Git and GitHub

# Create a Git Depository

To create a Git depository:

* Click on ‘**Start A Project**’
* Enter a name for ‘**Repository Name**’
* Leave the project as ‘**Public**’
* Check ‘**Initialize this repository with a README**’
* Click ‘**Create a repository**’

**Note: Within the README, you can click on it in the repository and type whatever you want to describe the project.**

To clone a repository:

* Click on the ‘**Clone or Download**’ button’
* Copy the Git url to the repository within the text field
* Within your terminal program ‘cd’ (change directory) into the directory on your  
   computer, in which you would like to clone the repository.
* Run ‘**$ git clone** “**url to repository**” **folder-name (optional)**
* Copy your files to the newly to newly created local git reposity folder on your   
   machine

**Note: If cloning someone else’s repository and you want to edit/update the files without altering the original; run $ git remote remove origin. To add the cloned files to your own repository, run**

**$ git remote add origin** [**https://github.com/your-depository/folder-name.git**](https://github.com/your-depository/folder-name.git)

**One quick note about cloning GitHub projects from the links provided throughout this course. If you wish to make commits and push code up after cloning down from any of the GitHub links, you should actually fork the project first. The reason for this is because the GitHub project links provided are pointing to repositories you do not own, in order to own the repository, you have to fork it so you get a copy of the project in your own GitHub account that you do own!**

To get the status of modified files on your local machine:

* Run ‘**git status**’. This will display of modified files in **red**; which needs to be  
   ‘added’ for tracking

To add files to be tracked:

* Run ‘**$ git add**’ + **filename** or **folder** or ‘**$ git add -A**’

To commit the modifications of the tracked files:

* Run ‘git commit -m ‘**add a descriptive message here**’
* If account validation is returned run:
* **$ git config --global user.email** ‘**you@your email.com**’
* **$ git config --global user.name** ‘**repository name**’
* Re-run the ‘**commit**’ script

To push your commit files to your GitHub repository:

* **$ git push**

To get the latest version of any changes:

* **$ git pull**

**Branching**

In real-world situations, use ‘branching’ to make edits/changes to files for approval before

merging them to the master branch.

To create a new branch, make changes and push:

* **$ git branch NewBranchName**
* **$ git checkout NewBranchName**
* Make your changes
* **$ git add .**
* **$ git commit -m ‘your message’**
* **$ git push**

From within GitHub, you can view branch changes under ‘Your recently pushed branches’. To view updates and changes, click the green button, ‘**Compare and pull request**’. From here you can enter a message to alert other users to open a pull request to view your updates for approval.

Under ‘Open a pull request’:

* Enter your message to the user(s) to approve changes.
* Click the green “**Create pull request**” button

As an approver, to view changes and merge pull requests from within GitHub:

* Click on the **Pull Request** tab
* View changes/updates
* If approved, enter a message and click the green ‘**Merge pull reques**t’ button
* From within the terminal, on the Master branch, run **$ git pull** to get the latest updates

### Deploying Applications

* Within Github, clone the repository.
* Within the terminal run ‘$ git add remote origin ’ + the copied clone url
* Within ‘package.json’, above dependencies, add ‘homepage’:
* “homepage”: “<https://desean-ward.github.io/monster-rolodex>
* Add two scripts under ‘scripts’:
* “predeploy”: “npm run build”
* “deploy”: “gh-pages -d build”
* Within the terminal:
* run ‘npm run deploy”
* git add -A
* git commit -m ‘adding files for github pages’
* git push