

**PROFESSIONAL TRAINING REPORT**  
**at**  
**Sathyabama Institute of Science and Technology**  
**(Deemed to be University)**

Submitted in partial fulfillment of the requirements for the award of  
Bachelor of Engineering Degree in Computer Science and Engineering

By  
**TILAK CHAUHAN**  
**REG. NO. 39120129**



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**  
**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**  
**CHENNAI – 600119, TAMILNADU**

**NOVEMBER 2021**



**SATHYABAMA**  
**INSTITUTE OF SCIENCE AND TECHNOLOGY**  
**(DEEMED TO BE UNIVERSITY)**  
**Accredited with Grade "A" by NAAC**  
(Established under Section 3 of UGC Act, 1956)  
JEPPIAAR NAGAR, RAJIV GANDHI SALAI  
CHENNAI- 600119  
[www.sathyabama.ac.in](http://www.sathyabama.ac.in)



---

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **TILAK CHAUHAN (Reg. No: 39120129)** who carried out the project entitled "**WEBSERVER AS CONTAINER**" under my supervision from June 2021 to November 2021.

**Internal Guide**

**Dr. V. MARIA ANU – IT M.E., Ph.D**

**Head of the Department**

**Dr.R.Subhashini – IT M.E., Ph.D**

---

**Submitted for Viva voce Examination held on \_\_\_\_\_**

**InternalExaminer**

**ExternalExaminer**

## **DECLARATION**

I, **TILAK CHAUHAN** hereby declare that the project report entitled **WEBSERVER AS CONTAINER** done by me under the guidance of **Dr.V. MARIA ANU** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Information Technology.

**DATE:**13/11/2021

**PLACE:**CHENNAI

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. R. Subhashini, M.E., Ph.D., Heads of the Department of Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. V. MARIA ANU**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project



**Red Hat**

# Certificate of Participation

Tilak Chauhan

has attended and successfully completed

Red Hat Certified System Administrator, ON-LINE course

at Sathyabama Institute of Science and Technology

held from 1st Sep 2021 to 22nd Sep 2021



  
Authorised Signatory

S.No : 2021/ADV/11985

Date : 27th Oct 2021

# **ABSTRACT**

## Create a Web Server as Container

### **Description**

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

Docker is an open-source lightweight virtualization tool. It is containerizing platform in which user can run and deploy application and its dependencies and form containers to run over any linux infrastructure.

Apache web server is popular open source http web server tool which is widely used for deployment of webpages. It can be installed in any operating system.

### **Overview of Web services**

Web services, in the general meaning of the term, are services offered via the Web. In a typical Web services scenario, a user application sends a request to a service at a given URL using the protocol of HTTP. The service receives the request, processes it, and returns a response. An often-cited example of a Web service is that of static or dynamic pages in the web server and the response gives as what a user requested, it can be company information, advertisement of product and

services or connection to external application running in different machines.

Web services and consumers of Web services are typically businesses, making Web services predominantly business-to-business (B-to-B) transactions. An enterprise can be the provider of Web services and also the consumer of other Web services. For example, a wholesale distributor of spices could be in the consumer role when it uses a Web service to check on the availability of specific products and in the provider role when it supplies prospective customers with different vendors' prices of those products.

## TABLE OF CONTENTS

| CHAPTER<br>No. | TITLE                                                           | PAGE<br>No |
|----------------|-----------------------------------------------------------------|------------|
|                | ABSTRACT                                                        | 6          |
| <b>1</b>       | <b>INTRODUCTION</b>                                             | <b>10</b>  |
| <b>2.</b>      | <b>AIM AND SCOPE OF THE<br/>PRESENT PROJECT</b>                 | <b>11</b>  |
|                | 2.1 Aim                                                         | 11         |
|                | 2.2 Scope                                                       | 11         |
|                | 2.3 Accesebility                                                | 12         |
| <b>3.</b>      | <b>EXPERIMENTAL OR MATERIALS AND<br/>METHODS, COMMANDS USED</b> | <b>12</b>  |
|                | 3.1 Software Required                                           | 12         |
|                | 3.1.1 Redhat Enterprise Server 8.2                              |            |
|                | 3.1.2 Apache server                                             |            |
|                | 3.1.3 Docker installed on system                                |            |
|                | 3.1.4 Podman                                                    |            |
|                | 3.1.5 Virtual Box                                               |            |
|                | 3.2 Hardware Required                                           | 13         |
|                | 3.2.1 Intel machines                                            |            |
|                | 3.3 Commands                                                    | 14         |
| <b>4.</b>      | <b>RESULTS AND DISCUSSION</b>                                   | <b>15</b>  |
|                | 4.1 Result and discussion                                       |            |
|                | 4.3 Screen shots                                                |            |

|          |                                |           |
|----------|--------------------------------|-----------|
| <b>5</b> | <b>SUMMARY AND CONCLUSIONS</b> | <b>26</b> |
|          | <b>REFERENCES</b>              | <b>26</b> |

# CHAPTER - 1

## INTRODUCTION

### 1.1 ABOUT THE CONCEPT

Linux containers are technologies that allow you to package and isolate applications with their entire runtime environment—all of the files necessary to run. This makes it easy to move the contained application between environments (dev, test, production, etc.) while retaining full functionality. Containers are also an important part of IT security. By building security into containers pipeline and defending your infrastructure, you can make sure your containers are reliable, scalable, and trusted.

A container image comprises **system libraries, dependencies & other platform settings needed by the application** to run in diverse environments. Red Hat Linux provides a set of useful container tools that you can leverage to work directly with Linux containers using requiring docker commands.

Things you can do with containers:

You can deploy containers for a number of workloads and use cases—big to small. Containers give your team the underlying technology needed for a cloud-native development style, so you can get started with DevOps, CI/CD (continuous integration and continuous deployment), and even go serverless.

Container-based applications can work across highly-distributed cloud architectures. Application runtimes middleware provides tools to support a unified environment for development, delivery, integration, and automation.

You can also deploy integration technologies in containers, so you can easily scale how you connect apps and data, like real-time data streaming through Apache Kafka. If you're building a microservices architecture, containers are the ideal deployment unit for each microservice and the service mesh network that connects them.

Using containers might be the easiest decision ever.

## **CHAPTER - 2**

### **AIM AND SCOPE OF PRESENT PROJECT**

#### **2.1 AIM**

To create and manage the Apache webserver as a container and the container should be system startup service, it should host a webpage at the port 80 for all the client machine.

#### **2.2 SCOPE**

Linux containers help reduce conflicts between your development and operations teams by separating areas of responsibility. Developers can focus on their apps and operations teams can focus on the infrastructure. And, because Linux containers are based on open source technology, you get the latest and greatest advancements as soon as they're available. Container technologies—including CRI-O, Kubernetes, and Docker—help your team simplify, speed up, and orchestrate application development and deployment.

This project is going to be achieved by using a oracle virtual box, redhat linux 8.2, podman commands, apache webserver, dockerfile, internet.

#### **2.3 ACCESSIBILITY**

- Linux Containers are designed to support isolation of one or more applications.
- System-wide changes are visible in each container. For example, if you upgrade an application on the host machine, this change will apply to all sandboxes that run instances of this application.

- Since containers are lightweight, a large number of them can run simultaneously on a host machine. The theoretical maximum is 6000 containers and 12,000 bind mounts of root file system directories.

## CHAPTER 3

### EXPERIMENTAL OR MATERIALS AND METHODS

#### 3.1 SOFTWARE REQUIREMENT

- ❑ Red Hat Enterprise server 8.2
- ❑ Apache httpd server
- ❑ Docker installed on system
- ❑ Podman
- ❑ Virtual Box

##### 3.1.1 Red Hat Enterprise server 8.2

Red Hat Enterprise Linux is an enterprise Linux operating system, certified on hundreds of clouds and with thousands of vendors. Red Hat Enterprise Linux provides a consistent foundation across environments and the tools needed to deliver services and workloads faster for any application. Red Hat Enterprise Linux reduces deployment friction and costs while speeding time to value for critical workloads, enabling development and operations teams to innovate together in any environment.

##### 3.1.2 Apache server

HTTPD - Apache2 Web Server. Apache is the most commonly used Web server on Linux systems. Web servers are used to serve Web pages requested by client computers. Clients typically request and view Web pages using **Web** browser applications such as Firefox, Opera, Chromium, or Internet Explorer.

### 3.1.3 Docker Installed on the system

**Docker** is an open source project that automates the deployment of applications inside Linux Containers, and provides the capability to package an application with its runtime dependencies into a container. It provides a Docker CLI command line tool for the lifecycle management of image-based containers. Linux containers enable rapid application deployment, simpler testing, maintenance, and troubleshooting while improving security. Using Red Hat Enterprise Linux 7 with Docker allows customers to increase staff efficiency, deploy third-party applications faster, enable a more agile development environment, and manage resources more tightly.

### 3.1.4 Podman

Podman is now the **default container engine used by Red Hat Enterprise Linux** (RHEL), and it allows you to develop, run, and manage OCI containers on Linux — all without a daemon! Podman also has the ability to share containers and pods with Kubernetes, as well as create systemd containers and pods.

### 3.1.5 Oracle VM Virtual Box

Oracle VM VirtualBox is **cross-platform virtualization software**. It allows users to extend their existing computer to run multiple operating systems including Microsoft Windows, Mac OS X, Linux, and Oracle Solaris, at the same time.

## 3.2 HARDWARE REQUIREMENT

- ❑ Intel machines with 8 GB RAM/ 40 GB HDD - 5 NOS

### 3.2.1 Intel Machines

It is the **world's largest semiconductor chip manufacturer by revenue**, and is the developer of the x86 series of microprocessors, the processors found in most personal computers (PCs). Intel supplies microprocessors for computer system manufacturers such as Lenovo, HP, and Dell.

### 3.3 Commands Used

1. # yum module install -y container-tools[to install all the container related tools]
2. # dnf install podman[to install the podman, podman is the container manager]
3. # podman search httpd[to pull the httpd]
4. # podman images[to see all the images and select the required image]
5. # podman pull docker.io/library/httpd[to pull the docker container image]
6. # podman ps[to see all the containers]
7. # podman run -dt -p 8083:80/tcp[to run the apache server port 80]
8. # podman inspect docker.io/library/httpd[to inspect]
9. # setsebool -P container\_manage\_cgroup on[to use podman and systemctl to set Boolean between se linux]
10. # podman run -d --name httpd-server -p 8080:80 httpd[to run servername httpd-server on port 80]
11. # podman ps
12. # vim /etc/systemd/system/httpd-container.service[to edit the httpd-container.service]
13. # systemctl start httpd-container.service[to start the httpd container service]
14. # systemctl enable httpd-container.service[to enable the httpd container service]
15. # vim Dockerfile[to edit the dockerfile]
16. # podman build -t mysysd[to build the mysysd localhost]
17. # podman run -d --name=mysysd\_run -P 80:80 mysysd
18. #podman ps -a

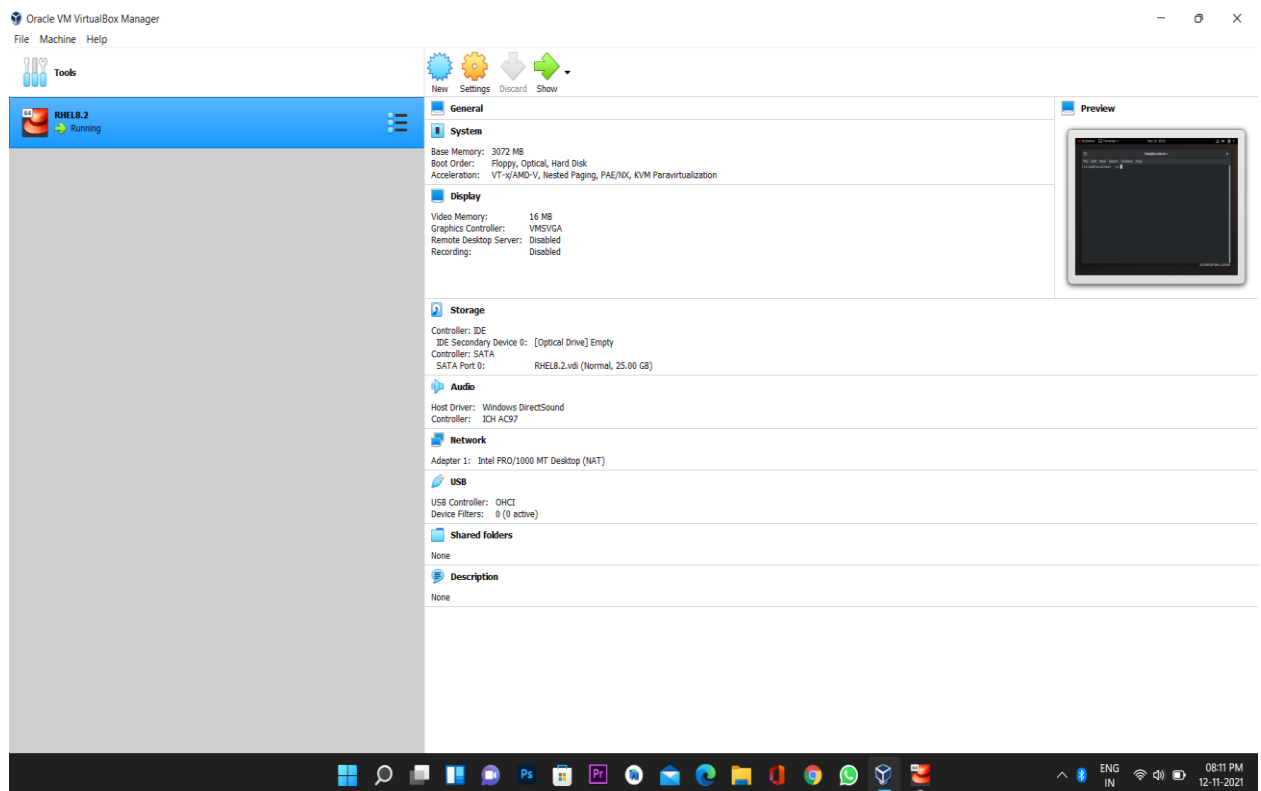
## CHAPTER 4

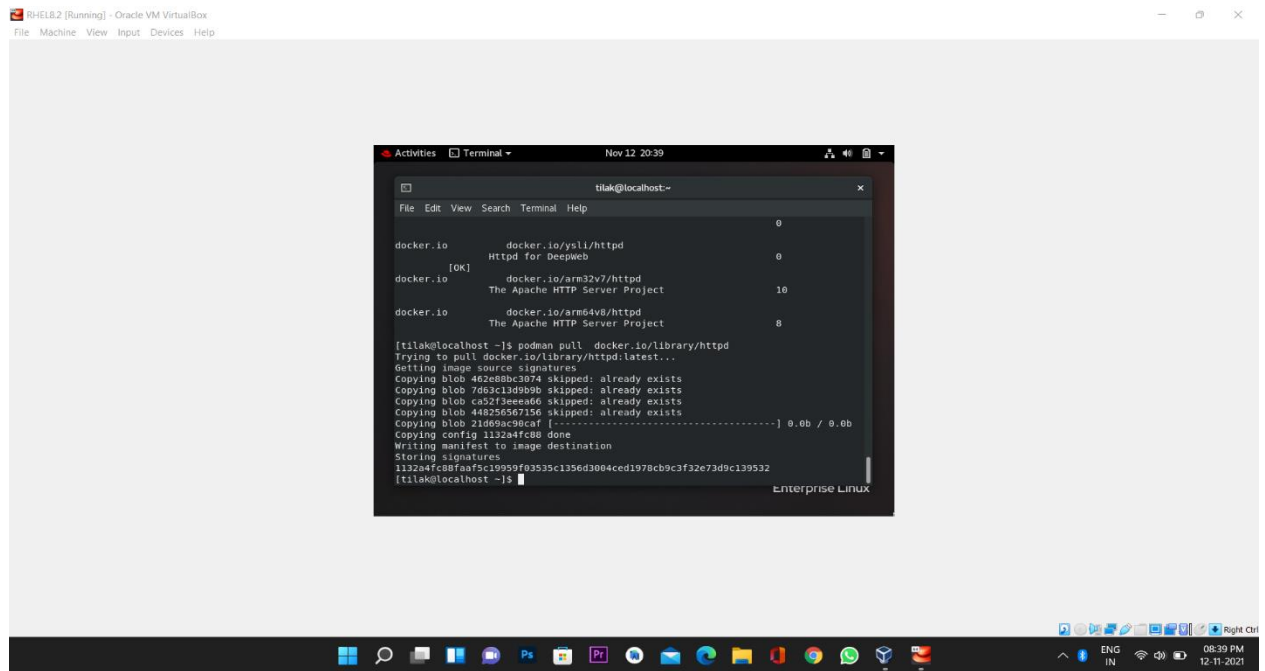
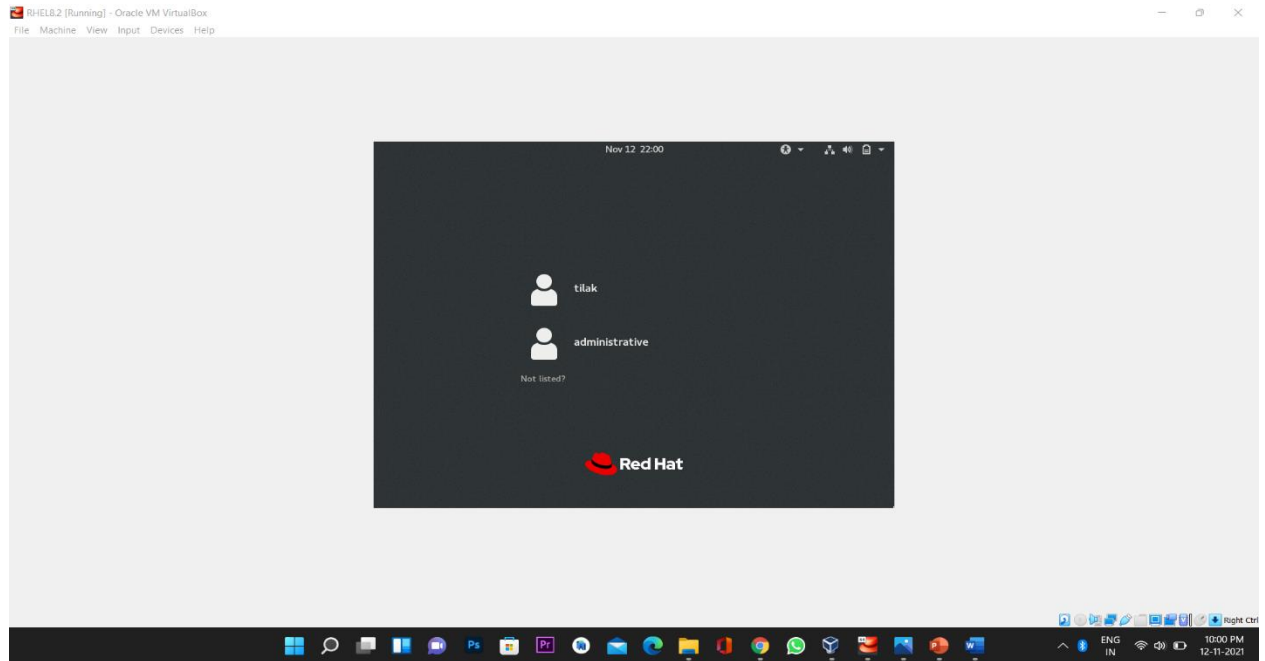
# RESULT AND DISCUSSION

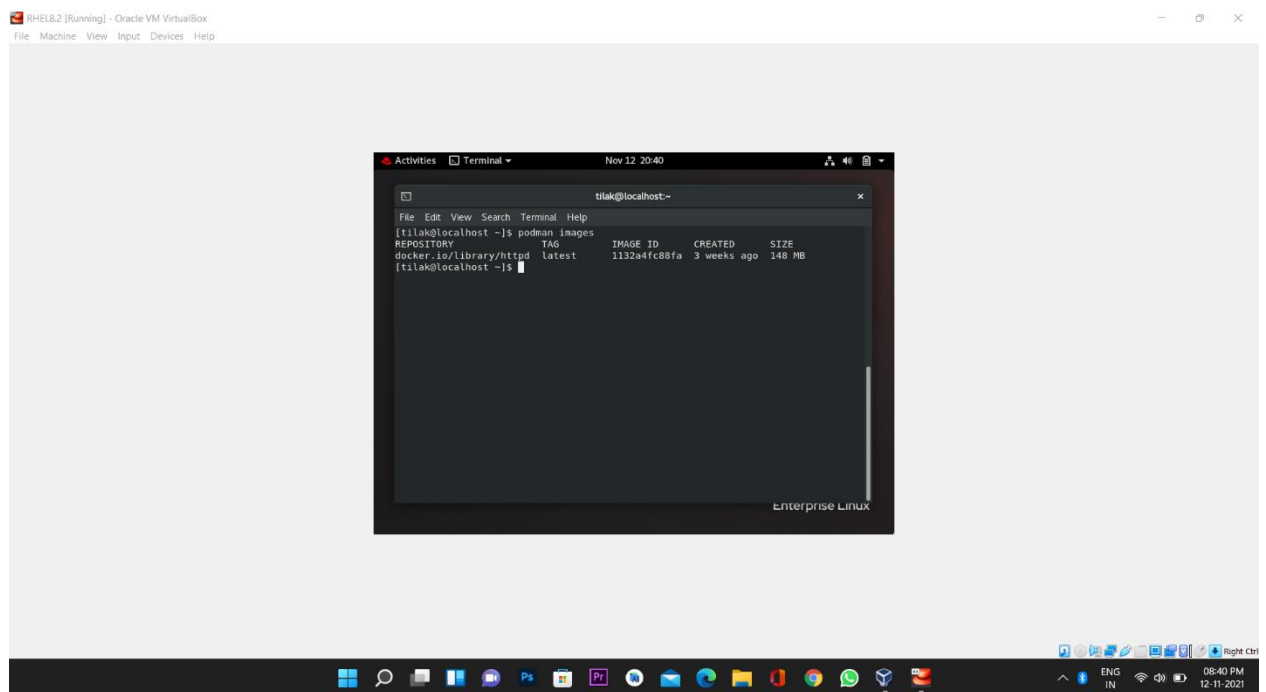
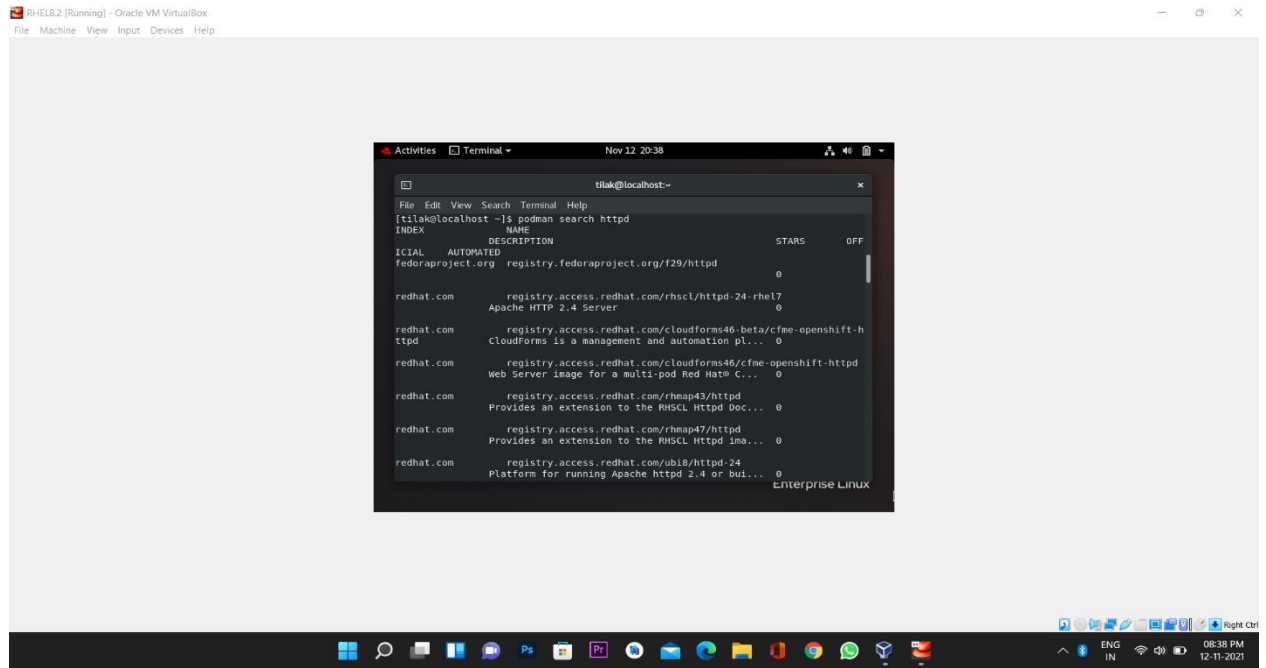
## 4.1 Result & Discussion

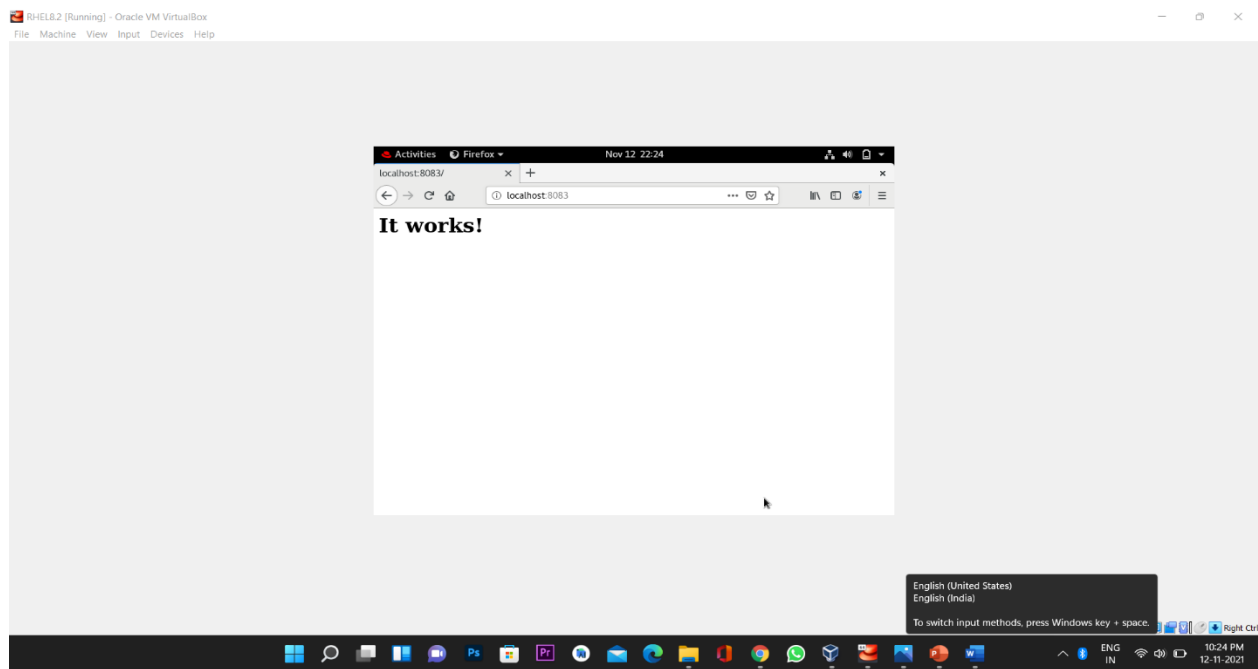
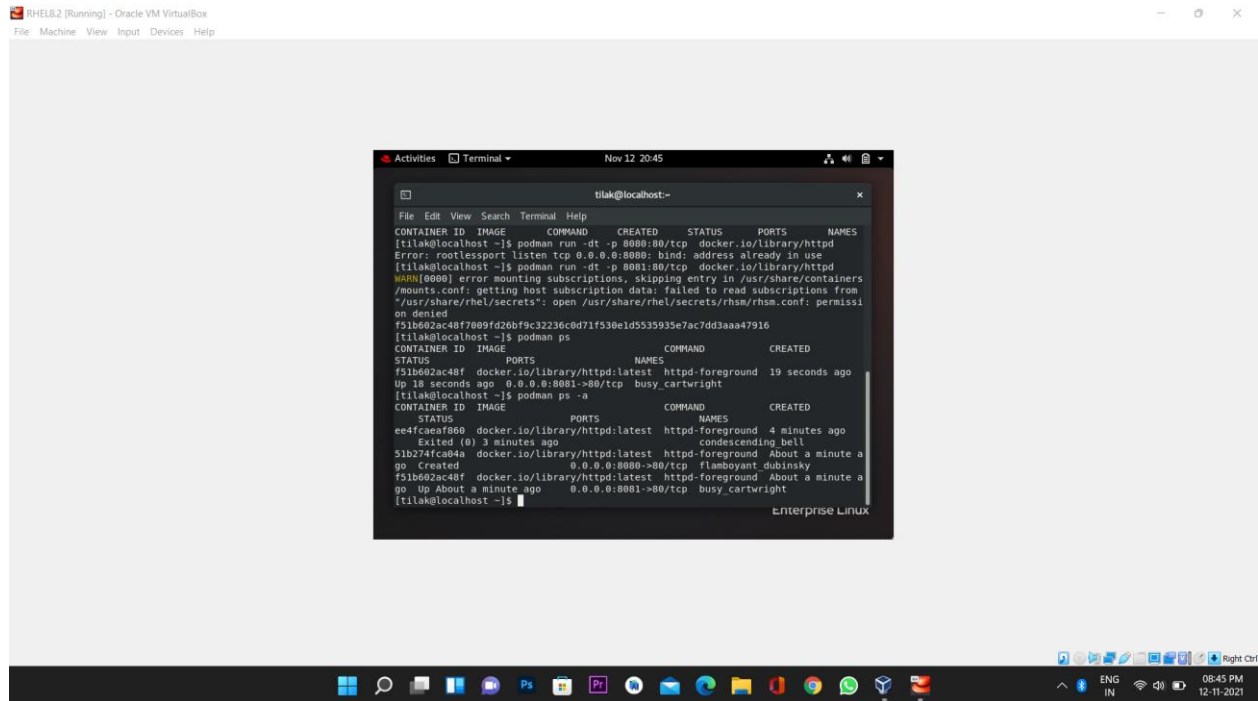
- By the end of the project I was able to create a webserver container and understood the complete concept of containers.
- Got to know how the technology behind the server application, linux administration works.
- Faced many technical challenges and simultaneously overcame it with great effort.
- Containers is very vast topic as well as interesting.

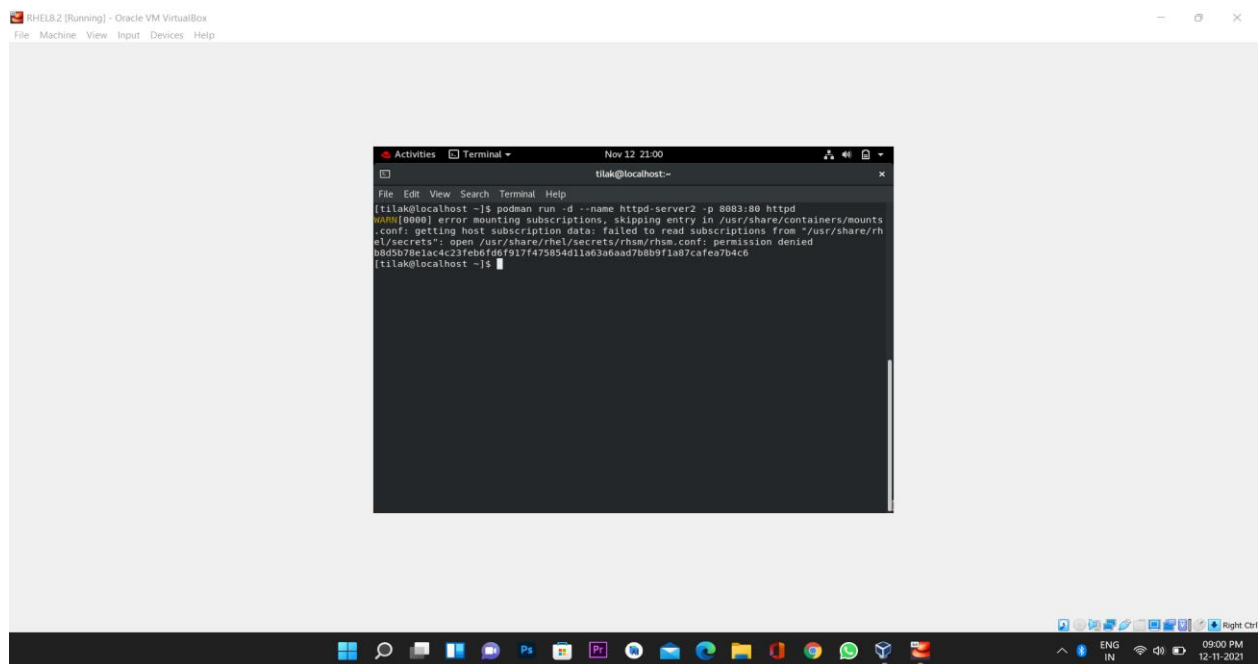
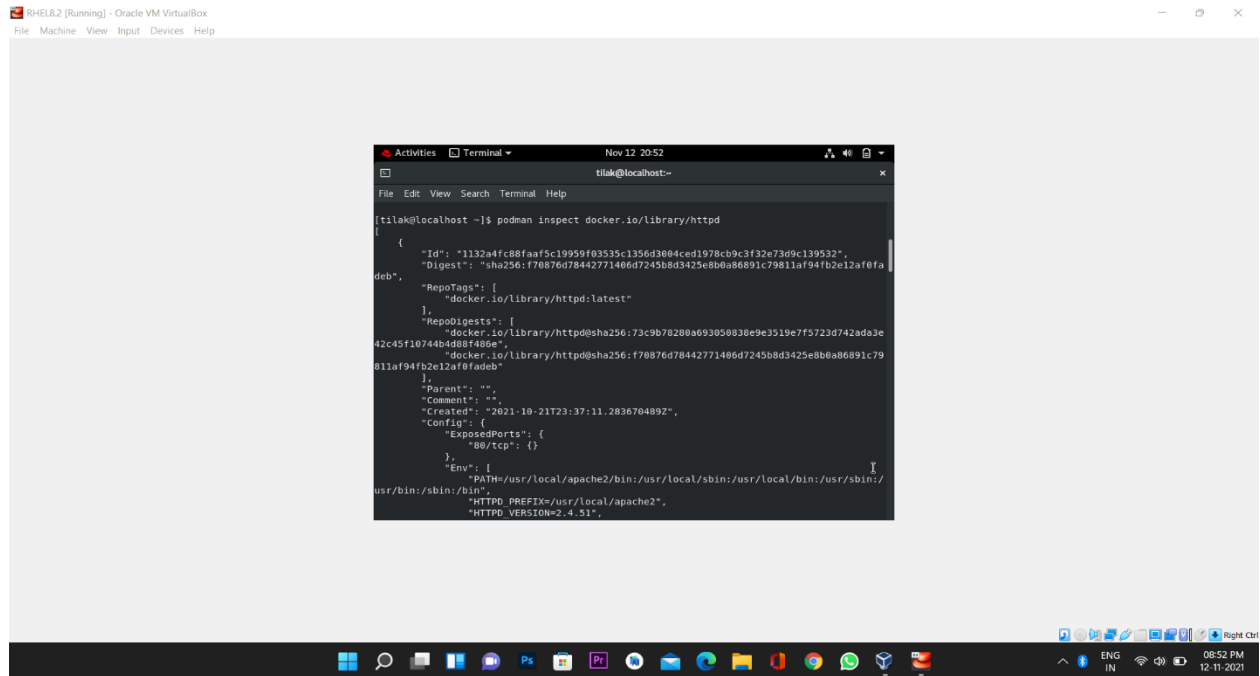
## 4.3 SCREENSHOTS

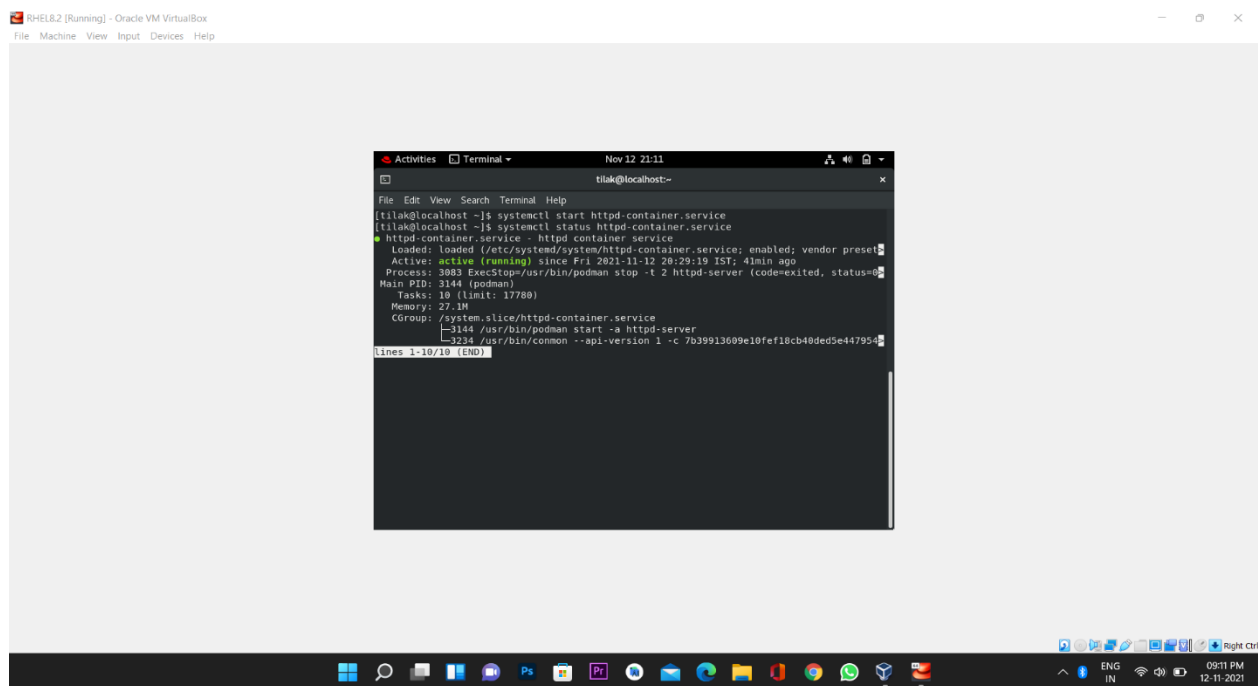
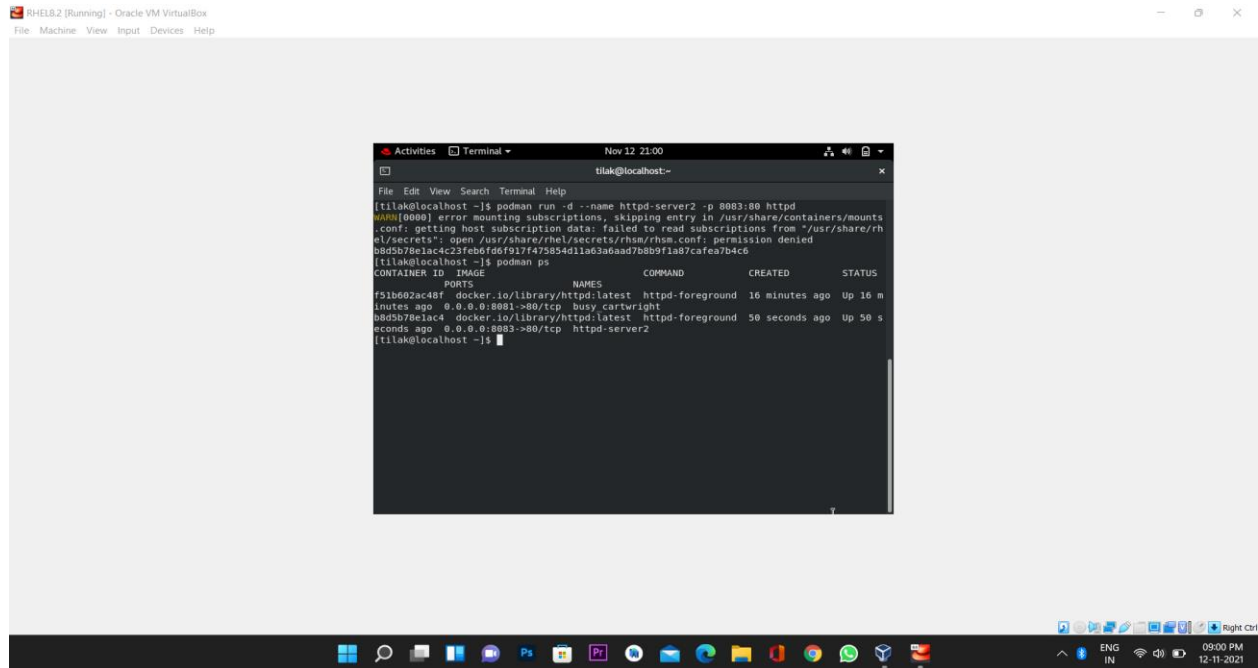


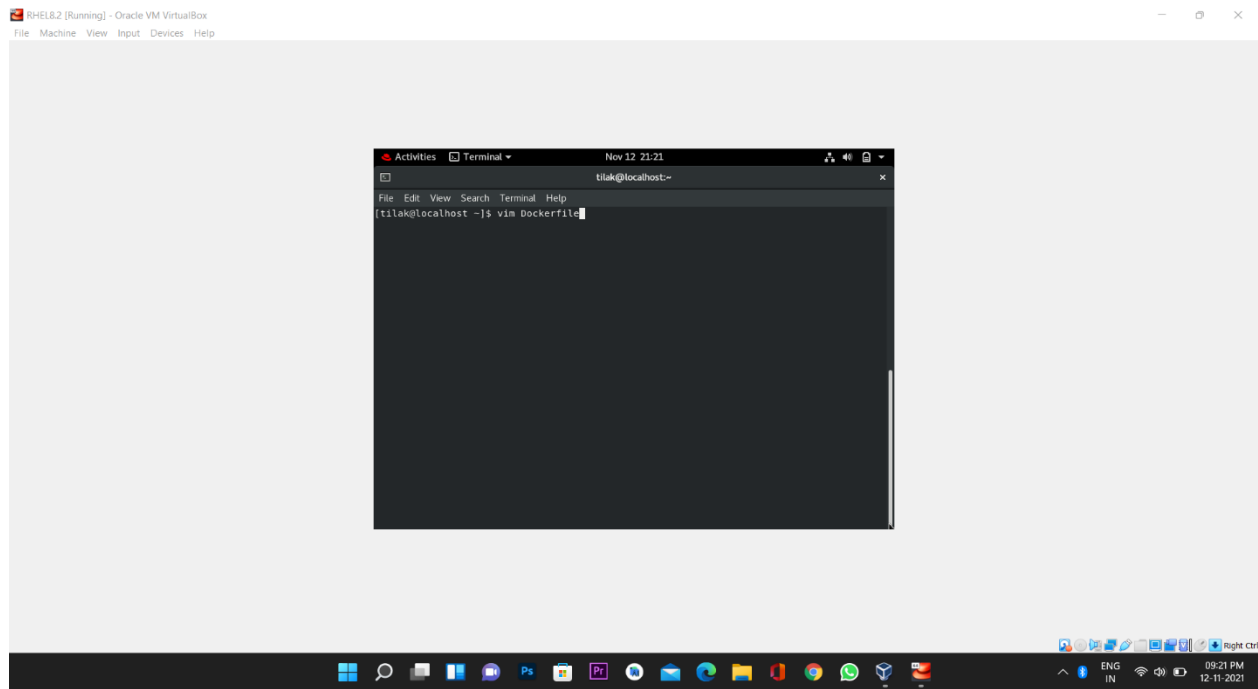
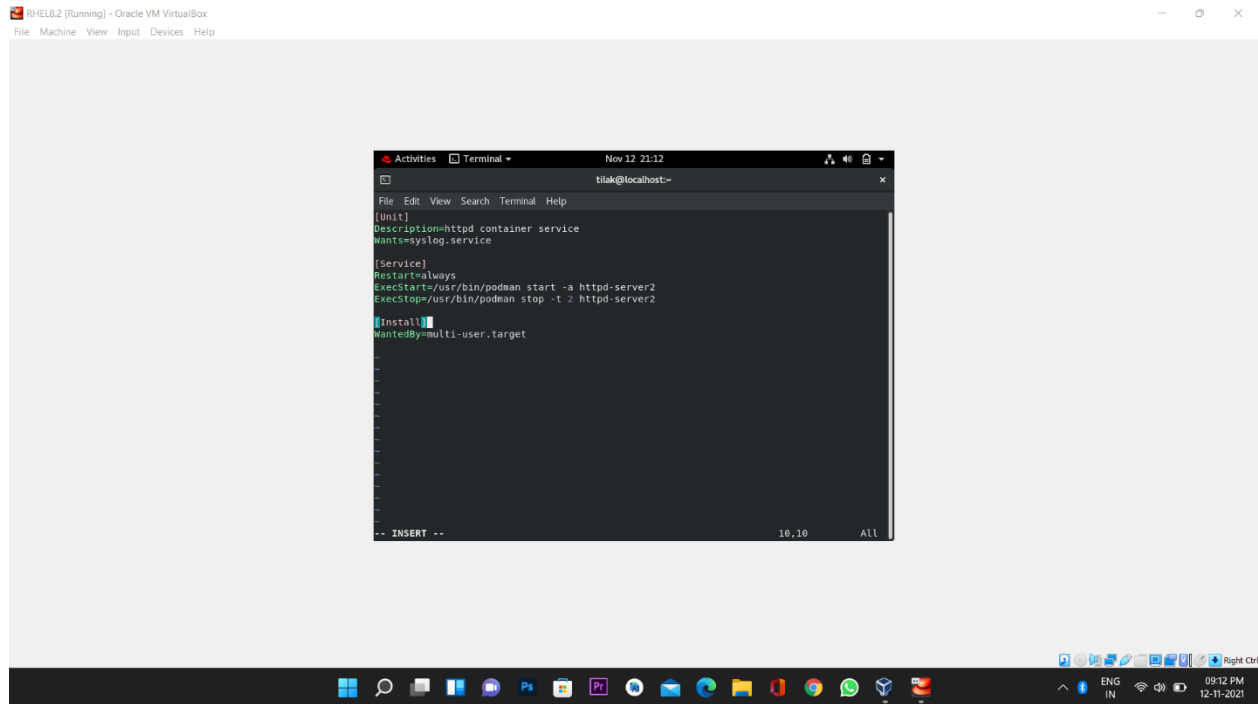






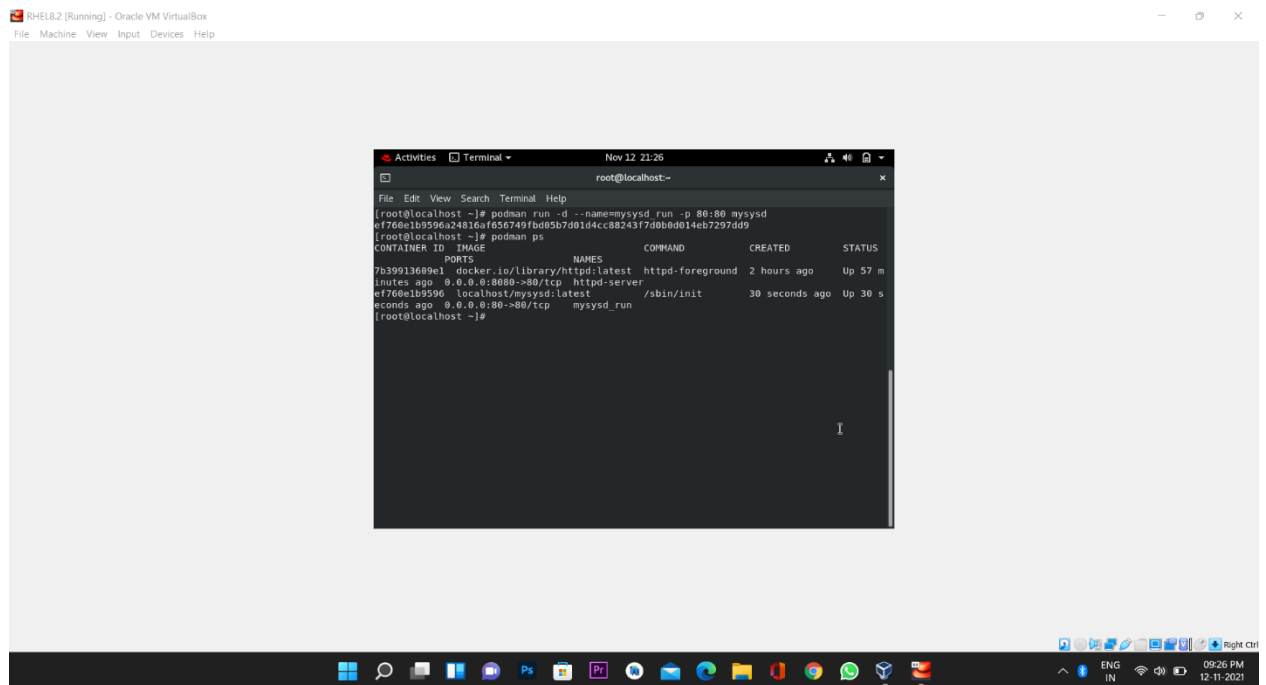
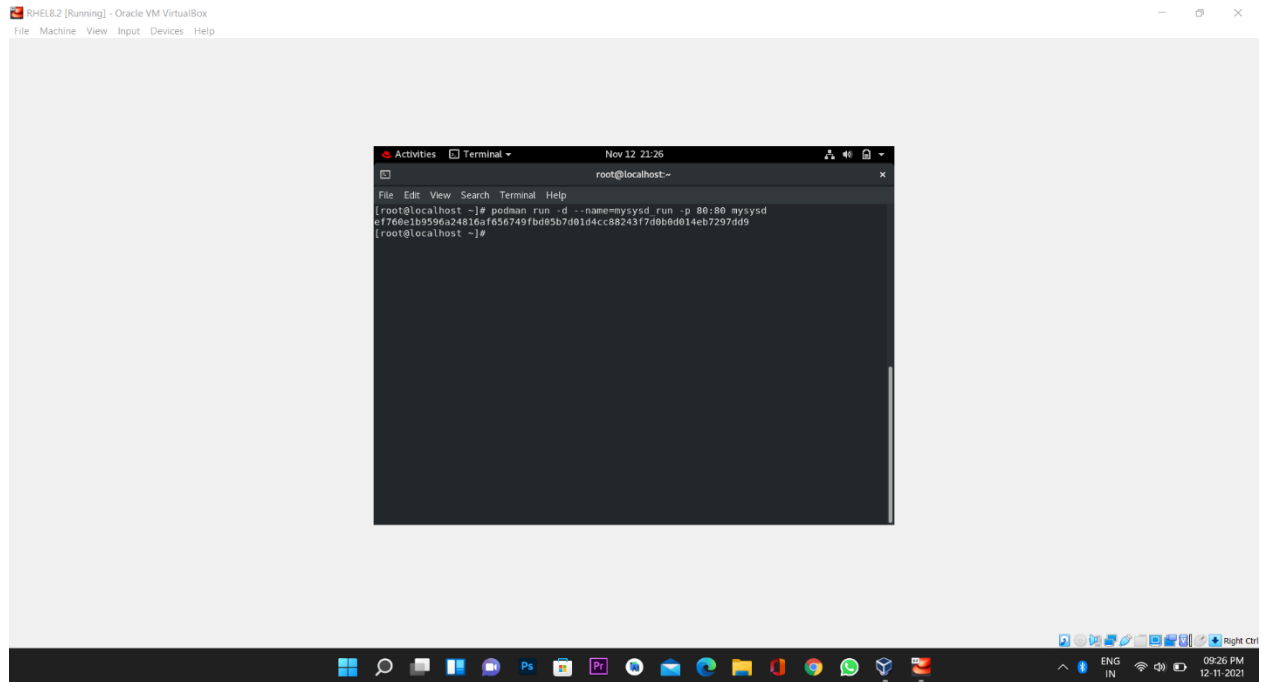


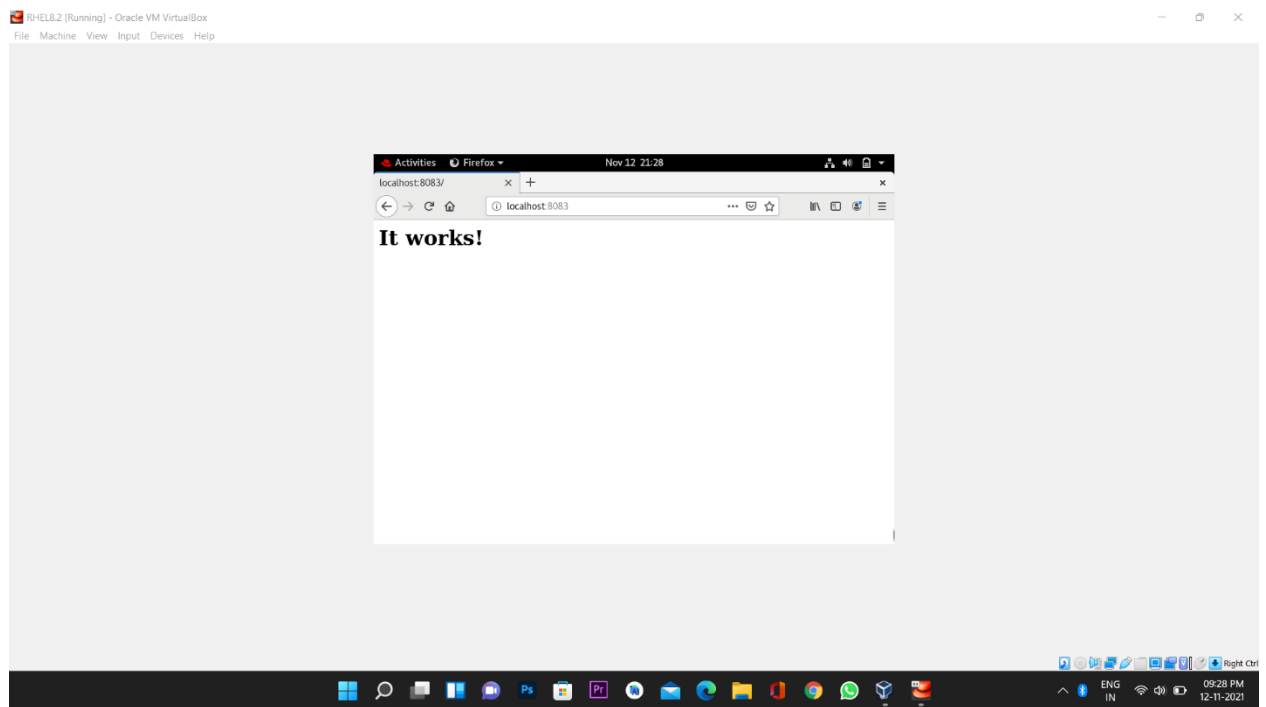
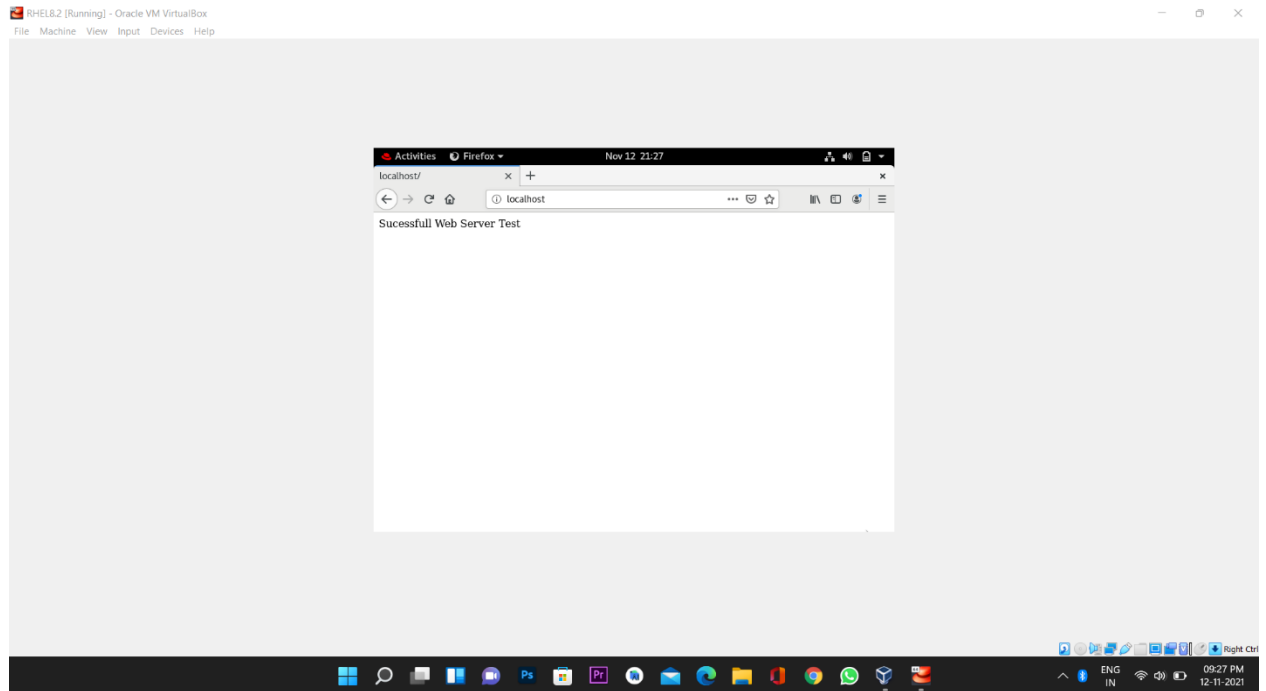




```
Nov 12 21:20
titak@localhost:~
File Edit View Search Terminal Help
FROM registry.access.redhat.com/ubi8/ubi-init
RUN yum -y install httpd; yum clean all; systemctl enable httpd;
RUN echo "Successful Web Server Test" > /var/www/html/index.html
RUN mkdir /etc/systemd/system/httpd.service.d; echo -e '[Service]\nRestart=always' > /
etc/systemd/system/httpd.service.d/httpd.conf
EXPOSE 80
-- INSERT --
```

```
Nov 12 21:24
root@localhost:~
File Edit View Search Terminal Help
[root@localhost ~]# podman build -t mysysd
STEP 1/5: FROM registry.access.redhat.com/ubi8/ubi-init
--> Using cache 92244713b2851c137b325e8d2a3932994d59dfe9926c9ca61a95f8dba0c2a22
--> 92244713b28
STEP 2/5: RUN yum -y install httpd; yum clean all; systemctl enable httpd;
--> Using cache b312dea333b3ce3d66c093f807473e3ecb4b4aac2005a8230a09ea122a0e3001
--> b312dea333b
STEP 3/5: RUN echo "Successful Web Server Test" > /var/www/html/index.html
--> Using cache dfb38a96c48027ad684d3395623da99d43e349ee78d4b046ab57c1f07cc38ad7
--> dfb38a96c48
STEP 4/5: RUN mkdir /etc/systemd/system/httpd.service.d; echo -e '[Service]\nRestart=a
lways' > /etc/systemd/system/httpd.service.d/httpd.conf
--> Using cache b102ba23e9a4277ec66609089bedcb4db04194cf19b943f21ae74b69957bbd88
--> b102ba23e9a
STEP 5/5: EXPOSE 80
--> Using cache b102ba23e9a4277ec66609089bedcb4db04194cf19b943f21ae74b69957bbd88
COMMIT mysysd
--> b102ba23e9a
Successfully tagged localhost/mysysd:latest
b102ba23e9a4277ec66609089bedcb4db04194cf19b943f21ae74b69957bbd88
[root@localhost ~]#
```





## CHAPTER 5

## **SUMMARY AND CONCLUSIONS**

At the last I would like to conclude that this project webserver as container just shows the beginning of the container technology and its capability, it depicts how curiously different project and technology of containers is. There are many day to day life example where containers play a huge role such as deploying cloud applications, daemon less tech, managing big cloud services, resources etc. This reduces the stress and opens a huge amount of opportunities to full fill the IT agenda.

## **REFERENCES**

- [Redhat.com](https://www.redhat.com)
- [Linux Command library](https://linuxcommandlibrary.com/)
- Practical sessions notes of the RHCSA training

