Lab: Container Images

Use Podman to create and pull an image from a container registry.

Outcomes

You should be able to:

- Create a container image from a Containerfile by using Quay.
- Pull an image from the Quay container registry.
- Add a tag to a container image.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

This command verifies that Podman is available and provides a Containerfile template.

The lab script continuously evaluates the objectives of this lab. Keep the script running in a terminal window and complete the objectives of this lab from a new terminal window.

After each objective, return to the lab script evaluation to see if you have finished the objective successfully. When you finish all objectives, the `lab` command prompts you to execute the `finish` function.

```
[student@workstation ~]$ lab start images-lab
```

Instructions

1. Build a container image that uses the `/home/student/DO188/labs/images-lab/Containerfile` file.

   Call the resulting image `images-lab` and push the image into the `registry.ocp4.example.com:8443` registry in the `developer` user repository.

   Use the `developer` user with the `developer` password to authenticate with the `registry.ocp4.example.com:8443` registry.

   Authenticate Podman with the `registry.ocp4.example.com:8443` registry.

   ```
   [student@workstation ~]$ podman login -u developer -p developer \
     registry.ocp4.example.com:8443
   Login Succeeded!
   ```

   In a terminal, change to the `/home/student/DO188/labs/images-lab` directory.

   ```
   [student@workstation ~]$ cd ~/DO188/labs/images-lab
   no output expected
   ```

Build the Containerfile with the `registry.ocp4.example.com:8443/developer/images-la`

```
[student@workstation images-lab]$ podman build --file Containerfile \
  --tag registry.ocp4.example.com:8443/developer/images-lab
...output omitted...
Successfully tagged registry.ocp4.example.com:8443/developer/images-lab:latest
8d14...dd5a
```

Push the image to the `registry.ocp4.example.com:8443` registry.

```
[student@workstation images-lab]$ podman push \
  registry.ocp4.example.com:8443/developer/images-lab
...output omitted...
Writing manifest to image destination
```

2. Add the `grue` tag to the `images-lab` container image, and push it to the `registry.ocp4.example.com:8443` registry.

   Add the `grue` tag to the image.

   ```
   [student@workstation images-lab]$ podman tag \
     registry.ocp4.example.com:8443/developer/images-lab \
     registry.ocp4.example.com:8443/developer/images-lab:grue
   no output expected
   ```

   Push the new image tag.

   ```
   [student@workstation images-lab]$ podman push \
     registry.ocp4.example.com:8443/developer/images-lab:grue
   ...output omitted...
   Writing manifest to image destination
   ```

3. Create a container by using the `images-lab:grue` image. Use the `images-lab` container name, and bind the `8080` container port to the `8080` host port. Start the container in the background.

   Optionally, use the `curl` command to verify that the HTTP server is running.

   Start the container.

   ```
   [student@workstation images-lab]$ podman run -d --name images-lab \
     -p 8080:8080 images-lab:grue
   2422...e7b1
   ```

Connect to the HTTP server by using `curl`. Note that this step is not required to make th

```
[student@workstation images-lab]$ curl localhost:8080
It is pitch black. You are likely to be eaten by a grue.
```

Finish

As the `student` user on the `workstation` machine, use the `lab` command to complete this exercise. This is important to ensure that resources from previous exercises do not impact upcoming exercises.

Press `y` when the `lab start` command prompts you to execute the `finish` function. Alternatively, execute the following command:

```
[student@workstation ~]$ lab finish images-lab
```