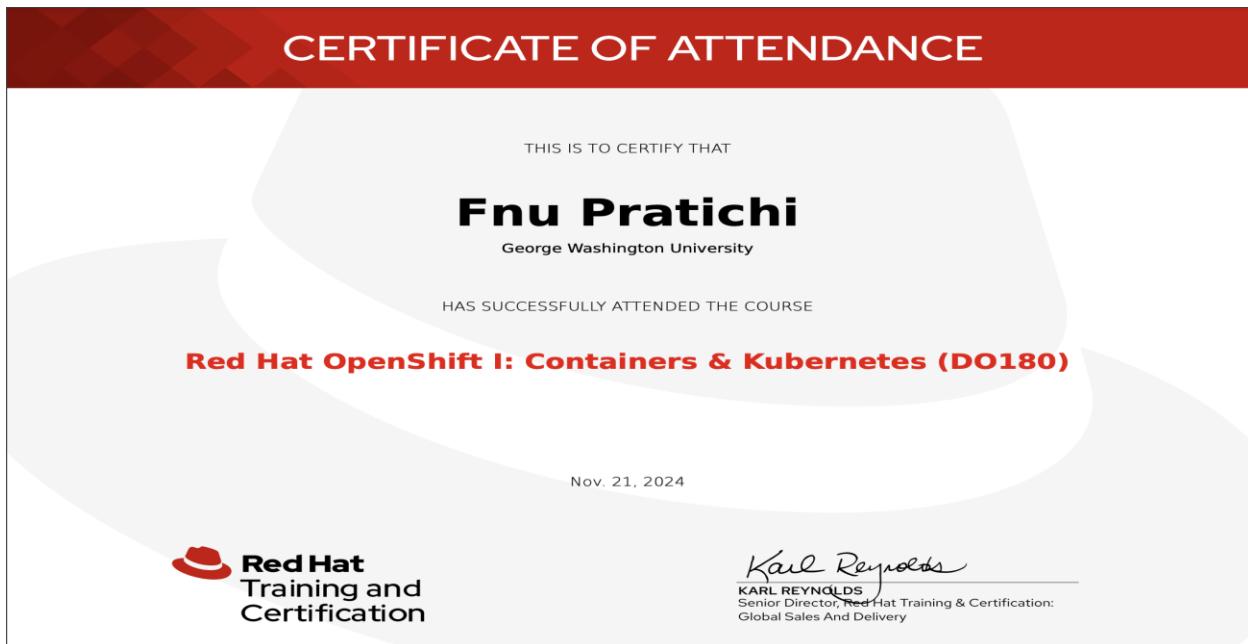


# Red Hat OpenShift I: Containers & Kubernetes 4.10



## Chapter 1 Introducing Container Technology

No labs

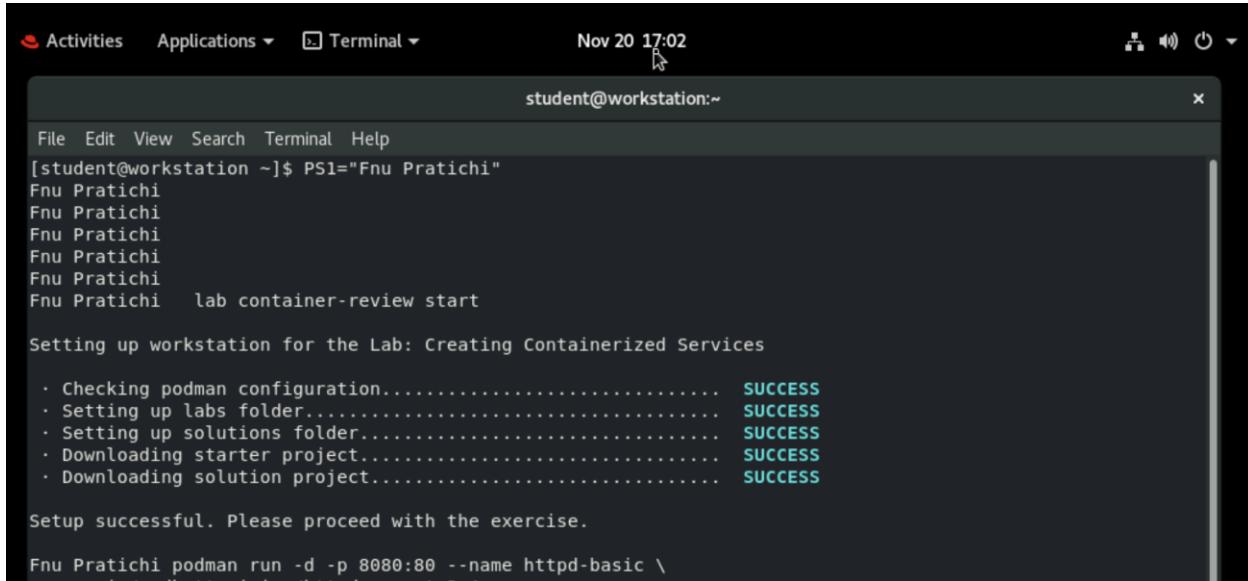
Summary

Chapter 1 introduces container technology, highlighting its advantages over traditional application deployment methods. Containers isolate applications and their dependencies, ensuring they run consistently across different environments without being impacted by OS changes. They offer benefits like low resource usage, quick deployment, environment isolation, and reusability, making them ideal for microservices. However, they may not be suitable for applications requiring direct hardware interaction.

-

## Chapter 2

### Lab: Creating Containerized Services



A screenshot of a Linux desktop environment showing a terminal window. The terminal title is "student@workstation:~". The window contains the following text:

```
File Edit View Search Terminal Help
[student@workstation ~]$ PS1="Fnu Pratichi"
Fnu Pratichi
Fnu Pratichi
Fnu Pratichi
Fnu Pratichi
Fnu Pratichi
Fnu Pratichi lab container-review start

Setting up workstation for the Lab: Creating Containerized Services

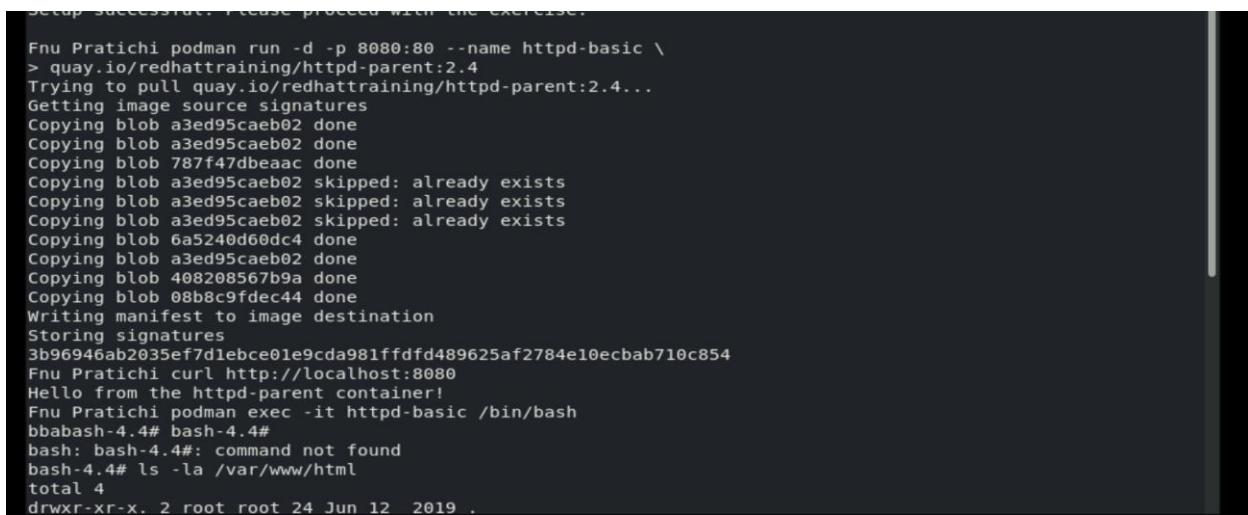
· Checking podman configuration..... SUCCESS
· Setting up labs folder..... SUCCESS
· Setting up solutions folder..... SUCCESS
· Downloading starter project..... SUCCESS
· Downloading solution project..... SUCCESS

Setup successful. Please proceed with the exercise.

Fnu Pratichi podman run -d -p 8080:80 --name httpd-basic \
```

### Screen Picture

Figure 1: Starting a container



A screenshot of a Linux terminal window showing the process of starting a container. The terminal title is "Setup successful! Please proceed with the exercise.". The window contains the following text:

```
Setup successful! Please proceed with the exercise.

Fnu Pratichi podman run -d -p 8080:80 --name httpd-basic \
> quay.io/redhattraining/httpd-parent:2.4
Trying to pull quay.io/redhattraining/httpd-parent:2.4...
Getting image source signatures
Copying blob a3ed95caeb02 done
Copying blob a3ed95caeb02 done
Copying blob 787f47dbeac done
Copying blob a3ed95caeb02 skipped: already exists
Copying blob a3ed95caeb02 skipped: already exists
Copying blob a3ed95caeb02 skipped: already exists
Copying blob 6a5240d60dc4 done
copying blob a3ed95caeb02 done
Copying blob 408208567b9a done
Copying blob 08b8c9fdec44 done
Writing manifest to image destination
Storing signatures
3b96946ab2035ef7d1ebce01e9cda981ffdfd489625af2784e10ecbab710c854
Fnu Pratichi curl http://localhost:8080
Hello from the httpd-parent container!
Fnu Pratichi podman exec -it httpd-basic /bin/bash
bbabash-4.4# bash-4.4#
bash: bash-4.4#: command not found
bash-4.4# ls -la /var/www/html
total 4
drwxr-xr-x. 2 root root 24 Jun 12 2019 .
```

Figure Description: The command `podman run -d -p 8080:80 --name httpd-basic` starts a container in detached mode using the `httpd` image. It maps port 8080 on the host to port 80 inside the container, allowing access to the container's web service through the host's port 8080. The container is named `httpd-basic`, making it easy to reference for management tasks.

```
-rw-r--r--- 1 root root 39 Jun 12 2019 index.html
bash-4.4# echo "Hello World" > /var/www/html/index.html
bash-4.4# exit
Fnu Pratichi curl http://localhost:8080
Hello World
Fnu Pratichi lab container-review grade

Grading the student's work for the Lab: Creating Containerized Services

· The httpd container image was pulled..... PASS
· The container was started with the correct name..... PASS
· The container was started with the correct image..... PASS
· Check the content of index.html..... PASS

Fnu Pratichi
Fnu Pratichi
Fnu Pratichi
Fnu Pratichi
Fnu Pratichi lab container-review finish

Completing the Lab: Creating Containerized Services

· Removing "httpd-basic" container..... SUCCESS
· Removing "redhattraining/httpd-parent:2.4" image..... SUCCESS
student@workstation:~
```

1 / 2

## Summary

This chapter covered how Podman enables searching and downloading images, running containers in detached or interactive modes, setting environment variables, and using the Red Hat Container Catalog for managing container images.

## Chapter 3

### Lab: Managing Containers

```
student@workstation:~ x

File Edit View Search Terminal Help
[student@workstation ~]$ ps1="Fnu Pratichi"
[student@workstation ~]$ lab manage-review start
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory

Setting up workstation for the Lab: Managing Containers

· Checking podman configuration..... SUCCESS
· Check that /home/student/local/mysql does not exist..... SUCCESS
· Setting up labs folder..... SUCCESS
· Setting up solutions folder..... SUCCESS
· Downloading starter project..... SUCCESS
· Downloading solution project..... SUCCESS

Setup successful. Please proceed with the exercise.

[student@workstation ~]$ PS1="FNU PRATICHI"
FNU PRATICHI mkdir -vp /home/student/local/mysql
mkdir: created directory '/home/student/local'
```

## Screen picture

Figure 2: Loading and Verifying Data in the Items Database

```
mysql> USE items;
Database changed
mysql> SELECT * FROM Item;
+----+-----+----+
| id | description | done |
+----+-----+----+
| 1  | Pick up newspaper | 0 |
| 2  | Buy groceries | 1 |
+----+-----+----+
2 rows in set (0.01 sec)

mysql> exit
```

Figure Description: The command loads the items database using the db.sql script. After loading, a SELECT query is executed to verify the contents of the Item table, displaying two records.

```
Grading the student's work for the Lab: Managing Containers
.
. Checking if the /home/student/local/mysql folder exists..... PASS
. Checking if owner was changed..... PASS
. Checking if the container mysql-1 was created..... PASS
. Checking if the mysql-2 container is running..... PASS
. Checking tables for the 'Finished lab' row..... PASS
FNU PRATICHI lab manage-review finish
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory

Completing the Lab: Managing Containers
.
. Removing the /home/student/local/mysql folder..... SUCCESS
. Removing the fcontext for /home/student/local/mysql..... SUCCESS
. Stopping mysql-1 container..... SUCCESS
. Removing mysql-1 container..... SUCCESS
. Stopping mysql-2 container..... SUCCESS
. Removing mysql-2 container..... SUCCESS
. Removing MySQL container image..... SUCCESS
. Removing temporary file..... SUCCESS
. Removing the project directory..... SUCCESS
. Removing the solution directory..... SUCCESS
FNU PRATICHI
```

## Summary

## Chapter 4

### Lab: Managing Images

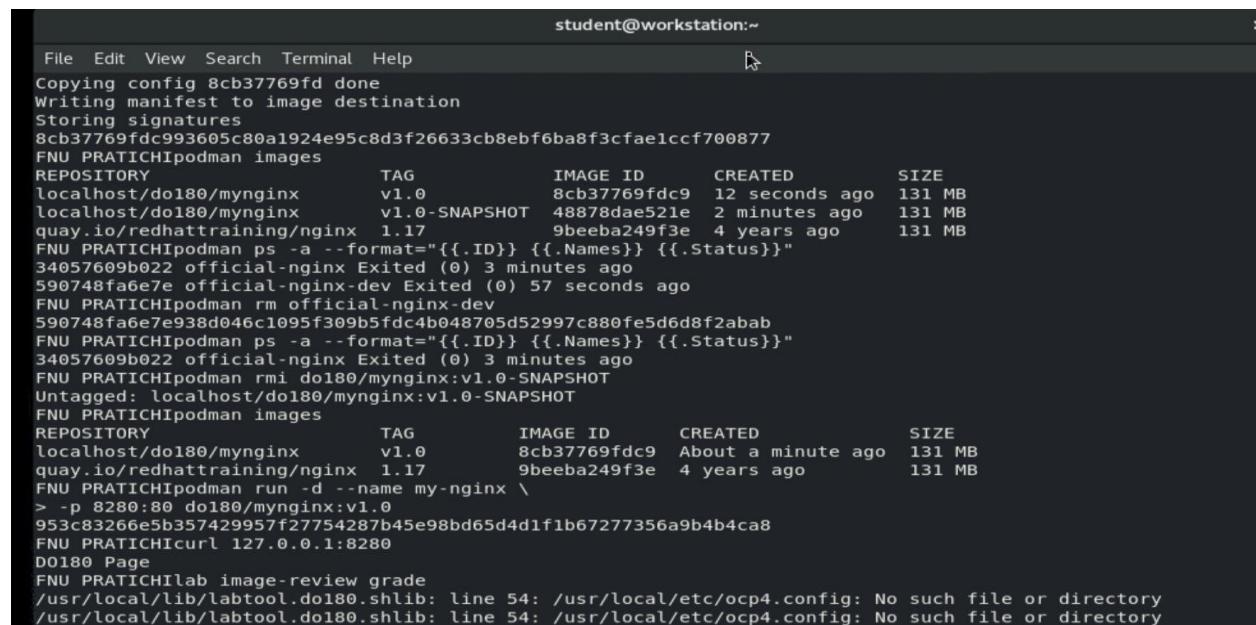
```
FNU PRATICHI lab image-review start
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory

Setting up workstation for the Lab: Managing Images

· Checking podman configuration..... SUCCESS
FNU PRATICHI podman pull quay.io/redhattraining/nginx:1.17
```

### Screen Picture

Figure 3: Managing Containers and Images with Podman



The screenshot shows a terminal window titled 'student@workstation:~'. The terminal displays the following sequence of commands and their outputs:

```
Copying config 8cb37769fd done
Writing manifest to image destination
Storing signatures
8cb37769fdc93605c80a1924e95c8d3f26633cb8ebf6ba8f3cfaf700877
FNU PRATICHI podman images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
localhost/do180/mynginx    v1.0      8cb37769fdc9  12 seconds ago  131 MB
localhost/do180/mynginx    v1.0-SNAPSHOT 48878dae521e  2 minutes ago  131 MB
quay.io/redhattraining/nginx  1.17      9beeba249f3e  4 years ago   131 MB
FNU PRATICHI podman ps -a --format="{{.ID}} {{.Names}} {{.Status}}"
34057609b022 official-nginx Exited (0) 3 minutes ago
590748fa6e7e official-nginx-dev Exited (0) 57 seconds ago
FNU PRATICHI podman rm official-nginx-dev
590748fa6e7e938d046c1095f309b5fdc4b048705d52997c880fe5d6d8f2abab
FNU PRATICHI podman ps -a --format="{{.ID}} {{.Names}} {{.Status}}"
34057609b022 official-nginx Exited (0) 3 minutes ago
FNU PRATICHI podman rmi do180/mynginx:v1.0-SNAPSHOT
Untagged: localhost/do180/mynginx:v1.0-SNAPSHOT
FNU PRATICHI podman images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
localhost/do180/mynginx    v1.0      8cb37769fdc9  About a minute ago  131 MB
quay.io/redhattraining/nginx  1.17      9beeba249f3e  4 years ago   131 MB
FNU PRATICHI podman run -d --name my-nginx \
> -p 8280:80 do180/mynginx:v1.0
953c83266e5b357429957f27754287b45e98bd65d4d1f1b67277356a9b4b4ca8
FNU PRATICHI curl 127.0.0.1:8280
D0180 Page.
FNU PRATICHI lab image-review grade
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
```

Figure Description: This figure demonstrates the process of listing all containers, removing a stopped container using podman rm, and deleting an image with podman rmi. It shows the verification steps before and after removal of both the container and image.

```
Grading the student's work for the Lab: Managing Images

· Nginx container image is pulled..... PASS
· Container official-nginx is created..... PASS
· Container official-nginx is stopped..... PASS
· Tag do180/mynginx:v1.0-SNAPSHOT is removed..... PASS
· Tag do180/mynginx:v1.0 is created..... PASS
· Container my-nginx is created and running..... PASS
· index.html is available with the custom content..... PASS

FNU PRATICHI lab image-review finish
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory

Completing the Lab: Managing Images

· Stopping official-nginx-dev container..... SUCCESS
· Removing official-nginx-dev container..... SUCCESS
· Stopping official-nginx container..... SUCCESS
· Removing official-nginx container..... SUCCESS
· Removing do180/mynginx:v1.0 image..... SUCCESS
· Removing quay.io/redhattraining/nginx:1.17 image..... SUCCESS

FNU PRATICHI
```

```
[student@workstation:~]
```

## Summary

In this chapter, we learned that the Red Hat Container Catalog offers certified images, Podman interacts with remote registries for managing images, and tags support multiple releases. We can also create images using podman commit.

## Chapter 5

### Lab: Creating Custom Container Images

```
student@workstation:~/DO180/labs/dockerfile-review

File Edit View Search Terminal Help
FNU PRATICHIclear

FNU PRATICHI lab dockerfile-review start
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory

Setting up workstation for the Lab: Creating Custom Container Images

· Checking podman configuration..... SUCCESS
· Checking if the container exists or is running..... SUCCESS
· Downloading starter project..... SUCCESS
· Downloading solution project..... SUCCESS

Setup successful. Please proceed with the exercise.
```

## Screen Picture

Figure 4: Verifying the Server Response with curl Command

```
FNU PRATICHI curl -s 127.0.0.1:20080
<html>
<header><title>D0180 Hello!</title></header>
<body>
    Hello World! The containerfile-review lab works!
</body>
</html>
```

Figure Description: The output from the curl command executed on 127.0.0.1:20080 shows the HTML page returned by the server, confirming the server's proper functionality.

```
Grading the student's work for Lab: Creating Custom Container Images
· Check for apache container..... PASS
· Check if the container is running..... PASS
· Verifying if httpd was installed correctly in apache image.. PASS
· Verifying if hello world is running on server..... PASS

FNU PRATICHI lab dockerfile-review finish
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory
/usr/local/lib/labtool.do180.shlib: line 54: /usr/local/etc/ocp4.config: No such file or directory

Completing the Lab: Creating Custom Container Images
· Stopping grade container..... SUCCESS
· Removing grade container..... SUCCESS
· Stopping running container..... SUCCESS
· Removing the container..... SUCCESS
· Removing do180/custom-apache container image..... SUCCESS
· Removing ubi8 container image..... SUCCESS

FNU PRATICHI
```

## Summary

In this chapter, we learned that a Containerfile outlines the instructions to build a container image. Red Hat Container Catalog or Quay.io images serve as good bases for custom images. The Source-to-Image (S2I) process simplifies building container images from application source code, allowing developers to focus on app development rather than Containerfile creation.

## Chapter 6

### Lab: Deploying Containerized Applications on OpenShift

```
student@workstation:~  
File Edit View Search Terminal Help  
[student@workstation ~]$ PS1="Fnu pratichi"  
Fnu pratichi lab openshift-review start  
Setting up workstation for the Lab: Deploying Containerized Applications on OpenShift  
Verifying the OpenShift cluster is running:  
· Log in on OpenShift..... SUCCESS  
· Check the internal registry is up and running..... SUCCESS  
· Ensuring the 'vadqmd-ocp' project does not exist..... SUCCESS
```

### Screen Picture

Figure 5: Creating and Deploying a Temperature Converter Application in OpenShift

```
student@workstation:~  
File Edit View Search Terminal Help  
Getting image source signatures  
Copying blob sha256:dedbd3f1d6397be3ec8dfc8bcc580a6baae0a5e14098540e03149a31de588bf  
Copying blob sha256:22839c7f91536af8647d130eb072139effc9a907d602e62cdab9a615144f5039  
Copying blob sha256:006ebe10a6353ffe3643faa0c514e8c0d44c8e9304c92ff91500d40e4ca32122  
Copying blob sha256:982238823a7311bb20522ced89715e7696d8c16891525844a5b8667cc9e874c2  
Copying blob sha256:ec121c671d0c70cf8d56aef452dbeec1508bc2b40c4515051987805796698fc21  
Copying config sha256:cae7a758ea324f02cdb6a66ba9983c9873942bb2574695d190481ff9b3c34082  
Writing manifest to image destination  
Storing signatures  
Successfully pushed image-registry.openshift-image-registry.svc:5000/vadqmd-ocp/temps@sha256:b699b3067366611c9  
41a7932944e7930947e118073a77aa6c1049ce8a69eb1fe  
Push successful  
Fnu pratichi oc get pods -w  
NAME READY STATUS RESTARTS AGE  
temps-1-build 0/1 Completed 0 57s  
temps-787f57dd8c-cksv4 1/1 Running 0 30s  
  
^C  
Fnu pratichi oc expose svc/temps  
route.route.openshift.io/temps exposed  
Fnu pratichi oc get route/temps  
NAME HOST/PORT PATH SERVICES PORT TERMINATION WILDCARD  
temps temps-vadqmd-ocp.apps.na410r.prod.ole.redhat.com temps 8080-tcp None
```

Figure Description: This figure demonstrates the process of creating a temperature converter application named "temps" using the php:7.3 image stream tag in OpenShift. It includes steps for setting up the application from the Git repository, tracking build progress, verifying deployment, and exposing the service for external access. The application is successfully built and deployed, and an external route is created for user access.

```
Fnu pratichilab openshift-review grade
Grading the student's work for the Lab: Deploying Containerized Applications on OpenShift
  · Log in on OpenShift..... SUCCESS
  Accessing the web application..... PASS
Fnu pratichilab openshift-review finish

Completing the Lab: Deploying Containerized Applications on OpenShift

  · Log in on OpenShift..... SUCCESS
  · Deleting the vadqmd-ocp project..... SUCCESS
Fnu pratichi
student@workstation:~ 1 / 2
```

## Summary:

In this chapter, we learned that OpenShift stores resource definitions in etcd, which include Pods, Persistent Volumes (PVs), Routes, and Build Configurations (BCs). You can use the OpenShift oc command-line tool to manage projects, create application resources, and monitor logs. The oc new-app command facilitates application deployment from container images, Containerfiles, or source code using the Source-to-Image (S2I) process. Routes provide public access to services, connecting internal pods with external users. Additionally, you can manage and monitor applications through both the OpenShift CLI and web console.

## Chapter 7

### Lab: Deploying Multi-Container Applications

```
Fnu pratichilab openshift-review grade
Grading the student's work for the Lab: Deploying Containerized Applications on OpenShift
  · Log in on OpenShift..... SUCCESS
  Accessing the web application..... PASS
Fnu pratichilab openshift-review finish

Completing the Lab: Deploying Containerized Applications on OpenShift

  · Log in on OpenShift..... SUCCESS
  · Deleting the vadqmd-ocp project..... SUCCESS
Fnu pratichi clear

Fnu pratichi lab multicontainer-review start
Setting up workstation for the Lab: Deploying Multi-container Applications

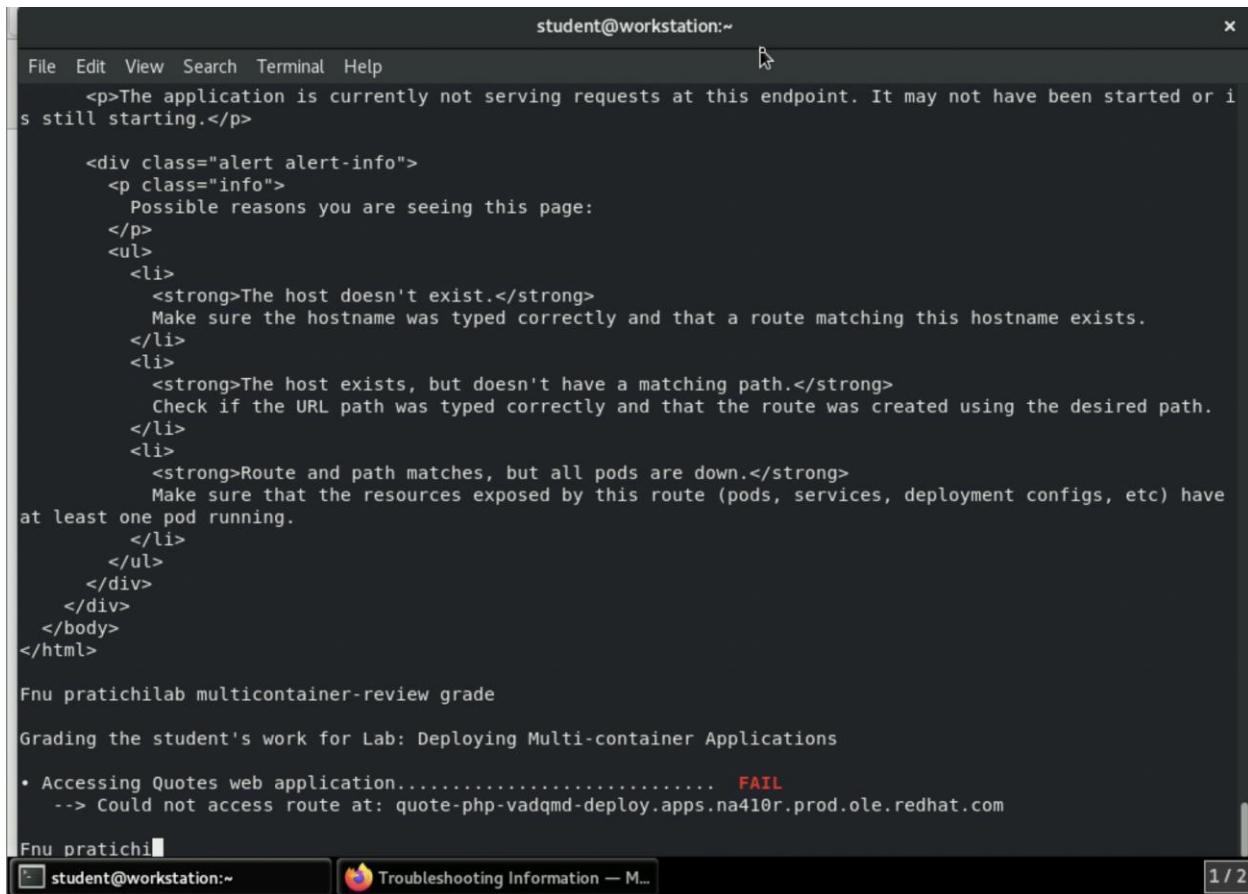
  · Setting up labs folder..... SUCCESS
  · Setting up solutions folder..... SUCCESS
  · Downloading starter project..... SUCCESS
  · Downloading solution project..... SUCCESS

Setup successful. Please proceed with the exercise.

Verifying the OpenShift cluster is running:
  · Log in on OpenShift..... SUCCESS
  · Check the internal registry is up and running..... SUCCESS
Fnu pratichicd ~/D0180/labs/multicontainer-review
Fnu pratichimulticontainer-review$ source /usr/local/etc/ocp4.config
student@workstation:~ 1 / 2
```

## Screen Picture

Figure 6: curl command to test the REST API



The screenshot shows a terminal window titled "student@workstation:~". The user has run a curl command to test a REST API endpoint. The output is as follows:

```
<p>The application is currently not serving requests at this endpoint. It may not have been started or it's still starting.</p>

<div class="alert alert-info">
  <p class="info">
    Possible reasons you are seeing this page:
  </p>
  <ul>
    <li>
      <strong>The host doesn't exist.</strong>
      Make sure the hostname was typed correctly and that a route matching this hostname exists.
    </li>
    <li>
      <strong>The host exists, but doesn't have a matching path.</strong>
      Check if the URL path was typed correctly and that the route was created using the desired path.
    </li>
    <li>
      <strong>Route and path matches, but all pods are down.</strong>
      Make sure that the resources exposed by this route (pods, services, deployment configs, etc) have at least one pod running.
    </li>
  </ul>
</div>
</div>
</body>
</html>

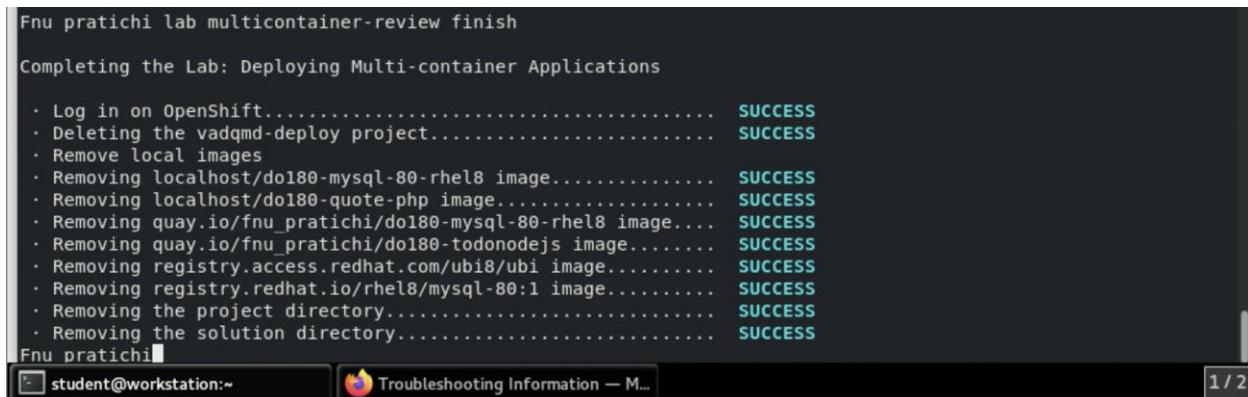
Fnu pratichilab multicontainer-review grade

Grading the student's work for Lab: Deploying Multi-container Applications

• Accessing Quotes web application..... FAIL
  --> Could not access route at: quote-php-vadqmd-deploy.apps.na4l0r.prod.ole.redhat.com
```

The terminal window also displays a "Troubleshooting Information" section and a status bar indicating "1 / 2".

Figure Description: Most of the students are getting errors in this. There are some issues related to endpoint. Error says: Host does not exist.



The screenshot shows a terminal window titled "student@workstation:~". The user has run a command to finish a lab review. The output is as follows:

```
Fnu pratichi lab multicontainer-review finish

Completing the Lab: Deploying Multi-container Applications

. Log in on OpenShift..... SUCCESS
. Deleting the vadqmd-deploy project..... SUCCESS
. Remove local images
. Removing localhost/do180-mysql-80-rhel8 image..... SUCCESS
. Removing localhost/do180-quote-php image..... SUCCESS
. Removing quay.io/fnu_pratichi/do180-mysql-80-rhel8 image.... SUCCESS
. Removing quay.io/fnu_pratichi/do180-todonodejs image..... SUCCESS
. Removing registry.access.redhat.com/ubi8/ubi image..... SUCCESS
. Removing registry.redhat.io/rhel8/mysql-80:1 image..... SUCCESS
. Removing the project directory..... SUCCESS
. Removing the solution directory..... SUCCESS

Fnu pratichi
```

The terminal window also displays a "Troubleshooting Information" section and a status bar indicating "1 / 2".

## Summary

In this chapter, you learned that software-defined networks (SDNs) enable communication between containers, requiring them to be on the same network. Containerized

applications do not rely on fixed IPs or host names. Podman utilizes Container Network Interface (CNI) to create SDNs, while Kubernetes and OpenShift create networks between containers within a pod. Kubernetes also injects service-specific variables into pods within the same project. OpenShift templates help automate the creation of applications with multiple resources, allowing template parameters to be reused across resources.

## Chapter 8.

### Lab: Troubleshooting Containerized Applications

```
Fnu pratichilab troubleshoot-review start
Checking prerequisites for Lab: Troubleshooting Containerized Applications

Checking local clone of the applications repository:
· Folder '/home/student/D0180-apps' is a git repo..... SUCCESS
· Git repo '/home/student/D0180-apps' has no pending changes.. SUCCESS
· Project 'nodejs-app' exists in student's GitHub fork..... SUCCESS
Checking for conflicts with existing OpenShift projects:
· Project 'vadqmd-nodejs-app' is absent..... SUCCESS
Checking for conflicts with existing local containers:
· Container 'test' is absent..... SUCCESS
Verifying the OpenShift cluster is running:
· Log in on OpenShift..... SUCCESS
· Check the internal registry is up and running..... SUCCESS

Setting up the classroom for Lab: Troubleshooting Containerized Applications
· Downloading starter project..... SUCCESS
· Downloading solution project..... SUCCESS

Setup successful. Please proceed with the exercise.

student@workstation:~ | Troubleshooting Information — M... | 1 / 2
```

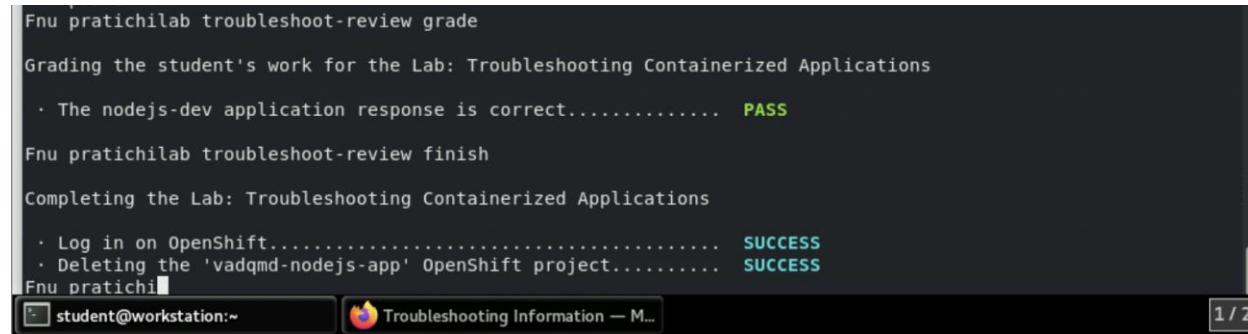
### Screen Picture

Figure 7: Viewing Logs for Node.js Application Deployment in OpenShift

```
Fnu pratichiooc logs -f deployment/nodejs-dev
Found 2 pods, using pod/nodejs-dev-79b9c8c4b4-bcngb
Environment:
  DEV_MODE=false
  NODE_ENV=production
  DEBUG_PORT=5858
Launching via npm...
npm info using npm@8.19.3
npm info using node@v16.19.1
npm timing npm:load:whichnode Completed in 0ms
npm timing config:load:defaults Completed in 2ms
npm timing config:load:file:/usr/lib/node_modules/npm/.npmrc Completed in 1ms
npm timing config:load:builtin Completed in 1ms
npm timing config:load:cli Completed in 4ms
npm timing config:load:env Completed in 0ms
npm timing config:load:project Completed in 6ms
npm timing config:load:file:/opt/app-root/src/.npmrc Completed in 0ms
npm timing config:load:user Completed in 0ms
npm timing config:load:file:/opt/app-root/src/.npm-global/etc/.npmrc Completed in 0ms
npm timing config:load:global Completed in 0ms
npm timing config:load:validate Completed in 1ms
npm timing config:load:credentials Completed in 1ms
npm timing config:load:setEnvs Completed in 1ms
npm timing config:load Completed in 17ms
npm timing npm:load:configload Completed in 17ms
npm timing npm:load:mkdirpcache Completed in 2ms
npm timing npm:load:mkdirplogs Completed in 0ms
npm timing npm:load:setTitle Completed in 2ms
npm timing config:load:flatten Completed in 3ms
npm timing npm:load:display Completed in 5ms
npm timing npm:load:logfile Completed in 5ms
npm timing npm:load:timers Completed in 0ms
npm timing npm:load:configScope Completed in 0ms
student@workstation:~ | Troubleshooting Information — M... | 1 / 2
```

Figure Description: This output shows the logs of the Node.js application deployment, providing details about the environment and the status of the deployment process in

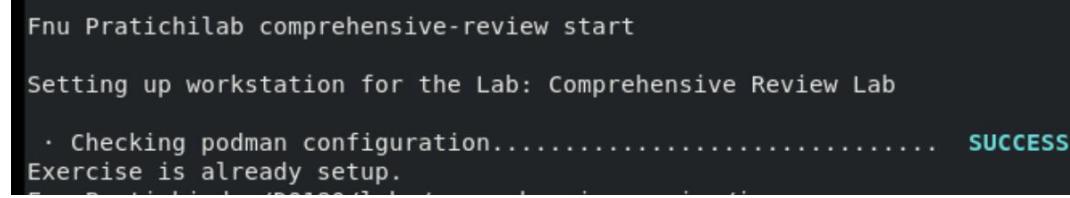
OpenShift. The `oc logs -f` command is used to stream and view real-time logs of the deployment.



```
Fnu pratichilab troubleshoot-review grade
Grading the student's work for the Lab: Troubleshooting Containerized Applications
  · The nodejs-dev application response is correct..... PASS
Fnu pratichilab troubleshoot-review finish
Completing the Lab: Troubleshooting Containerized Applications
  · Log in on OpenShift..... SUCCESS
  · Deleting the 'vadqmd-nodejs-app' OpenShift project..... SUCCESS
Fnu pratichi[1/2]
```

## Chapter 9

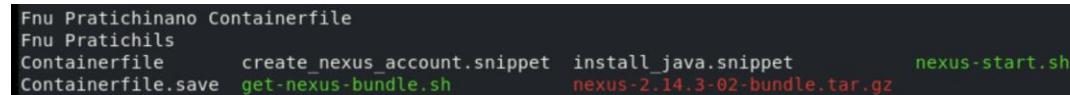
### Lab: Containerizing and Deploying a Software Application



```
Fnu Pratichilab comprehensive-review start
Setting up workstation for the Lab: Comprehensive Review Lab
  · Checking podman configuration..... SUCCESS
Exercise is already setup.
Fnu Pratichi[1/2]
```

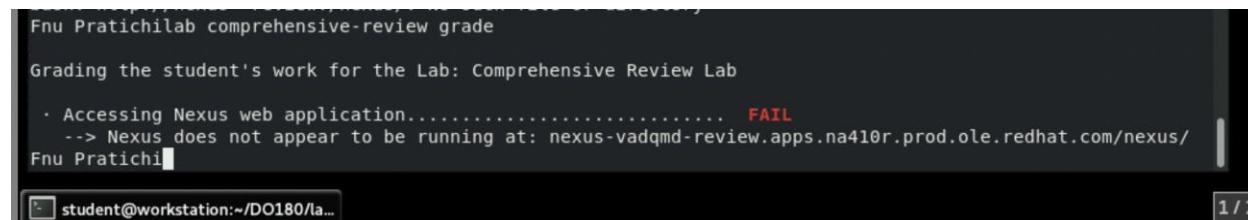
### Screen Picture

Figure 8: Editing a Containerfile Using Nano Editor



```
Fnu Pratichinano Containerfile
Fnu Pratichile
Containerfile      create_nexus_account.snippet  install_java.snippet      nexus-start.sh
Containerfile.save  get-nexus-bundle.sh           nexus-2.14.3-02-bundle.tar.gz
```

Figure Description: This figure shows the process of editing a Containerfile using the nano text editor. The `nano` command is used to open and modify the `Containerfile`, which contains instructions to build a container image.



```
Fnu Pratichilab comprehensive-review grade
Grading the student's work for the Lab: Comprehensive Review Lab
  · Accessing Nexus web application..... FAIL
    --> Nexus does not appear to be running at: nexus-vadqmd-review.apps.na410r.prod.ole.redhat.com/nexus/
Fnu Pratichi[1/2]
```

### Summary

In this chapter, you learned to containerize a Nexus server, build and test the image using Podman, and deploy it to OpenShift. You also explored how to expose the service externally and evaluate the deployment.