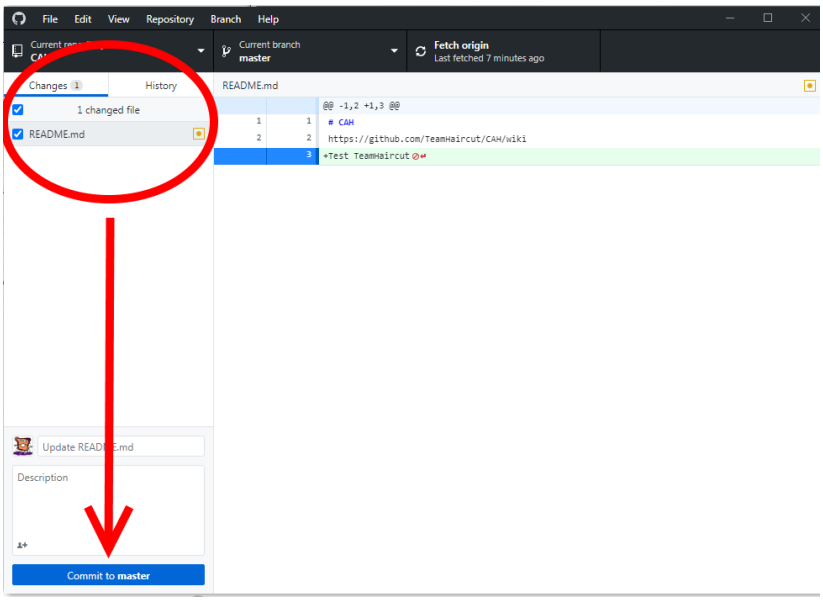
Dev-Environment Setup

First you need to download and install the GitHub Desktop Application. This application allows you to set up a local repository that you will use for development. The application gives you ability to make changes to the files located in your root working directory, and push them to the central repository on the web.

1. Download GitHub Desktop Application
    a. go to https://desktop.github.com/
        i. Click on link "Download for Windows (64bit)"
    b. Save GitHubDestopSetup.exe to downloads
2. Install GitHub Desktop Application
    a. go to downloads folder
    b. double click GitHubDesktopSetup.exe to run file
    c. Sign In to github
    d. Click the "Authorize desktop" button
    e. Allow x-github-desktop-auth popup
    f. Click "Continue"
    g. Click "Finish"
3. Set up the local repository
    a. Click "Clone a repository from the Internet"
    b. Select the URL tab
        i. In the "URL or username/repository" input field paste "https://github.com/TeamHaircut/CAH"
        ii. Click "Choose" to select the directory where you want the local repository (your working files) to reside
            1. I put mine under "...\ Desktop\CAH\CAH-dev"
            2. We will refer to this as your root working directory
        iii. Click "Clone"
            1. This will pull all the files located from the central repository (https://github.com/TeamHaircut/CAH), into your local working directory.
4. Test the GitHub connection
    a. Open README.md in notepad
        i. Recommend using a more robust text editor (like notepad++), for development
    b. Append the text "UserName
" to the bottom of the file
        i. Where "UserName" is your github username
    c. Save the file
    d. Note that the 1 changed file can be found in the GitHub Desktop Application

i. 

e. Click" Commit to master"
f. Then click "Fetch Origin" to sync with the central repository
g. Confirm that README.md has been updated with your username on the central repository by going to https://github.com/TeamHaircut/CAH

The next step is to install NodeJS onto your development machine. The software architecture will use NodeJS to support the application's back-end server. NodeJS allows you to start the server locally on your machine, and test the code you develop in a web browser.

5. Download NodeJS
    a. go to https://nodejs.org/en/
        i. Click link for "12.8.3 LTS"
        ii. Save the msi in the downloads folder
6. Install NodeJS
    a. Double click the msi to run the file and the installation wizard should start.

b. Once you click on the "Next" button, the installation should begin. A few moments later, you'll see a confirmation screen similar to the following screenshot, telling you NodeJS was successfully installed:

The HelloWorldExpressApp is a simple web app you can use to test that NodeJS has been properly installed. It was added to your root working directory when you initially set up your local repository. From your root working directory, its source code is located at...\CAH\examples\HelloWorldExpressApp\HelloWorldExpressApp\ . To test this application you will need to open a command prompt window, start the node server, and open a web browser.

7. Start the node server
    a. Open a command prompt
        i. Click "Windows Start" buttom, and search for "cmd". Then select "cmd.exe" from results list
    b. Using the cd command traverse the filesystem to the HelloWorldExpressApp source directory (see example below)
        i. cd C:\Users\~UsrName\Desktop\CAH\CAH-dev\CAH\examples\HelloWorldExpressApp\HelloWorldExpressApp
        ii. Then press "Enter"
    c. Type "node server" in the command prompt window to start the server
        i. expected output: "Server running at http://localhost:3000/"
8. Test the HelloWorldExpressApp
    a. go to http://localhost:3000/, and you should see the words "Hello World" printed in the web browser.
    b. Note: If you refresh the browser after closing the cmd prompt that is running the server, you will get a refused connection error. You get this error because the browser acts as a client, and it was not able to locate the server it was coded to connect to. To avoid this, always make sure the server is running, before you test your code.