Creating an Ingress Resource with Default Backend

In this lesson, we will first define and then create an Ingress resource with a default backend.

WE'LL COVER THE FOLLOWING

Non-Matching Requests

In some cases, we might want to define a default backend. We might want to forward requests that do not match any of the Ingress rules.

Let’s take a look at an example.

```
curl -I -H "Host: acme.com" \
    "http://$IP"
```

So far, we have two sets of Ingress rules in our cluster. One accepts all requests with the base path `/demo`. The other forwards all requests coming from the `devopstoolkitseries.com` domain. The request we just sent does not match either of those rules, so the response was once again `404 Not Found`.

Default Backend Ingress Resource

Let’s first define and then create a default backend resource.

Looking into the Definition

Let’s imagine that it would be a good idea to forward all requests with the wrong domain to the `devops-toolkit` application. Of course, by “wrong domain”, I mean one of the domains we own, and not one of those that are already included in Ingress rules.

```
cat ingress/default-backend.yml
```

The **output** is as follows.

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: default
  annotations:
    kubernetes.io/ingress.class: "nginx"
    ingress.kubernetes.io/ssl-redirect: "false"
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
spec:
  backend:
    serviceName: devops-toolkit
    servicePort: 80
```

There’s no Deployment, nor is there a Service. This time, we’re creating only an Ingress resource.

The `spec` has no rules, but only a single `backend`.

> When an Ingress `spec` is without rules, it is considered a default backend. As such, it will forward all requests that do not match paths and/or domains set as rules in the other Ingress resources.

We can use the default backend as a substitute for the default `404` pages or for any other occasion that is not covered by other rules.

You’ll notice that the `serviceName` is `devops-toolkit`. The example would be much better if we created a separate application for this purpose but it does not matter for this example. All we want, at the moment, is to see something other than `404 Not Found` response.

Creating the Resource

```
kubectl create \
    -f ingress/default-backend.yml
```

We created the Ingress resource with the default backend, and now we can test whether it truly works.

```
curl -I -H "Host: acme.com" \
    "http://$IP"
```

This time, the output is different. We got `200 OK` instead of the `404 Not Found` response.

```
HTTP/1.1 200 OK
...
```

That’s it for now. Coming up next is a quick quiz to test your understanding of Ingress resources.