**Practical No: 4**

**Date: 11/04/2023**

**Aim: Working with Docker Desktop App**

**Description:**

Give Overview of Docker Desktop App

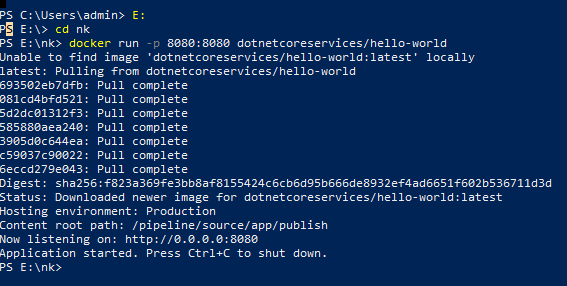
**Container**: A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. By default, a container is relatively well isolated from other containers and its host machine. You can control how isolated a container’s network, storage, or other underlying subsystems are from other containers or from the host machine.

**Image**: An image is a read-only template with instructions for creating a Docker container. You might create your own images or you might only use those created by others and published in a registry. To build your own image, you create a Dockerfile with a simple syntax for defining the steps needed to create the image and run it.

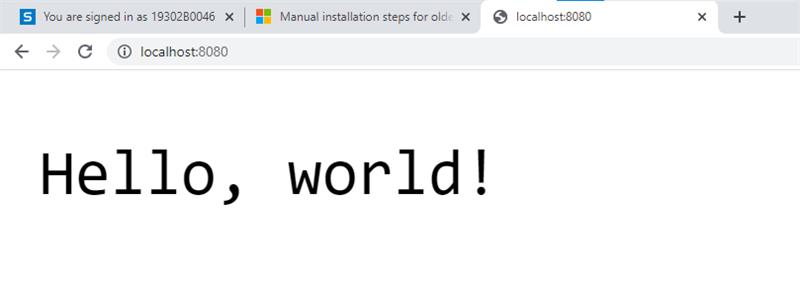
**Volumes**: Volumes are the preferred mechanism for persisting data generated by and used by Docker containers. Volumes are easier to back up or migrate than bind mounts. You can manage volumes using Docker CLI commands or the Docker API. Volumes work on both Linux and Windows containers. Volumes can be more safely shared among multiple containers. Volume drivers let you store volumes on remote hosts or cloud providers, to encrypt the contents of volumes, or to add other functionality. New volumes can have their content pre-populated by a container. Volumes on Docker Desktop have much higher performance than bind mounts from Mac and Windows hosts.

**Code & Output:**

**Commands**

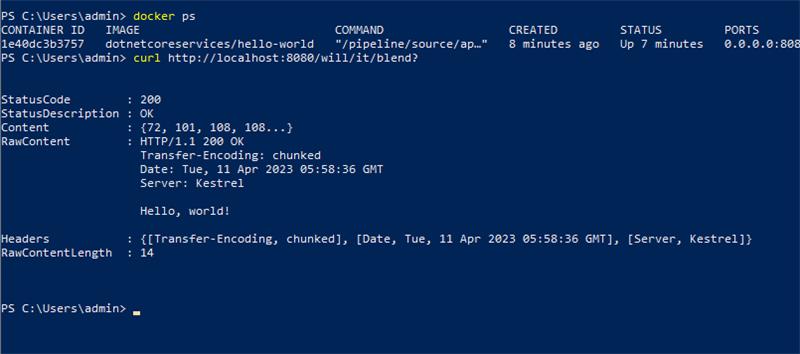
$ docker run -p 8080:8080 dotnetcoreservices/hello-world   


Run Localhost in browser

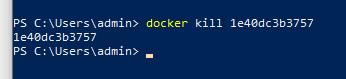
$ docker ps



curl <http://localhost:8080/will/it/blend>?



$ docker kill PID (process id of application)



Process Id terminated

