Devops expt 8

Install docker in ubuntu aws

<https://www.devopsschool.com/blog/lifecycle-of-docker-containers/>

To understand Docker architecture, container lifecycle, and how to manage Docker images and interact with containers, you can follow these steps to install Docker on an AWS EC2 instance running Ubuntu.

1. Launch an AWS EC2 Instance:

- Log in to your AWS Management Console.

- Launch an EC2 instance with an Ubuntu AMI.

- Ensure that the security group associated with your EC2 instance allows incoming traffic on port 22 (SSH) and any additional ports you may need for Docker.

2. Connect to your EC2 Instance:

- Use an SSH client like `ssh` to connect to your EC2 instance. Replace `your-instance-ip` with your actual EC2 instance's public IP address.

```bash

ssh ubuntu@your-instance-ip

```

3. Update the package list and install necessary packages:

```bash

sudo apt update

sudo apt install -y apt-transport-https ca-certificates curl software-properties-common

```

4. Add Docker's official GPG key:

```bash

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

```

5. Add the Docker repository:

```bash

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

```

6. Install Docker:

```bash

sudo apt update

sudo apt install -y docker-ce

```

7. Start the Docker service and enable it to start on boot:

```bash

sudo systemctl start docker

sudo systemctl enable docker

```

8. Verify Docker installation by running the following command:

```bash

docker --version

```

9. To understand Docker's architecture, you can explore its components:

- Docker Daemon: The background service that manages Docker containers.

- Docker Client: The command-line tool used to interact with the Docker daemon.

- Docker Images: Pre-built filesystems used to create containers.

- Docker Containers: Running instances of Docker images.

- Docker Registry: A repository for Docker images (e.g., Docker Hub).

10. To manage Docker images and interact with containers, you can use the following Docker commands:

- Pull a Docker image from Docker Hub (e.g., Ubuntu):

```bash

docker pull ubuntu

```

- List downloaded images:

```bash

docker images

```

- Run a Docker container:

```bash

docker run -it ubuntu /bin/bash

```

- Inside the container, you can execute commands or install software as needed.

- To exit the container without stopping it, use `Ctrl + P` followed by `Ctrl + Q`.

- To stop and remove a container:

```bash

docker stop <container\_id\_or\_name>

```

- To remove an image:

```bash

docker rmi <image\_id\_or\_name>

```

These are some basic Docker commands to get you started. Docker is a powerful tool for containerization and can be used for various purposes, including deploying applications and services in a containerized environment. Make sure to explore Docker's documentation and tutorials for more in-depth knowledge and best practices.

Use sudo for priveleges permission denied error

List running containers:

docker ps

List all containers (including stopped ones):

docker ps -a

Inspect a specific container:

To get detailed information about a specific container, including its Container ID and various attributes, you can use the **docker inspect** command with either the Container Name or Container ID:

docker inspect container-name-or-id

docker stop container-id

docker pull alpine

docker pull nginx

docker pull mysql

EXPERIMENT 9

* Go to the AWS Management Console and launch a new Ubuntu EC2 instance.
* Make sure to open the necessary ports for SSH (port 22) and, if needed, for your web application (e.g., port 80 for HTTP).

If you have a simple "Hello, World!" application, you can create a Docker image for it using the following steps:

1. Create a directory for your application and navigate to it on your AWS Ubuntu EC2 instance:

```bash

mkdir helloworld-app

cd helloworld-app

```

2. Create a simple "Hello, World!" application. You can use a basic HTML file for this example:

```html

<!-- index.html -->

<!DOCTYPE html>

<html>

<head>

<title>Hello, World!</title>

</head>

<body>

<h1>Hello, World!</h1>

</body>

</html>

```

3. Create a Dockerfile for your "Hello, World!" application:

```Dockerfile

# Use an official Nginx image as a parent image

FROM nginx:latest

# Copy the HTML file to the default Nginx web server directory

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

# Expose port 80 to the outside world

EXPOSE 80

```

4. Build the Docker image from the Dockerfile:

```bash

docker build -t helloworld-app .

```

This will create a Docker image named `helloworld-app` based on the Nginx web server and your "Hello, World!" HTML file.

5. Run a container using the created image:

```bash

docker run -d -p 80:80 helloworld-app

```

This command starts a Docker container in the background (-d) and maps port 80 on the host to port 80 in the container.

6. Access your "Hello, World!" application in a web browser by using your EC2 instance's public IP or DNS name. Open your web browser and enter the EC2 instance's public IP address or DNS name.

You should see the "Hello, World!" message displayed in your web browser. This demonstrates a simple web application running in a Docker container on your AWS Ubuntu EC2 instance.