|  |
| --- |
| **Name:** WASIMA QAYYUMUDDIN SHAIKH  **RollNo:**6220071  **Class:** T.E.I.T  **Sem:** V  **Subject:** ADVACE DEVOPS LAB **(Addevops**)  **EXPERIMENT NO: 12**  **Q1. Install docker on AWS EC2 –Ubuntu by using curl**  **curl -fsSLhttps://get.docker.com -o get-docker.sh;**  **sh get-docker.sh**   1. Create EC2 instance Service → EC2 →Launch Instance 2. Then choose AMI here we are choosing Ubuntu Server 20. 3. Then in Configure Security Group → Add Rule → (HTTP) 4. Then review and launch by downloading key-value pair 5. After launching the instance, we need to connect to our instance by clicking on connect. Then we need to select SSH client. 6. After that open Termux qpp -> locate private key file by using cd command 🡪 Execute chmod 400 wasima12.pem for publically viewable to connect instance execute command ssh -I “wasimaubuntu.pem” and it’s Public DNS.      1. Then type ‘sudo su’ for entering root user. Then type curl command for installing docker To install docker write the command. #curl –fsSL https://get.docker.com -o get-docker.sh.     #sh get-docker.sh    **Q2. Run a Flask Application with HTML page printing colorful and decorated message inside a Docker Container and explain the steps .**   1. To create flask application first Create directory demo- dockerflask then change director to which recently created In demo-dockerflask directory create on more directory named app🡪Change directory to app 🡪In app directory create app.py fileby nano editor      1. In app.py write python code which we have to run.      1. In app folder Create requirements.txt file by using nano editor or vi editor .      1. In recruitments file write flask Only it automatically take latest version .      1. In app folder create one more folder named templates in template create html file using nano editor.      1. In html file write our code which printing colour ful and decorated messages.      1. Then go to demo-dockerflash directory create Docker file by using nano editor.        1. To build docker execute command docker build -t demo-dockerflash:latest .      1. To run docker execute command docker run -d -p 80:80 demo-dockerflask.      1. After that go to instance 🡪 details 🡪 Public DNS 🡪 Copy that Address and paste in browser     **Note: After successfully running docker image, terminate the EC2 Instance**    **Q3. What is docker file? explain all line of docker file.**  Dockerfile is a first step to containerize an application. Dockerfile contains a list of commands to assemble an image.  Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.When you run the Docker run command and specify WordPress, Docker uses this file to build the image itself. The Dockerfile is essentially the build instructions to build the image.  The advantage of a Dockerfile over just storing the binary image (or a snapshot/template in other virtualization systems) is that the automatic builds will ensure you have the latest version available. This is a good thing from a security perspective, as you want to ensure you’re not installing any vulnerable software.  **FROM python**  The first part is the FROM command, which tells us what image to base this off of. This is the multi-layered approach that makes Docker so efficient and powerful. In this instance, it’s using the python Docker image, which again references a Dockerfile to automate the build process.  **WORKDIR /opt/demo/**  **COPY /app .**  The COPY command is simply as it sounds. It can copy a file (in the same directory as the Dockerfile) to the container. You can do this for things like custom configuration files or like in this instance, a file to run commands after the container has been set up.  **RUN pip install -r requirements.txt**  The next set of calls are the RUN commands. This is what runs within the container at build time. and it then installs the libraries which specified in requirements.txt file.  **ENTRYPOINT python app.py**  If the ENTRYPOINT isn’t specified, Docker will use /bin/sh -c as the default. However, if you want to override some of the system defaults, you can specify your own entrypoint and therefore manipulate the environment. The ENTRYPOINT command also allows arguments to be set from the Docker run command as well |