## Run a Web Application using Deployment

## Create namespace

**kubectl create namespace test**

**$ kubectl config set-context $(kubectl config current-context) --namespace=<insert-namespace-name-here>**

***# Validate it***

**$ kubectl config view | grep namespace:**

## YAML

apiVersion: apps/v1beta1

kind: Deployment

metadata:

name: nginx-deployment

spec:

selector:

matchLabels:

app: nginx

replicas: 2 # tells deployment to run 2 pods matching the template

template: # create pods using pod definition in this template

metadata:

# unlike pod-nginx.yaml, the name is not included in the meta data as a unique name is

# generated from the deployment name

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.7.9

ports:

- containerPort: 80

## Create a Deployment based on the YAML file:

**kubectl apply -f nginx-deploy.yaml -–namespace=test**

## Display information about the Deployment:

**kubectl describe deployment nginx-deployment -–namespace=test**

## The output is similar to this:

**user@computer:~/kubernetes.github.io$ kubectl describe deployment nginx-deployment -–namespace=test**

**Name: nginx-deployment**

**Namespace: default**

**CreationTimestamp: Tue, 30 Aug 2016 18:11:37 -0700**

**Labels: app=nginx**

**Annotations: deployment.kubernetes.io/revision=1**

**Selector: app=nginx**

**Replicas: 2 desired | 2 updated | 2 total | 2 available | 0 unavailable**

**StrategyType: RollingUpdate**

**MinReadySeconds: 0**

**RollingUpdateStrategy: 1 max unavailable, 1 max surge**

**Pod Template:**

**Labels: app=nginx**

**Containers:**

**nginx:**

**Image: nginx:1.7.9**

**Port: 80/TCP**

**Environment: <none>**

**Mounts: <none>**

**Volumes: <none>**

**Conditions:**

**Type Status Reason**

**---- ------ ------**

**Available True MinimumReplicasAvailable**

**Progressing True NewReplicaSetAvailable**

**OldReplicaSets: <none>**

**NewReplicaSet: nginx-deployment-1771418926 (2/2 replicas created)**

**No events.**

1. List the pods created by the deployment:

**kubectl get pods -l app=nginx -–namespace=test**

The output is similar to this:

**NAME READY STATUS RESTARTS AGE**

**nginx-deployment-1771418926-7o5ns 1/1 Running 0 16h**

**nginx-deployment-1771418926-r18az 1/1 Running 0 16h**

1. Display information about a pod:

**kubectl describe pod <pod-name>**

where **<pod-name>** is the name of one of your pods

## Updating the deployment

You can update the deployment by applying a new YAML file. This YAML file specifies that the deployment should be updated to use nginx 1.8.

| [**deployment-update.yaml**](https://raw.githubusercontent.com/kubernetes/kubernetes.github.io/master/docs/tutorials/stateless-application/deployment-update.yaml) |
| --- |
| **apiVersion: apps/v1beta1**  **kind: Deployment**  **metadata:**  **name: nginx-deployment**  **spec:**  **selector:**  **matchLabels:**  **app: nginx**  **replicas: 2**  **template:**  **metadata:**  **labels:**  **app: nginx**  **spec:**  **containers:**  **- name: nginx**  **image: nginx:1.8 *# Update the version of nginx from 1.7.9 to 1.8***  **ports:**  **- containerPort: 80** |

1. Apply the new YAML file:

**kubectl apply -f nginx-deploy.yaml -–namespace=test**

1. Watch the deployment create pods with new names and delete the old pods:

**kubectl get pods -l app=nginx -–namespace=test**

## Scaling the application by increasing the replica count

You can increase the number of pods in your Deployment by applying a new YAML file. This YAML file sets repicas to 3 which specifies that the Deployment should have 3

**apiVersion: apps/v1beta2**

**kind: Deployment**

**metadata:**

**name: nginx-deployment**

**spec:**

**selector:**

**matchLabels:**

**app: nginx**

**replicas: 4 *# Update the replicas from 2 to 4***

**template:**

**metadata:**

**labels:**

**app: nginx**

**spec:**

**containers:**

**- name: nginx**

**image: nginx:1.8**

**ports:**

**- containerPort: 80**

1. Apply the new YAML file:

**kubectl apply -f nginx-deploy.yaml**

1. Verify that the Deployment has four pods:

**kubectl get pods -l app=nginx**

The output is similar to this:

**NAME READY STATUS RESTARTS AGE**

**nginx-deployment-148880595-4zdqq 1/1 Running 0 25s**

**nginx-deployment-148880595-6zgi1 1/1 Running 0 25s**

**nginx-deployment-148880595-fxcez 1/1 Running 0 2m**

**nginx-deployment-148880595-rwovn 1/1 Running 0 2m**

## Creating a Service for an application

1.List the replica set for the two Hello World pods:

**kubectl get replicasets --selector="app=nginx"**

The output is similar to this:

**NAME DESIRED CURRENT AGE**

**hello-world-2189936611 2 2 12m**

2. Create a Service object that exposes the replica set:

**kubectl expose rs <your-replica-set-name> --type="LoadBalancer" --name="example-service"**

where **<your-replica-set-name>** is the name of your replica set.

1. Display the IP addresses for your service:

**kubectl get services example-service**

The output shows the internal IP address and the external IP address of your service. If the external IP address shows as **<pending>**, repeat the command.

## This will create a Load balancer automatically , you can view the Load balancer in AWS console -> EC2 -> Load Balancer Click on the LB name and get description.