NAME : PAREENITA A.SHIRSATH PRN : 221101062 PRN : 221101062 T.E.A.I.&.D.S.



**SEPM EXPERIMENT 8**

**Aim:** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers.

**Theory:**

**Docker Architecture**

* The Core components of Docker consist of
  + Docker daemon
  + Docker client
  + Docker Desktop
  + Docker registry
  + Docker images
  + Docker containers

A diagram of a docker host

Description automatically generated

* Docker enables efficiency and reduces operational overheads so that any developer, in any development environment, can build stable and reliable applications. Docker provides the ability to package and run an application in a loosely isolated environment called a container.
* Isolation and security allow you to run many containers simultaneously on a given host. You can easily share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.
* Docker Container Lifecycle Management
  + The lifecycle of a docker container consists of five states:
  + Created state
  + Running state
  + Paused state/unpaused state
  + Stopped state
  + Killed/Deleted state

A diagram of a running process

Description automatically generated

* Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications.
* Volumes are the preferred mechanism for persisting data generated by and used by Docker containers. While bind mounts are dependent on the directory structure and OS of the host machine, volumes are completely managed by Docker. Volumes have several advantages over bind mounts:
  + 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.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA white background with black and white clouds

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen with white text

Description automatically generated

A screenshot of a computer

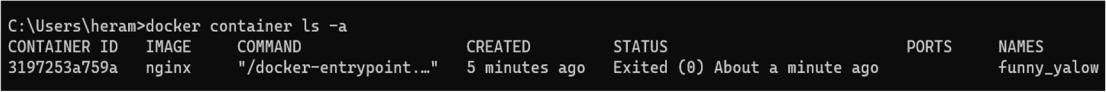
Description automatically generated

1. **Listing Out Running Containers:**

****

1. **Stopping a running containerA black background with white text

   Description automatically generated**
2. **Listing Out Containers that are running and stopped.**

****

1. **Exploring Specific Logs Of a Container:**

A computer screen with white text

Description automatically generated

**Conclusion:**

In this experiment, we learned to set up and run Selenium tests in Jenkins using Maven. We also gained an overview of Docker architecture, understanding its core components and how it enables efficient application development through containerization. Docker provides isolation and security for running applications in containers, allowing easy sharing and consistent behavior across different environments.