Git – 5 commands to success

Table of Contents

[Command 1 (and 1.1) 2](#_Toc21191560)

[init & cd 2](#_Toc21191561)

[Command 2 4](#_Toc21191562)

[Pull 4](#_Toc21191563)

[Command 3 (and 3.1) 5](#_Toc21191564)

[Add 5](#_Toc21191565)

[Command 4 7](#_Toc21191566)

[Commit 7](#_Toc21191567)

[Command 5 8](#_Toc21191568)

[Push 8](#_Toc21191569)

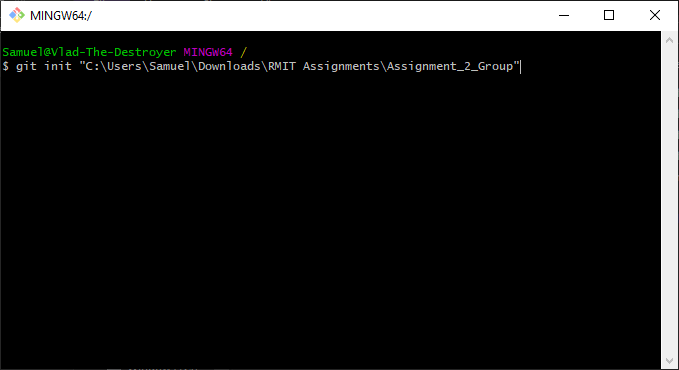
# Command 1 (and 1.1)

## init & cd

There are two things to do when first starting with git.

1. The command git init “<your directory path to the local repository>” (note the double quotes)

**Example:** git init “C:\Users\Samuel\Downloads\RMIT Assignments\Assignment\_2\_Group”

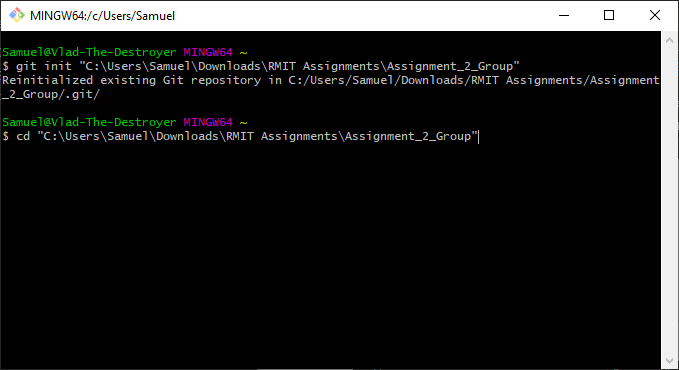


This initialises the local repository (repo) – basically, it creates the relevant files in a hidden folder at the directory location you specified, so git will work.

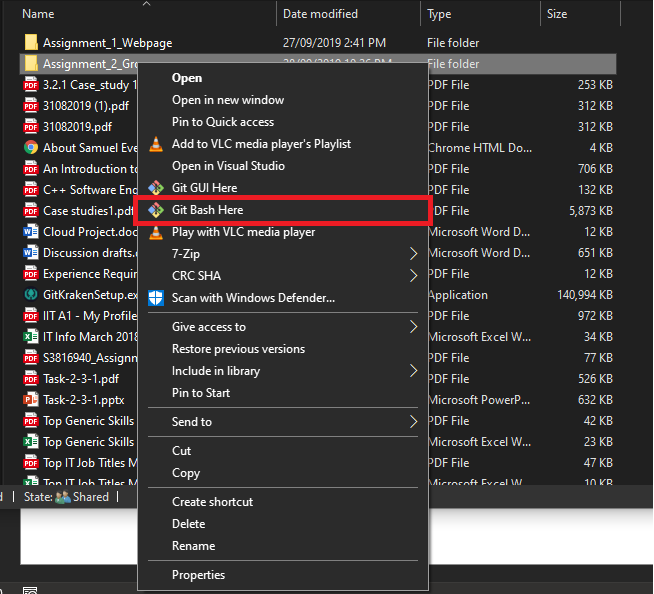
You only need to do this the very first time you start a local repo. The next time/s you access this local repo you will simply need to ensure your Git Bash console is set to the correct directory (covered next!).

**Note:**   
*You can choose anywhere you want on your computer to contain your project and initialise git to, just make it somewhere you remember!*

1. Set the directory path to your local repo in the git bash console – There are two ways to do this:
   1. Type cd “<your local repo pathway here>” (note the double quotes)
      1. **Example:** cd “C:\Users\Samuel\Downloads\RMIT Assignments\Assignment\_2\_Group“



* 1. Locate the folder in windows explorer or mac finder, right click the folder and click ‘Git Bash Here’.



Now your local repository location is selected. Whatever we do next, this is the location that will be affected. Any files in this location will now be managed by git so any changes made, additional files created, or files removed etc. will be tracked.

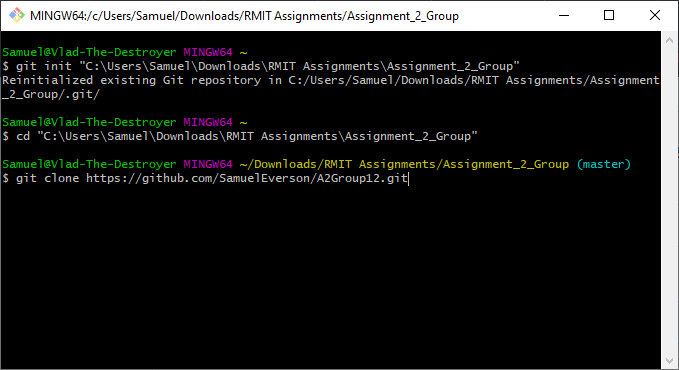
# Command 2

## Pull

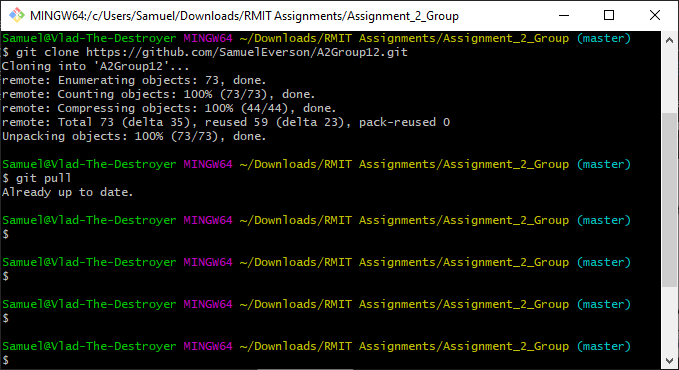
Now that we have initialised our local git repo and set the path in git bash, we can pull a repo from the web, such as a GitHub repo.

We first need to set up our repository (repo) location to push and pull from.

To do this you can get the URL from the remote repo and use it in the command git clone, e.g.;  
 git clone https://github.com/SamuelEverson/A2Group12.git



Once that is set up simply type ‘git pull’ and it will pull or ‘download’ the files stored in the GitHub repo.



Remember these files are as up to date as the latest push from each user (which we will cover shortly).

Now you’ve done a pull command you will also notice if you navigate to the folder in windows explorer or mac finder, any files updated with the pull will be available (hence why I sometime refer to a pull as a download).

# Command 3 (and 3.1)

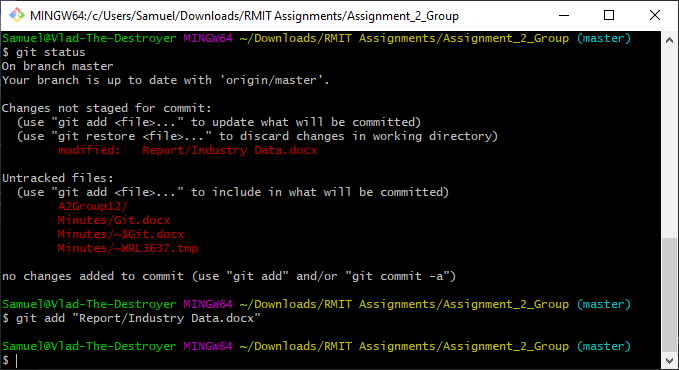
## Add

So, we’ve set the git bash to the correct directory and pulled the latest repo from GitHub. Now assume we’ve made some changes to a document and it’s time add the changes to the ‘staging’ area – in other words we are going to make git recognize these changes are going to take effect.

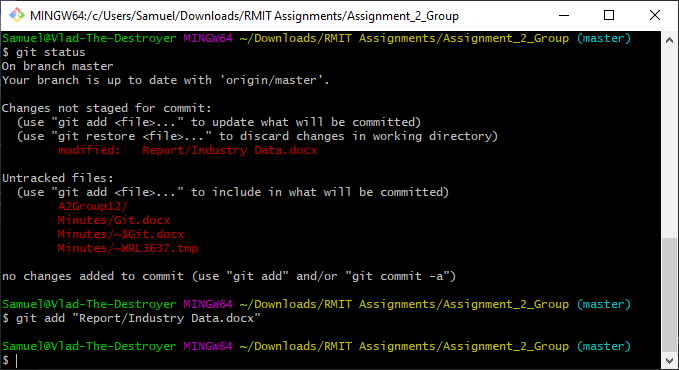
**Command:** git add <file name or wildcard>

**Note:**  
Command 3.1 would be git status. Git status shows you what files in your local repo either have changes and are ready to be added to the staging area OR which files have already been added.

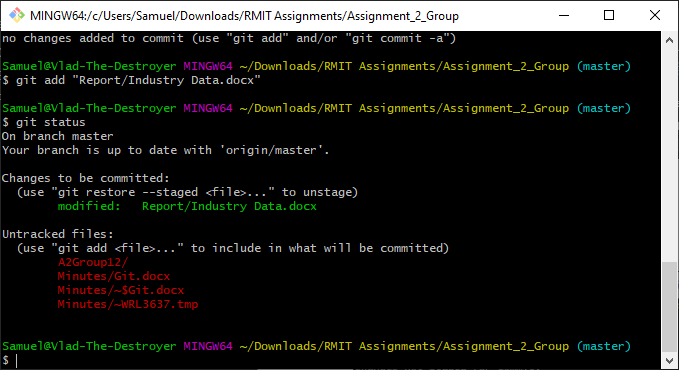
First I like to check what’s happening in the staging area with git status:



You can either type the specific file name/s you want to add to the staging area or you can use wildcards, for example, git add \*.docx will add all files with the .docx file extension (regular Microsoft Word document) that have changes (including new and removed files). Another example is simply git add . which will add all files with changes in the directory regardless of file type.



**Note** there is no confirmation that the file is added but we can input another status command to see the changes:



Notice how there is now a green text for ‘changes to be committed’ which is the file we just added!

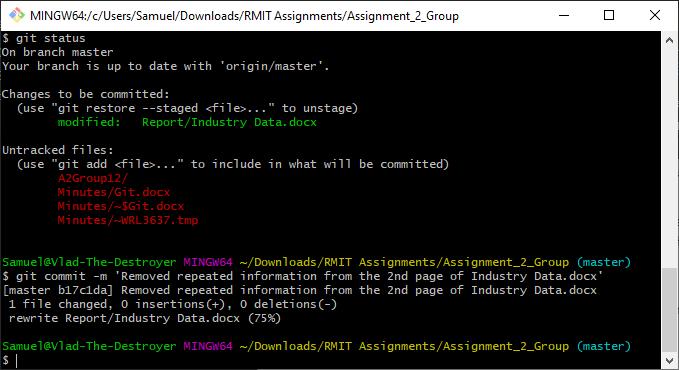
# Command 4

## Commit

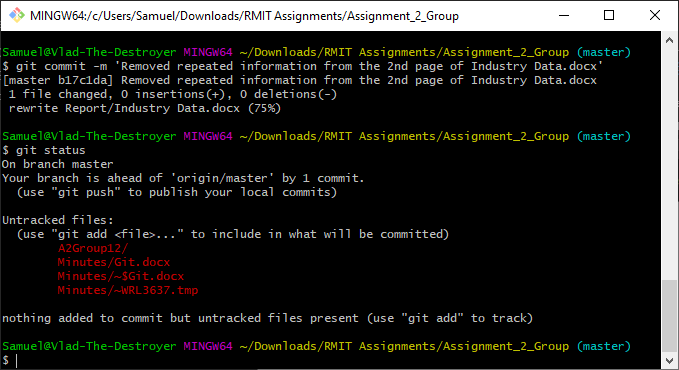
Git commit simply commits to the changes made and updates the files in git, which will then allow us to push the changes to the remote repo (GitHub).

If you just type ‘git commit’, a notepad window will open which allows you to leave notes regarding the changes made – think of it as being able to add a change log entry each time you commit to a change in one or more files.

To make things a bit easier (unless you have a big note to leave) you can add the option -m <insert note here> to add your note in the command which prevents the notepad window opening – the note is still added all the same!



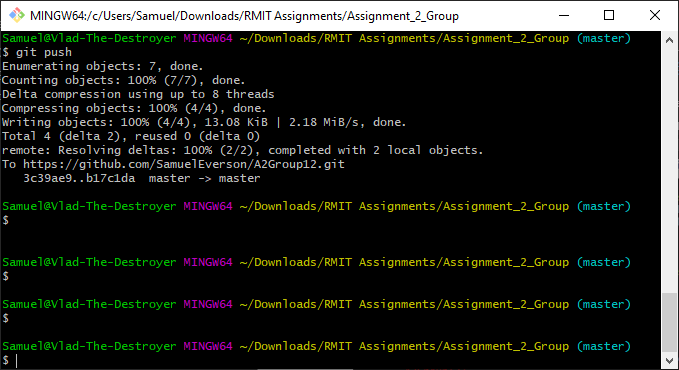
If we look at git status again now, we will see our local repo is ahead of the remote repository (GitHub) and thus is ready to be pushed (uploaded) or published to the remote repo.



# Command 5

## Push

Finally, as mentioned earlier, git push will publish or upload your changes to the remote repo. This them makes your changes visible to other members of the remote repo (or anyone that accesses the files). For other users to see your changes, they would need to pull the repo again, after you have pushed your changes that is.



And that’s it!

5 (and a few other) commands to master git!