Experiment 2

6 Docker Commands Are:

1. Build A Docker Image: Builds a Docker image from a Dockerfile

`docker build -t [image_name] .`

2. Run A Docker Container: Runs a container from an image

`docker run -d -p 8080:80 --name [container_name] [image_name]`

3. List All Containers: Lists all containers (running & stopped)

`docker ps -a`

4. Stop A Container: Stops a running container

`docker stop [container_name]`

5. Remove A Container: Deletes a stopped container

`docker rm [container_name]`

6. Remover A Image: Deletes a Docker image

`docker rmi [image_name]`

Experiment 3

Steps:

1. Create a empty folder.
2. Open VS Code in that folder.
3. Create Your index.html file (or whatever content you want) in your (working directory) folder.
4. Create a *Dockerfile*. If you want to run HTML files in your container then your dockerfile will be as follows:

FROM nginx:alpine

RUN rm -rf /usr/share/nginx/html/*

COPY index.html /usr/share/nginx/html/

EXPOSE 80

5. Open VS code's integrated terminal.
6. Run the command "docker --v" to ensure that docker is up and running.
7. To build a image, run the following command:

`docker build -t [image_name] .`

8. Your image has been created. To confirm run

`docker images`

You should see your image name.

9. To create a container and run it, run this command:

`docker run -d -p 8080:80 --name [container_name] [image_name]`

10. To check if your container is running you can execute the command "docker ps -a" or you can go to localhost:8080 to check if that port is running the output of the container or not.
11. To stop the container you need to execute:

`docker stop [container_name]`

12. To remove the container you need to execute:

`docker rm [container_name]`

Experiment 4

Steps:

1. Open your terminal in your working directory.
2. Then, to pull nginx, run the following command:

   `docker pull nginx`
3. To ensure that the image is pulled, you can run:

   `docker images`
4. To create and run the container, execute the following command:

   `docker run -d -p 8080:80 --name [container_name] nginx`
5. To check if your container is running, go to localhost:8080. You should see this page:

6. To stop a docker container, just run:

   `docker stop [container_name]`

Experiment 5

Steps:

1. Go to KillerCoda And log in.
2. Open Kubernetes 1.32
3. The master and the worker node will already be created. To view these nodes, run:

   kubectl get nodes

4. The 6 Commands in Kubernetes are:
   - To Get Nodes:

     kubectl get nodes

   - To Create a Pod:

     kubectl run [pod_name] --image=nginx

   - To Get Pod details:

     kubectl describe pod [pod_name]

   - To expose pod as a service:

     kubectl expose pod [pod_name] –port=80 --type=NodePort

   - To Get service:

     kubectl get svc

Experiment 6

Steps:

1. Go to KillerCoda and open a Kubernetes playground.
2. To create a pod, you need to create a pod YAML file. So, run the following command:

nano mypod.yaml

3. Paste the following configuration in that file:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
  - name: mycontainer
    image: nginx
    ports:
    - containerPort: 80
```

apiVersion: v1
kind: Pod
metadata:
name: mypod
spec:
containers:
- name: mycontainer
image: nginx
ports:
- containerPort: 80

4. To save and exit nano: Ctrl + X → Y → Enter
5. Deploy the pod:

kubectl apply -f mypod.yaml

6. To ensure pod is running, execute:

kubectl get pods

7. To get the pod IP Address, run:

kubectl get pod mypod -o wide

8. To get the logs:

kubectl logs mypod

9. To describe a pod:

kubectl describe pod mypod