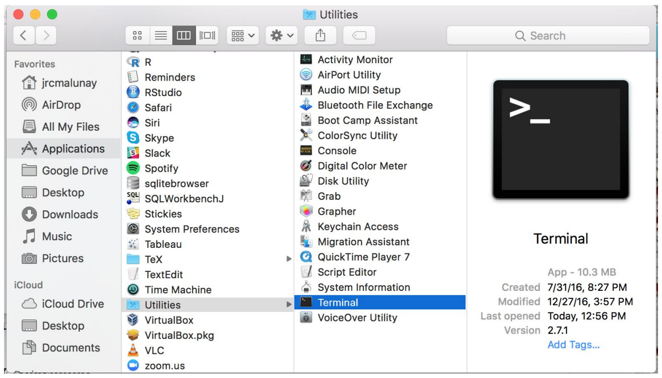
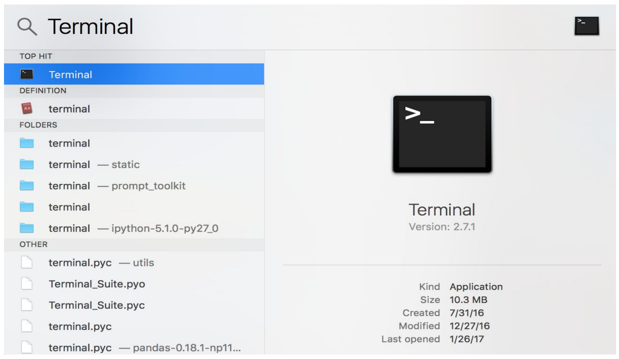
Part 1: the Terminal

Terminal: is the application that allows you to navigate through your computer without using your mouse. You can open the Terminal by either:

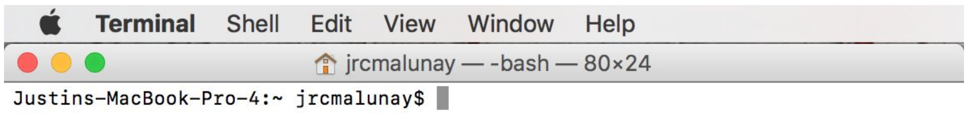
- opening Finder, then click Applications fromt the sidebar, then open Utilities, then open Terminal



- holding Command key then press the Spacebar on your computer, then you can simply type Terminal and press Return

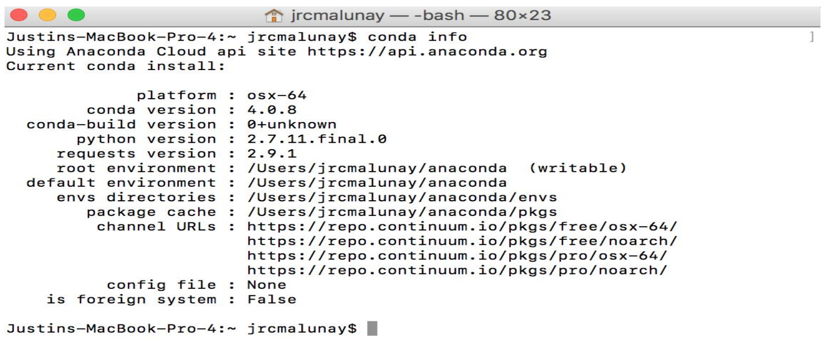


Once you open the terminal, your screen should now look similar to:



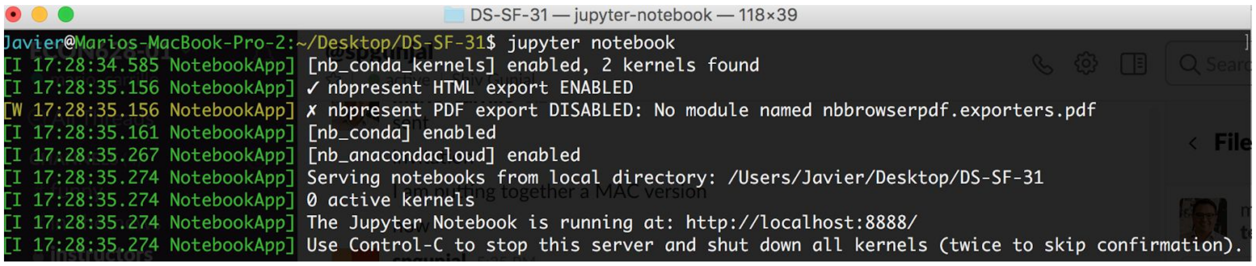
```
Justins-MacBook-Pro-4:~ jrcmalunay$ 
```

To check if Anaconda is installed, type: **conda info** and Return in your *Terminal*. It should output something similar to:



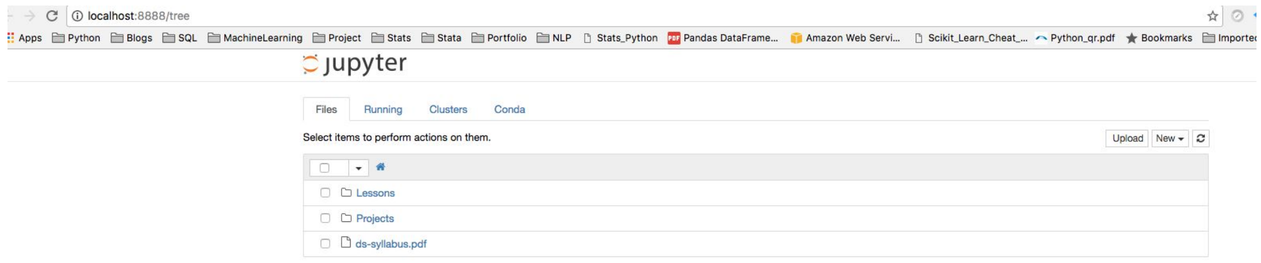
```
Justins-MacBook-Pro-4:~ jrcmalunay$ conda info
Using Anaconda Cloud api site https://api.anaconda.org
Current conda install:

             platform : osx-64
        conda version : 4.0.8
  conda-build version : 0+unknown
       python version : 2.7.11.final.0
     requests version : 2.9.1
     root environment : /Users/jrcmalunay/anaconda  (writable)
  default environment : /Users/jrcmalunay/anaconda
     envs directories : /Users/jrcmalunay/anaconda/envs
        package cache : /Users/jrcmalunay/anaconda/pkgs
         channel URLs : https://repo.continuum.io/pkgs/free/osx-64/
                        https://repo.continuum.io/pkgs/free/noarch/
                        https://repo.continuum.io/pkgs/pro/osx-64/
                        https://repo.continuum.io/pkgs/pro/noarch/
          config file : None
    is foreign system : False

Justins-MacBook-Pro-4:~ jrcmalunay$ 
```

- Then in the terminal type: **jupyter notebook** and **press Enter** (see picture below)

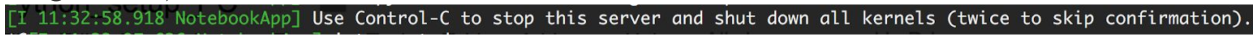


```
DS-SF-31 — jupyter-notebook — 118×39
Javier@Marios-MacBook-Pro-2:~/Desktop/DS-SF-31$ jupyter notebook
[I 17:28:34.585 NotebookApp] [nb_conda_kernels] enabled, 2 kernels found
[I 17:28:35.156 NotebookApp] ✓ nbpresent HTML export ENABLED
[W 17:28:35.156 NotebookApp] ✗ nbpresent PDF export DISABLED: No module named nbbrowserpdf.exporters.pdf
[I 17:28:35.161 NotebookApp] [nb_conda] enabled
[I 17:28:35.267 NotebookApp] [nb_anacondacloud] enabled
[I 17:28:35.274 NotebookApp] Serving notebooks from local directory: /Users/Javier/Desktop/DS-SF-31
[I 17:28:35.274 NotebookApp] 0 active kernels
[I 17:28:35.274 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/
[I 17:28:35.274 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
```

- You should eventually see a **new** tab open up in your browser for you to begin using Jupyter Notebooks.

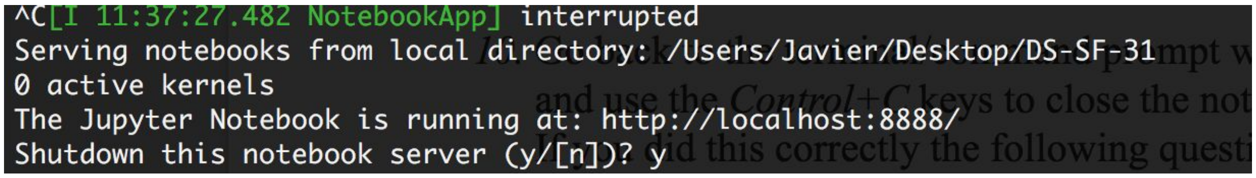


- Now we need to learn how to close the new tab on your browser. *Note the following lines were taken directly from the jupyter notebook beginner guide http://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/execute.html#shut-down-the-jupyter-notebook-app:'*
    a. In a nutshell, closing the browser (or the tab) **will not close** the Jupyter Notebook App. To completely shut it down you need to **close the associated terminal**. In more detail, the Jupyter Notebook App is a server that appears in your browser at a default address (*http://localhost:8888*). Closing the browser will not shut down the server.
- Go back to the terminal/command prompt where you previously typed jupyter notebook and use the ***Control+C*** keys to close the notebook (both keys should be press at the same time => see image below)



```
[I 11:32:58.918 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
```

- If you did this correctly the following question will appear in your terminal: **Shutdown this notebook server (y/[n])?** => then type **y** and press **enter** => your notebook is now closed (see image below):



```
^C[I 11:37:27.482 NotebookApp] interrupted
Serving notebooks from local directory: /Users/Javier/Desktop/DS-SF-31
0 active kernels
The Jupyter Notebook is running at: http://localhost:8888/
Shutdown this notebook server (y/[n])? y
```

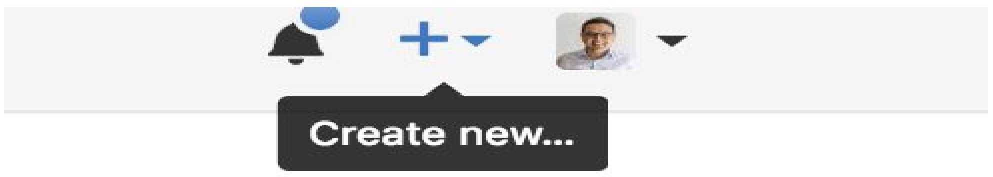
- For more information on the Jupyter Notebook system see the official documentation.
    a. Additional information on Jupyter Notebook Manual https://athena.brynmawr.edu/jupyter/hub/dblank/public/Jupyter%20Notebook%20Users%20Manual.ipynb

Part 2: Cloning/creating your class directory/folder

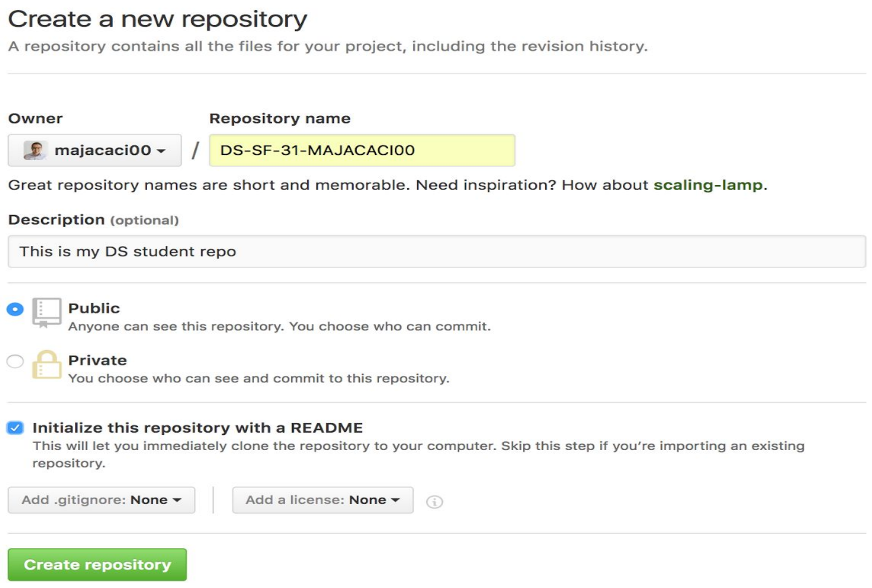
1. We already worked on this step the first day of class, but in case you need a refresher on how we did it here are the steps:
    a. In the **Terminal** type: **pwd** (remember that pwd = print working directory)
    b. We **suggested** to create/clone the DS-SF-31 github repo into your Desktop. Some of you did it and some of you have this folder/directory saved somewhere else. I am going to follow the steps to allocate the folder/directory into your Desktop:
        b.1 type: **cd Desktop** (remember that cd = change directory, in this case we need to change the directory to your Desktop , and it is in here where we are going to allocate the DS-SF-31 folder)
    c. type: **git clone** https://github.com/ga-students/DS-SF-31.git
    d. type: **cd DS-SF-31**
    e. type: **git pull** (git pull is the command you will need to use to update your **DS-SF-31** folder, do this as frequently as you need)

Part 3: Create your student repo

1. Access your github account by going github.com
2. On the top right corner of your github page, identify the + sign click on it and select create on *New repository* (see image)



3. A new window will open. In this window go to *Repository Name* and type DS-SF-31-*MAJACACI00* (**replace** *MAJACACI00* with **your github username**)
4. On the *description* section type: "This is my DS student repo"
5. Scroll to the bottom and make sure that *Public* is selected
6. *Select* Initialize this repository with a README
7. Click on *Create repository* (see image below)



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: majacaci00 / Repository name: DS-SF-31-MAJACACI00

Great repository names are short and memorable. Need inspiration? How about scaling-lamp.

Description (optional): This is my DS student repo

● Public — Anyone can see this repository. You choose who can commit.

○ Private — You choose who can see and commit to this repository.

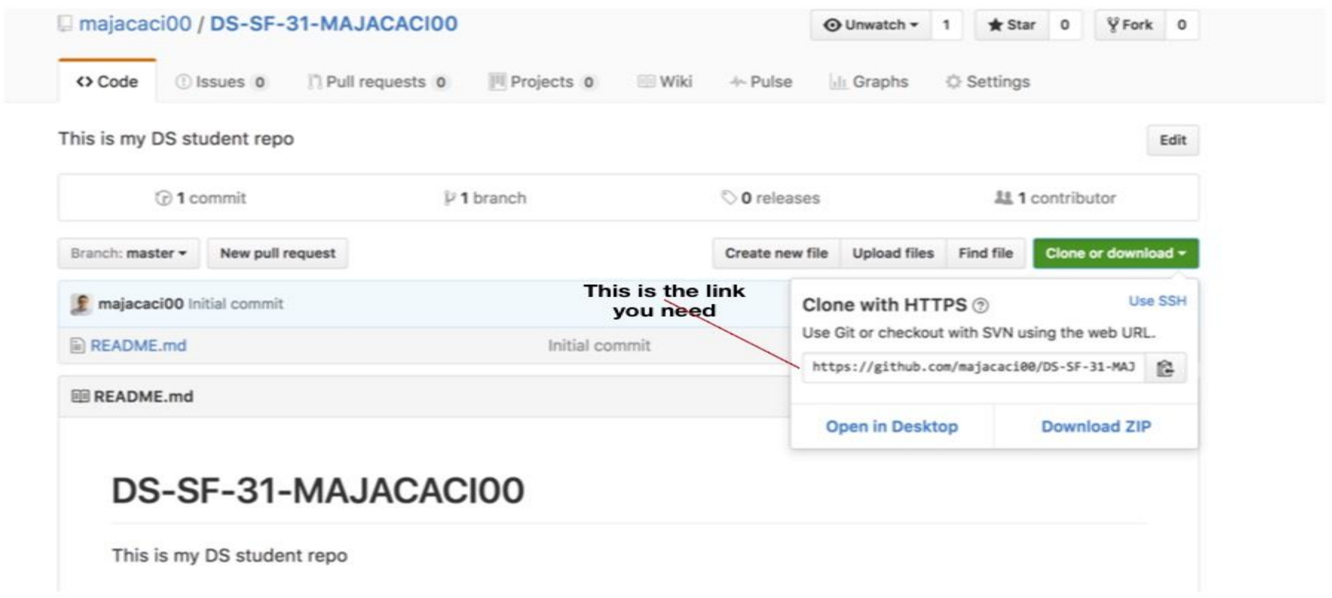
✓ Initialize this repository with a README — This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

Create repository

Part 4: Cloning your student repo to your computer

1. Open the Terminal and type **pwd** (make sure you are in your Desktop, if not follow step 2)
2. type: **cd Desktop** and press return/enter
3. type: **pwd** and press return, it should say something similar to **/Users/yourname/Desktop**
4. Once you are in your desktop type: **git clone** https://github………….(this is the url from your DS-SF-31-githubusername=> you can find this link if you click on the green button "Clone or download" => see picture below)



This is my DS student repo

Use Git or checkout with SVN using the web URL.

```
https://github.com/majacaci00/DS-SF-31-MAJ
```

DS-SF-31-MAJACACI00

This is my DS student repo

Part 5: Pushing a document to your student repo

1. **MANUALLY** copy a **ANY *pdf*** document into your *DS-SF-31-MAJACACI00* (remember that you in your case the *MAJACACI00* should be *your github username*)
2. **MANUALLY** rename your ***pdf*** document to ***test***
3. Go back to your Terminal and ***make*** sure you are in the *DS-SF-31-githubusername* folder. => you can check this by typing **pwd**
4. type: **git add test.pdf**
5. type: **git status** (you should see your test.pdf under *Changes to be committed*: *blablabla*, also your file should appear *highlighted in green)*
6. type: **git commit -m "testing my student repo"**
7. type: **git push origin master**
8. go to github.com (*refresh the site*) and you should see your test.pdf document
9. SUCCESS!

Part 6: Copying files from folder to folder => via the Terminal

How to copy files from the ***DS-SF-31*** folder to your ***DS-SF-31-MAJACACI00*** (remember that you in your case the *MAJACACI00* should be *your github username*) folder using the terminal (**GET USE TO DO THIS, IS BEST PRACTICE AND YOU WILL BE DOING THIS FREQUENTLY)**

1. in your terminal => navigate to your ***DS-SF-31****-githubusername* folder

2. type: pwd => and **make** sure your are in the ***DS-SF-31****-githubusername*
3. once you are in the ***DS-SF-31****-githubusername* => type: **mkdir Lessons**
4. type : **ls -l** => (you should *see* the **Lessons** folder)
5. cd to your Desktop
6. type: pwd => and make sure you are in your Desktop
7. type: cp -i /Users/YOURNAME/Desktop/***DS-SF-31***/*Lessons* /Users/YOURNAME/Desktop/***DS-SF-31****-githubusername*/*Lessons*
    a. **<u>Note</u>**: the lines above are just *one* line. There is a *space* between \Lessons and /Users
    b. the command cp => copies the content from the *Lessons* folder in ECON628-01 to *ECON628-01-githubusername*)
    c. **<u>Note</u>**: YOURNAME => refers to your mac user name
    d. **<u>Note:</u>** This is the process of copying files and folders from your local repository (***DS-SF-31***) to your local student repository (***DS-SF-31****-githubusername*)
    e. You can make sure you have copied the files to your local student repository (***DS-SF-31****-githubusername*) by ***manually*** opening "Finder => Desktop => ***DS-SF-31****-githubusername*)
    f. See the 2 images below for more details:

- In some versions of **Dolphin** and managers for KDE, you can enter wildcards directly on the location bar. For example, if you want to see all the files starting with a lowercase *u* in the */usr/bin* directory, enter **/usr/bin/u*** in the location bar, and it will display the result.

Many ideas originally found in the command line interface make their way into the graphical interface, too. It is one of the many things that make the Linux desktop so powerful.

mkdir—Create Directories

The `mkdir` command is used to create directories. It works like this:

```
mkdir directory...
```

A note on notation: In this book, when three periods follow an argument in the description of a command (as above), it means that the argument can be repeated; thus, in this case,

```
mkdir dir1
```

would create a single directory named *dir1*, while

```
mkdir dir1 dir2 dir3
```

would create three directories named *dir1*, *dir2*, and *dir3*.

cp—Copy Files and Directories

The `cp` command copies files or directories. It can be used two different ways:

```
cp item1 item2
```

to copy the single file or directory *item1* to file or directory *item2* and:

```
cp item... directory
```

to copy multiple items (either files or directories) into a directory.

Tables 4-4 and 4-5 list some of the commonly used options (the short option and the equivalent long option) for cp.

Table 4-4: cp Options

Option	Meaning
`-a, --archive`	Copy the files and directories and all of their attributes, including ownerships and permissions. Normally, copies take on the default attributes of the user performing the copy.
`-i, --interactive`	Before overwriting an existing file, prompt the user for confirmation. **If this option is not specified, cp will silently overwrite files.**
`-r, --recursive`	Recursively copy directories and their contents. This option (or the -a option) is required when copying directories.
`-u, --update`	When copying files from one directory to another, copy only files that either don't exist or are newer than the existing corresponding files in the destination directory.
`-v, --verbose`	Display informative messages as the copy is performed.

Table 4-5: cp Examples

Command	Results
`cp file1 file2`	Copy *file1* to *file2*. **If *file2* exists, it is overwritten with the contents of *file1*.** If *file2* does not exist, it is created.
`cp -i file1 file2`	Same as above, except that if *file2* exists, the user is prompted before it is overwritten.
`cp file1 file2 dir1`	Copy *file1* and *file2* into directory *dir1*. *dir1* must already exist.
`cp dir1/* dir2`	Using a wildcard, all the files in *dir1* are copied into *dir2*. *dir2* must already exist.
`cp -r dir1 dir2`	Copy directory *dir1* (and its contents) to directory *dir2*. If directory *dir2* does not exist, it is created and will contain the same contents as directory *dir1*.

Part 7: Get more familiar with Github

1. For information on Git please revise this links
   a. https://guides.github.com/activities/hello-world/
   b. https://help.github.com/articles/git-and-github-learning-resources/