**Lesson 6 Demo 5**

**Setting up an Ingress**

**Objective:** To create an Ingress controller on a Kubernetes cluster

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

**Prerequisites:** A Kubernetes cluster must be set up (follow steps of Lesson 2 Demo 1)

Steps to be followed:

1. Creating an Nginx Ingress controller
2. Adding a rule
3. Verifying the Ingress

**Step 1: Creating an Nginx Ingress controller**

1. To create an **ingress-nginx-controller**, run the following command:

**kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.2.0/deploy/static/provider/cloud/deploy.yaml**

**Text

Description automatically generated**

1. Use the following commands to verify the **ingress-nginx-controller**:

**kubectl get all -n ingress-nginx**

**kubectl get pods -n ingress-nginx**

**kubectl get deployment -n ingress-nginx**

**kubectl get svc -n ingress-nginx**

Graphical user interface

Description automatically generated with low confidence

**Graphical user interface

Description automatically generated**

The **ingress-nginx-controller** has been successfully created.

**Step 2: Adding a rule**

1. Write the following code in the **rule.yaml** file:

**apiVersion: networking.k8s.io/v1**

**kind: Ingress**

**metadata:**

**name: test-ingress**

**annotations:**

**nginx.ingress.kubernetes.io/rewrite-target: /**

**spec:**

**ingressClassName: nginx**

**rules:**

**- http:**

**paths:**

**- path: /**

**pathType: Prefix**

**backend:**

**service:**

**name: app1**

**port:**

**number: 80**

**Text

Description automatically generated**

1. Run the command below to add a **rule** for the **ingress-nginx-controller**:

**kubectl create -f rule.yaml**

**Text

Description automatically generated**

**Step 3: Verifying the Ingress**

1. Use the following command to verify the **Ingress** state:

**kubectl get ingress**

Text

Description automatically generated

As seen in the above screenshot, the **test-ingress** is successfully configured to manage external access to cluster services and accept traffic from outside the Kubernetes platform, and load balance it to Containers.