**VERSION CONTROL SYSTEM USE REPORT**

* **Difficulties or challenges during development**.

*During the development process of Musoplan Software, after the initial challenge of getting used to the chosen VCS software which is Git and Github, as per organization requirements. There were no other challenges faced through the development process, as most of the commands are repeated steps and pretty straightforward to use.*

* **How have you complied with the organizational requirements?**

*One of the organization requirements for the VCS for the project requires a distributed version control system that most developers are familiar with and has a good online supporting ecosystem, to be utilized for all software development.*

*Also, another part of the requirements stated that the VCS must take place on feature branches, so the main branch is exclusively initialized and the merges. In other words, the VCS chosen must be able to support the branch features.*

*Git and Github met this requirement.*

* **How did you verify the VCS was performing as expected, behaviors you should test and describe include**
* *Branches: Two branches were created and used for the development of the software. ‘docs' and ‘jsfiles’. The ‘docs branch’ was used for the software documentation and the ‘jsfiles branch’ was used for the software JavaScript files.*
* *Stages: The three stages (modified, staged, and committed) were noted all through the software development process. When files were modified, added with git add ‘filename’ and committed to the branch.*
* *Commits: The commit scope was kept to a single change. That is when a new function was added, it was committed immediately, when a function or file was modified, it was committed immediately as well.*
* *Merges: Merge was used after the development of the software was completed. The two branches created were merged with the main branch.*
* *push/pull to share commits with remotes: using the ‘git push -u remote-name branch name’ command, I was able to push the commit from my local repository to my remote repository.*
* *pull requests: The pull request was used to ask our instructor to review changes that were made after a forked repository was forked from her Github account.*
* *commit log: The commit log was used to track the time and change of committed files*
* If your repository was forked from another repository, how would you go about having them integrate your changes to the central repository?

There are two ways to integrate changes made on a forked repository into the central repository. Using the command line with Git or on the Github website.

I used the Github website to perform this function. I used the ‘pull-request’ or the ‘reverse-pull request’ feature on the site.