

# git cheat sheet

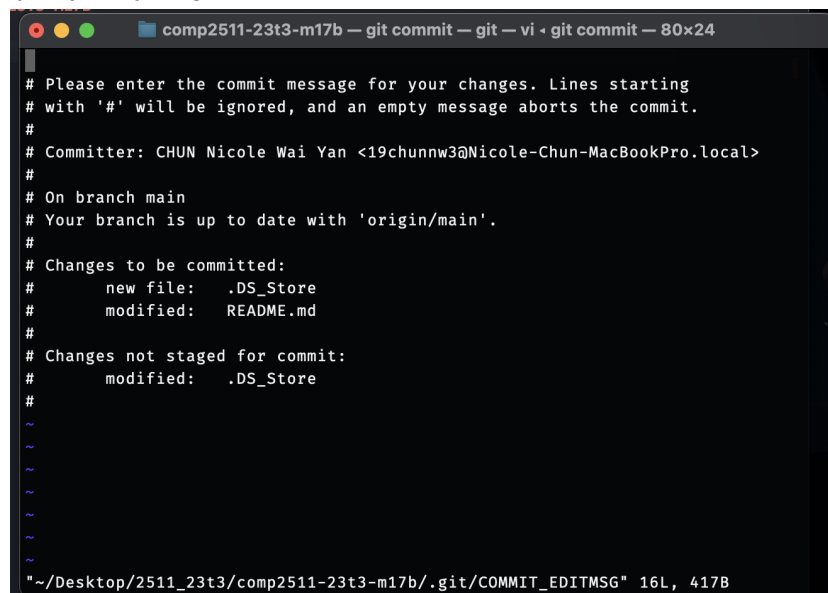
Also useful: [1511 Linux command cheat sheet](#)

<code>git clone [link]</code>	On the page of the repository you want to clone, click the 'clone' button in the top right corner (blue). I don't think it matters too much if you clone with SSH or HTTPS.
<code>git pull</code>	Fetches the most recent version of our work, recommended to do every time before you work on something. And also pull frequently, that's probably a good idea too.  <code>git pull</code> only takes stuff from master, if you want to pull anything from a specific branch use: <code>git pull origin [branch_name]</code>
<code>git status</code>	Shows you (in red) the files that you have made changes to but not yet updated on gitlab. You do not need to add/commit files that are green.
<code>git log</code>	Shows commit history, but in your terminal! Type 'q' to return.
<code>git add</code>	You can also use <code>git add --all</code> OR <code>git add .</code> To add all files in your current directory, but it isn't recommended because you might add files that you don't want.  Make sure you're in the correct branch !!  Also, we never really have to add the node modules folder (which is why <code>git add --all</code> is a bad idea) (it should usually be <code>git ignored</code> tho)
<code>git reset [file name]</code>	Un-adds a file that you have committed. There are other variants of <code>git reset</code> , but I wouldn't recommend using them unless you are very sure of what you are doing or don't mind accidentally purging your work.
<code>git commit -m "[commit message here]"</code>	Commits your changes – your commit message should be descriptive of what changes you have made.  <i>** check bottom of doc for how to exit vim</i>

<code>git push</code>	<p>Uploads stuff to gitlab, make sure to click the link in the terminal to create a merge request</p> <p>Sometimes terminal will output:  fatal: The current branch  branch_name has no upstream  branch.  To push the current branch and  set the remote as upstream, use</p> <pre>git push --set-upstream origin branch_name</pre> <p>In which case just copy the <code>git push --set-upstream origin branch_name</code> and it should work</p>
<code>git checkout -b "[branchname]"</code>	Creates a new branch called whatever you name it
<code>git checkout [branchname]</code>	Moves you to your desired branch
<code>git branch</code>	Lists all the branches. Use 'q' to return.

## Exiting vim

This screen may pop up if you forget to add a closing quotation mark (" ") to your commit message, or if you just type git commit.



```
comp2511-23t3-m17b — git commit — git — vi · git commit — 80x24
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Committer: CHUN Nicole Wai Yan <19chunnw3@Nicole-Chun-MacBookPro.local>
#
# On branch main
# Your branch is up to date with 'origin/main'.
#
# Changes to be committed:
#   new file:   .DS_Store
#   modified:   README.md
#
# Changes not staged for commit:
#   modified:   .DS_Store
#
~
~
~
~
~
~/Desktop/2511_23t3/comp2511-23t3-m17b/.git/COMMIT_EDITMSG" 16L, 417B
```

Basically doing this will open a vim editor to edit your commit message. If you know how to use vim that's cool! But to quit vim, press 'esc'/'escape', then type `:q!`