Git Version Control Document

GitLab is one of the largest and most advanced code hosting platforms for version control and collaboration. It allows users to work together on projects from anywhere and manages different versions of an applications development.

In our development process we will be using GitLab for code sharing as well as ensuring that everyone works on the most up to date code to avoid continuity issues. A single repository has been created to organise our project under the schools GitLab. Here in this repository is where we are storing our files and will organise version control.

To do this we will be taking advantage of branching, this is the best solution to work on different versions of the repository at one time. A branch has been created called *'master'* this is the definitive branch which will contain completed versions of code. Each member can create a branch from the *'master'* branch to experiment, implement user stories, or make changes after testing before merging them with the *'master'* branch. When a branch is created it creates a copy of the project at that time, this is effective during the development process as if new lines of code are added which do not function as intended, they can be kept separate from the *'master'* branch to avoid further issues. Those issues can then be then corrected before committing and pushing back to the *'master'* branch. When updates are made to branches a pull request can be made which will get the latest code from the repository to ensure everyone is working on the most up to date versions of the project.

When a branch has been committed to the repository a pull request can be made for another member of the group to pull other members' work into their own as a new branch, proposing the changes that they have made and merging content from both branches into a single branch. This process continues as branches are created until branches are merged into the *'master'* branch which will form the final project files.

For our development process changes will be pushed to GitLab when significant changes have been made to warrant being viewed by another member of the group, or when a particular feature set has been implemented as is needed for testing purposes. For example branches from *'master'* will be created, such as *'login'* and *'view_bookings'.* Both of these branches correlate to the user stories that we are working on for the first sprint. When testing these methods, several branches will be created from these to add the test classes. Once the testing is complete each test branch is merged with the original branch for which its methods were tested and then that branch will be merged into master.

In one of our weekly scrum meetings we will be resolving issues relating to merge conflicts on branches that have been committed but not merged at that point. In our final scrum meeting for the first sprint, we will merge both the *'login'* and *'view_bookings'* back into *'master'* in preparation for the next set of branches which will be created to work on the next user stories in the second sprint.