# Using Terminal, Git and Github

**What you need**

- Access to a terminal
- Access to VSCode (and you can also use terminal in that) or another code editor

**Open terminal on VSCode, Gitpod or CodeAnywhere**

Mac: Press `control` + `` ` ``
PC: Press `Ctrl` + `` ` ``

**Pushing and pulling from the correct repo**

First, you should add an upstream branch - this is the ORIGINAL repo that all the other forks are being merged into. That means the upstream branch will **always** have the most up-to-date code. When you pull from the upstream branch, you are using the most up-to-date code and will not cause a merge conflict unless two people are working on the same part of the code. To set the upstream branch, you should run the command `git remote add upstream [remote url]` - if using GitPod/CodeAnywhere, this may be set up already you can type `git remote -v` to confirm.

You should now be able to pull from the original repo's main using `git pull upstream main`. It's good to pull frequently to ensure you have the most up-to-date branch.When working on a feature, you should create a new branch for the feature. Normally, looks something like this:

1. While on the main branch, `git pull upstream main` to get the most recent version of the main branch.
2. From `main`, `git checkout -b [name-of-new-feature-branch]` to create (and switch to a new branch)
3. Carry out some work on the branch
4. Commit your changes to your feature branch using `git commit -m "[short description of change]"`
5. Push your branch using `git push origin [name-of-new-feature-branch]`
6. Open the Pull request on Github where someone will review it and resolve merge conflicts
7. Merge approved branch into main
8. Locally, `git checkout main` to switch back to the main branch

9. Repeat steps 1-8.

**Basic Terminal Commands**

Anywhere with `[something]` means that you add the name of the file/folder you want to navigate to - you don't need to add the `[ ]` around it.

| | |
|---|---|
| Show files in current folder | `ls` |
| Go back/up a folder | `cd ..` |
| Change to another folder | `cd [name of folder]` (you can use tab to autocomplete the same of the folder) |
| Clear the terminal | `clear` |

**Basic Git Commands**

| | |
|---|---|
| Show which files have changes made to them and if they have been "staged" to be committed | `git status` |
| Pull the most recent version of the repo to your local project | `git pull upstream main` |
| Stage your commits - any files with changes that are staged will be included in your commit | `git add .` (`.` means it selects all the files with changes)<br>`git add [name of file]` |
| Commit (save your changes) to the branch - these are the changes that will be pushed later | `git commit -m "[your message that describes the changes]"` |
| Create a new branch | `git checkout -b [name-of-branch]`<br>The name of the branch does not have any quotation marks around it and it should not have spaces. You should use hyphens (-) instead. |
| List the branches that exist | `git branch` |
| Switch to another branch | `git checkout [name-of-branch]` |
| Pushing a branch<br><br>The **first time** you push a branch, you will want to add `origin` to the command so that it creates it on the repo. | `git push origin [name-of-branch]` |

| | |
|---|---|
| Pull a branch from the repo to work on it locally. | `git push origin [name-of-branch]` |
| If you accidentally start making changes on the wrong branch, you can carry over all your changes by using this command | `git switch -c [name-of-branch]` |