### Version Control Software Evaluation

#### Overview

At the conclusion of the project you are to write a 300-word (minimum) report on the version control software you used to maintain backups of your project source code for the duration of your project. Your report is to cover:

* What version control software was used;
* An evaluation of its performance. This may cover responsiveness, user feedback, support, ease-of use, speed, or any other relevant measures of performance;
* An evaluation of the benefits of this specific version control software to a game studio or software organisation; and
* Identification of any environmental considerations.

#### Template

**Version Control Software Evaluation Report**

**Math for Games**

**<Blake Peck>**

**<29/03/2021>**

**Version Control Software:**

The version control software used was:

**The version control software is Git.**

The client program was:

**Client program is GitKraken**

**Performance:**

List the performance criteria. For example, how easy the software was to use; how easy it was to integrate into your daily workflow; or how responsive the user interface was.

**Using GitKraken was a bit hard to understand for the first time using a version control. But it very easy to learn how to use it with the amount of help it gives to understand how to use it better. It very simply to make a new repository and clone it to a location that best suits you. It was quick easy to start using GitKraken now every day did not ruin the workflow but made it better as it help look over my work and see what I have done over the days working on assessment that I may have forgotten about doing.**

Measure the performance of the software against the performance criteria. This may be a subjective evaluation (for example, ‘I feel the user interface was not intuitive’).

**I feel the management of files that are put in the repository are organises very well, keeping the files together and not making a mess having trouble with searching for a file. Starting to use GitKraken every day came with the user interface being confusing for me to understand at first with it being hard to learn at first, being not nice to newcomers. When it comes to the cloning of the repository and having multiple clones of it was difficult to use, by having files not transferring over which could be cause of not one button pressed. Making that not doing one step hard to work on the same repository hard.**

You may wish to identify any specific features or aspects of the software that significantly affected your evaluation of its performance (i.e., what were the specific good or bad points).

**The part about cloning allowing you to work on it on different computers are helpful as the files are saved on all computer when saved to GitKraken. This is much better than having a USB which leads to having to copy the files on a it to move it to the other computer which could mess up files. Worse situation you lose the UBS so you cannot transfer, compared to GitKraken it stored online.**

**Evaluation:**

What are the benefits (if any) for organisations? Would this software be useful in a game development environment? Give your opinion of the software.

**The main benefit that I say is allowing multiple users to work on the repository at the same time and not cause problems if there are more than one person working on the file. GitKraken software world 100% be useful in a game development environment as users can branch off working on there own local, write long description about what you just saved to GitKraken so other can read, and bring all branches together into a single branch that holds all the work.**

**Environmental Considerations:**

What environmental considerations should be considered when integrating this technology into an organisation or project?

**The more people using GitKraken will cause the use of USB being use lower the chance of computers being destroyed by the USB overheating in the port. Which would lead to throwing out the computer onto a landfill where the chemicals pollute the soil and enter the groundwater contaminating it.**

Consider an organisation implementing a self-hosted version control server. You may want to research and discuss:

* Power usage

**The power usage will be a lot as the organisation will been having to hold up to millions of users and their severs that they make.**

* Cost (environmental or otherwise) of hardware end-of-life

**Up to a year or three maybe more is when you will need to upgrade the hardware as it will be better and the support for the old hardware will stop. Meaning lots of hardware is thrown out and hardware made that will go in a cycle of throwing out and making.**

* Cooling costs associated with servers

**The cooling costs will increase depending on the size of the sever room and the temperature of the room**