Lab: Multi-container Applications with Compose

Create a compose file to deploy your application in a testing environment. The application uses three components: a UI, a back end, and an external service that the back end uses. Because the development team does not own the external service, they decided to use a mock server called `Wiremock` to mock the external service interactions with the back end.

Outcomes

You should be able to:

- Create a multi-container compose file with the following features:
  - Bind mounts
  - Environment variables
  - Networks
  - Published ports
- Reload the compose file after modifying it.

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

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

The `start` function copies a file called `compose.yaml` that you must complete throughout this exercise. It also copies the configuration for `wiremock` to mock the `quotes-provider` service. The `compose.yaml` file has the `name` field with the `compose-lab` value, which guarantees the lab works correctly. Do not change this field name or value.

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.

Instructions

1. Change to the `/home/student/DO188/labs/compose-lab` directory. This directory contains the compose file called `compose.yaml`, which you must complete. Then, start the containers in the background.

Container Name	Image
`quotes-provider`	`registry.ocp4.example.com:8443/redhattraining/wiremock`
`quotes-api`	`registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose`
`quotes-ui`	`registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui`

> **NOTE**
>
> The WireMock container image provides the ability to create a fake API service. This exercise uses this image to run the `quotes-provider` API. In a real-world scenario, your API image is built from your application source code.

Change to the `/home/student/DO188/labs/compose-lab` directory.

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

Update the `compose.yaml` file to look like the following code block:

```
name: compose-lab

services:
  wiremock:
    container_name: "quotes-provider"
    image: "registry.ocp4.example.com:8443/redhattraining/wiremock"
  quotes-api:
    container_name: "quotes-api"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose
  quotes-ui:
    container_name: "quotes-ui"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui"
```

From the directory that contains the `compose.yaml` file, run `podman-compose up` to create run the containers in the background.

```
[student@workstation compose-lab]$ podman-compose up -d
quotes-provider
...output omitted...
quotes-api
...output omitted...
quotes-ui
...output omitted...
```

2. Configure the `quotes-provider` mock service by providing the Wiremock mappings and responses in the `/home/student/DO188/labs/compose-lab/wiremock/stubs` directory.

   Wiremock expects two directories to configure the mock server:

   - `mappings`: contains files that define the HTTP endpoints.
   - `__files`: contains files with fixed responses.

To provide the Wiremock configuration, mount the `/home/student/DO188/labs/compose-lab/wiremock/stubs` directory as the `/home/wiremock` directory in the container.

To apply the changes, delete and restart the containers by using the `podman-compose` command.

Use a bind mount to map `/home/student/DO188/labs/compose-lab/wiremock/stubs` to t the `z` option to add the correct SELinux permissions.

```
name: compose-lab

services:
  wiremock:
    container_name: "quotes-provider"
    image: "registry.ocp4.example.com:8443/redhattraining/wiremock"
    volumes:
      - ~/DO188/labs/compose-lab/wiremock/stubs:/home/wiremock:Z
  quotes-api:
    container_name: "quotes-api"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose
  quotes-ui:
    container_name: "quotes-ui"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui"
```

Apply the changes by running the `podman-compose down` command followed by `podman-c`

Run `podman-compose down` to stop and remove the containers in the `compose.yaml` file.

```
[student@workstation compose-lab]$ podman-compose down
quotes-ui
quotes-provider
quotes-api
quotes-ui
quotes-api
quotes-provider
c08e...78de
compose-lab_default
```

Run `podman-compose up` to re-create the containers with the new changes.

```
[student@workstation compose-lab]$ podman-compose up -d
7c6f...be17
7ddb...f6c3
quotes-provider
b448...4746
quotes-api
01d9...8807
quotes-ui
```

3. Expose the `quotes-api` service to the host machine on port `8080`. The `quotes-api` service listens on port `8080` in the container.

   To apply the changes, delete and restart the containers by using the `podman-compose` command.

   Use the `ports` property to expose the port `8080` in the container to port `8080` on the host

   ```
   name: compose-lab

   services:
     wiremock:
       container_name: "quotes-provider"
       image: "registry.ocp4.example.com:8443/redhattraining/wiremock"
       volumes:
         - ~/DO188/labs/compose-lab/wiremock/stubs:/home/wiremock:Z
     quotes-api:
       container_name: "quotes-api"
       image: "registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose
       ports:
         - "8080:8080"
     quotes-ui:
       container_name: "quotes-ui"
       image: "registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui"
   ```

   Apply the changes by running the `podman-compose down` command followed by `podman-c`

   Run `podman-compose down` to stop and remove the containers in the `compose.yaml` file.

   ```
   [student@workstation compose-lab]$ podman-compose down
   ...output omitted...
   ```

   Run `podman-compose up` to re-create the containers with the new changes.

   ```
   [student@workstation compose-lab]$ podman-compose up -d
   ...output omitted...
   ```

4. Isolate together the `quotes-provider` and `quotes-api` services by creating a Podman network called `lab-net` in the `compose.yaml` file.

   To apply the changes, delete and restart the containers by using the `podman-compose` command.

   Use the `networks` top-level property to create the `lab-net` network from the `compose.ya` under the `wiremock` and `quotes-api` services to connect both containers to the `lab-net`

```
name: compose-lab

services:
  wiremock:
    container_name: "quotes-provider"
    image: "registry.ocp4.example.com:8443/redhattraining/wiremock"
    volumes:
      - ~/DO188/labs/compose-lab/wiremock/stubs:/home/wiremock:Z
    networks:
      - lab-net
  quotes-api:
    container_name: "quotes-api"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose
    ports:
      - "8080:8080"
    networks:
      - lab-net
  quotes-ui:
    container_name: "quotes-ui"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui"

networks:
  lab-net: {}
```

Apply the changes by running the `podman-compose down` command followed by the `podma`

Run the `podman-compose down` command to stop and remove the containers in the `compo`

```
[student@workstation compose-lab]$ podman-compose down
...output omitted...
```

Run the `podman-compose up` command to re-create the containers with the new changes

```
[student@workstation compose-lab]$ podman-compose up -d
...output omitted...
```

5. The `quotes-api` and `quotes-provider` services share a network, but the `quotes-api` service is missing the hostname configuration of the provider endpoint. Set the `QUOTES_SERVICE` environment variable to configure the `quotes-provider` URL. Use the default name resolution, the `quotes-provider` configuration, the `http` protocol, and the port `8080` to configure this value.

   To apply the changes, delete and restart the containers by using the `podman-compose` command.

   Add the `QUOTES_SERVICE="http://quotes-provider:8080"` environment variable to the `c`

```
name: compose-lab

services:
  wiremock:
    container_name: "quotes-provider"
    image: "registry.ocp4.example.com:8443/redhattraining/wiremock"
    volumes:
      - ~/DO188/labs/compose-lab/wiremock/stubs:/home/wiremock:Z
    networks:
      - lab-net
  quotes-api:
    container_name: "quotes-api"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose
    ports:
      - "8080:8080"
    networks:
      - lab-net
    environment:
      QUOTES_SERVICE: "http://quotes-provider:8080"
  quotes-ui:
    container_name: "quotes-ui"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui"

networks:
  lab-net: {}
```

Apply the changes by running the `podman-compose down` command followed by the `podma`

Run the `podman-compose down` command to stop and remove the containers in the `compo`

```
[student@workstation compose-lab]$ podman-compose down
...output omitted...
```

Run the `podman-compose up` command to re-create the containers with the new changes

```
[student@workstation compose-lab]$ podman-compose up -d
...output omitted...
```

6. Expose the `quotes-ui` service on port `3000` so that the host can access the application on http://localhost:3000. The `quotes-ui` is listening on port `8080` in the UI container.

   To apply the changes, delete and restart the containers by using the `podman-compose` command.

   Use the previous URL to visit the quotes application by using a web browser.

   Use the `ports` property in the `quotes-ui` service to expose the port `8080` in the container

```
name: compose-lab

services:
  wiremock:
    container_name: "quotes-provider"
    image: "registry.ocp4.example.com:8443/redhattraining/wiremock"
    volumes:
      - ~/DO188/labs/compose-lab/wiremock/stubs:/home/wiremock:Z
    networks:
      - lab-net
  quotes-api:
    container_name: "quotes-api"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotesapi-compose
    ports:
      - "8080:8080"
    networks:
      - lab-net
    environment:
      QUOTES_SERVICE: "http://quotes-provider:8080"
  quotes-ui:
    container_name: "quotes-ui"
    image: "registry.ocp4.example.com:8443/redhattraining/podman-quotes-ui"
    ports:
      - "3000:8080"

networks:
  lab-net: {}
```

Apply the changes by running the `podman-compose down` command followed by the `podma`

Run the `podman-compose down` command to stop and remove the containers in the `compo`

```
[student@workstation compose-lab]$ podman-compose down
...output omitted...
```

Run the `podman-compose up` command to re-create the containers with the new changes

```
[student@workstation compose-lab]$ podman-compose up -d
...output omitted...
```

Go to http://localhost:3000 to verify that the quotes application shows famous quote

Finish

As the `student` user on the `workstation` machine, use the `lab` command to complete this exercise. This step 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 compose-lab
```