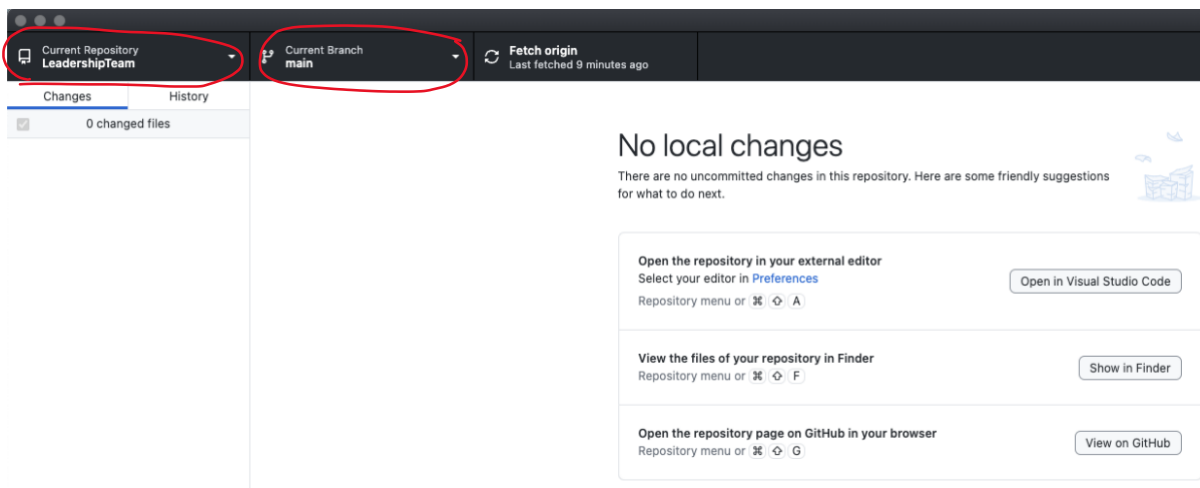


choose the URL or Github.com

And click 'Clone.'

You can see the Current Repository at the top left corner.

Then you need to create a branch from the main branch to start editing the file in your branch. After that, the leadership team will merge your branch with the main branch.



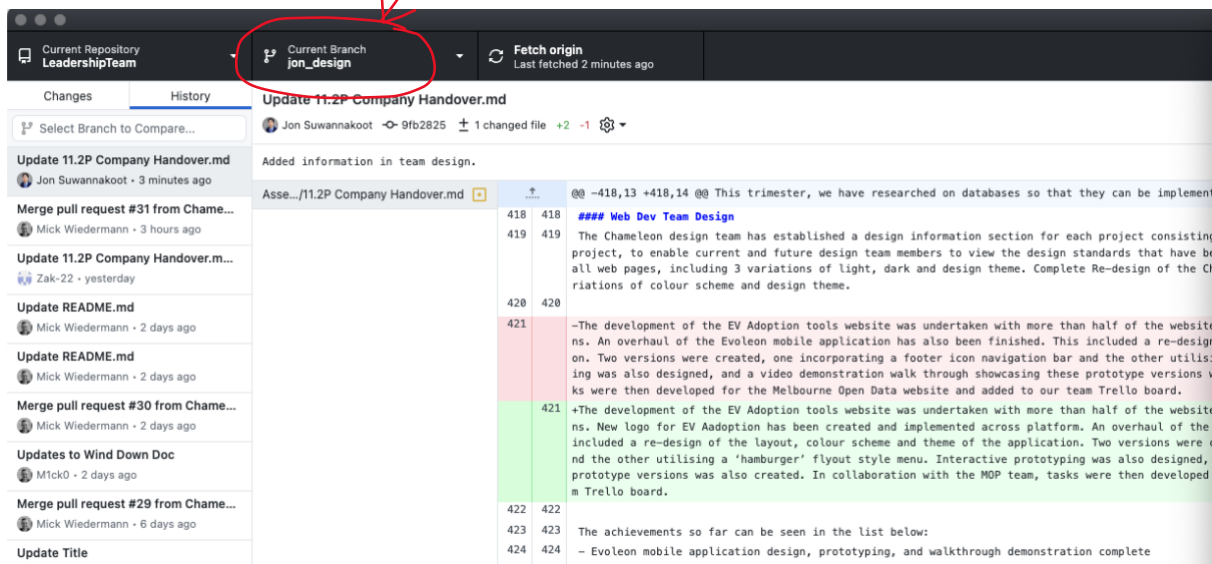
From the picture above, you can see that you're still in the main branch of the leadership team.

Create the Branch

In GitHub Desktop, go to Branch -> New Branch

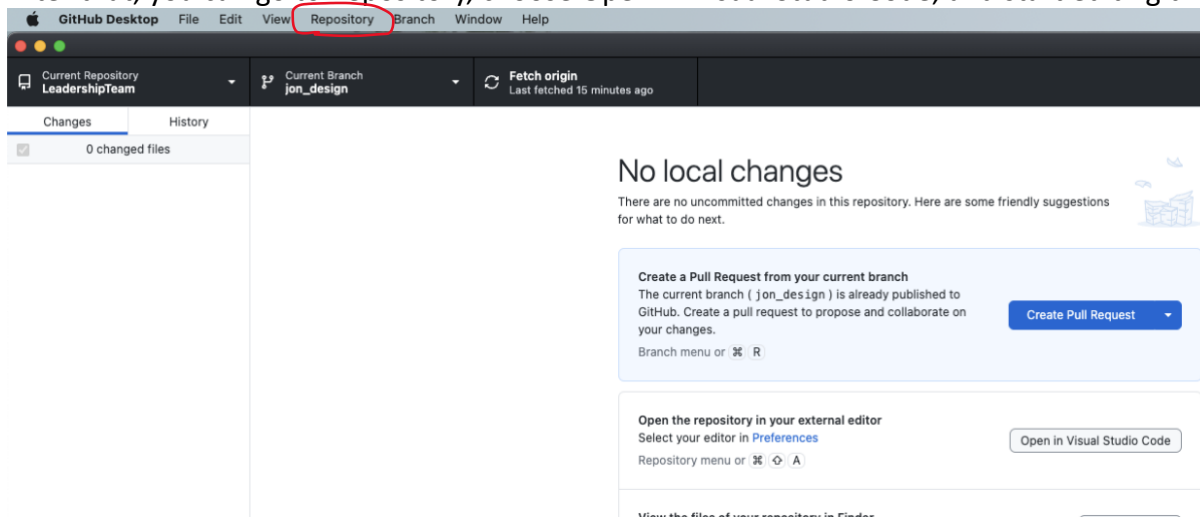
Then you will see that you are in your new branch in the current repository.

Your new branch



Then you can make the pull request to download the files to your local directory.

After that, you can go to Repository, choose Open in Visual Studio Code, and start editing the files.



Important: To prevent conflict, make sure that you rebase before committing to the changes you make. Follow the step below.

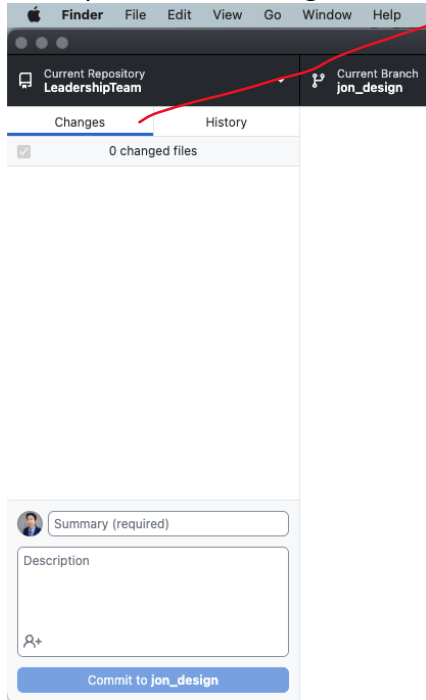
In GitHub, the "rebase" feature is used to integrate the changes from one branch into another by incorporating the commits of one branch onto the tip of another branch. It allows you to modify the commit history and make it appear as if the changes were made directly on top of the target branch. The primary purposes of using the rebase feature in GitHub are:

1. **Keeping Commit History Clean:** Rebasing helps maintain a linear and cleaner commit history by avoiding unnecessary merge commits. It allows you to incorporate changes from another branch directly onto the tip of the target branch, resulting in a more streamlined and readable commit history.
2. **Simplifying Merge Commits:** Instead of creating a merge commit with the default merge operation, rebasing allows you to integrate the changes seamlessly by replaying the commits on top of the target branch. This can make the branch history easier to follow and understand.
3. **Resolving Conflicts:** When rebasing, if there are conflicts between the changes in your branch and the target branch, you will have an opportunity to resolve those conflicts during the rebase process. This helps in addressing conflicts early and ensures a cleaner and conflict-free integration.
4. **Preparing for Pull Requests:** Rebasing can be useful when you want to ensure your branch is up to date with the latest changes from the target branch before submitting a pull request. By rebasing, you can incorporate those changes into your branch, resolve conflicts, and present a more cohesive and up-to-date set of commits for review.

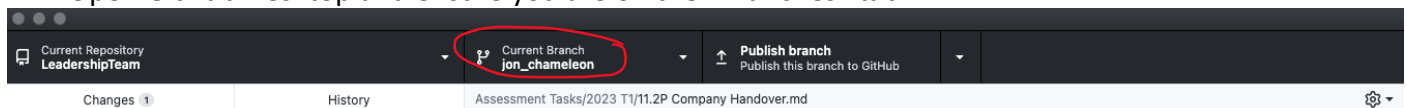
It's important to note that rebasing modifies the commit history of your branch, which can affect collaboration with others if the branch is shared. It's generally recommended to communicate and coordinate with your team when using the rebase feature to avoid any unintended consequences and conflicts.

To rebase your branch using GitHub Desktop, follow these steps:

After finishing editing the files or your code in Visual Studio code, you need to commit and push to your branch. This is done by going to Changes (picture below), writing about your changes in summary and description, and clicking commit.

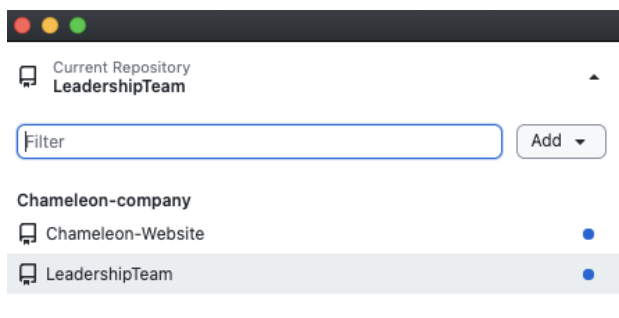
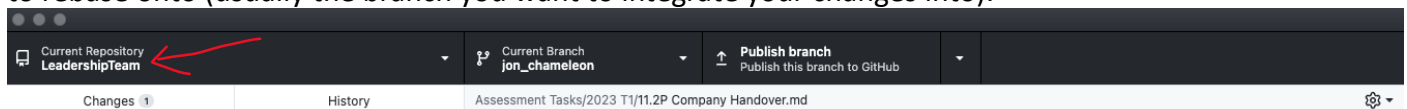


1. Open GitHub Desktop and ensure you are on the "Branches" tab.



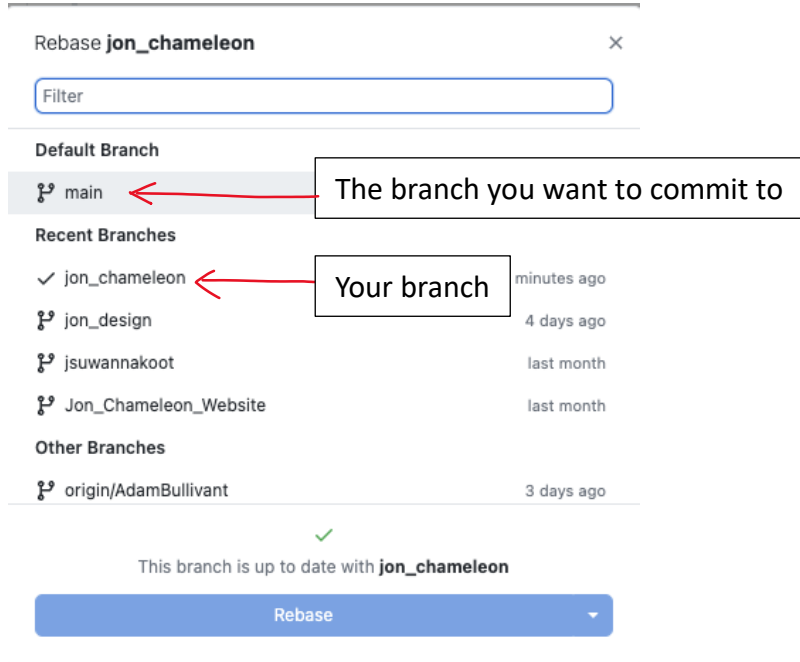
2. Select the branch you want to rebase from the list of branches on the left-hand side.

3. Click on the "Current Branch" dropdown button located at the top left and select the branch you want to rebase onto (usually the branch you want to integrate your changes into).



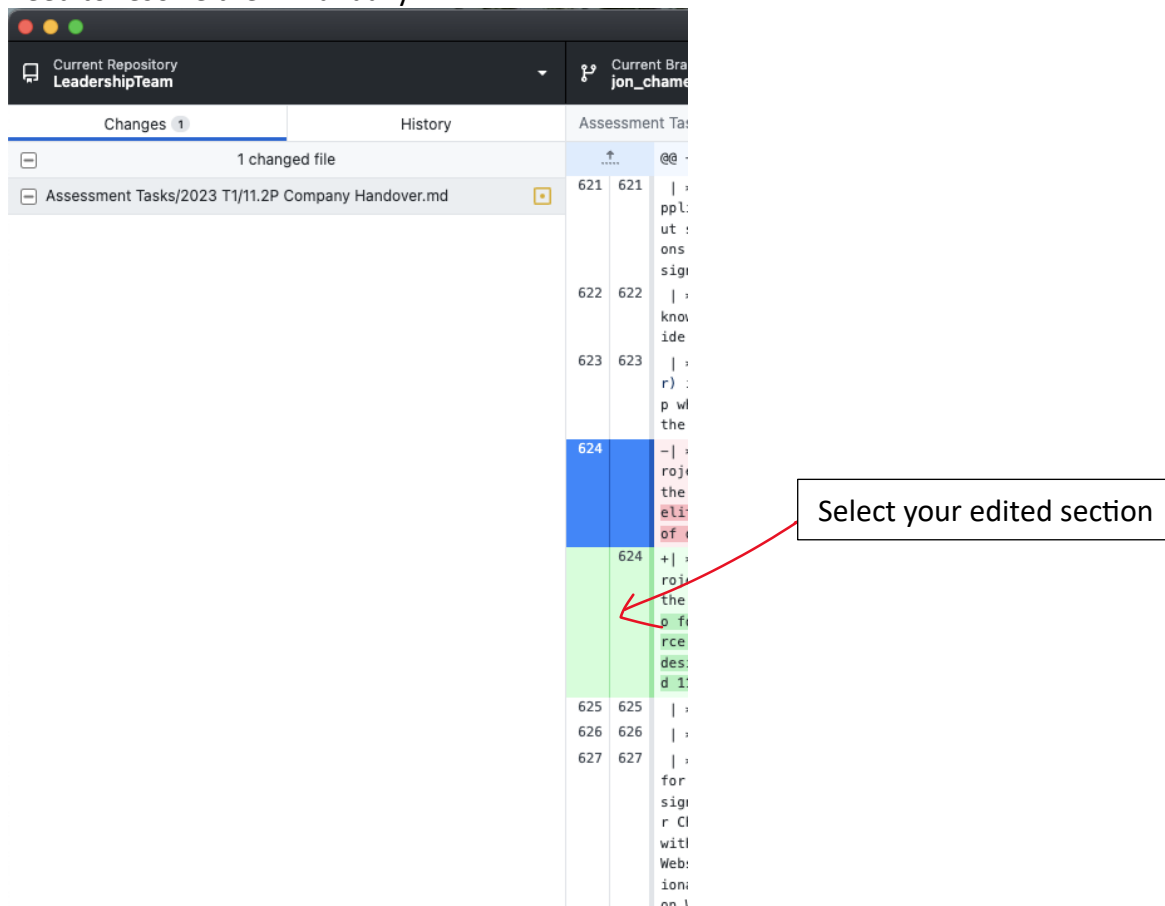
4. With the branch selected, click on the "Branch" menu in the menu bar and choose "Rebase current branch...".

5. In the Rebase dialogue, you will see a list of recent commits on your branch. Click on the checkbox next to each commit you want to include in the rebase process. You can also drag and drop commits to reorder them if needed.



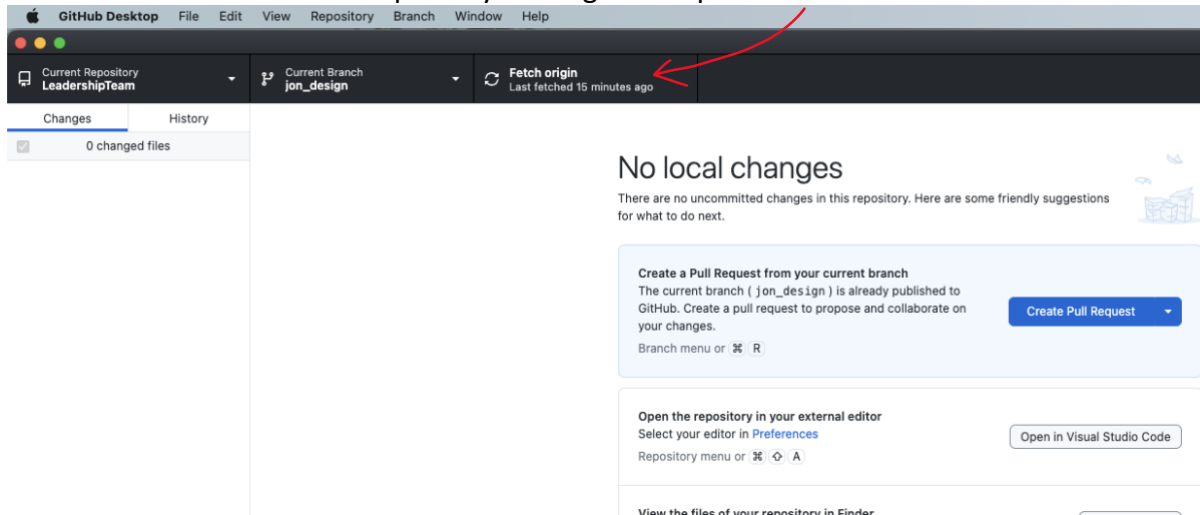
6. Then you can open the document in Visual Studio code and edit and save.

7. Go back to GitHub Desktop. Then select the desired commits, and click on the commit button. GitHub Desktop will then perform the rebase operation, incorporating your selected commits onto the target branch. This may involve resolving any conflicts that arise during the process. If conflicts occur, you will need to resolve them manually.

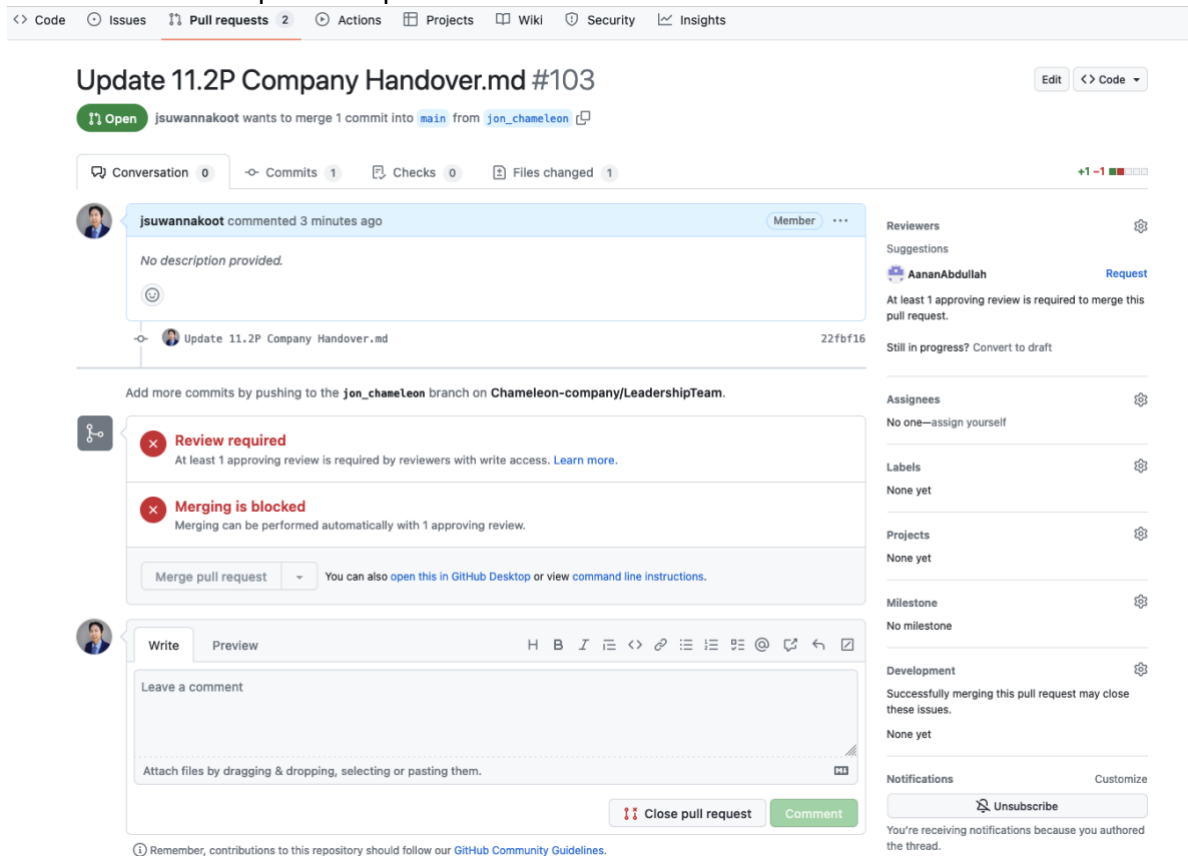


8. After the rebase is complete, you can push the rebased branch to the remote repository by clicking the "Push" button at the top right of the GitHub Desktop interface.

9. Then click Create Pull Request by clicking on the push button.



10. Then go to the GitHub website. You will see your commit to the Chameleon company. Fill in the information to complete the process as below.



After that, the leadership team will review and merge it. Notice that there is number #103 of your commit, ready for the leadership team to review and merge.

If there is a problem, you will be asked to edit and commit/push. After your branch is merged, if you want to edit the same document, you should create a new branch and commit/push with rebase.