**Operating System project:**

**Dockerizing DBMS application**

|  |  |
| --- | --- |
| Title of the Project: **Dockerizing of the DBMS Project** | |
| USN: **1MS20CS117** | NAME: **Skanda S Kumar** |
| USN: **1MS20CS110** | NAME: **Shivalingesh J Patil** |
| USN: **1MS20CS114** | NAME: **Shridhar D** |
| USN: **1MS20CS119** | NAME: **Somaning Cheeli** |

**Description of our DBMS Project**

During situations like COVID, going out and buying pharmaceutical products is challenging as there is greater probability of ourselves getting exposed to virus. And also, old age people cannot go out and buy due to their age-related issues. So, our project focuses on overcoming these scenarios by creating pharmaceutical database to order medicines online.

A **Pharmaceutical** **Database Application** is our DBMS project.

A place where a user can add his orders for medicine to be delivered safely.

He is able to easily order medicines through our user-friendly interface.

Anyone can add their orders provided they have account(registered) with this app.

Like ordering Dolo 650, Garnier facewash, etc.

It is possible for an adult to add his orders he is wishing to be delivered and delete the order before delivery if not required.

Can be used by anybody as a virtual medical store.

**Dockerizing our DBMS project**

1. **Creating the Docker files and Docker CLI**

Let’s explore the Dockerfile and what each of the commands do.

* FROM python:3.10 : This installs a Python image into the Docker image. This is also the version of Python that will run the application in the container
* WORKDIR /code: This explicitly tells Docker to set the provided directory as the location where the application will reside within the container
* COPY . CODE: This copies every other necessary file and its respective contents into the app folder that is the root directory of the application within the container
* RUN pip install -r requirements.txt: This command installs all the dependencies defined in the requirements.txt file into your application within the container
* EXPOSE 8000: This command releases port 8000 within the container, where the Django app will run
* CMD python manage.py runserver: This command starts the server and runs the application

1. **Running the app in Docker**

To run the app, you need to perform two steps:

1. Build the image: This is done using the build command, which uses the Dockerfile you just created. To build the image, run the command below: docker build . -t dbms-project . This command should be executed in the directory where the Docker file lives. The -t flag tags the image so that it can be referenced when you want to run the container.
2. Run the image: This is done using the docker run command. This will convert the built image into a running container. To run the app, execute the below command:

docker run -it -p 8000:8000 dbms-project

You can proceed to view your app in the browser at localhost:8000.

1. **To check the containers which are in running state**

docker ps -a

Here a refers to all images which are in running state

1. **Uploading Docker Image to Docker Hub**

STEP 1: Login to Docker Hub from Command Line

To begin with, in order to login to docker hub from command line, make sure that the Docker service is enabled.

Open your Commandline and execute the following command:

docker login

This would ask you for your Docker Hub username and password. Just login by entering your credentials and it will prompt “Login Successful”.

STEP 2: Tag Docker Image

Next, you would tag Docker image so that it is uniquely identifiable when pulling the image. Let’s take the Docker image that I already have on my computer, named ubuntu. To attach a tag name, execute the following command:

docker tag dbms-project skanda18/newname:tag

* USERNAME is the username of the Docker Hub
* NEW\_NAME is the new name you may want to give to the image. It can remain the same too.
* TAG is the unique identifier. Usually, it’s given the version number of the project.

STEP 3: Push an Image to Docker Hub

Finally, we are ready to push local image to docker hub. Simply execute the following command to publish it to your repository.

Docker push skanda18/dbms-project:0.0.1

And you are done! Open Docker Hub account and see the published image there, as shown in the image below:

STEP 4: How to Pull Image from Docker Hub

To pull the image to any other machine, simply go to the image repository you pushed on Docker Hub. On the right-hand side, you will see the command to pull image from docker hub into your machine.

Simply copy and paste the command to your terminal or CMD and the image would get downloaded. From there just execute the ‘docker run IMAGE\_NAME:TAG’ command to run its container.