**What is Git:**

It is a version control tool that allows you create snapshots of your code, allowing you to be able to switch between different snapshots. This is quite handy if you, for example, have found out that your new updated code has introduced a vulnerability that can cause disruption to business operations. Having a previous snapshot to roll back on allows you to have the business operating again while you debug the new piece of code and update the code in order to patch the vulnerability.

It also has other features such as being able to see who was responsible for a certain version of the code, which keeps everyone accountable of their work.

I have already mentioned above on why git is important, but another reason is because (due to better management of code) companies in certain indurstries may need to show that their IT environments are compliant and in accordance to different regulatory bodies.

**What are branches:**

Branches are variations of the original code i.e. the main branch. It allows developers to edit the original code without affecting it. It also allows multiple developers to work on the original without there being any conflict. Once developers have updated the code, and it has gone through appropriate testing and approvals, the variation of the code can be used to update the original code.

Main Branch – The ‘live’ branch

Sub Branch – The ‘development and testing’ branch

**What is a ReadMe.MD file:**

This is essentially a file that gives an introduction and explanation of the repository and it’s content. It is useful for anyone viewing the repo in order for them to understand what it is about.

Workflow when making changes to repository/code:

Add – This will add the change onto your Local Repo (i.e. your computer), NOT your Remote Repo (i.e. The GitHub website)

Commit – A description of the change that you had made

Push – Pushes the change from LOCAL Repo to Remote Repo

Pull – Syncs the Local Repo and Remote Repo so that they are both are up to date

<quick note: If you push changes from Local to Remote Repo, on the SAME branch (i.e. main branch), it won’t require the request to be accepted on GitHub website.

If you push a change from a sub-branch to the main branch, it will come in the form of a pull request and will need to be accepted (on the GitHub website) before being merged into the main branch.>

**Commit best practices:**

* Commit often so it is easier to rollback code versions to more specific points when needed
* Commit descriptions should be clear and concise, mentioning WHY the change was made, and NOT what the change was
* Have one commit for each new/modified feature/function. Do not have multiple commits for each new/modified feature/function
* Test your changes before making a Commit
* Have tickets (i.e. on a ticketing system) raised for each commit so it is simple to refer to the ticket on details on commit if the need arises

**Merge Conflicts, what are they:**

These are essentially conflicting pieces of code between the Local Repo and Remote Repo. They appear when the sub-branch is based off an old version of the main branch (which can happen if you updated the main branch after creating the sub branch), resulting in a conflict when you attempt to merge the sub branch back into the main branch

**Squashing Commits:**

Combining commits together to form one commit. This is useful when you have made multiple commits for one feature and now you want to combine all of these commits into one single commit (if you go up to the best practices for making commits you can see this being one of the points).