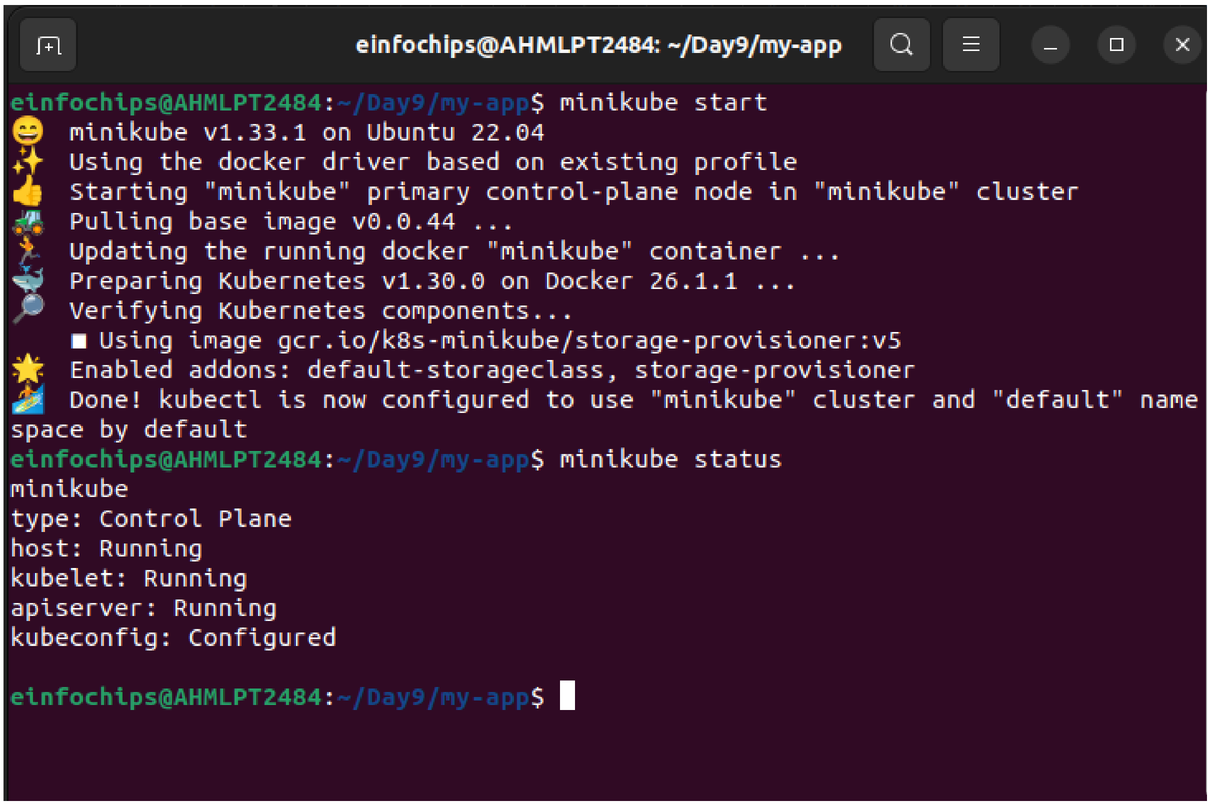
Project Overview

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

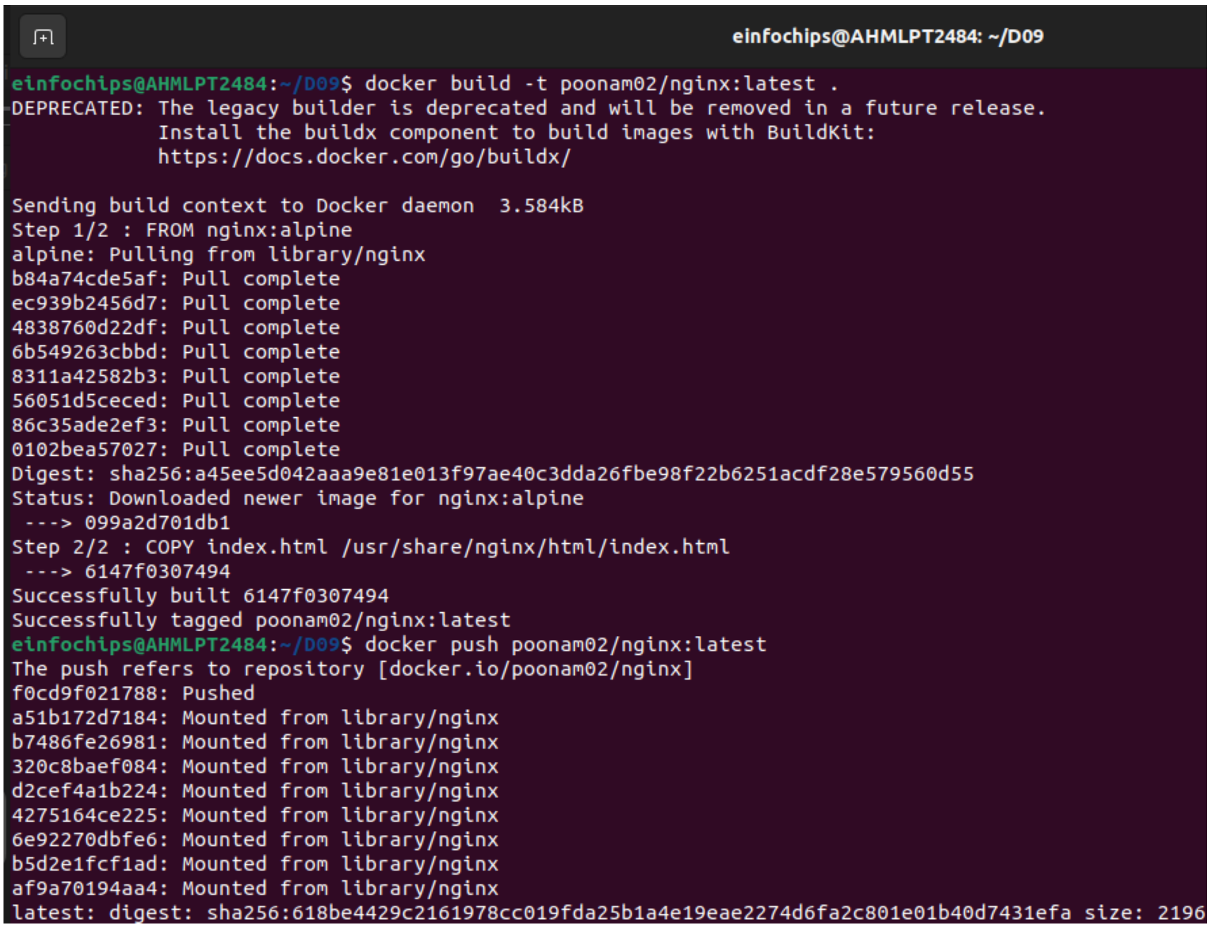
1. **Set Up Minikube:**
    - Ensure Minikube is installed and running on the local Ubuntu machine.
    - Verify the Kubernetes cluster is functioning correctly.



```
einfochips@AHMLPT2484:~/Day9/my-app$ minikube start
😄  minikube v1.33.1 on Ubuntu 22.04
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.44 ...
🏃  Updating the running docker "minikube" container ...
🐳  Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: default-storageclass, storage-provisioner
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" name
space by default
einfochips@AHMLPT2484:~/Day9/my-app$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

einfochips@AHMLPT2484:~/Day9/my-app$ 
```

2. **Deploy Static Web App:**
    - Create a Dockerfile for a simple static web application (e.g., an HTML page served by Nginx).
    - Build a Docker image for the static web application.

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

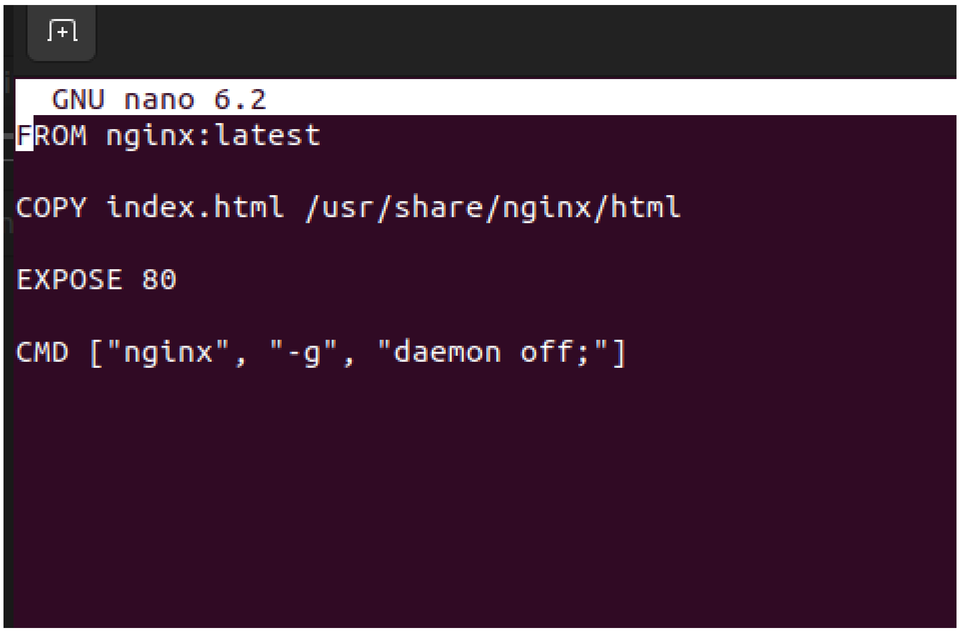


```
einfochips@AHMLPT2484: ~/D09
einfochips@AHMLPT2484:~/D09$ docker build -t poonam02/nginx:latest .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.584kB
Step 1/2 : FROM nginx:alpine
alpine: Pulling from library/nginx
b84a74cde5af: Pull complete
ec939b2456d7: Pull complete
4838760d22df: Pull complete
6b549263cbbd: Pull complete
8311a42582b3: Pull complete
56051d5ceced: Pull complete
86c35ade2ef3: Pull complete
0102bea57027: Pull complete
Digest: sha256:a45ee5d042aaa9e81e013f97ae40c3dda26fbe98f22b6251acdf28e579560d55
Status: Downloaded newer image for nginx:alpine
 ---> 099a2d701db1
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
 ---> 6147f0307494
Successfully built 6147f0307494
Successfully tagged poonam02/nginx:latest
einfochips@AHMLPT2484:~/D09$ docker push poonam02/nginx:latest
The push refers to repository [docker.io/poonam02/nginx]
f0cd9f021788: Pushed
a51b172d7184: Mounted from library/nginx
b7486fe26981: Mounted from library/nginx
320c8baef084: Mounted from library/nginx
d2cef4a1b224: Mounted from library/nginx
4275164ce225: Mounted from library/nginx
6e92270dbfe6: Mounted from library/nginx
b5d2e1fcf1ad: Mounted from library/nginx
af9a70194aa4: Mounted from library/nginx
latest: digest: sha256:618be4429c2161978cc019fda25b1a4e19eae2274d6fa2c801e01b40d7431efa size: 2196
```

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.

Deliverables:

- Dockerfile for the static web app



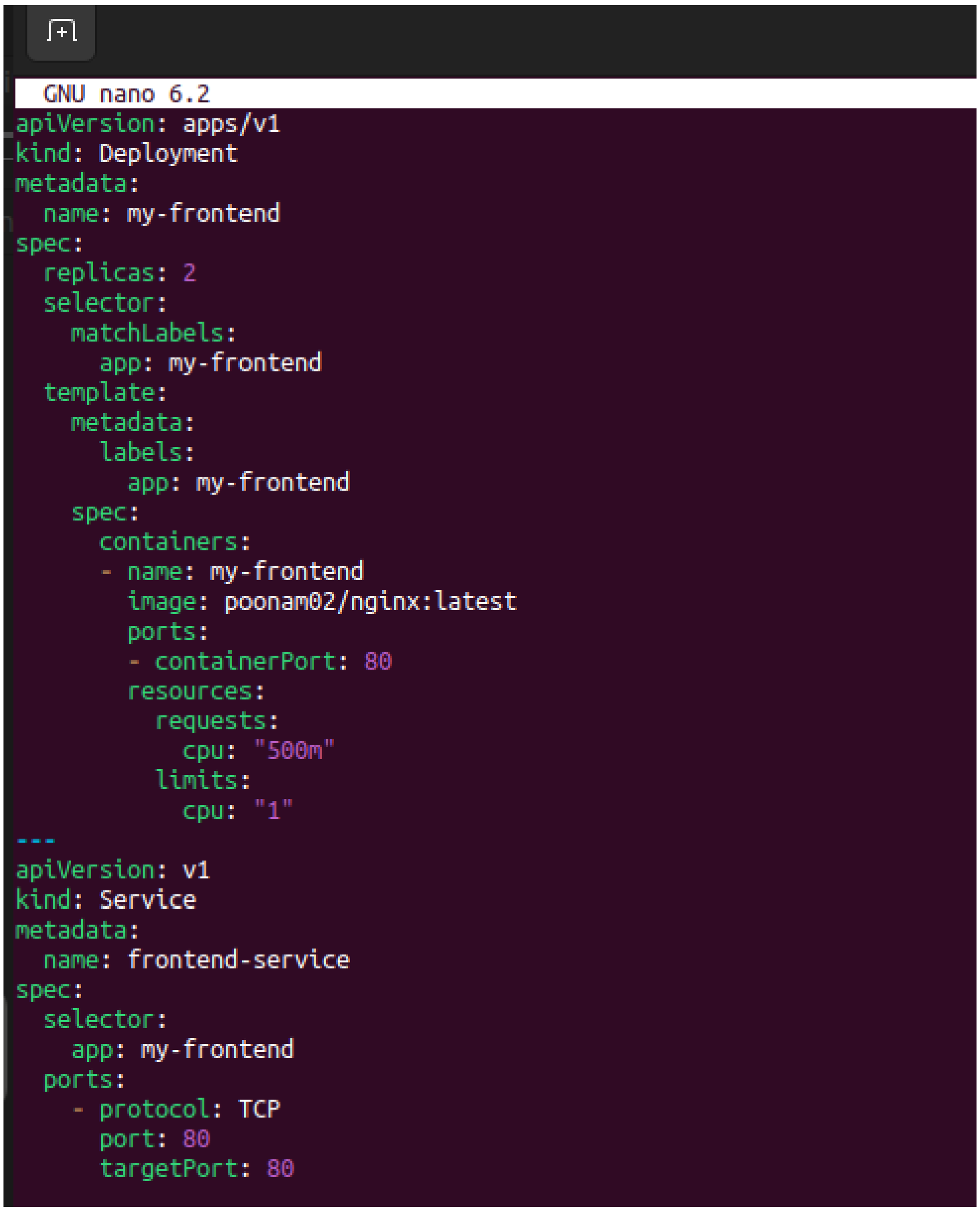
```
  GNU nano 6.2
FROM nginx:latest

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

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

- Docker image URL

  https://hub.docker.com/repository/docker/poonam02/nginx/general

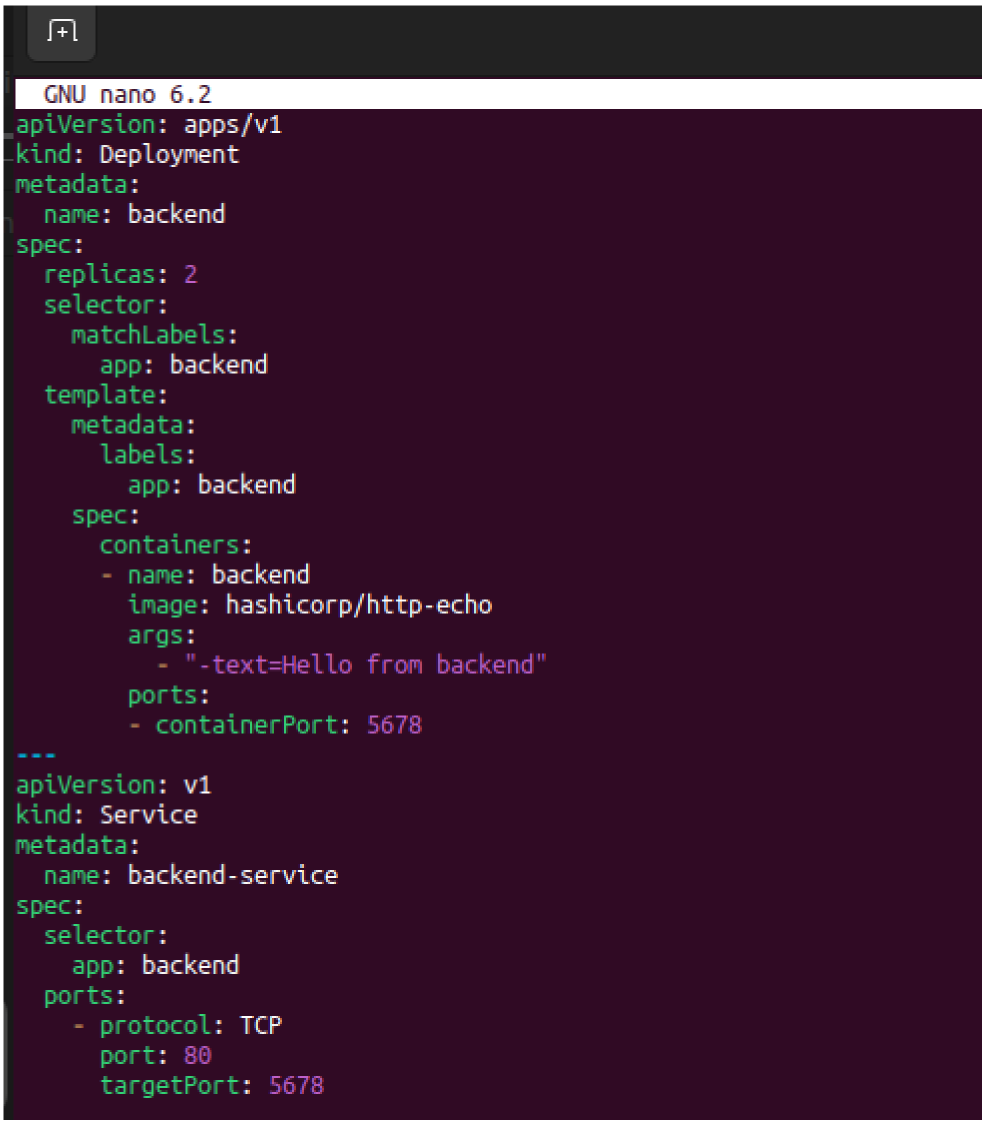
- Kubernetes deployment and service YAML files

  1. Frontend deployment and service yaml file



```
  GNU nano 6.2
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-frontend
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-frontend
  template:
    metadata:
      labels:
        app: my-frontend
    spec:
      containers:
      - name: my-frontend
        image: poonam02/nginx:latest
        ports:
        - containerPort: 80
        resources:
          requests:
            cpu: "500m"
          limits:
            cpu: "1"
---
apiVersion: v1
kind: Service
metadata:
  name: frontend-service
spec:
  selector:
    app: my-frontend
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

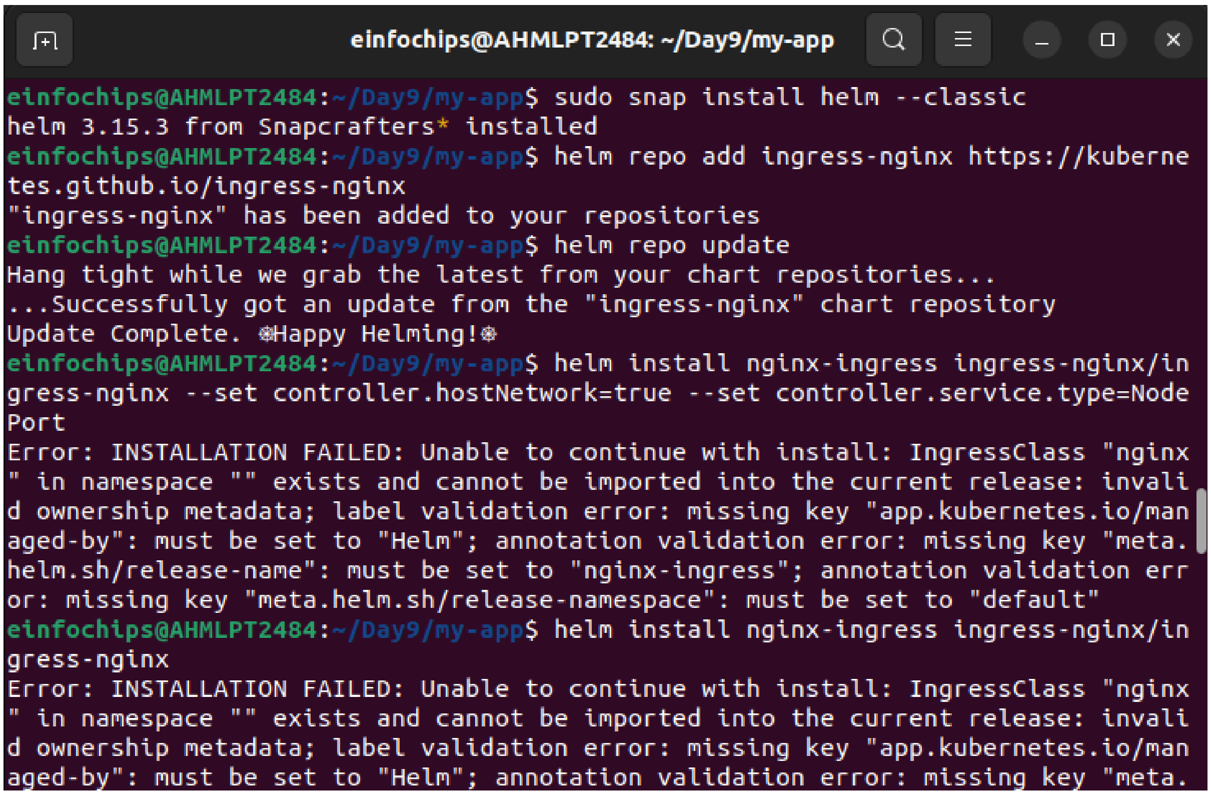
2. Backend deployment and service yaml file



```
  GNU nano 6.2
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend
spec:
  replicas: 2
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
      labels:
        app: backend
    spec:
      containers:
      - name: backend
        image: hashicorp/http-echo
        args:
          - "-text=Hello from backend"
        ports:
        - containerPort: 5678
---
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  selector:
    app: backend
  ports:
    - protocol: TCP
      port: 80
      targetPort: 5678
```

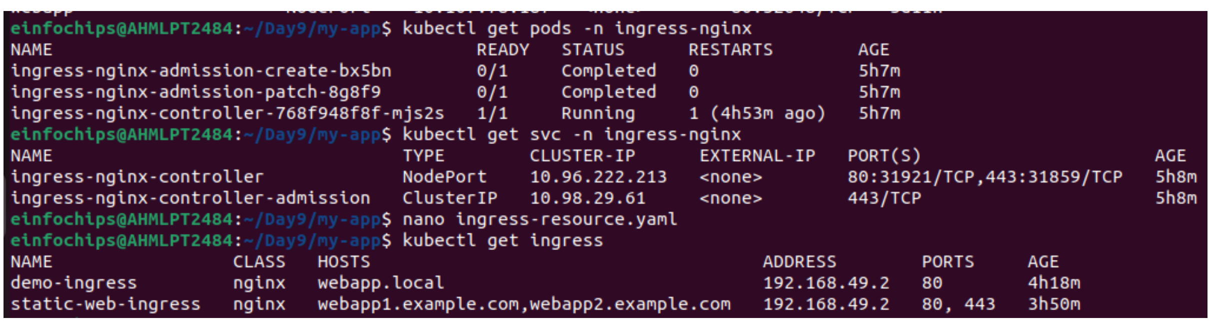
Stage 2: Configuring Ingress Networking

4. **Install and Configure Ingress Controller:**
    - Install an ingress controller (e.g., Nginx Ingress Controller) in the Minikube cluster.



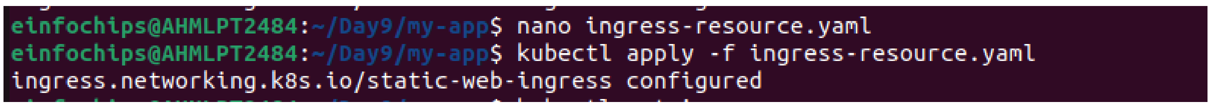
```
einfochips@AHMLPT2484: ~/Day9/my-app
einfochips@AHMLPT2484:~/Day9/my-app$ sudo snap install helm --classic
helm 3.15.3 from Snapcrafters* installed
einfochips@AHMLPT2484:~/Day9/my-app$ helm repo add ingress-nginx https://kuberne
tes.github.io/ingress-nginx
"ingress-nginx" has been added to your repositories
einfochips@AHMLPT2484:~/Day9/my-app$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "ingress-nginx" chart repository
Update Complete. ⎈Happy Helming!⎈
einfochips@AHMLPT2484:~/Day9/my-app$ helm install nginx-ingress ingress-nginx/in
gress-nginx --set controller.hostNetwork=true --set controller.service.type=Node
Port
Error: INSTALLATION FAILED: Unable to continue with install: IngressClass "nginx
" in namespace "" exists and cannot be imported into the current release: invali
d ownership metadata; label validation error: missing key "app.kubernetes.io/man
aged-by": must be set to "Helm"; annotation validation error: missing key "meta.
helm.sh/release-name": must be set to "nginx-ingress"; annotation validation err
or: missing key "meta.helm.sh/release-namespace": must be set to "default"
einfochips@AHMLPT2484:~/Day9/my-app$ helm install nginx-ingress ingress-nginx/in
gress-nginx
Error: INSTALLATION FAILED: Unable to continue with install: IngressClass "nginx
" in namespace "" exists and cannot be imported into the current release: invali
d ownership metadata; label validation error: missing key "app.kubernetes.io/man
aged-by": must be set to "Helm"; annotation validation error: missing key "meta.
```

    - Verify the ingress controller is running and accessible.



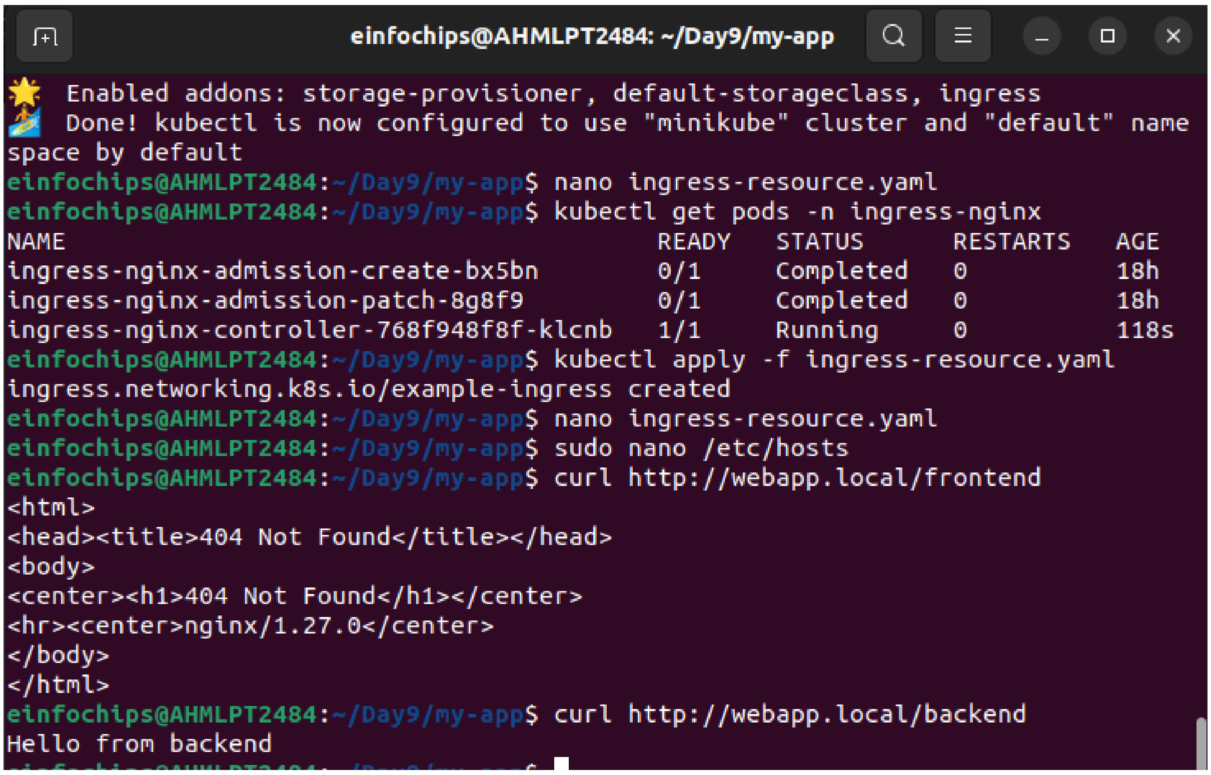
```
einfochips@AHMLPT2484:~/Day9/my-app$ kubectl get pods -n ingress-nginx
NAME                                       READY   STATUS      RESTARTS        AGE
ingress-nginx-admission-create-bx5bn       0/1     Completed   0               5h7m
ingress-nginx-admission-patch-8g8f9        0/1     Completed   0               5h7m
ingress-nginx-controller-768f948f8f-mjs2s  1/1     Running     1 (4h53m ago)   5h7m
einfochips@AHMLPT2484:~/Day9/my-app$ kubectl get svc -n ingress-nginx
NAME                                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)                      AGE
ingress-nginx-controller             NodePort    10.96.222.213   <none>        80:31921/TCP,443:31859/TCP   5h8m
ingress-nginx-controller-admission   ClusterIP   10.98.29.61     <none>        443/TCP                      5h8m
einfochips@AHMLPT2484:~/Day9/my-app$ nano ingress-resource.yaml
einfochips@AHMLPT2484:~/Day9/my-app$ kubectl get ingress
NAME                 CLASS   HOSTS                                   ADDRESS        PORTS     AGE
demo-ingress         nginx   webapp.local                            192.168.49.2   80        4h18m
static-web-ingress   nginx   webapp1.example.com,webapp2.example.com 192.168.49.2   80, 443   3h50m
```

5. **Create Ingress Resource:**
    - Write an ingress resource manifest to route external traffic to the static web application.



```
einfochips@AHMLPT2484:~/Day9/my-app$ nano ingress-resource.yaml
einfochips@AHMLPT2484:~/Day9/my-app$ kubectl apply -f ingress-resource.yaml
ingress.networking.k8s.io/static-web-ingress configured
```

- ○ Configure advanced ingress rules for path-based routing and host-based routing (use at least two different hostnames and paths).



einfochips@AHMLPT2484: ~/Day9/my-app

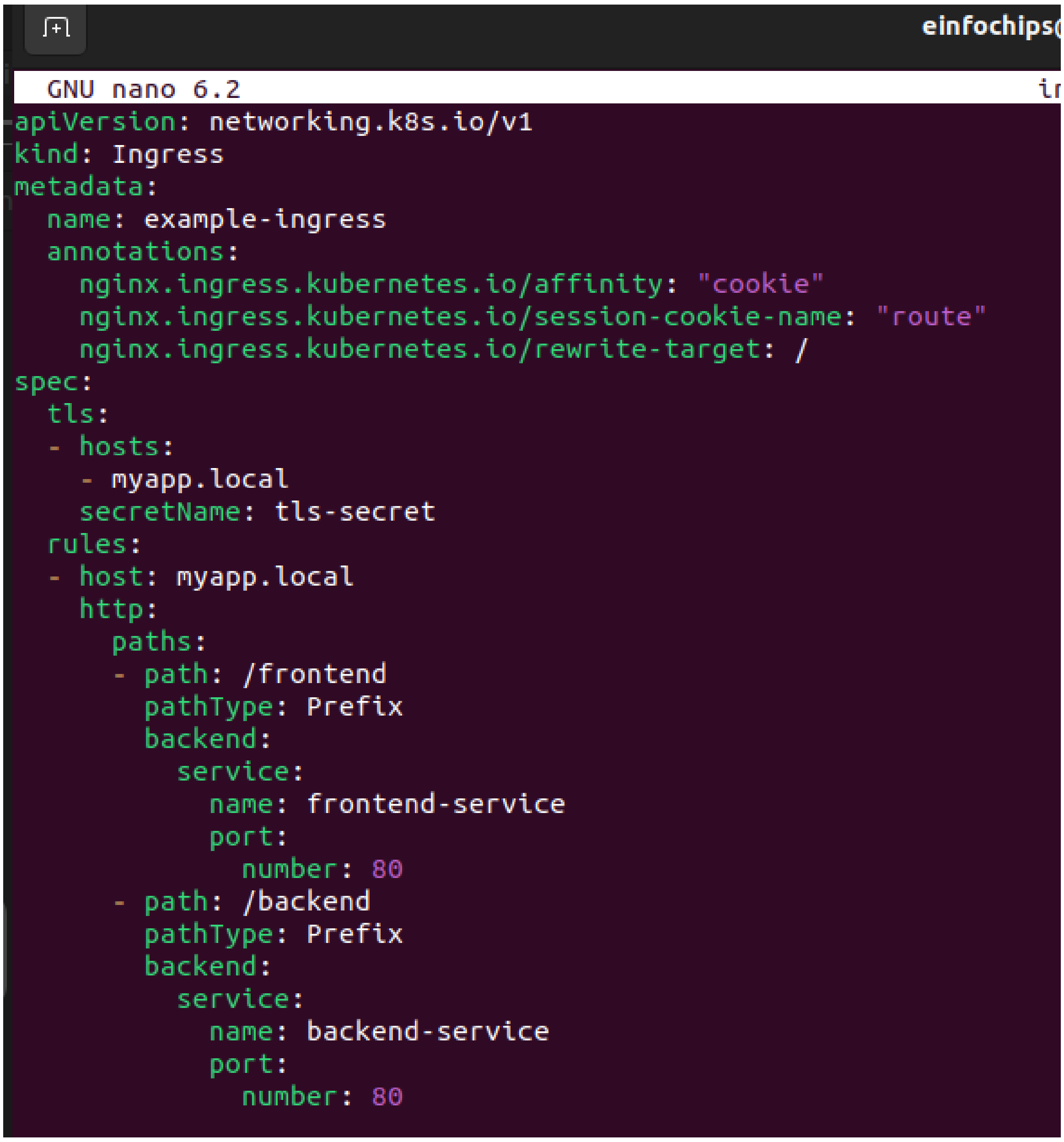
```
🌟  Enabled addons: storage-provisioner, default-storageclass, ingress
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" name
space by default
einfochips@AHMLPT2484:~/Day9/my-app$ nano ingress-resource.yaml
einfochips@AHMLPT2484:~/Day9/my-app$ kubectl get pods -n ingress-nginx
NAME                                        READY   STATUS      RESTARTS   AGE
ingress-nginx-admission-create-bx5bn        0/1     Completed   0          18h
ingress-nginx-admission-patch-8g8f9         0/1     Completed   0          18h
ingress-nginx-controller-768f948f8f-klcnb   1/1     Running     0          118s
einfochips@AHMLPT2484:~/Day9/my-app$ kubectl apply -f ingress-resource.yaml
ingress.networking.k8s.io/example-ingress created
einfochips@AHMLPT2484:~/Day9/my-app$ nano ingress-resource.yaml
einfochips@AHMLPT2484:~/Day9/my-app$ sudo nano /etc/hosts
einfochips@AHMLPT2484:~/Day9/my-app$ curl http://webapp.local/frontend
<html>
<head><title>404 Not Found</title></head>
<body>
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.27.0</center>
</body>
</html>
einfochips@AHMLPT2484:~/Day9/my-app$ curl http://webapp.local/backend
Hello from backend
```

- ○ Implement TLS termination for secure connections.
- ○ Configure URL rewriting in the ingress resource to modify incoming URLs before they reach the backend services.
- ○ Enable sticky sessions to ensure that requests from the same client are directed to the same backend pod.

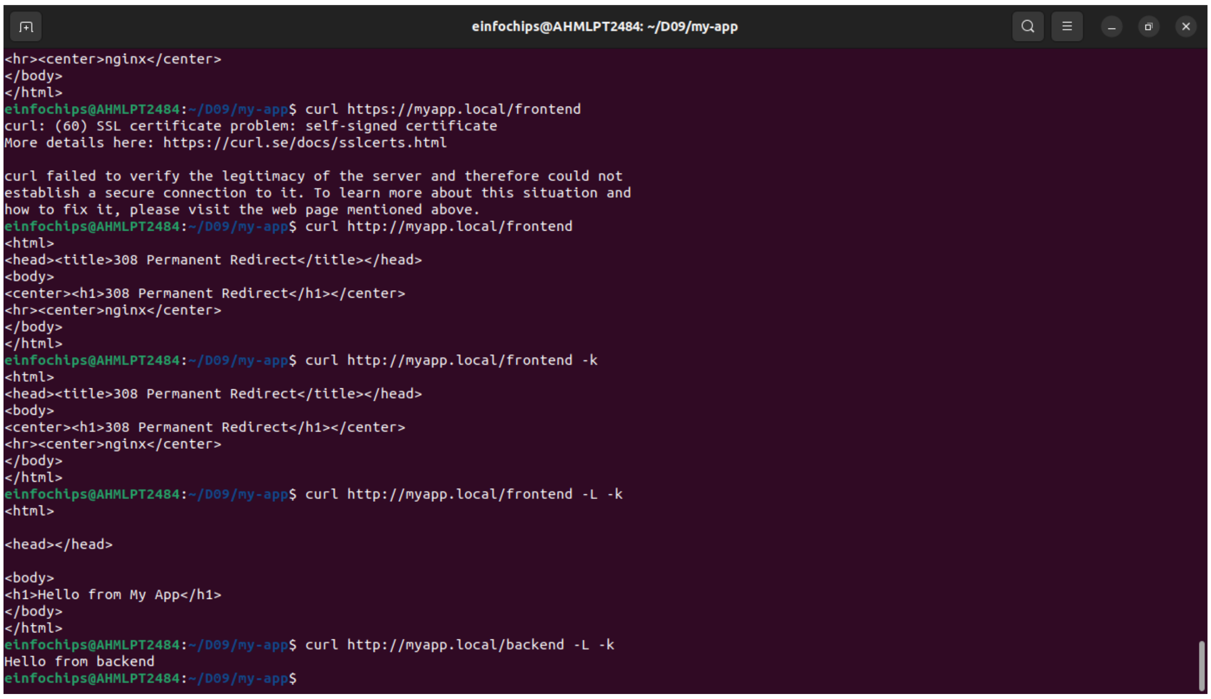
Deliverables:

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

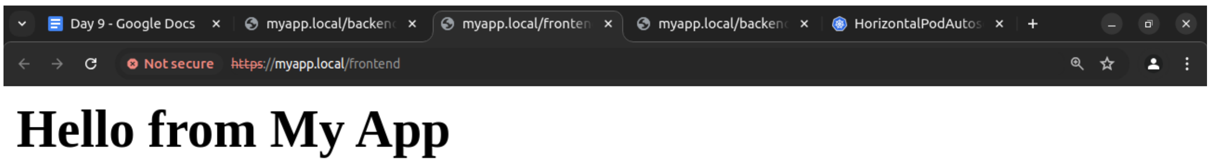
Ingress resource.yaml file



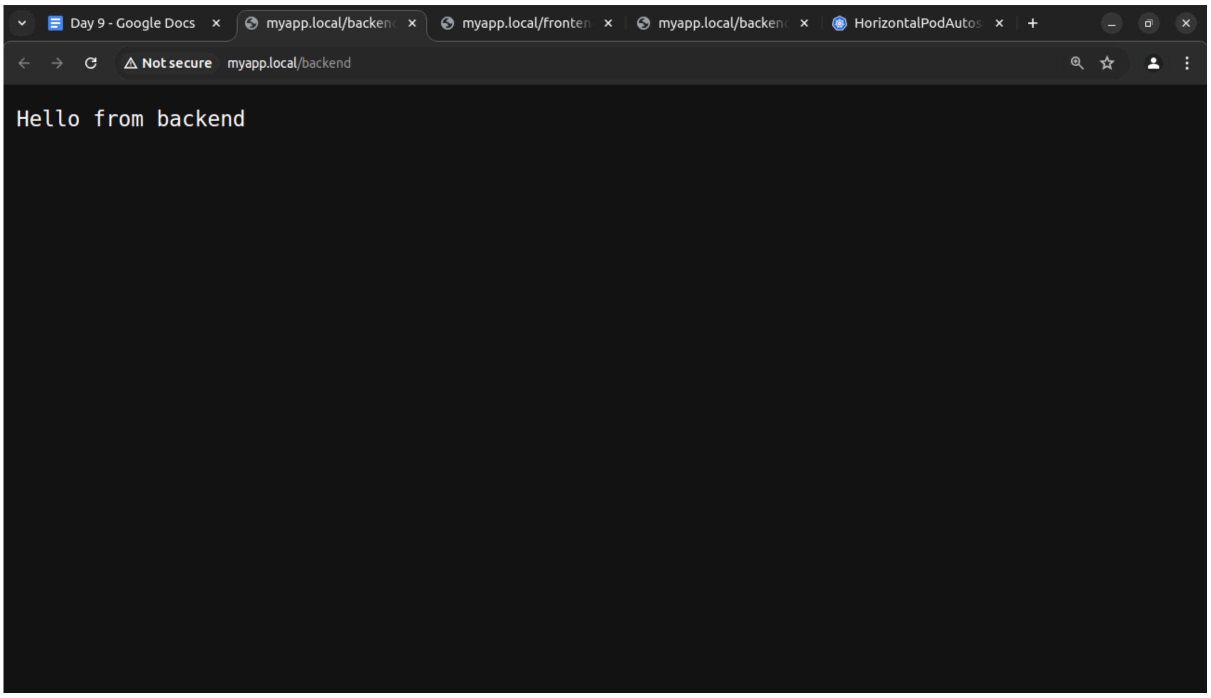
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
  annotations:
    nginx.ingress.kubernetes.io/affinity: "cookie"
    nginx.ingress.kubernetes.io/session-cookie-name: "route"
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  tls:
  - hosts:
    - myapp.local
    secretName: tls-secret
  rules:
  - host: myapp.local
    http:
      paths:
      - path: /frontend
        pathType: Prefix
        backend:
          service:
            name: frontend-service
            port:
              number: 80
      - path: /backend
        pathType: Prefix
        backend:
          service:
            name: backend-service
            port:
              number: 80
```



```
<hr><center>nginx</center>
</body>
</html>
einfochips@AHMLPT2484:~/D09/my-app$ curl https://myapp.local/frontend
curl: (60) SSL certificate problem: self-signed certificate
More details here: https://curl.se/docs/sslcerts.html

curl failed to verify the legitimacy of the server and therefore could not
establish a secure connection to it. To learn more about this situation and
how to fix it, please visit the web page mentioned above.
einfochips@AHMLPT2484:~/D09/my-app$ curl http://myapp.local/frontend
<html>
<head><title>308 Permanent Redirect</title></head>
<body>
<center><h1>308 Permanent Redirect</h1></center>
<hr><center>nginx</center>
</body>
</html>
einfochips@AHMLPT2484:~/D09/my-app$ curl http://myapp.local/frontend -k
<html>
<head><title>308 Permanent Redirect</title></head>
<body>
<center><h1>308 Permanent Redirect</h1></center>
<hr><center>nginx</center>
</body>
</html>
einfochips@AHMLPT2484:~/D09/my-app$ curl http://myapp.local/frontend -L -k
<html>

<head></head>

<body>
<h1>Hello from My App</h1>
</body>
</html>
einfochips@AHMLPT2484:~/D09/my-app$ curl http://myapp.local/backend -L -k
Hello from backend
einfochips@AHMLPT2484:~/D09/my-app$
```



Hello from My App



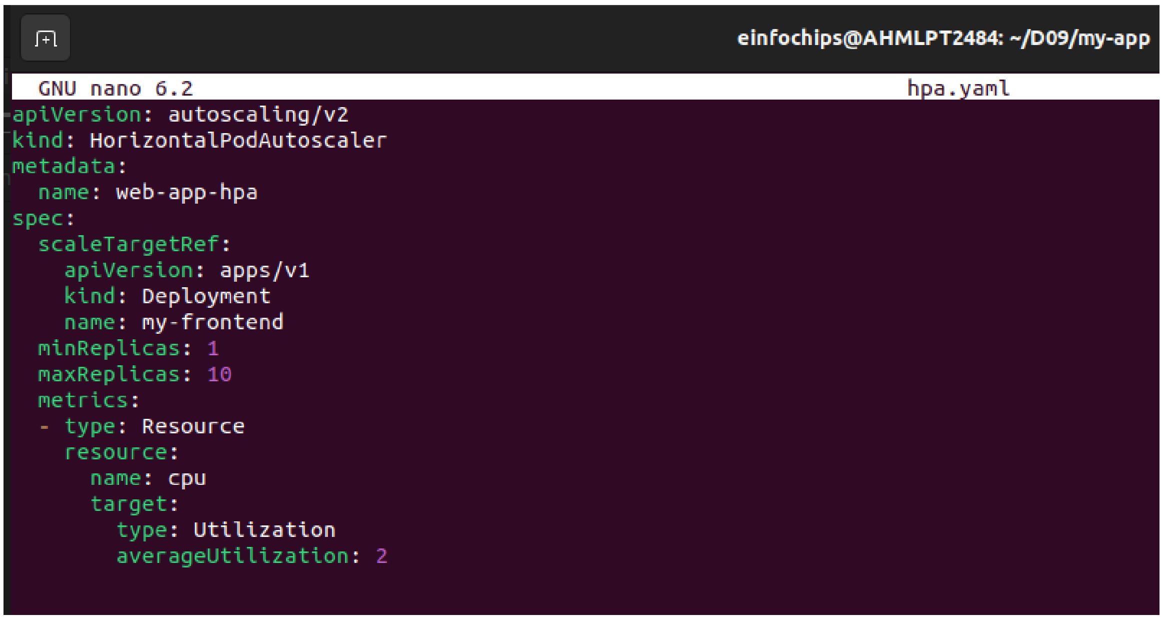
```
Hello from backend
```

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.
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.

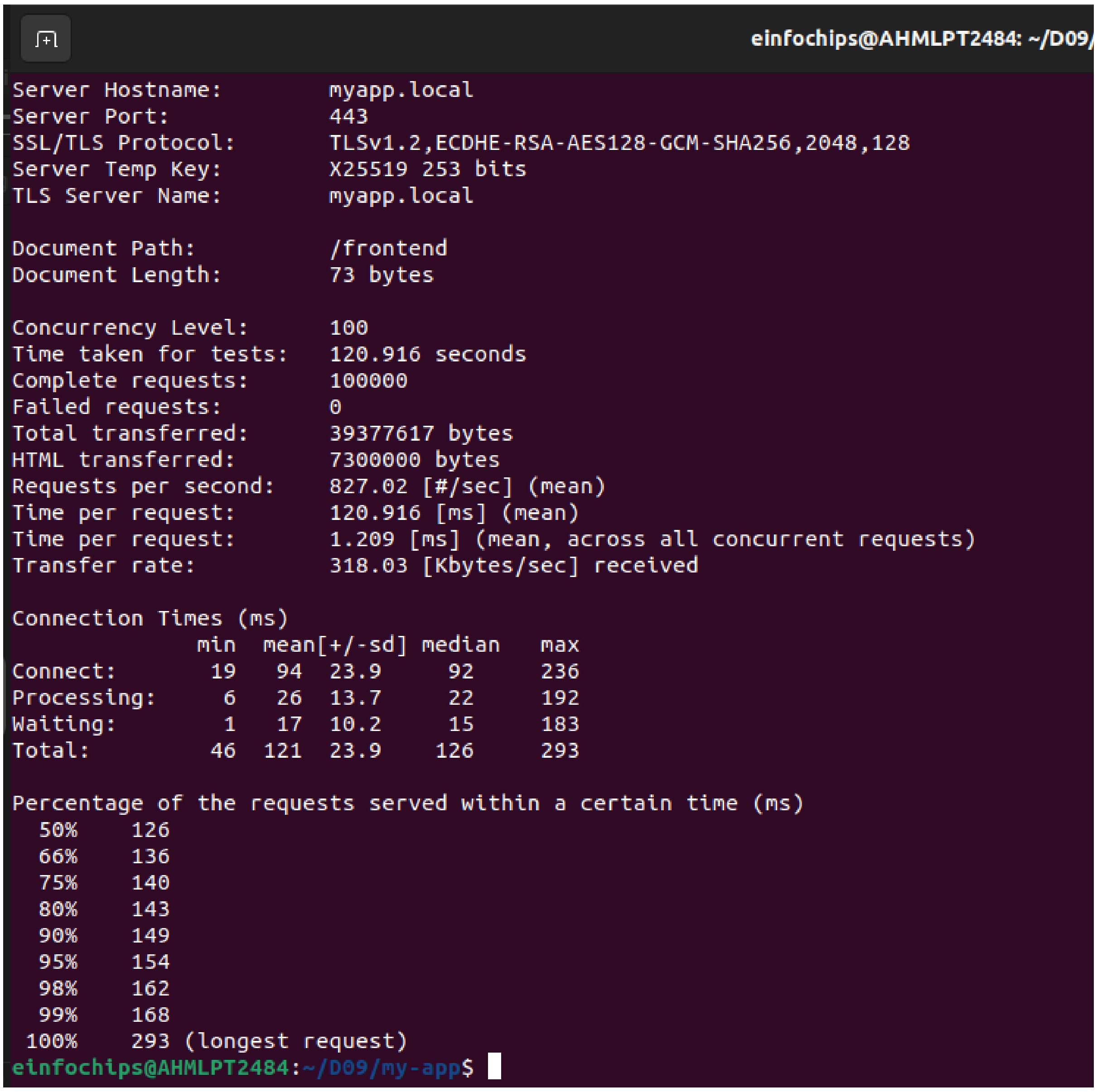
Deliverables:

- Horizontal pod autoscaler YAML file

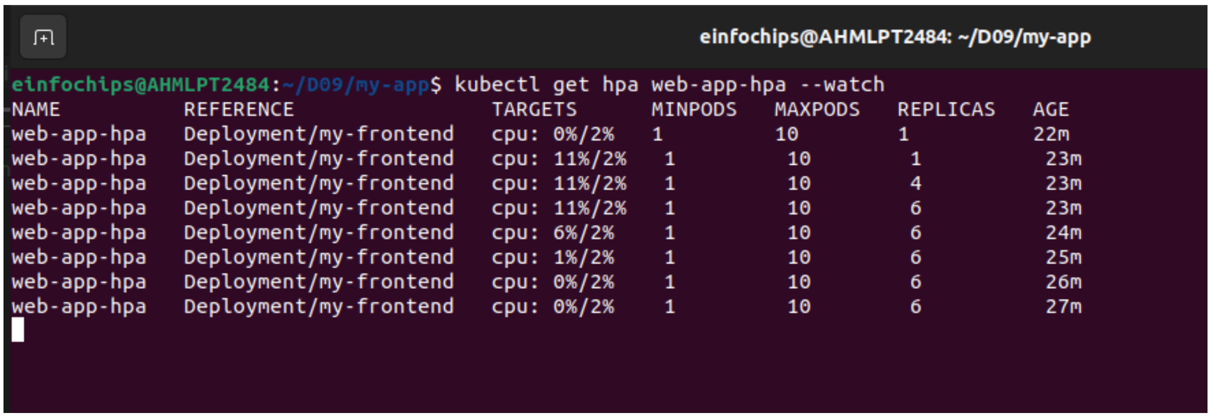


```
einfochips@AHMLPT2484: ~/D09/my-app
  GNU nano 6.2                                                        hpa.yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: web-app-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: my-frontend
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 2
```

- Documentation or screenshots of the stress testing process and scaling behavior



```
Server Hostname:        myapp.local
Server Port:            443
SSL/TLS Protocol:       TLSv1.2,ECDHE-RSA-AES128-GCM-SHA256,2048,128
Server Temp Key:        X25519 253 bits
TLS Server Name:        myapp.local

Document Path:          /frontend
Document Length:        73 bytes

Concurrency Level:      100
Time taken for tests:   120.916 seconds
Complete requests:      100000
Failed requests:        0
Total transferred:      39377617 bytes
HTML transferred:       7300000 bytes
Requests per second:    827.02 [#/sec] (mean)
Time per request:       120.916 [ms] (mean)
Time per request:       1.209 [ms] (mean, across all concurrent requests)
Transfer rate:          318.03 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       19   94  23.9     92     236
Processing:     6   26  13.7     22     192
Waiting:        1   17  10.2     15     183
Total:         46  121  23.9    126     293

Percentage of the requests served within a certain time (ms)
  50%    126
  66%    136
  75%    140
  80%    143
  90%    149
  95%    154
  98%    162
  99%    168
 100%    293 (longest request)
einfochips@AHMLPT2484:~/D09/my-app$
```



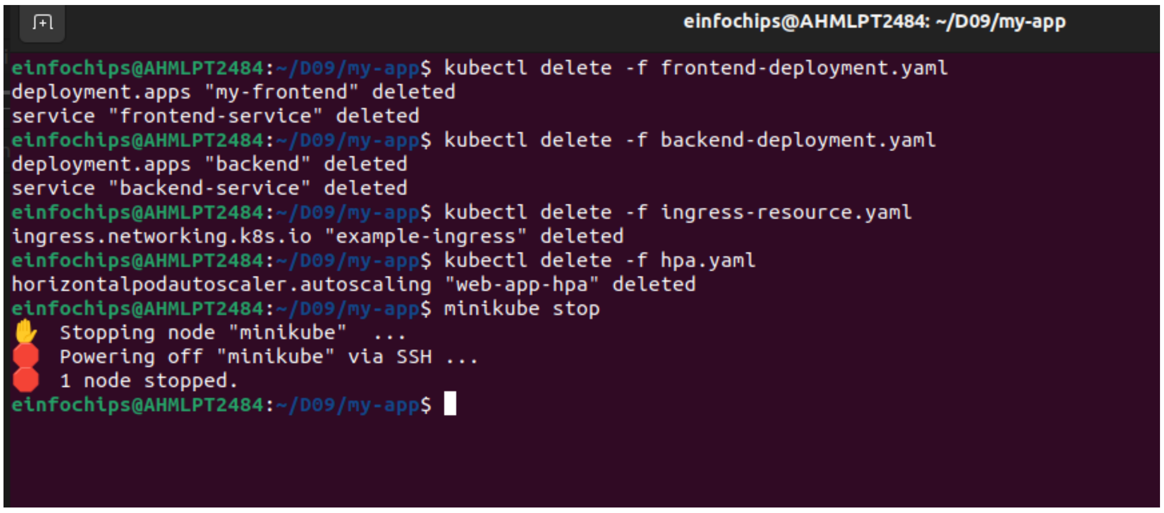
```
einfochips@AHMLPT2484:~/D09/my-app$ kubectl get hpa web-app-hpa --watch
NAME          REFERENCE                 TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
web-app-hpa   Deployment/my-frontend    cpu: 0%/2%  1         10        1          22m
web-app-hpa   Deployment/my-frontend    cpu: 11%/2%   1         10        1          23m
web-app-hpa   Deployment/my-frontend    cpu: 11%/2%   1         10        4          23m
web-app-hpa   Deployment/my-frontend    cpu: 11%/2%   1         10        6          23m
web-app-hpa   Deployment/my-frontend    cpu: 6%/2%    1         10        6          24m
web-app-hpa   Deployment/my-frontend    cpu: 1%/2%    1         10        6          25m
web-app-hpa   Deployment/my-frontend    cpu: 0%/2%    1         10        6          26m
web-app-hpa   Deployment/my-frontend    cpu: 0%/2%    1         10        6          27m
```

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.
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



```
einfochips@AHMLPT2484: ~/D09/my-app
einfochips@AHMLPT2484:~/D09/my-app$ kubectl delete -f frontend-deployment.yaml
deployment.apps "my-frontend" deleted
service "frontend-service" deleted
einfochips@AHMLPT2484:~/D09/my-app$ kubectl delete -f backend-deployment.yaml
deployment.apps "backend" deleted
service "backend-service" deleted
einfochips@AHMLPT2484:~/D09/my-app$ kubectl delete -f ingress-resource.yaml
ingress.networking.k8s.io "example-ingress" deleted
einfochips@AHMLPT2484:~/D09/my-app$ kubectl delete -f hpa.yaml
horizontalpodautoscaler.autoscaling "web-app-hpa" deleted
einfochips@AHMLPT2484:~/D09/my-app$ minikube stop
✋  Stopping node "minikube"  ...
🛑  Powering off "minikube" via SSH ...
🛑  1 node stopped.
einfochips@AHMLPT2484:~/D09/my-app$
```