**ASSIGNMENT – GIT TUTORIAL**

**What is GitHub? When was it created? Why? By who? What similar platforms exist? Why would you use such a platform?**

GitHub is a repository hosting service. It can be described as the "cloud" for code. GitHub will host your source code projects in a variety of different programming languages and keep track of the various changes made to every iteration. It is able to do this by using Git, a revision control system that runs in the command line interface.

The development of GitHub began in October 2007 and it was launched in April 2008 by Tom Preston-Werner, Chris Wanstrath, and PJ Hyett.

There are plenty of GitHub alternatives if you’re looking for distributed version control software (DVCS). Some reasons for looking at GitHub alternatives include: Your team needs the security and privacy of an in-house system, you have a large team, and GitHub’s pricing doesn’t fit or you work in a version control language other than Git. A few examples are:

1. Bitbucket
2. GitLab
3. DevHub
4. Beanstalk
5. SourceForge
6. Apache Allure
7. Cloud Source by Google
8. AWS CodeCommit
9. GitKraken

Using GitHub has numerous benefits including easier collaboration with colleagues and peers, ability to look back on previous versions, and tons of easy integration options.

With an emphasis on speed, data integrity, and excellent support for distributed, non-linear workflows, Git is the gold standard for remote digital teams. Also, Git is convenient for working offline or without a VPN (virtual private network), making it easy for developers to work on the go and stay productive. For teams rapidly building sites and prototypes for customers, GitHub is an efficient, safe and seamless way to get projects reviewed, approved, and signed off on.

**Define the following terms in the context of Git:**

• Repository - A repository is simply a place where the history of your work is stored. It often lives in a .git subdirectory of your working copy - a copy of the most recent state of the files you're working on.

• Commit - Stores the current contents of the index in a new commit along with a log message from the user describing the changes.

• Push - Push the specified branch to <remote>, along with all of the necessary commits and internal objects. This creates a local branch in the destination repository.

• Branch - A branch represents an independent line of development. Branches serve as an abstraction for the edit/stage/commit process

• Fork - A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

• Merge - Merge the specified branch into the current branch. Git will determine the merge algorithm automatically.

• Clone - Clones a repository into a newly created directory, creates remote-tracking branches for each branch in the cloned, and creates and checks out an initial branch that is forked from the cloned repository’s currently active branch.

• Pull - Fetch the specified remote’s copy of the current branch and immediately merge it into the local copy.

• Pull request - pull requests are a mechanism for a developer to notify team members that they have completed a feature.

**Git Tutorial**









































