**CS 623 Summer I**

**Exercise on GitHub and Git**

In this project, we will be using GitHub for assignment submissions and code versioning.

The goal of this exercise is to get you started with Git and GitHub. Even if you are using Git and GitHub regularly you need to do this exercise and submit it. If you already did it, you need to create the required repository and fill out the required documents.

Please follow the instructions completely. Work that does not follow the instructions (including naming conventions) will NOT be accepted and will result in a grade of 0.

While there is a GUI to use Git and GitHub, developers use the command line!

**Part 3:**

**Answer the following questions.**

**(Answer between 2 and 3 lines)**

Answer these questions in a Word file called *LastnameFirstnameGitTutorial-mm-dd-yyyy.docx*. Please respect the naming conventions!

**What is GitHub?** GitHub can be defined as a software development platform. It allows developers to host and review their codes and manage projects alongside millions of other developers.

**When was it created?**

GitHub was created in 2008.

**Why?**

GitHub’s purpose was to make it easier for all coders novice or experience to use Git for version control and collaboration, hence its super user-friendly interface.

**By who?**

GitHub was created by Chris Wanstrath, P. J. Hyett, Scott Chacon and Tom Preston-Werner.

**What similar platforms exist?**

A list of similar platforms would include but is not limited to Beanstalk, Bitbucket, CodePen and SourceForge.

**Why would you use such a platform?**

I would use GitHub to store my projects and get help as I work on my projects. For instance, I can connect with some of my classmates, view their codes (their projects) and communicate with them on how to better my work.

**Part 5:**

Become familiar with the following terms:

* **Repository**

A directory or storage space where your projects can live. Sometimes GitHub users shorten this to “repo.” It can be local to a folder on your computer, or it can be a storage space on GitHub or another online host. You can keep code files, text files, image files, you name it, inside a repository.

* **Commit**

This is the command that gives Git its power. When you commit, you are taking a “snapshot” of your repository at that point in time, giving you a checkpoint to which you can reevaluate or restore your project to any previous state.

* **Push**

If you’re working on your local computer and want your commits to be visible online on GitHub as well, you “push” the changes up to GitHub with this command.

* **Branch**

Working with multiple collaborators and want to make changes on your own? This command will let you build a new branch, or timeline of commits, of changes and file additions that are completely your own. Your title goes after the command. If you wanted a new branch called “cats,” you’d type git branch cats.

* **Fork**

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Most commonly, forks are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea. A great example of using forks to propose changes is for bug fixes. Rather than logging an issue for a bug you've found, you can fork the repository, make the fix and submit a pull request to project owner. If the project owner likes your work, they might pull your fix into the original repository

* **Merge**

When you’re done working on a branch, you can merge your changes back to the master branch, which is visible to all collaborators. git merge cats would take all the changes you made to the “cats” branch and add them to the master.

* **Clone**

When you create a repository on GitHub, it exists as a remote repository. You can clone your repository to create a local copy on your computer and sync between the two locations.

* **Pull**

If you’re working on your local computer and want the most up-to-date version of your repository to work with, you “pull” the changes down from GitHub with this command.

* **Pull Request**

Pull requests let you tell others about changes you've pushed to a GitHub repository. Once a pull request is sent, interested parties can review the set of changes, discuss potential modifications, and even push follow-up commits if necessary.