Introduction/Purpose

The purpose of this guide is not to be an all encompassing guide to Git bash, but as a quick reference for those that are either new to Git or those that just need a refresher. In this guide we will cover some commands that come in handy when using Git bash. These commands include some tips on how to navigate your file system using Git bash as well as how to make a commit to a remote repository.

Navigating your File System

We feel that it is pretty safe to say that pretty much everyone is capable of navigating a file system using a Graphical User Interface(GUI). In fact, this is the exclusive method of navigation for many people. To make full use of the Git bash we will be required to learn how to use commands in the Git bash shell to navigate the file system on your computer. Luckily for us Git bash shares many of the same commands as the Linux shell. This will make our lives much easier. We will make a list of the commands necessary along with a short description of their function.

- **Pwd** - This will print the working directory, the folder that you are currently in
- **Cd** - Changes the directory to one of your choosing
  - Now for some tips for using this command
  - Typically when using cd you need to include the entire file path for example:
    Cd /root/documents/pictures
    This will change your directory to picture directory but what if you were already in the documents directory it seems kind of redundant to copy the entire file path luckily you it is not necessary to have the whole file path instead use:
    Cd ./pictures
  - Sometimes you may want to go down a level in your file system to do this without having to write the whole file path use:
    Cd ..
  - To move all the way back to the root folder of your file system you can use the command:
    Cd ~
- **Touch** - This command will create a file with the name of your choosing in the current directory. For example:
  Touch python_script.py
- **Mkdir** - This command will make a directory with the name of your choosing in your current directory. For example:
  Mkdir New_directory
- **Rm** - This command will remove a file or directory. For example:
  Rm New_directory
- **Cp** - this command will copy a file or directory from one destination to another. For example:
  Cp ./copy_from_here ./copy_to_here

Using Git

Now that we have covered how to use the Git bash to navigate your file system we will cover on how to incorporate the Git Distributed Concurrency Control system. In this section we will cover just the bare essentials to get a remote Gitlab repository working. Before we begin there are two things that we want to mention. First make sure that you have a Gitlab account setup. To do this go to the Gitlab website and follow the onscreen directions. Second thing that we wish to mention is that for Git it is highly recommended that you create a folder just for your Git repository. We will cover the sequence of commands that are necessary to start working with a remote Gitlab repository.

1. Change directory into the directory that you created for your Git repository and enter the command:

   ```
   Git init
   ```

   This will initialize your repository
2. Follow the instructions in the following link to create an SSH sign in. Doing this will allow you to bypass having to enter your username and password every time you want to make a commit.

   http://docs.gitlab.com/ce/ssh/README.html
3. Now it is time to add the remote server to your Git repository that you created. To do this go to the project home page on Gitlab and copy the SSH information provided. Once you have done this go into the Git Bash and type in the following command:

   ```
   git remote add HI git@gitlab.com:HISummer2016/DataAnalytics.git
   ```

   In the command above the HI is what we have named the remote repository, you could choose to name it something else.
4. Now that we have added the remote repository we now want to know how to make commit changes to it. We will cover the sequence to commit changes to the remote repository.

   ```
   git add Your_directory
   ```

   We recommend making a directory for the files that you are working on. In this way you can add all of your work by adding the directory.

   ```
   git status
   ```

   This will list all of the files that are staged and ready to be committed.

   ```
   git commit -m “Commit message”
   ```

   If you leave of the **-m** option you will be taken to the VI editor to escape this type in **:x** to save and exit alternatively you could use **:q!** to exit without saving. This commit is to your local repository to sync to the remote repository we need to run the following:

   ```
   git pull HI branch_you_want
   ```

   Doing this will update any changes that have been made to the master branch by other users. This step isn’t always necessary for example if you were the last one to make a commit. If your local repository is not synced with the remote repository it will not let you make changes, if this is the case then just use this command

git push HI branch_you_want
This is the last step and this command will commit your changes to the remote repository.