#This File is written by Sayali Jadhav

**Day17: Docker Project for DevOps Engineers**

**Dockerfile:**

Docker is a tool that makes it easy to run applications in containers. Containers are like small packages that hold everything an application needs to run. To create these containers, developers use something called a Dockerfile.

A Dockerfile is like a set of instructions for making a container. It tells Docker what base image to use, what commands to run, and what files to include. For example, if you were making a container for a website, the Dockerfile might tell Docker to use an official web server image, copy the files for your website into the container, and start the web server when the container starts.

**Task:**

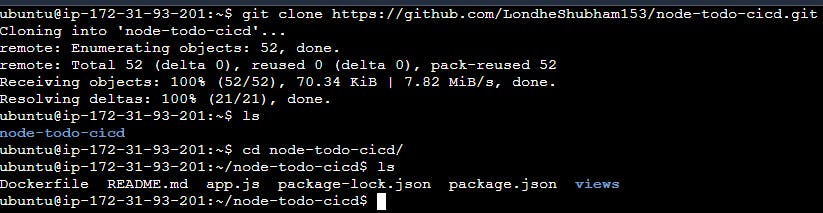
**1)Create a Dockerfile for a simple web application (e.g., a Node.js or Python app)**

**Steps:**

**1)Launch AWS ubuntu ec2 instance.**

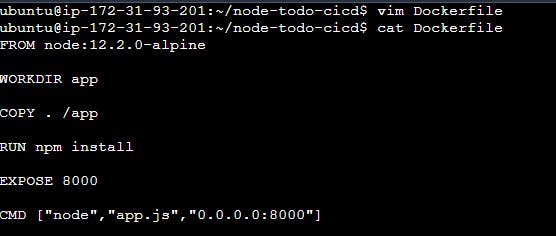
**2)Clone the repository from GitHub to your ubuntu server using command:**

git clone <repository\_url>



**3)Create a Dockerfile:**

vim Dockerfile



The specific commands you can use in a dockerfile are:

* **FROM:** The first thing we need to do is define from which image we want to build from. Creates a layer from the node 12.2.0-alphine
* Next we create a directory to hold the application code inside the image, this will be the working directory for your application. **WORKDIR** a command is used to define the working directory of a Docker container at any given time.
* **COPY** is a dockerfile command that copies files from a local source location to a destination in the Docker container.
* **RUN** command to execute the command npm install and Builds your container
* **EXPOSE** instruction informs Docker that the container listens on the specified network ports at runtime.
* **CMD** Specifies what command we want to run when our image is run inside of a container.

**2)Build the image using the Dockerfile and run the container**

To build an image using Dockerfile, Go to the directory that has your Dockerfile and run the following command:

**docker build -t <image-name> .**

Text

Description automatically generated

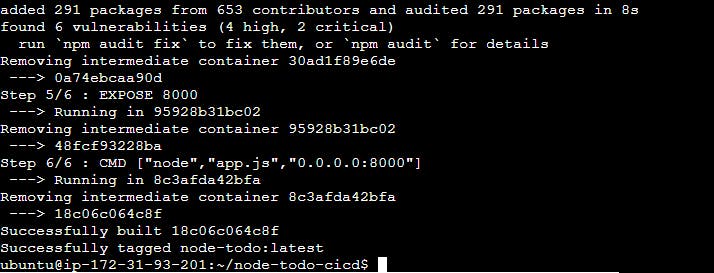
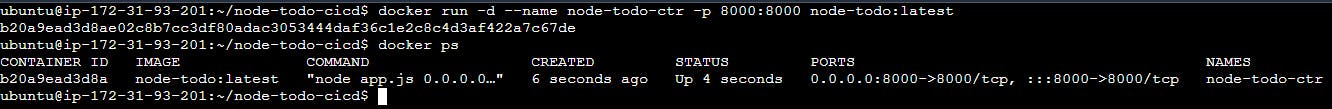


Image created, now can create container using command-

**docker run -d - -name <container-name> -p 8000:8000 <image-name>**

Running your image with -d runs the container in detached mode, leaving the container running in the background. The -p flag redirects a public port to a private port inside the container and - - name flag for assigning name to docker container.

**docker ps** command is used to list all the containers.



**3)Verify that the application is working as expected by accessing it in a web browser**

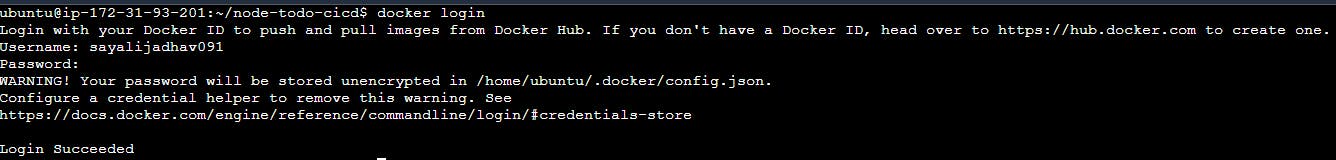
we have created container successfully, now go to your Ec2 instance and check security group port 8000 is open or not. If not,then go to security imbound rule and add new rule and open 8000 port.

Now copy public IPv4 address paste into brower url and add 8000 port.



**4)Push the image to a public or private repository (e.g. Docker Hub )**

Use command “docker login” to login to dockerhub.



* Push docker image to docker hub using command:
* docker push <image-name>

Text

Description automatically generated

Graphical user interface, application, email

Description automatically generated