**Assignments Sessions:**

Git: 12/05/2020, 01/04/2020

Kubernetes: 22/04/2020,22/05/2020,07/04/2020

Ansible: 21/05/2020, 31/03/2020

Docker-1: 27/03/2020, 19/05/2020, 24/04/2020

Docker-2: 27/03/2020

Jenkins: 23/04/2020

Nagios: 30/04/2020

Selenium: 29/04/2020

Puppet: 31/03/2020

Capstone Project: 13/05/2020,16/04/2020

**Docker Notes:**

**To remain as a Sudo user:**

sudo -s, I think that will always work.  
  
I've had issues using sudo -s before, because it keeps your environment profiles, and if you're not careful, it can end up changing permissions to items in your home directory.  
  
Use  
sudo -i instead, it'll take roots environment variables, but other than that, is essentially the same as sudo -s

**Another way to run without sudo:**

sudo usermod -aG docker $USER

And then relogin into your session.

1. **Commands:**
2. To clear the screen: clear
3. Install a Ubuntu: sudo apt-get install docker.io
4. Version: docker --version
5. To pull an image: sudo docker pull docker-hub-id/ubuntu
6. To know all the Images: sudo docker images
7. Running an image: sudo docker run -it -d ubuntu

-it means to make the terminal interactive

-d make it daemon to run it in background until I stop the container.

1. To remove an image: sudo docker rmi <image id>
2. To know the running containers: sudo docker ps
3. To know the all the containers: sudo docker ps -a
4. To stop the container: sudo docker stop <container id>
5. To kill a container: sudo docker kill <container id>

-kill is to forcefully kill a container if not responsive or hanged, as equal to doing a powercut.

1. To remove a container: sudo docker rm <container id>
2. To force remove a container which is running: sudo docker rm -f <container id>
3. To remove all the containers: sudo docker rm -f $(sudo docker ps -a -q)
4. To get inside a container: sudo docker exec -it <container id> bash

-it means to make the terminal interactive

bash to run on the current bash.

1. To update inside a container: apt-get update
2. To come out of a container: exit
3. login, it can be done using the following command: docker login
4. pushing your container on DockerHub, use the following command

$ docker push <username>/<container-id>

docker push jatmakur/ubuntu

1. To know all the directories inside the container: ll (small L twice)
2. Creating a directory: mkdir <directory\_name>
3. Saving or Committing a container: sudo docker commit <container-id> <new\_name>
4. Saving or Committing a container with user-id or docker hub id: sudo docker commit <container-id> <user\_name/new\_name>
5. Installing a new software: sudo apt-get install apache2

To check apache2 is running: service apache2 status

To run the apache2: service apache2 start

1. To run an image with port mapping:

sudo docker run -it -p <OS-port>:<container-port> -d <image-name>

Ex: sudo docker run -it -p 82:80 -d jatmakur/apache

Apache runs on default port 80 on container and mapping it with OS port 82.

1. To get the history of commands: history
2. **Creating a Docker File:**

**FROM:**

* Specify the base image. Ex: FROM ubuntu

**ADD:**

* ADD keyword is used to add files to the container being built.
* ADD <source> <destination in container>
* Ex: ADD . /var/www/html (.) means current directory

**RUN:**

* The RUN keyword is used to add layers to the base image, by installing components. Each RUN statement adds a new layer to the docker image.
* Ex: RUN apt-get update
* RUN apt-get -y install apache2

**CMD:**

* The CMD keyword is used to run commands on the start of the container.
* These commands run only when there is no argument specified while running the container.
* Ex: CMD apachectl -D FOREGROUND

**ENTRYPOINT:**

* The ENTRYPOINT keyword is used strictly run commands the moment the container initializes.
* The difference between CMD and ENTRYPOINT is, ENTRYPOINT will run irrespective of the fact whether the argument is specified or not.
* Ex: ENTRYPOINT apachectl -D FOREGROUND

**ENV:**

* The ENV keyword is used to define environment variables in the container run-time.
* Ex: ENV name DevOps Intellipaat

**Build the docker file:**

* Get into the docker file folder and run below command.

docker build . -t new\_dockerfile

**To view the Docker file with out entering inside the file:** cat dockerfile

**To edit or enter into the file:** vim dockerfile or nano dockerfile