

Lab: Deploy Managed and Networked Applications on Kubernetes

Deploy a database server and a web application that connects to that database, and expose the web application to external access.

Outcomes

- Deploy a MySQL database from a container image.
- Deploy a web application from a container image.
- Configure environment variables for a deployment.
- Expose the web application for external access.

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

This command ensures that the cluster is accessible and that all exercise resources are available. It also creates the `database-applications` project.

```
[student@workstation ~]$ lab start deploy-review
```

Instructions

The API URL of your OpenShift cluster is `https://api.ocp4.example.com:6443`, and the `oc` command is already installed on your workstation machine.

Log in to the OpenShift cluster as the `developer` user with `developer` as the password.

Use the `database-applications` project for your work.

1. Log in to the OpenShift cluster, and change to the `database-applications` project.

Log in to the OpenShift cluster as the `developer` user with `developer` as the password.

```
[student@workstation ~]$ oc login -u developer -p developer \
https://api.ocp4.example.com:6443
Login successful
...output omitted...
```

Change to the `database-applications` project.

```
[student@workstation ~]$ oc project database-applications
Now using project "database-applications" on server "https://api.ocp4.example.com:6443"
...output omitted...
```

2. Create a MySQL database deployment named `mysql-app` by using the `registry.ocp4.example.com:8443/redhattraining/mysql-app:v1` image, and identify the root cause of the failure.

Create the MySQL database deployment.

```
[student@workstation ~]$ oc create deployment mysql-app \
  --image registry.ocp4.example.com:8443/redhattraining/mysql-app:v1
deployment.apps/mysql-app created
```

Verify the deployment status. The pod name might differ in your output.

```
[student@workstation ~]$ oc get pods
NAME                                READY  STATUS   ...
mysql-app-75dfd58f99-5xfqc         0/1    Error    ...
[student@workstation ~]$ oc status
...output omitted...
Errors:
  pod/mysql-app-75dfd58f99-5xfqc is crash-looping

1 error, 1 info identified, use 'oc status --suggest' to see details.
```

Identify the root cause of the deployment failure.

```
[student@workstation ~]$ oc logs mysql-app-75dfd58f99-5xfqc
...output omitted...
You must either specify the following environment variables:
  MYSQL_USER (regex: '\^[a-zA-Z0-9_]+$')
  MYSQL_PASSWORD (regex: '\^[a-zA-Z0-9_~!@#%&*()-=<>,.;:|]+$')
  MYSQL_DATABASE (regex: '\^[a-zA-Z0-9_]+$')
Or the following environment variable:
  MYSQL_ROOT_PASSWORD (regex: '\^[a-zA-Z0-9_~!@#%&*()-=<>,.;:|]+$')
...output omitted...
```

3. Configure the environment variables for the `mysql-app` deployment by using the following information:

Field	Value
<code>MYSQL_USER</code>	<code>redhat</code>
<code>MYSQL_PASSWORD</code>	<code>redhat123</code>
<code>MYSQL_DATABASE</code>	<code>world_x</code>

Then, execute the following command in the `mysql-app` deployment pod to load the `world_x` database:

```
/bin/bash -c "mysql -uredhat -predhat123 </tmp/world_x.sql"
```

Update the environment variables for the mysql-app deployment.

```
[student@workstation ~]$ oc set env deployment/mysql-app \
  MYSQL_USER=redhat MYSQL_PASSWORD=redhat123 MYSQL_DATABASE=world_x
deployment.apps/mysql-app updated
```

Verify that the mysql-app application pod is in the RUNNING state. The pod name might di

```
[student@workstation ~]$ oc get pods
NAME                                READY   STATUS   ...
mysql-app-57c44f646-5qt2k          1/1     Running  ...
```

Load the world_x database.

```
[student@workstation ~]$ oc exec -it mysql-app-57c44f646-5qt2k \
  -- /bin/bash -c "mysql -uredhat -predhat123 </tmp/world_x.sql"
mysql: [Warning] Using a password on the command line interface can be insecure.
[student@workstation ~]$
```

Confirm that you can access the MySQL database.

```
[student@workstation ~]$ oc rsh mysql-app-57c44f646-5qt2k
```

```
sh-4.4$ mysql -uredhat -predhat123 world_x
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with ; or \g.
...output omitted...
mysql>
```

Exit the MySQL database, and then exit the container.

```
mysql> exit
Bye
sh-4.4$ exit
exit
```

4. Create a service for the mysql-app deployment by using the following information:

Field	Value
Name	mysql-service
Port	3306
Target port	3306

Expose the mysql-app deployment.

```
[student@workstation ~]$ oc expose deployment mysql-app --name mysql-service \
  --port 3306 --target-port 3306
service/mysql-service created
```

Verify the service configuration. The endpoint IP address might differ in your output.

```
[student@workstation ~]$ oc get services
NAME                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)        AGE
mysql-service       ClusterIP     172.30.146.213 <none>         3306/TCP       10s
[student@workstation ~]$ oc get endpoints
NAME                ENDPOINTS          AGE
mysql-service       10.8.0.102:3306    19s
```

5. Create a web application deployment named php-app by using the registry.ocp4.example.com:8443/redhattraining/php-webapp:v1 image.

Create the web application deployment.

```
[student@workstation ~]$ oc create deployment php-app \
  --image registry.ocp4.example.com:8443/redhattraining/php-webapp:v1
deployment.apps/php-app created
```

Verify the deployment status. Verify that the php-app application pod is in the RUNNING state.

```
[student@workstation ~]$ oc get pods
NAME                READY   STATUS    ...
php-app-725...      1/1     Running   ...
mysql-app-57c...    1/1     Running   ...
```

```
[student@workstation ~]$ oc status
...output omitted...
deployment/php-app deploys registry.ocp4.example.com:8443/redhattraining/php-webapp
deployment #1 running for about a minute - 1 pod
...output omitted...
```

6. Create a service for the php-app deployment by using the following information:

Field	Value
Name	php-svc
Port	8080

Field	Value
Target port	8080

Then, create a route named `phpapp` to expose the web application to external access.

Expose the `php-app` deployment.

```
[student@workstation ~]$ oc expose deployment php-app --name php-svc \
  --port 8080 --target-port 8080
service/php-svc exposed
```

Verify the service configuration. The endpoint IP address might differ in your output.

```
[student@workstation ~]$ oc get services
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
mysql-service       ClusterIP      172.30.146.213   <none>           3306/TCP         7m47s
php-svc             ClusterIP      172.30.228.80    <none>           8080/TCP         4m34s
[student@workstation ~]$ oc get endpoints
NAME                ENDPOINTS          AGE
mysql-service       10.8.0.102:3306    7m50s
php-svc             10.8.0.107:8080    4m37s
```

Expose the `php-svc` service.

```
[student@workstation ~]$ oc expose service/php-svc --name phpapp
route.route.openshift.io/phpapp exposed
```

```
[student@workstation ~]$ oc get routes
NAME          HOST/PORT          ...
phpapp        phpapp-database-applications.apps.ocp4.example.com ...
```

7. Test the connectivity between the web application and the MySQL database. In a web browser, go to the `phpapp-database-applications.apps.ocp4.example.com` route, and verify that the application retrieves data from the MySQL database.

Go to the `phpapp-database-applications.apps.ocp4.example.com` route in the web browser.

Connected successfully

1	Kabul	AFG Kabul	{"Population": 17
2	Qandahar	AFG Qandahar	{"Population": 23
3	Herat	AFG Herat	{"Population": 18
4	Mazar-e-Sharif	AFG Balkh	{"Population": 12
5	Amsterdam	NLD Noord-Holland	{"Population": 73
6	Rotterdam	NLD Zuid-Holland	{"Population": 59
7	Haag	NLD Zuid-Holland	{"Population": 44
8	Utrecht	NLD Utrecht	{"Population": 23
9	Eindhoven	NLD Noord-Brabant	{"Population": 20
10	Tilburg	NLD Noord-Brabant	{"Population": 19
11	Groningen	NLD Groningen	{"Population": 17
12	Breda	NLD Noord-Brabant	{"Population": 16
13	Apeldoorn	NLD Gelderland	{"Population": 15

Evaluation

As the student user on the workstation machine, use the `lab` command to grade your work. Correct any reported failures and rerun the command until successful.

```
[student@workstation ~]$ lab grade deploy-review
```

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.

```
[student@workstation ~]$ lab finish deploy-review
```