**Section 5.2 – Introduction to GitHub**

**Github**

If you only want to keep track of your code locally, you don't need to use GitHub. But if you want to work with a team, you can use GitHub to collaboratively modify the project's code.

To create a new repo on GitHub, log in and go to the GitHub home page. You should see a green '+ New repository' button:

A screenshot of a cell phone

Description automatically generated

After clicking the button, GitHub will ask you to name your repo and provide a brief description:

A screenshot of a cell phone

Description automatically generated

To create the repo you’ve press the ‘Create repository’ button at the bottom.

GitHub will ask if you want to create a new repo from scratch or if you want to add a repo you have created locally. In this case, since we've already created a new repo locally, we want to push that onto GitHub so follow the **'** an existing repository from the command line' section:

**Note:** Change the *<your URL>* in the first command line (git remote add origin <your URL>) to what GitHub lists as your URL.

user$ git remote add origin <*your URL>*

user$ git push -u origin master

Counting objects: 3, done.

Writing objects: 100% (3/3), 263 bytes | 0 bytes/s, done.

Total 3 (delta 0), reused 0 (delta 0)

To <*your URL>*

\* [new branch] master -> master

Branch master set up to track remote branch master from origin.

**Pushing branches to GitHub**

Now lets push the commit in the branch you previously made to the GitHub repo. This will allow other people to see any changes that have been made. If they're approved by the repository's owner, the changes can then be merged into the master branch.

To push changes onto a new branch on GitHub, use the ‘git push origin <yourbranchname> command. GitHub will automatically create the branch for you on the remote repository:

user$ git push origin branch\_example

Counting objects: 3, done.

Delta compression using up to 8 threads.

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 313 bytes | 0 bytes/s, done.

Total 3 (delta 0), reused 0 (delta 0)

To *<your URL>*

\* [new branch] branch\_example -> branch\_example

**Pull Requests**

A pull request (or PR) is a way to alert a repo's owners that you want to make some changes to their code. It allows them to review the code and make sure it looks good before putting your changes on the master branch.

This is what the PR page looks like before you've submitted it:

A screenshot of a cell phone

Description automatically generated

And this is what it looks like once you've submitted the PR request:

A screenshot of a cell phone

Description automatically generated

If the button entitled ‘Merge pull requests’ is green then you are good to go. If it is grey, this means there is a merge conflict. This happens when changes in one file don’t fit with changes to another file, and git is unsure which version to use.

**Merging private repos**

If you click the 'Merge pull request' button your changes will merge into a master branch.

When you're done delete the branch. This saves things getting cluttered.

You can double check that your commits were merged by clicking on the 'Commits' link on the first page of your new repo.

This shows a list of all the commits in that branch.

You can also see the hash code of the commit on the right-hand side. A hash code is a unique identifier for that specific commit. It's useful for referring to specific commits and when undoing changes. You can do this using the command: git revert <your hash code number>

**Linking changes to GitHub back to your local machine**

Your local Git wont automatically update when you merge branches (or make any changes) on GitHub. To get the most recent changes use the git pull origin master command.

user$ git pull origin master

remote: Counting objects: 1, done.

remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (1/1), done.

From https://github.com/user/mynewrepository

\* branch master -> FETCH\_HEAD

b345d9a..5381b7c master -> origin/master

Merge made by the 'recursive' strategy.

mnelson.txt | 1 +

1 file changed, 1 insertion(+)

This shows you all the files that have changed and how they've changed.

You can then use git log to see any and all changes.

**Cloning a repository**

If you want to get a repository from GitHub onto your local machine (this could be your teams project, or an open source piece of software you want to use) you can ‘clone’that repo from GitHub to your local machine

Cloning a repository pulls down a full copy of all the repository data that GitHub has at that point in time, including all versions of every file and folder for the project. At any point, you can push your changes to the remote repository on GitHub, or pull other people's changes from GitHub.

To do this, do the following:

$ git clone https://github.com/THE-USERNAME/THE-REPOSITORY

This will create a local clone on your computer.