Vaishnavi Vishwas Pawar
vpawar@iu.edu

Assignment Report - Docker Practice

Q. Your previous familiarity with Docker

- Before starting this assignment, I had a basic understanding of Docker and its use for containerizing applications, but I had never used it in a real-world setting. I learned about Docker's role in containerizing applications and realized how critical it is for creating consistent development environments. I was also aware that Docker containers isolate an application and its dependencies, allowing it to run on multiple platforms, but I had never set up or managed a Docker environment myself. This assignment provided me with my first hands-on experience using Docker to pull images, create containers, and run applications in a virtualized environment, allowing me to gain a better understanding of the platform.

For this assignment, I successfully created an instance named Docker-practice-vpawar with the parameters mentioned in the assignment. I started a virtual machine on Jetstream2 using the **instance Docker-practice-vpawar (Fig. 1)** and installed Docker to begin practicing.

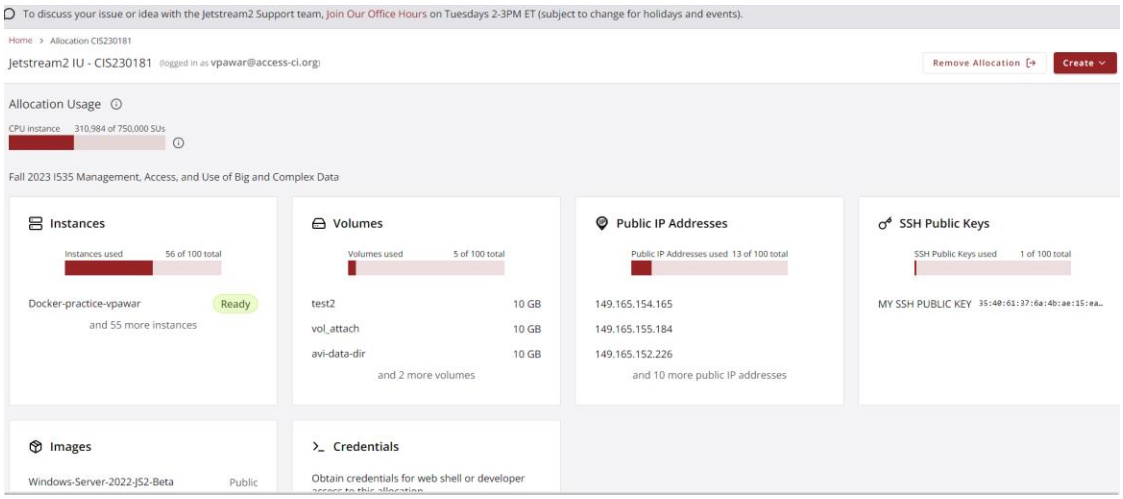

Fig. 1: Successfully created Instance Docker-practice-vpawar

My first task was to run the "hello-world" Docker container with the docker pull and docker run commands. This allowed me to test Docker's functionality on the virtual machine **(Fig. 2).** This initial setup was simple, and it helped me become more comfortable with Docker commands such as docker images for viewing locally stored images and docker run for creating and running containers **(Fig. 3)**.

Vaishnavi Vishwas Pawar
vpawar@iu.edu

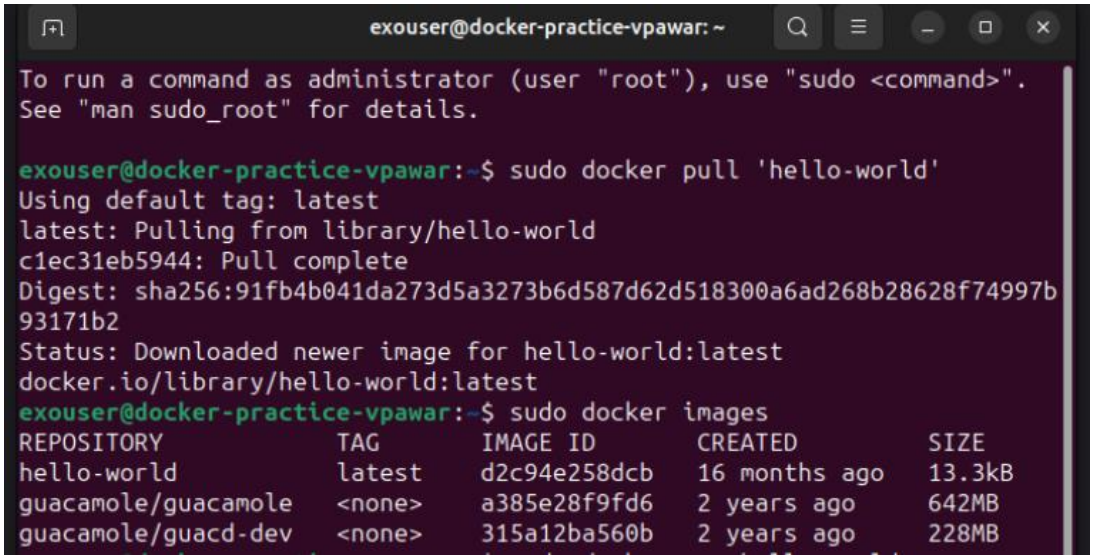

Fig. 2: Successfully ran Docker applications & checked the images present locally.

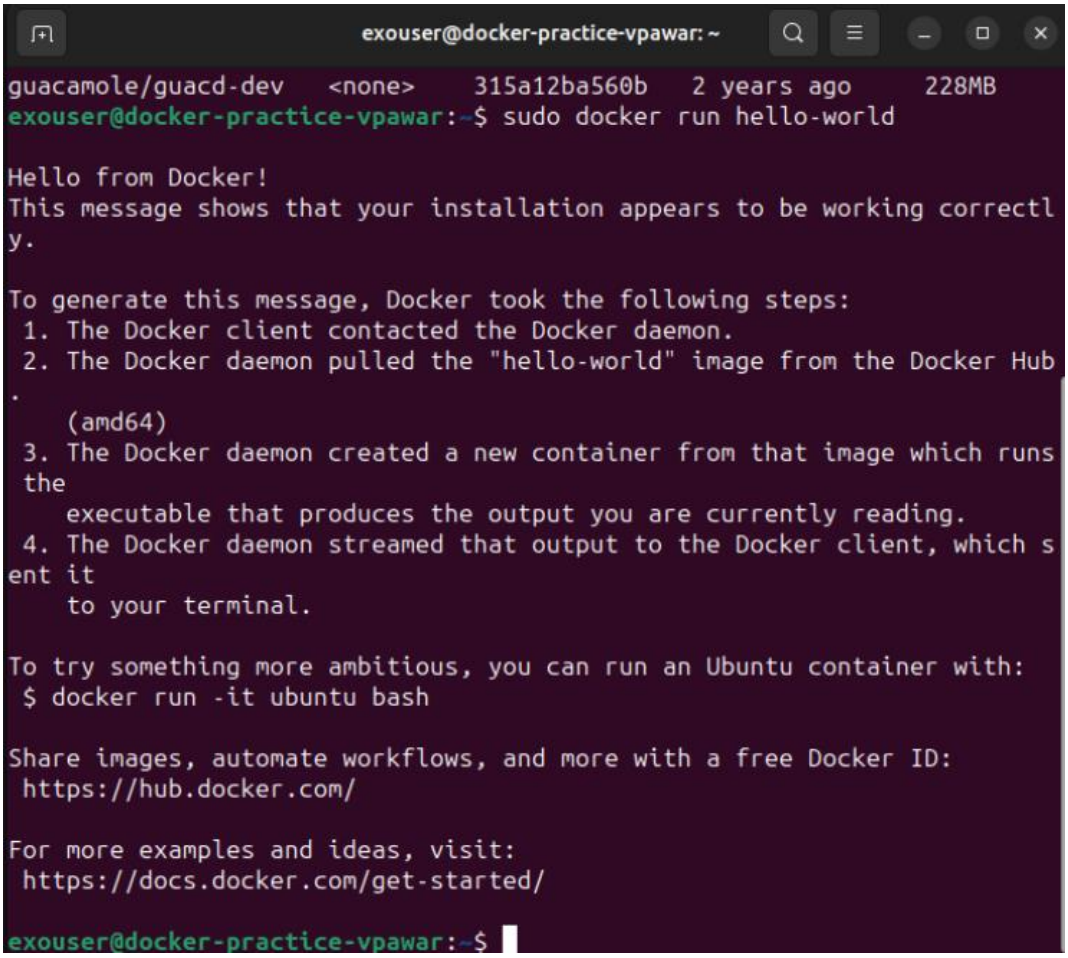

Fig. 3: Successfully created first container

Vaishnavi Vishwas Pawar
vpawar@iu.edu

Next, I followed the tutorial to create a simple containerized bulletin board application. I created a new **directory named docker-assignment and cloned the sample app repository from GitHub (Fig. 4)**. After navigating to the appropriate directory, I created the Docker image using the docker build command **(Fig. 5.1). Docker image build was successfully created (Fig. 5.2).**

```
exouser@docker-practice-vpawar:~$ mkdir docker-assignment
exouser@docker-practice-vpawar:~$ git clone https://github.com/dockersam
ples/node-bulletin-board.git
Cloning into 'node-bulletin-board'...
remote: Enumerating objects: 152, done.
remote: Total 152 (delta 0), reused 0 (delta 0), pack-reused 152 (from 1
)
Receiving objects: 100% (152/152), 190.11 KiB | 2.64 MiB/s, done.
Resolving deltas: 100% (69/69), done.
exouser@docker-practice-vpawar:~$ cd node-bulletin-board/bulletin-board-
app
```

Fig. 4: Created a directory and cloned the sample app from GitHub

```
exouser@docker-practice-vpawar: ~/node-bulletin-board/bull...
exouser@docker-practice-vpawar:~/node-bulletin-board/bulletin-board-app$
 sudo docker build .
DEPRECATED: The legacy builder is deprecated and will be removed in a fu
ture release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  45.57kB
Step 1/7 : FROM node:current-slim
current-slim: Pulling from library/node
a2318d6c47ec: Pull complete
faa3ebc12ad3: Pull complete
9f59ffbb720b: Pull complete
ef58851168f3: Pull complete
6aba5acc95e5: Pull complete
Digest: sha256:903eaf1ae555002624d07066b7ce506dc2fb67b6da3121255b40ff4dc
8e7e1b8
Status: Downloaded newer image for node:current-slim
 ---> 9729f62b36c7
Step 2/7 : WORKDIR /usr/src/app
 ---> Running in c571c78274b1
Removing intermediate container c571c78274b1
 ---> 88c0957acda1
Step 3/7 : COPY package.json .
 ---> dfc40b3994d8
Step 4/7 : RUN npm install
 ---> Running in 54fc1578551f
```

Fig. 5.1: Build the docker image using docker file in the directory using build command

Vaishnavi Vishwas Pawar
vpawar@iu.edu

```
15 packages are looking for funding
  run `npm fund` for details

2 vulnerabilities (1 moderate, 1 critical)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
Removing intermediate container 54fc1578551f
 ---> be67f28981b8
Step 5/7 : EXPOSE 8080
 ---> Running in 351848cbf856
Removing intermediate container 351848cbf856
 ---> f05736e2264a
Step 6/7 : CMD [ "npm", "start" ]
 ---> Running in 472f28164e72
Removing intermediate container 472f28164e72
 ---> 92c9638e7214
Step 7/7 : COPY . .
 ---> 7a5aaafbc339
Successfully built 7a5aaafbc339
```

Fig. 5.2: Docker images build successful

After the image was successfully built, I started the container with the **docker run -d -p 8080:8080 command (Fig. 6)**, which mapped the **host port 8080 to the container's port 8080**. This allows me to access the app through my browser.

```
exouser@docker-practice-vpawar:~/node-bulletin-board/bulletin-board-app$ sudo docker images
REPOSITORY          TAG           IMAGE ID       CREATED         SIZE
<none>              <none>        7a5aaafbc339   2 minutes ago   242MB
node                current-slim  9729f62b36c7   5 days ago      215MB
hello-world         latest        d2c94e258dcb   16 months ago   13.3kB
exouser@docker-practice-vpawar:~/node-bulletin-board/bulletin-board-app$
 sudo docker run -d -p 8080:8080 7a5aaafbc339
2880aca8a8c205e05505a531f20aa36d167106193978c35ff622d39ffb809815
exouser@docker-practice-vpawar:~/node-bulletin-board/bulletin-board-app$ sudo docker container ls
CONTAINER ID   IMAGE                 COMMAND                  CREATED             STATUS                    PORTS
                                        NAMES
2880aca8a8c2   7a5aaafbc339          "docker-entrypoint.s…"   39 seconds ago      Up 38 seconds             0.0.0.
0:8080->8080/tcp, :::8080->8080/tcp     gracious_mahavira
548360c6670c   guacamole/guacamole   "/opt/guacamole/bin/…"   About an hour ago   Up 36 minutes             0.0.0.
0:49528->8080/tcp, :::49528->8080/tcp   guacamole-exo-guac-guacamole-1
9ccdc09078c8   guacamole/guacd-dev   "/bin/sh -c '/usr/lo…"   About an hour ago   Up 36 minutes (healthy)   4822/t
```

Fig. 6: Verification of image build & successfully started application container

After successfully building and running the Docker container, I needed to ensure that the application was running properly. To accomplish this, I used the wget package, a command-line tool for retrieving content from web servers. Running the command wget http://localhost:8080/ confirmed that the application was responding, as the index.html file was successfully retrieved **(Fig. 7)**. This step **verified that the application was running** and accessible from within the virtual machine.

Vaishnavi Vishwas Pawar
vpawar@iu.edu

```
exouser@docker-practice-vpawar:~/node-bulletin-board/bulletin-board-app
$ wget http://localhost:8080/
--2024-09-23 01:26:44--  http://localhost:8080/
Resolving localhost (localhost)... 127.0.0.1
Connecting to localhost (localhost)|127.0.0.1|:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1826 (1.8K) [text/html]
Saving to: 'index.html.1'

index.html.1        100%[============>]   1.78K  --.-KB/s    in 0s

2024-09-23 01:26:44 (318 MB/s) - 'index.html.1' saved [1826/1826]
```

Fig. 7: Application is up and running using wget package

Next, I tried to **access the application through a web browser**. First, I launched a browser from the Jetstream2 desktop and navigated to http://localhost:8080. The application loaded successfully, indicating that the containerized app was running properly within the virtual machine **(Fig. 8.1)**. Following that, I checked the **app's accessibility from my local machine using the virtual machine's public IP address**. I entered http://149.165.154.82:8080 in my browser, 149.165.154.82 being my virtual machine's public IP Address, and the app loaded without issues **(Fig. 8.2)**. This demonstrated that the app was properly mapped to the appropriate ports and could be accessed externally, which was one of the assignment's primary objectives.

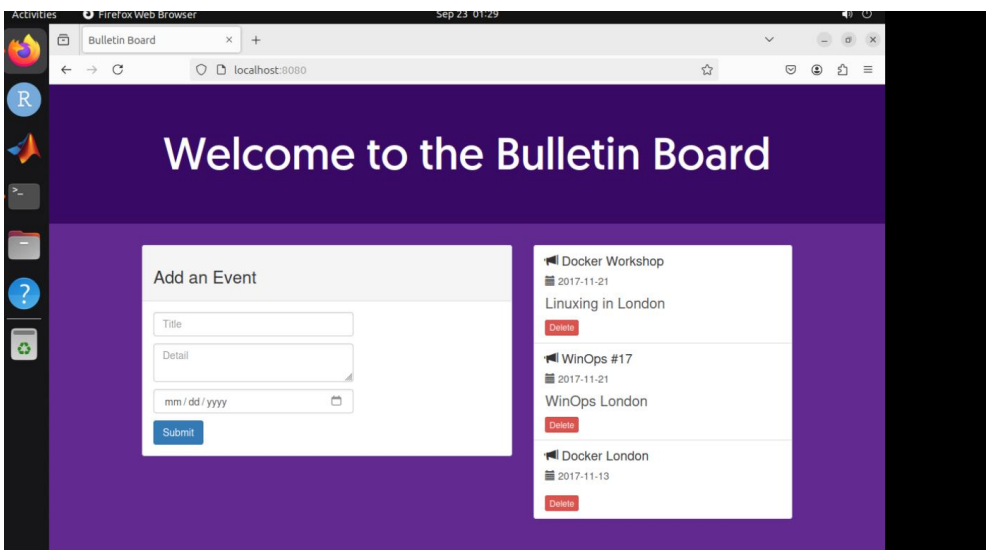

Fig. 8.1: Application is accessible on a browser using Jetstream desktop

Vaishnavi Vishwas Pawar
vpawar@iu.edu

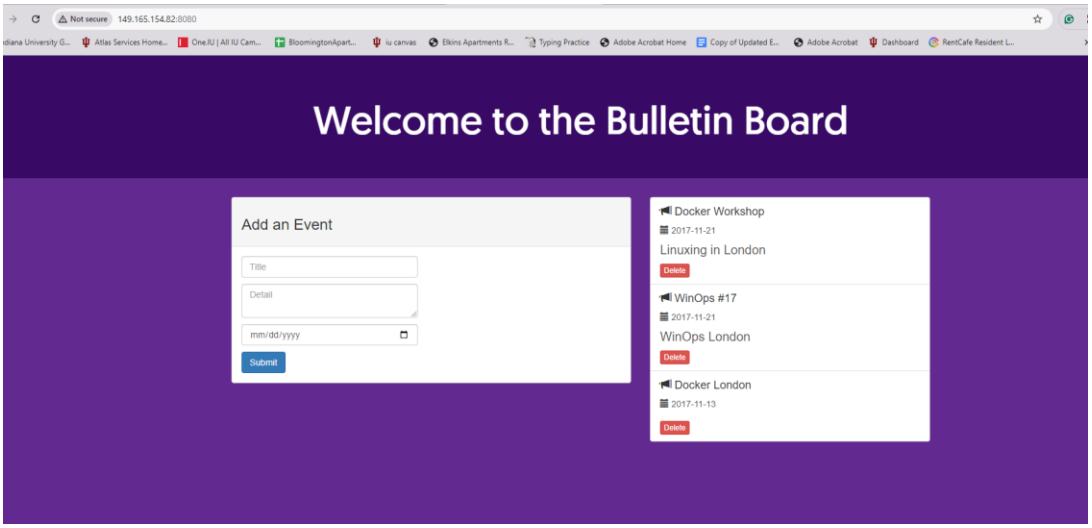

Fig. 8.2: Application is accessible on my own browser using IP address

After successfully completing the assignment, I shelved my instance Docker-practice-vpawar to free up resources for others use **(Fig. 9).**

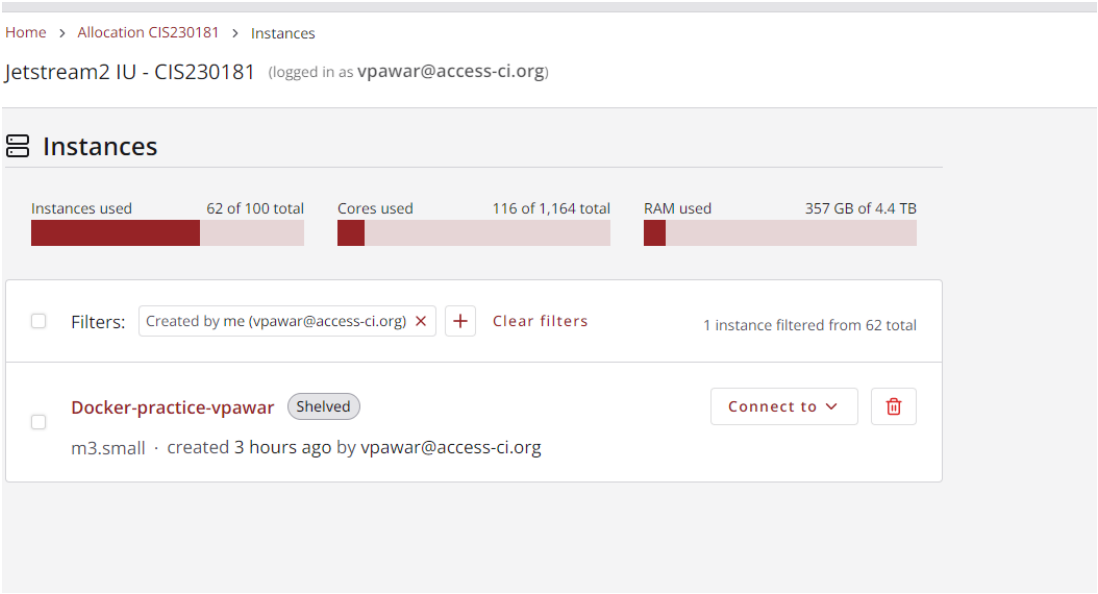

Fig. 9: Shelved the instance Docker-practice-vpawar

Vaishnavi Vishwas Pawar
vpawar@iu.edu

Q. Your experience with the assignment and difficulties you faced

- My overall experience with this assignment was positive and educational. Setting up Docker in a virtual machine on Jetstream2 gave me hands-on experience with Docker's core features, such as pulling images, creating containers, and developing a containerized application. The first steps, such as pulling and running the "hello-world" container, were simple and straightforward. However, I ran into some difficulties while creating the Docker image for the bulletin board app. The build process initially failed because the required dependencies were not installed on the virtual machine. This was due to some missing system-level packages that I had to manually install before the build could continue. This issue required some troubleshooting, as I had to search for the right package names and install them. After the dependencies were resolved, the build was completed successfully.

Q. Any troubleshooting that you needed to do

- One of the main troubleshooting tasks was to fix the Docker image build process. I ran into an error while building the bulletin board app because some system-level dependencies were missing. After reviewing the error logs, I discovered that some of the packages required by the app were not installed on the virtual machine. I used sudo apt-get install to install the missing packages, which resolved the issue and allowed me to finish the Docker build successfully. This experience demonstrated the importance of properly configuring the base environment before building an app within a container.

Q. Your understanding of the benefits of running a dockerized app

- My experience with this assignment has taught me to appreciate Docker's convenience and power. One of the most significant advantages is that Docker packs everything an application requires, including dependencies, libraries, and even the operating system, into a single container. This means that the app will function exactly the same regardless of where it is deployed, eliminating the frustration of "it works on my machine" issues. I also discovered that Docker makes managing and deploying apps much easier. Instead of manually installing everything on a new server, you can simply pull and launch a container in seconds. Furthermore, Docker is less resource-intensive than full virtual machines, allowing multiple containers to run on a single host with minimal overhead. Overall, Docker streamlines the process, whether testing locally or deploying to production, and it's easy to see why it's such a popular tool among developers.