1. Setup git account -
   git config user.name 'your user name'
   git config user.email 'your email name'

2. Get user info - **nano ~/.gitconfig**

References:
https://www.liquidweb.com/kb/install-git-ubuntu-16-04-lts/
http://rogerdudler.github.io/git-guide/

Linux Commands:

free -h -s 1 -- see RAM usage

AWS S3 commands: (https://docs.aws.amazon.com/cli/latest/reference/sagemaker/index.html)

- aws s3 ls

- To upload to S3, go to the directory whose **contents** you want to upload from your terminal. The run this command:
  - aws s3 sync **.** s3://yours3storagefolder/ (--dryrun to check cmd)

- Sync data from S3 storage in SageMaker. Go to directory where you want to save data
  - aws s3 sync s3://thinkbricks-testdata-us-east-2/data-train/ **.** --dryrun

- Download all jupyter notebook files
  - !tar cvfz allfiles.tar.gz *

CONDA CHEAT SHEET
- https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c49671267e40c689e0bc00ca/conda-cheatsheet.pdf

Docker stuffs

- https://docs.docker.com/install/linux/docker-ce/ubuntu/#install-using-the-repository
- https://docs.docker.com/get-started/
- https://docs.docker.com/get-started/part2/ -- make docker image

```
## List Docker CLI commands
docker
docker container --help

## Display Docker version and info
docker --version
docker version
docker info

## Execute Docker image
docker run hello-world

## List Docker images
docker image ls

## List Docker containers (running, all, all in quiet mode)
docker container ls
docker container ls --all
docker container ls -aq

## Remove container all
docker container prune

## Run docker image
sudo docker run -p 4000:80 friendlyhello
## Run app in background
docker run -d -p 4000:80 friendlyhello

## Build docker image
docker build -t friendlyhello .  # Create image using this directory's Dockerfile
docker run -p 4000:80 friendlyhello  # Run "friendlyhello" mapping port 4000 to 80
docker run -d -p 4000:80 friendlyhello         # Same thing, but in detached mode
docker container ls                                # List all running containers
docker container ls -a             # List all containers, even those not running
docker container stop <hash>           # Gracefully stop the specified container (hash=ID of
container)
docker container kill <hash>         # Force shutdown of the specified container
docker container rm <hash>        # Remove specified container from this machine
docker container rm $(docker container ls -a -q)        # Remove all containers
docker image ls -a                             # List all images on this machine
docker image rm <image id>            # Remove specified image from this machine
docker image rm $(docker image ls -a -q)   # Remove all images from this machine
docker login             # Log in this CLI session using your Docker credentials
docker tag <image> username/repository:tag  # Tag <image> for upload to registry
docker push username/repository:tag           # Upload tagged image to registry

docker run username/repository:tag                 # Run image from a registry
docker run -p 4000:80 hasibzunair/myapp:v0.0.1    # Run app

docker run -p 4000:80 myapp:v0.0.1

docker stack ls                                            # List stacks or apps
docker stack deploy -c <composefile> <appname>  # Run the specified Compose file
docker service ls                 # List running services associated with an app
docker service ps <service>                  # List tasks associated with an app
docker inspect <task or container>                   # Inspect task or container
docker container ls -q                                      # List container IDs
docker stack rm <appname>                              # Tear down an application
docker swarm leave --force      # Take down a single node swarm from the manager
```

SSH in a container:

```
docker exec -it <container name> /bin/bash
```

http://phase2.github.io/devtools/common-tasks/ssh-into-a-container/

Docker Compose

```
## docker-compose.yml
## docker-compose up
## Add volume in docker-compose to edit code and update instantly
## Run again with docker-compose up command
## Exit with docker-compose down
## docker-compose stop
```

Docker X Heroku:

(need to install heroku container registry)

- Heroku container:login
- GO to the root folder (app folder, dockerfile)
- Heroku create -a NAME
- Sudo heroku container:push web -a NAME (Dockerize)
- Sudo heroku container:release web -a NAME (Deploy app)

Pipenv setup

- Install pipenv
- pipenv # see stuff
- pipenv --python 3.6 #make env with python 3.6
- pipenv install requests # install package (auto generates piplock and pipfile)
- pipenv run python main.py
- pipenv install -r requirements.txt (import from requirements file)
- pipenv shell # enter virtual env (CTRL+D to exit or simply type exit)