GIT Commands:

* **cd ‘directory’**: navigate
* **git config --global user.name “username”**: add profile
* **git config --global user.email email**: add email
* **git clone URL**: clone URL to current directory
* **git status**: check directory
* **git add filename.ext**: stage changes ready for commit
* **git add** **.** : everything
* **touch ‘filename.ext’**: create new file from git
* **touch .gitignore**: create ignore file. To use in this file:
  + **\*.log** -> will ignore all .log files
* **git init**: setup repository
* **git commit:** commit changes (Using vim)
  + Press **I** for insert
  + Name commit
  + **Esc**
  + :wq > enter
* OR **git commit -m ‘commit name’**: quick commit
* OR **git commit -a -m ‘commit name’**: quick add of current tracked files and commit
* **git config –global core.editor “vim”**: change commit editor to vim
* **git log**: view commit history
* **git branch MyBranch**: create branch MyBranch
* **git checkout master**: use master branch
* **git checkout MyBranch**: use MyBranch branch
* **git merge MyBranch**: merge to destination branch (ensure destination branch checked out) the MyBranch (source branch) changes
* **git checkout -b old-project-state hash**: will create branch of an old version **hash**
* **git push -u origin master**: upload master branch (branch name) recent commits to ‘origin’ url (remote name)
* **git merge –no-ff MyBranch:** No fastforward ensures a new commit object is always created without fast-forward, avoiding lost information about the historical existance of a feature branch. Look into next merge.
* **git rebase master**: when child branch checked out this will update child branch to current version of master while keeping child branch unmerged changes also
* **git branch -d <feature-branch>:** to delete <feature-branch> when master checked out
* **git checkout -b <new online branch> origin/<new online branch>:** checkout remote branch not on
* **git push origin –delete <branchToDelete>:** deletes branch from remote repository