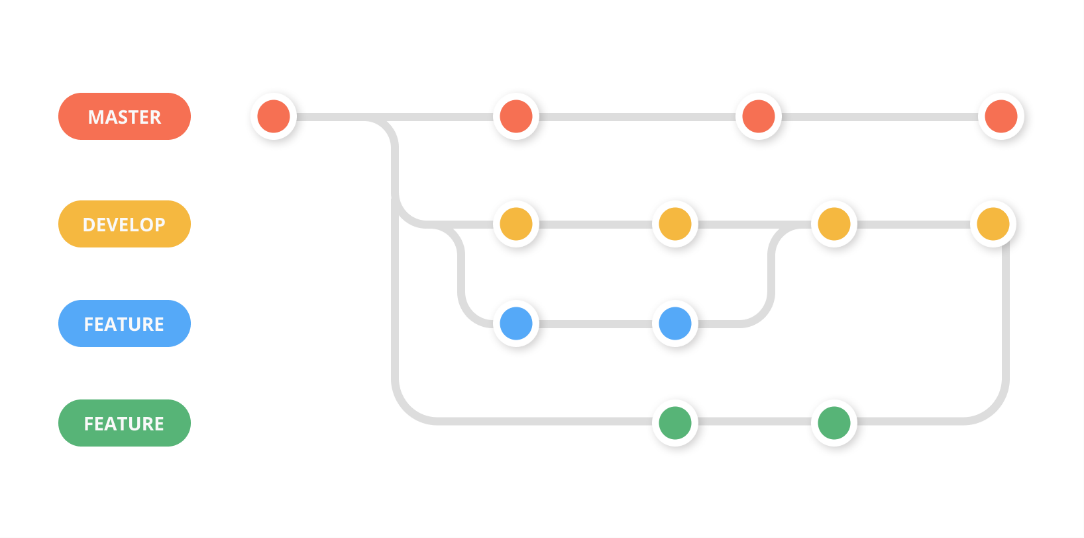
# Git and GitHub Guide

## How we will use GitHub

I will have set up the setting in the original unity project so they should work with GitHub, but if you have any problems related to this please let me know.

We will be using branches to control our project on GitHub, the structure of these branches will be similar to shown below.



The master (liklely to be called “main”) branch will contain the verison of the game produced from the previous sprint. Any work being done within the current sprint will be done from the development branch, each time someone is working on a specific feature they should always create a branch from the development branch, then create a pull request to the development branch. *I will explain this more with an example in the next meeting…*

## Installing Git

Before using GitHub you need to install git on your system, if you haven’t used it before

1. Download the correct version for you system at: <https://git-scm.com/download/win>
2. Click through the installation window, you shouldn’t need to change any of the settings

## Using GitHub for the first time

1. GitHub have made a desktop app which makes using GitHub a lot easier although you will still need Git installed, you can download the GitHub app here: <https://desktop.github.com/>
   1. If for some reason you can’t download it, I can talk you through how to use GitHub without the app
2. The first thing you need to do is clone the GitHub repository which I have set up.
   1. The URL you need to clone the repository is: <https://github.com/MarcusOWilliams/Software-Engineering-CW2.git>

* 1. Open the GitHub desktop app and click the “Clone a repository” option, on the URL tab, paste the URL set the local path to the directory where you want to save the project
  2. The project will be in a sub folder, you should now be able to open the project in unity

## Using GitHub when making changes throughout the project

1. **Before you start working on a new feature you should always pull/fetch the most recent version of the development branch from the repository**
   1. Make sure you are on the development branch and press fetch origin at the top of the GitHub app, if this then says “pull” then you should press pull



1. You should then create a new branch before you start working on a feature, give the branch a clear name to show what changes it makes

Graphical user interface, application

Description automatically generated

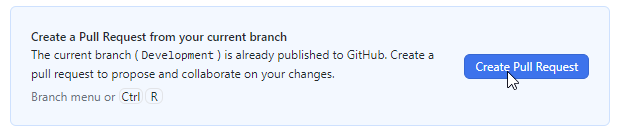
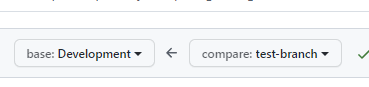
* 1. You should make this branch based on the development branch

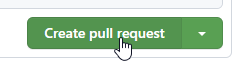
Graphical user interface, text, application

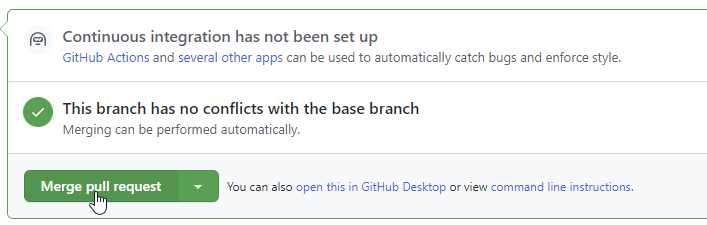
Description automatically generated

1. After creating your branch you need to publish your it, this creates it on the GitHub repository, click publish branch either at the top of the screen or the window in the centre.
2. After you have made changes to the unity project, or documentation, **make sure all of your changes are saved**, all changes should appear in the left hand side of the GitHub app.
3. Once you are ready to submit your changes press commit at the bottom of the changes column, make sure to give it a summary (just a very brief description of what you did)

****

1. Whenever you make a commit you then need to push it to origin, this just uploads your changes on the branch to the online repository.
2. You can commit and push changes multiple times throughout a development session, I would suggest doing it whenever you get to a suitable stopping point for anything you implement.
3. Next you need to create a pull request to merge your changes with the development branch, this will open up the remote (online) repository.
4. **It is important to make sure you change the pull request to the development branch**.
   1. You can then press create pull request, after updating the title/description if you want to



1. Hopefully the branch has no conflicts with the development branch so you can merge the pull requests. Sometimes there may be conflict, if multiple people have changed the same files, if this is the case, please let me know and I will try to sort it out ASAP.
2. Finally, when you are done for the session, let the team know that the changes have been made as they will need to pull the newest version of the development branch.
3. You can also delete the branch you created to add you feature (not the development branch!), this will help keep it tidy, you may need to delete it on both the remote and local repositories.
4. I am happy to do the rest of the updating and merging if the branches from this stage, I will update the main branch at the end of each sprint.