Standard Operation Procedure for collaborating through Github:

Master branch is the one that has the established code that everyone has seen; it's also the branch that our website is based off of.
-> When working on a feature, we can create a sub-branch so that we don't mess with the original branch.
***It's very crucial to communicate with each other and avoid working on the exact same thing; this can create conflicts.**

Step 1: open up terminal (on Mac) or search for cmd (on Windows)

Step 2: Type in git clone "The url on the repository" (ex: git clone http://ycpan1597.github.io/...)

Step 3: Type in cd "where you placed your cloned repository" (For example, I saved my repository on desktop so I type in: cd Desktop/DISplayWebsite)
(cd stands for change in directory)

Step 4: Type in git checkout master
This switches you into the master branch

Step 5: Type in git pull
This pulls all of the changes that have been merged to master since you last worked on it.

Step 6: Type in git branch <Your own branch> (ex: git branch newBranch)
This creates a branch for you that is identical to the master branch at the time you create it.

Step 7: Type in git checkout <Your own branch> (ex: git checkout newBranch)
This switches you into the new branch you just created

Step 8: start working on your branch (including html, css, file organization/uploading)

Step 9: Once you're done, go back to the terminal and type git add <the file name> (for example, git add index.html; or if you want to add everything, do git add -A)

Step 10: Type in git commit -m "message"
If you accidentally forgot the -m "message" part, you'll have to press the escape

key, and type in :wq and press enter; that should get you out of the window

Step 11: Type in git checkout master, followed by git pull
This switches you back into master so that you can pull any changes that others might have made to the master branch during the time that you've been working on your own branch.

Step 12: At this point, a conflict could arise if you and someone else were working on the exact same thing (which is why I recommend now working on the exact same thing at the beginning of this document)
If this happens, go into the problematic file(s) and make manual changes to accommodate the differences.

Step 13: if no error has occurred, or after you've fixed the conflict, then type in git checkout <Your own branch> (ex: git checkout newBranch)
This switches you back into your own branch

Step 14: Type in git push origin newBranch
This pushes whatever you added in step 9 to a new branch called "newBranch"; at that point, other people can also see this modification and merge it into the master for you .

Step 15: Once you have pushed your changes to the branch, and these changes have been verified, we use the following four commands to merge this branch back to the master branch:

- git checkout master (switches your to the master branch)
- git pull origin master (you should always pull to make sure no one else merged to the master branch recently)
- git merge newBranch (merge the two branches together -> there could be a merging conflict at this point
- git push origin master push this newest master up to the repository
- git branch -d newBranch delete other branches once merging is complete