

Project Overview

Participants are required to deploy a simple static web application on a Kubernetes cluster using Minikube, set up advanced ingress networking with URL rewriting and sticky sessions, and configure horizontal pod autoscaling to manage traffic efficiently. The project will be divided into stages, with each stage focusing on specific aspects of Kubernetes ingress, URL rewriting, sticky sessions, and autoscaling.

Requirements and Deliverables

Stage 1: Setting Up the Kubernetes Cluster and Static Web App

1. Set Up Minikube:

- Ensure Minikube is installed and running on the local Ubuntu machine.
- Verify the Kubernetes cluster is functioning correctly.

2. Deploy Static Web App:

- Create a Dockerfile for a simple static web application (e.g., an HTML page served by Nginx).

Index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>CustomPage</title>

</head>

<body>

<center><h1>Hello from Maaz Patel!</h1></center>

<center><p>This is a simple static front-end served by
Nginx.</p></center>

</body>
```

</html>

dockerfile

FROM nginx:latest

COPY index.html /usr/share/nginx/html/

EXPOSE 80

- Build a Docker image for the static web application.

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest
/staticWebapp$ docker build -t maazpatel24/nginx-k8s-ingress:latest .
[+] Building 0.2s (7/7) FINISHED                                docker:default
=> [internal] load build definition from dockerfile             0.0s
=> => transferring dockerfile: 103B                             0.0s
=> [internal] load metadata for docker.io/library/nginx:latest 0.0s
=> [internal] load .dockerignore                               0.0s
=> => transferring context: 2B                                    0.0s
=> [internal] load build context                                0.0s
=> => transferring context: 374B                                  0.0s
=> CACHED [1/2] FROM docker.io/library/nginx:latest            0.0s
=> [2/2] COPY index.html /usr/share/nginx/html/                0.0s
=> exporting to image                                           0.0s
=> => exporting layers                                           0.0s
=> => writing image sha256:ae8da0cb9c9f27de74718a60df81952553ad7f1f3adf8fb 0.0s
=> => naming to docker.io/maazpatel24/nginx-k8s-ingress:latest 0.0s
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest
/staticWebapp$
```

- Push the Docker image to Docker Hub or a local registry.

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest
/staticWebapp$ docker push maazpatel24/nginx-k8s-ingress:latest
The push refers to repository [docker.io/maazpatel24/nginx-k8s-ingress]
9ea759a8e104: Pushed
56b6d3be75f9: Mounted from library/nginx
0c6c257920c8: Mounted from library/nginx
92d0d4e97019: Mounted from library/nginx
7190c87a0e8a: Mounted from library/nginx
933a3ce2c78a: Mounted from library/nginx
32cfaf91376f: Mounted from library/nginx
32148f9f6c5a: Mounted from library/nginx
latest: digest: sha256:212e7cc50fb182abcd003d296d98d6feb6069fb1ecb10f357959c5fa33
e27ba size: 1985
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest
/staticWebapp$
```

3. Kubernetes Deployment:

- Write a Kubernetes deployment manifest to deploy the static web application.
- Write a Kubernetes service manifest to expose the static web application within the cluster.
- Apply the deployment and service manifests to the Kubernetes cluster.


```
Every 2.0s: kubectl get all

NAME                                READY   STATUS    RESTARTS   AGE
pod/nginx-webapp-deploy-74b7c77fd8-b9tsg  1/1     Running   0           112s
pod/nginx-webapp-deploy-74b7c77fd8-kj84g  1/1     Running   0           112s

NAME                                TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
service/kubernetes                  ClusterIP     10.96.0.1    <none>        443/TCP          63d
service/nginx-webapp-service        NodePort      10.101.48.136 <none>        80:30024/TCP     112s

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-webapp-deploy  2/2     2             2           112s

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx-webapp-deploy-74b7c77fd8  2         2         2       112s
```



Hello from Maaz Patel!

This is a simple static front-end served by Nginx.

Deliverables:

- Dockerfile for the static web app
- Docker image URL
- Kubernetes deployment and service YAML files

Stage 2: Configuring Ingress Networking

4. Install and Configure Ingress Controller:

- Install an ingress controller (e.g., Nginx Ingress Controller) in the Minikube cluster.

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest/kubernetes$ minikube start --addons=ingress
🐳 minikube v1.32.0 on Ubuntu 20.04
🌟 Using the docker driver based on existing profile
👍 Starting control plane node minikube in cluster minikube
🚚 Pulling base image ...
🔄 Updating the running docker "minikube" container ...
🎉 minikube 1.33.1 is available! Download it: https://github.com/kubernetes/minikube/releases/tag/v1.33.1
💡 To disable this notice, run: 'minikube config set WantUpdateNotification false'

🐳 Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
🔍 Verifying Kubernetes components...
  ▪ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
  ▪ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
  ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
  ▪ Using image registry.k8s.io/ingress-nginx/controller:v1.9.4
🔍 Verifying ingress addon...
🌟 Enabled addons: storage-provisioner, default-storageclass, ingress
🏁 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

- Verify the ingress controller is running and accessible.

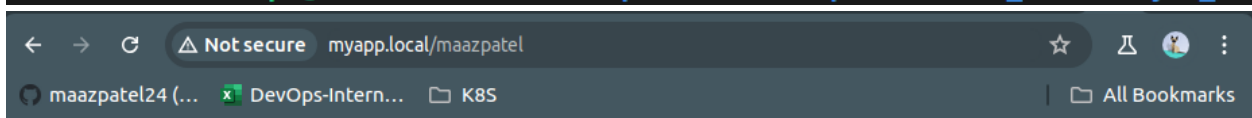
```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest/kubernetes$ kubectl get pods -n ingress-nginx
```

NAME	READY	STATUS	RESTARTS	AGE
ingress-nginx-admission-create-fzxhf	0/1	Completed	0	58s
ingress-nginx-admission-patch-gvs6z	0/1	Completed	0	58s
ingress-nginx-controller-7c6974c4d8-wptqt	1/1	Running	8 (2m21s ago)	58s

5. Create Ingress Resource:

- Write an ingress resource manifest to route external traffic to the static web application.
- Configure advanced ingress rules for path-based routing and host-based routing (use at least two different hostnames and paths).

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/Lecture_notes/Day-9_IngressManifest/kubernetes$ curl myapp.local/maazpatel
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CustomPage</title>
</head>
<body>
  <center><h1>Hello from Maaz Patel!</h1></center>
  <center><p>This is a simple static front-end served by Nginx.</p></center>
</body>
</html>
```



Hello from Maaz Patel!

This is a simple static front-end served by Nginx.

- Implement TLS termination for secure connections.
(`openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout tls.key -out tls.crt`)
- Configure URL rewriting in the ingress resource to modify incoming URLs before they reach the backend services.

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/Lecture_notes/Day-9_IngressManifest$ curl -k https://myapp.local/maazpatel
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CustomPage</title>
</head>
<body>
  <center><h1>Hello from Maaz Patel!</h1></center>
  <center><p>This is a simple static front-end served by Nginx.</p></center>
</body>
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/Lecture_notes/Day-9_IngressManifest
```

- Enable sticky sessions to ensure that requests from the same client are directed to the same backend pod.

Deliverables:

- Ingress controller installation commands/scripts
- Ingress resource YAML file with advanced routing, TLS configuration, URL rewriting, and sticky sessions

Stage 3: Implementing Horizontal Pod Autoscaling

6. Configure Horizontal Pod Autoscaler:

- Write a horizontal pod autoscaler (HPA) manifest to automatically scale the static web application pods based on CPU utilization.
- Set thresholds for minimum and maximum pod replicas.

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressManifest/kubernetes$ kubectl get hpa
```

NAME	REFERENCE	TARGETS	MINPODS	MAXPODS	REPLICAS	AGE
nginx-webapp-hpa	Deployment/nginx-webapp-deploy	0%/50%	2	10	2	18s

7. Stress Testing:

- Perform stress testing to simulate traffic and validate the HPA configuration.
- Monitor the scaling behavior and ensure the application scales up and down based on the load.

```
kubectl run -i --tty load-generator --rm --image=appropriate/curl --
restart=Never -- /bin/sh -c "while sleep 0.01; do curl -sk
https://myapp.local/maazpatel; done"
```

```
Every 2.0s: kubectl get hpa nginx-w... AHMLPT1108: Fri Jul 19 12:05:07 2024
```

NAME	REPLICAS	REFERENCE	TARGETS	MINPODS	MAXP
nginx-webapp-hpa	10	Deployment/nginx-webapp-deploy	10%/3%	1	10

```

einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressMa
nifest$ kubectl get deployments.apps nginx-webapp-deploy
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
nginx-webapp-deploy 10/10   10           10          17h
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressMa

```

Deliverables:

- Horizontal pod autoscaler YAML file
- Documentation or screenshots of the stress testing process and scaling behavior

Stage 4: Final Validation and Cleanup

8. Final Validation:

- Validate the ingress networking, URL rewriting, and sticky sessions configurations by accessing the web application through different hostnames and paths.
- Verify the application's availability and performance during different load conditions.

```
einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_IngressMa
nifest/kubernetes$ curl -k https://myapp.maaz/diffpath
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CustomPage</title>
</head>
<body>
  <center><h1>Hello from Maaz Patel!</h1></center>
  <center><p>This is a simple static front-end served by Nginx.</p></cente
r>
</body>
</html>einfochips@AHMLPT1108:~/Desktop/OneDrive/Repos/lecture_notes/Day-9_In
gressManifest/kubernetes$
```

9. Cleanup:

- Provide commands or scripts to clean up the Kubernetes resources created during the project (deployments, services, ingress, HPA).

Deliverables:

- Final validation report documenting the testing process and results
- Cleanup commands/scripts