Lab: Persisting Data

Use volumes to provide persistence to an application.

Outcomes

You should be able to create volumes, import data into volumes, and use volumes in an application.

This lab uses a URL shortener application, which consists of three components: a database container, a back-end container, and a front-end container.

The source code for the front end and back end is available at `/home/student/DO188/solutions/persisting-lab` after you execute the lab script.

Note the following:

- The back-end container uses the following information:
  - The back end uses default values for the user, password, and database for simplicity.
  - The back end uses the database container name to resolve the database IP address. Do not change the database container name.
- The front-end container uses the following information:
  - The front end uses an Nginx server to redirect requests from `localhost:8080` to the `persisting-backend:8080` host. Do not change the back-end container name.
  - If the front end exits after start, execute `podman logs persisting-frontend` to check the logs.
- The application can become unresponsive after you stop individual containers. If this happens, stop all containers and start the containers in order of database, back end, and front end.

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

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

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. Create a named volume with the following parameters:

   - The volume is called `postgres-vol`.
   - The volume contains the contents of the `/home/student/DO188/labs/persisting-lab/postgres-vol.tar.gz` file.

   Create the volume.

   ```
   [student@workstation ~]$ podman volume create postgres-vol
   postgres-vol
   ```

   Import the `/home/student/DO188/labs/persisting-lab/postgres-vol.tar.gz` file to the volume:

   ```
   [student@workstation ~]$ podman volume import postgres-vol \
     ~/DO188/labs/persisting-lab/postgres-vol.tar.gz
   ...no output expected...
   ```

2. Start the application database container with the following parameters:

   - Call the container `persisting-db`.
   - Start the container in the background.
   - Connect the container to the `persisting-net` network.
   - Use the following environment variables:
     - `POSTGRESQL_PASSWORD=pass`
     - `POSTGRESQL_USER=user`
     - `POSTGRESQL_DATABASE=db`
   - Mount the `postgres-vol` volume to the `/var/lib/pgsql/data` directory.

- Use the `registry.ocp4.example.com:8443/rhel9/postgresql-13:1` image.

Create the `persisting-net` network.

```
[student@workstation ~]$ podman network create persisting-net
persisting-net
```

Start the database container.

```
[student@workstation ~]$ podman run --name persisting-db -d \
  --net persisting-net -e POSTGRESQL_USER=user -e POSTGRESQL_PASSWORD=pass \
  -e POSTGRESQL_DATABASE=db \
  --mount='type=volume,src=postgres-vol,dst=/var/lib/pgsql/data' \
  registry.ocp4.example.com:8443/rhel9/postgresql-13:1
c97f...4a29
```

3. Start the back end with the following parameters:

- Call the container `persisting-backend`.
- Start the container in the background.
- Use the environment variable `DB_HOST=persisting-db`.
- Expose the port `8080` on the machine to route requests to port `8080` inside the container.
- Connect the container to the `persisting-net` network.
- Use the `registry.ocp4.example.com:8443/redhattraining/podman-urlshortener-backend` image.

```
[student@workstation ~]$ podman run --name persisting-backend -d \
  -e DB_HOST=persisting-db -p 8080:8080 --net persisting-net \
  registry.ocp4.example.com:8443/redhattraining/podman-urlshortener-backend
3a46...4e60
```

4. Start the front end with the following parameters:

- Call the container `persisting-frontend`.
- Start the container in the background.
- Connect the container to the `persisting-net` network.
- Expose the port `3000` on the machine to route requests to port `8080` inside the container.
- Use the `registry.ocp4.example.com:8443/redhattraining/podman-urlshortener-frontend` image.

```
[student@workstation ~]$ podman run --name persisting-frontend -d \
  --net persisting-net -p 3000:8080 \
  registry.ocp4.example.com:8443/redhattraining/podman-urlshortener-frontend
b10e...940f
```

5. Test the application.

   In a web browser, verify the functionality of the application at `http://localhost:3000`.

   In a web browser, test the database data import by navigating to `http://localhost:8080/api/shorturl/a9yi4rcl5uuzunv`.

   The `a9yi4rcl5uuzunv` short URL is a part of the database data that you imported in a previous step into the `postgres-vol` volume.

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 persisting-lab
```