

**Portfolio 6 Report**

**Student Name:** Ling Yee Kit

**Student ID:**  A4482

The Need for Version Control

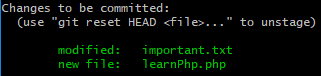
Imagine hundreds of people are working on the same project, a lack of management and coordination can have drastic implications on the project’s completion. Some might be working on something that already finished, some might be missed an important part of the project. There is more than one person in the organization so it has a high risk to create those file with a similar name or maybe exactly the same name, this unintentional mistake will lead to file path error or even the whole programme stop running. Besides, if there is no merge process for the project, the organization are easily overwriting each other’s file. If the file is overwritten by others, it might have a risk that cannot get back to the previous state of the project. This ends up slowing the project development time and spend even more labour to resolve the unintentional mistake. Software developers working in an organization are continually writing new source code and changing existing source code, the code is typically organized in a folder structure or file tree. Some developer may be working on a new feature while another developer fixes bug by changing code, each developer may make their changes in several parts of the file tree. By tracking and updating every changes made by the collaborator, it lowers the chances of making an error and effectively increases the productivity of the organization.

Version control systems are a software tool that helps a software team manage changes to source code over time. Version control systems keep track of every changes of the source code, every change that made to the source code is tracked, it keeps the data of who made the change, why they made the change, and when they made the change. Version control systems help each collaborator access to the most recent state of the project, it can work like an online database and it allowed the developer to upload any file to the repository and download the file from the repository at any time. Apart from that, there are no worries about making a mistake to the source code because visual control systems can get the project back to the previous state at any time. Version control systems ensure every collaborator understand what happened to the project. Whenever each collaborator saves a new version of the project, the version control systems require them to provide a short description of what was changed. Additionally, the collaborator can see what exactly was changed in the file's content. This helps every collaborator understand how the project evolved between versions and make the development process easier than ever.

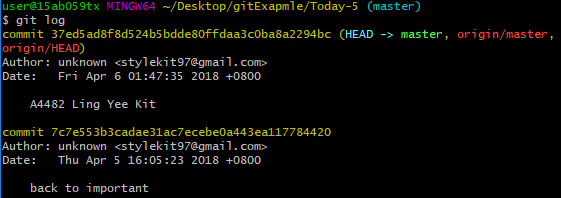
There are two types of version control systems: central or distributed. Git is the most popular example of a distributed version control system. To begin with git repository, user may cd to the project directory and run git init, this will add .git (contains the Git metadata for the new repository) subdirectory to the current directory. The current directory is now having a fully functional Git repository.

# Editing files and committing changes

The commands: git add, git status, and git commit are usually used in combination to save a project’s current snapshot. git add is the first command in a chain of operations that directs Git to save a snapshot of the project’s current state. Whenever the user is ready to save a copy of the current state of the project, use the git add command add the file to a staging area, this tell Git that user need to keep current version to the next commit. User can combine git status command to view the status of current directory and the staging area. File that in the staging area will display as diagram below:



Commits are created with the git commit command to capture the state of a project at that point in time. By typing git commit will open a text editor prompting user for a commit message. After user entered a commit message, save the file and close the text editor for actual commit. There is a shortcut command create commit and accept commit message at the same time: git commit -m “commit message”. User can use the git log command to display committed snapshots and project history, as diagram below:



# Interacting with remote repositories

Remote repository is located on a remote server on the internet or in your local network. The project teams are using remote repositories to share and exchange data. User can publish the file from local to remote repository by “pushing”, and see what others have contributed by “pulling” branches into local repository.

If user have only local repositoriy and no remote connection are save, user can visit to <https://github.com> to create one. Whenever a user clone a repository from a remote server, Git automatically remembers this connection, and saves it as a remote called "origin" by default. The git remote command allows user to create, vew, and delete the connection to the remote repositories. The remote is like a bookmarks, it saves the links connect into other repositories. Use git remote command to list the remote connection to other repositories. The git remote -v command same as git remote, but include the URL of each connection. The git remote add <name> <url> allows user to create a new connection to a remote repository. After adding a remote, user able to use <name> as a convenient shortcut to connect to the other repository.

# Merge conflicts and how to resolve them

M