Lesson 4 Lesson-End Project

**MySQL and WordPress Installation in Kubernetes**

**Project agenda:** To create a highly available MySQL and WordPress deployment using autoscaling.

**Description:** Your team is going to deploy MySQL and WordPress containers. All the users should be added using ConfigMaps and all the sensitive data should be added using Secrets. The service should be on the NodePort. The WordPress Pod should not deploy if the MySQL service is not deployed. Also, ensure autoscaling is on.

**Tools required:** kubeadm, kubectl, kubelet, and docker

**Prerequisites:** kubeadm, kubectl, kubelet, and docker should be installed

**Expected deliverables:** A Kubernetes cluster with high availability enabled

Steps to be followed:

1. Creating a cluster
2. Creating a Secret for storing the MySQL password securely
3. Creating a MySQL YAML manifest file for deploying MySQL
4. Deploying the WordPress application and designing the manifest YAML file
5. Accessing the WordPress application using the NodePort
6. Enabling autoscaling on a WordPress Pod to ensure scaling if the CPU utilization is greater than 50 percent

**Step 1: Creating a cluster**

|  |
| --- |
| **Note:** Refer Demo 1 of Lesson 2 to create a Kubernetes cluster |

**Step 2: Creating a Secret for storing the MySQL password securely**

1. Validate the connectivity between the master and worker nodes using the following command:

**kubectl get node**

A screenshot of a computer

Description automatically generated

1. Encrypt your plain password with **base64** encoding to make the password more secure and hard to remember:

**echo -n 'mysql@Temp123' | base64**

A screenshot of a computer

Description automatically generated

1. Once the encrypted password is generated, save that to your Secret using the below YAML code and pull this directly during the MySQL Deployment:

**vi secret.yaml**

**apiVersion: v1**

**kind: Secret**

**metadata:**

**name: mysql-secret-password**

**type: kubernetes.io/basic-auth**

**stringData:**

**username: root**

**password: bXlzcWxAVGVtcDEyMw==**

A screenshot of a computer

Description automatically generated

1. Create and describe the Secret using the following commands:

**kubectl apply -f secret.yaml**

**kubectl describe secret mysql-secret-password**

Text

Description automatically generated

**Step 3: Creating a MySQL YAML manifest file for deploying MySQL**

1. Create a MySQL YAML manifest file using the following code:

**vi mysql.yaml**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: mysql**

**labels:**

**app: mysql-wordpress**

**spec:**

**ports:**

**- port: 3306**

**selector:**

**app: mysql-wordpress**

**product: mysql**

**---**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: mysql**

**labels:**

**app: mysql-wordpress**

**spec:**

**selector:**

**matchLabels:**

**app: mysql-wordpress**

**product: mysql**

**strategy:**

**type: Recreate**

**template:**

**metadata:**

**labels:**

**app: mysql-wordpress**

**product: mysql**

**spec:**

**containers:**

**- image: mysql**

**name: mysql-container**

**env:**

**- name: MYSQL\_DATABASE**

**value: wordpress**

**- name: MYSQL\_ROOT\_PASSWORD**

**valueFrom:**

**secretKeyRef:**

**name: mysql-secret-password**

**key: password**

**ports:**

**- containerPort: 3306**

**name: mysql**

1. Create a Service and Deployment using the following command:

**kubectl apply -f mysql.yaml**

**A screenshot of a computer

Description automatically generated**

* 1. To verify that the Pods, Services, and Deployments are created, use the following commands:

**kubectl get deployments -o wide**

**kubectl get pods -o wide**

**kubectl get services -o wide**

**Text

Description automatically generated**

* 1. Once the MySQL Pods are deployed, we can check the logs of MySQL to ensure that the MySQL container is running without any issues:

**kubectl logs <mysql-pod-name>**

Graphical user interface, text

Description automatically generated

|  |
| --- |
| **Note:** This will show errors in case the container inside the Pod is not functioning properly. |

**Step 4: Deploying the WordPress application and designing the manifest YAML file**

1. Deploy the WordPress Pod using the YAML code given below, and ensure that the Pods are up and running:

**vi wordpress.yaml**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: wordpress**

**labels:**

**app: mysql-wordpress**

**spec:**

**ports:**

**- port: 80**

**selector:**

**app: mysql-wordpress**

**tier: frontend**

**type: NodePort**

**---**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: wordpress**

**labels:**

**app: mysql-wordpress**

**spec:**

**selector:**

**matchLabels:**

**app: mysql-wordpress**

**tier: frontend**

**strategy:**

**type: Recreate**

**template:**

**metadata:**

**labels:**

**app: mysql-wordpress**

**tier: frontend**

**spec:**

**containers:**

**- image: wordpress**

**name: wordpress**

**env:**

**- name: WORDPRESS\_DB\_HOST**

**value: mysql**

**- name: WORDPRESS\_DB\_USER**

**value: root**

**- name: WORDPRESS\_DB\_PASSWORD**

**valueFrom:**

**secretKeyRef:**

**name: mysql-secret-password**

**key: password**

**ports:**

**- containerPort: 80**

**name: wordpress**

1. Create the WordPress Service and Deployment:

**kubectl apply -f wordpress.yaml**

**A screenshot of a computer

Description automatically generated**

1. Verify the Services and Pods:

**kubectl get services**

**kubectl get pods**

**A screenshot of a computer

Description automatically generated**

**Step 5: Accessing the WordPress application using the NodePort**

1. Fetch the NodePort using the below command and access the WordPress application:

**kubectl get services**

**A screenshot of a computer

Description automatically generated**

1. Navigate to the desktop tab of the master node, open the Firefox browser, and then use the below URL to access the WordPress application:

**http://<node\_port>**

**A screenshot of a computer

Description automatically generated**

1. Once the WordPress application opens, choose **‘’English’’** and then **‘’Continue’’**:

**A screenshot of a computer

Description automatically generated**

**Graphical user interface, application

Description automatically generated**

1. Provide the website-related information, and proceed with the installation:

**A screenshot of a computer

Description automatically generated**

1. Once the installation is completed, log into the WordPress application:

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

**Step 6: Enabling autoscaling on a WordPress Pod to ensure scaling if the CPU utilization is greater than 50 percent**

1. Enable autoscaling HPA using the following command with CPU percentage as 50 and min and max value of the Pods as desired:

**kubectl autoscale deployment wordpress --cpu-percent=50 --min=1 --max=10**

**A screenshot of a computer

Description automatically generated**

1. To check the status of the newly created HPA, use the following command:

**kubectl get hpa**

**Text

Description automatically generated**

You have now successfully deployed the MySQL and WordPress application using autoscaling.