Assignment #2: Intro to containers

**Docker Simplified: guide for beginners**

***Notes from*** [***Docker Simplified: A Hands-On Guide for Absolute Beginners***](https://www.freecodecamp.org/news/docker-simplified-96639a35ff36/)

**Docker**

Software platform that simplifies the process of building, running, managing and distributing applications. It does this by **virtualizing the operating system of the computer on which it is installed and running**.

**Problem it solves**

Imagine we have three different Python-based applications that you plan to host on a single server (either physical or a virtual machine).

Each of these applications makes use of a different version of Python, as well as the associated libraries and dependencies, differ from one application to another.

Since **we cannot have different versions of Python installed on the same machine, this prevents us from hosting all three applications on the same compute**r.

**The Solution**

We could solve this problem without making use of Docker by having three physical machines, or a single physical machine, powerful enough to host and run three virtual machines on it- -> the costs associated with procuring and maintaining the hardware are quite expensive.

**How Docker Function**

**A close-up of a container

Description automatically generated**

*Docker Host:* the machine on which Docker is installed.

Whenever you plan to deploy an application on the host, it would create a logical entity on it to host that application: *Container/Docker Container*

**A Docker Container doesn’t have any operating system installed and running on it**. But it would have a virtual copy of the process table, network interface(s), and the file system mount point(s), inherited from the operating system of the host on which the container is hosted and running.

**The kernel of the host’s operating system is shared across all the containers that are running on it.**

This allows each container to be isolated from the other present on the same host. Thus it supports multiple containers with different application requirements and dependencies to run on the same host, **as long as they have the same operating system requirements.**

**In short, Docker would virtualize the operating system of the host on which it is installed and running, rather than virtualizing the hardware components.**

**Advantages of using Docker**

* Storage Optimized. A large number of applications can be hosted on the same host, as containers are usually few megabytes in size and consume very little disk space.
* Robustness. A container does not have an operating system installed on it. Thus, it consumes very little memory in comparison to a virtual machine (which would have a complete operating system installed and running on it). This also reduces the bootup time to just a few seconds, as compared to a couple of minutes required to boot up a virtual machine. (Cold start penalties)
* Reduces costs. Docker is less demanding when it comes to the hardware required to run it.

**Disadvantages of using Docker**

* Applications with different operating system requirements cannot be hosted together on the same Docker Host.

**Core components of Docker**

* Docker engine, client-server based application consisting of: a server, REST API, Client

A diagram of a server

Description automatically generated

The **Server** runs a daemon known as **dockerd** **(Docker Daemon)**, which is nothing but a process. It is responsible for creating and managing Docker Images, Containers, Networks and Volumes on the Docker platform.

The **REST API** specifies how the applications can interact with the Server, and instruct it to get their job done.

The **Client** is nothing but a command line interface, that allows users to interact with **Docker** using the commands.

**Docker Terminology**

**Docker Image**: template that contains the application, and all the dependencies required to run that application on Docker.

**Docker Container**: a logical entity. In more precise terms, it is a running instance of the Docker Image.

**What is Docker Hub?**

**Docker Hub** is the official online repository where you could find all the Docker Images that are available for us to use. It also allows us to store and distribute our custom images as well.

**Docker Commands**

docker create: allows us to create a new container.

docker ps: allows us to view all the containers that are running on the Docker Host.

docker start: starts any stopped container(s).

docker stop: stops any running container(s).

docker restart: restarts any running container(s).

docker run: first creates the container, and then it starts the container. It is a combination of the docker create and the docker start command.

docker rm: to delete a container. *Containers need to be in a stopped state in order to be deleted.*

docker images: lists out all the Docker Images that are present on your Docker Host.

docker rmi: allows us to remove an image(s) from the Docker Host.