**Exercise 2 - GitHub training**

For this exercise, you will need:

* A Github account <https://github.com/>
* Atom editor <https://atom.io/>
* GitHub Desktop <https://desktop.github.com/>

Please download Atom editor prior to GitHub Desktop so you can select Atom as your text editor during the installation procedure.

* Three images (jpeg preferably) that you are happy to show others and describe! (If you can’t access any immediately, just google something interesting and save as a jpeg.)

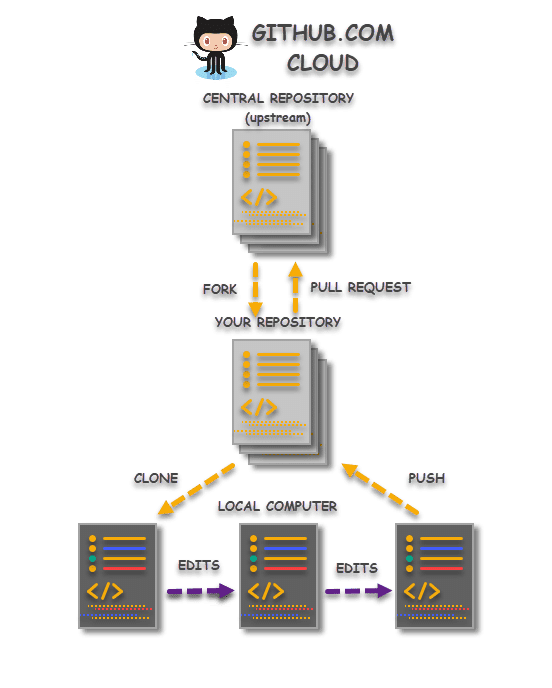
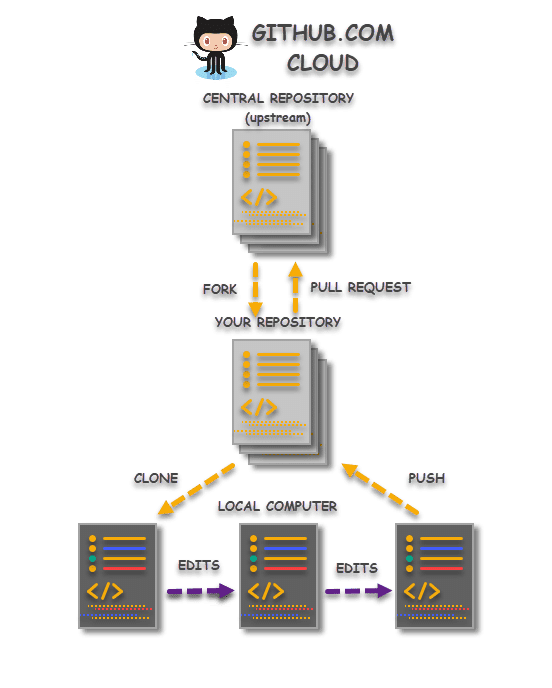
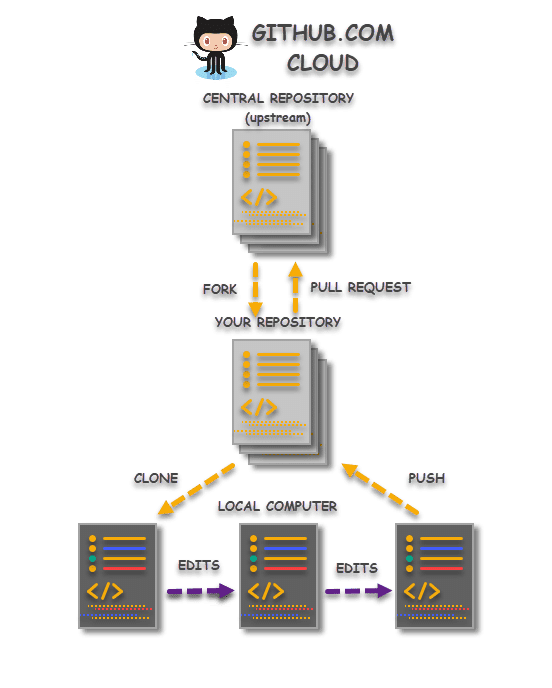
###############################################################################

1. Locate the ‘Example\_website\_2’ repo in my GitHub page: <https://github.com/SBurnard>
2. ‘Fork’ this repo to your own account (upper right screen)
3. ‘Clone’ to your desktop, via GitHub desktop:
   1. Click the arrow next to ‘current repository’.
   2. Add
   3. Clone repository…
   4. Either ‘filter for your repositories’ or use the ‘url’ to your repo!  
       i.e. https://github.com/<your.user.name>/Example\_website\_2
   5. Choose your local path (where to download it to)
   6. ‘Clone’ and wait!
4. Open this repo on the left bar.
5. Click ‘show in Explorer’. Have a look around. Double click on some of the html files!
6. Click ‘Open in Atom’. Have a look at the directories in the left bar and view the html and text files. You’re now actually seeing what’s ‘underneath’ the html files
7. Delete (or move) the three images pic02-04 in the directory /images/ via file explorer.
8. Locate your images, and change the name of your image names to:  
   pic02.jpg, pic03.jpg and pic04.jpg
9. Move or copy these renamed images into the ‘images’ directory.
10. Now to edit the webpage writing!
    1. Click back onto Atom
    2. We’re going to now the ‘index.html’ file which contains the main code that creates the webpage. Notice the different colours that are autogenerated by atom, which depends on the command being used. These can be changed, but the default setting in Atom have been made to work with GitHub, html, python and other types of code. 😊
    3. Look out for the light gray code - these are notes, similar to those in R when using the hashtag #. But in HTML, these are wrapped with <!-- personal comment not displayed on webpage -->
    4. On line 9, write you name (removing < >)
    5. Now we’re going to write titles and a brief description for the three images you’ve chosen (and recently renamed).  
       I’ve written a comment beside each of these, but for ease you can refer to the line numbers each of these are located. N.B. replace the white text, between the flags that are encased with arrow heads <>. A simple image header and interesting image description will suffice! We will share these with everyone at the end. 😊

~~62 Image\_2 File~~  
68 Image\_2 Title  
73 Image\_2 Description  
~~86 Image\_3 File~~  
89 Image\_3 Title  
92 Image\_3 Description  
~~102 Image\_4 File~~  
105 Image\_4 Title  
108 Image\_4 Description

* 1. Save these changes (notice any colour changes on Atom?) - either ‘ctrl s’ or file save.

1. We are now read to ‘commit’ then ‘push’ the changes back to your remote repo!



* 1. Close atom (to ensure changes have been saved!)
  2. Click back onto GitHub Desktop - you should now see a screen with green and red lines (with either - or +), on the left of that shows line numbers. This is showing the changes you’ve made.
  3. Before clicking ‘commit to main’ (bottom left) write a title and brief description of the changes, this serves as a reminder as to what you have done. (you can leave this blank though but is a good habit to fill these in with useful info).
  4. Click commit to main. Wait
  5. Now ‘push’ to origin.   
     Either Ctrl P or click ‘push origin’.
  6. Huzzah! These changes have now been sent to your online repo on GitHub!
  7. To confirm - Go to your GitHub repo and refresh. Notice any time-stamp changes on your files?

1. Now we want to make our webpage active!
   1. Click on settings in the repo page.
   2. Scroll down to ‘GitHub Pages’
   3. Under ‘Source’ change branch to ‘main’ and save.
   4. Above you should now see a link to your webpage - click it and have a look… (it may take a minute to generate).
2. Awesome! It worked (hopefully). But wait. We’re missing an image next to ‘The future has landed…’. So we’ll fill this in with an image of your favourite animal! Maybe, your spirit animal?!
   1. Google whatever your favourite animal is.
   2. Right click and ‘save as’ jpeg to a location on your computer you can easily access and name it something appropriate such as ‘My\_spirit\_animal’.
   3. Upload the file into the /images/ directory in the GitHub repo.   
      You have at least three different options to achieve this. What are they? (clue - two are directly via the GitHub website).
   4. Click back to the ‘top level’ of this repo and click on the ‘index.html’ file.  
        
      We’re now going to edit this file from within GitHub and point the circular image to the name of the image file you’ve created.
   5. Click the pen icon at the upper right side of the document on the webpage (when you hover over it, it should say ‘edit this file).
   6. On line 55, change the name of the file that ‘src’ is pointing towards to the name of the file you just uploaded i.e.  
      *<span class="image"><img src="images/pic01.jpg" alt="" /></span>*becomes  
      *<span class="image"><img src="images/My\_spirit\_animal.jpg" alt="" /></span>*
   7. *Commit those changes (Don’t forget to add some update details)*
3. Now go back to your webpage and refresh. Do you see your animal in the circle now?   
   If you’ve just refreshed the page soon after committing the changes the chances are you won’t see it… It will need a couple of minutes for these changes to then be pushed onto your webpage. Don’t worry, this happens automatically. Just come back to it in a couple more minutes, and, voila! 😊 Your own personal page has landed!

Congratulations on making your own website and learning some of the fundamentals of Git and GitHub!! ^^