DEPLOYING NGINX ON KUBERNETES

GROUP 4

GROUP MEMBERS

- STEPHEN AMIHERE
- INNOCENTIA AZAL
- BENEDICTA OPOKU-AMANKWAAH
- CLEMENT OWUSU BEMPAH
- ANGELA KESSE
- RIVERSON ATTA
- JOEL AFEDU

OBJECTIVES

- Deploy Nginx on Kubernetes.
- Manage deployments and scaling.
- Expose the app externally via Service.
- Document the process.

TOOLS USED

- Kubernetes Environment: Minikube
- CLI Tool: kubectl Sample
- App: Nginx Web Server System
- Requirements: Installed Docker Working Kubernetes CLI

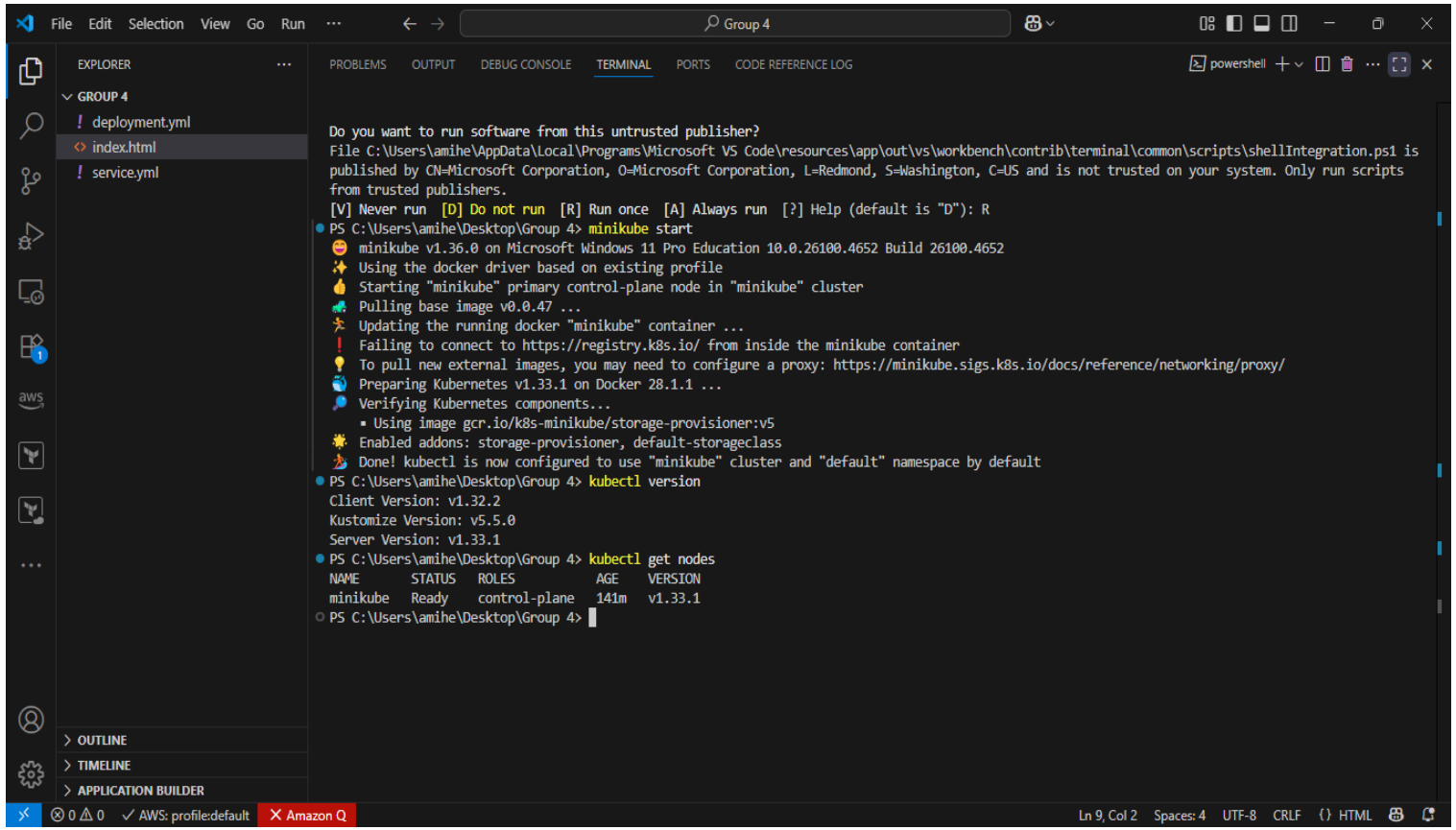
TASK 1: SELECTING A SAMPLE APPLICATION

Chosen App: Nginx

Why Nginx?:

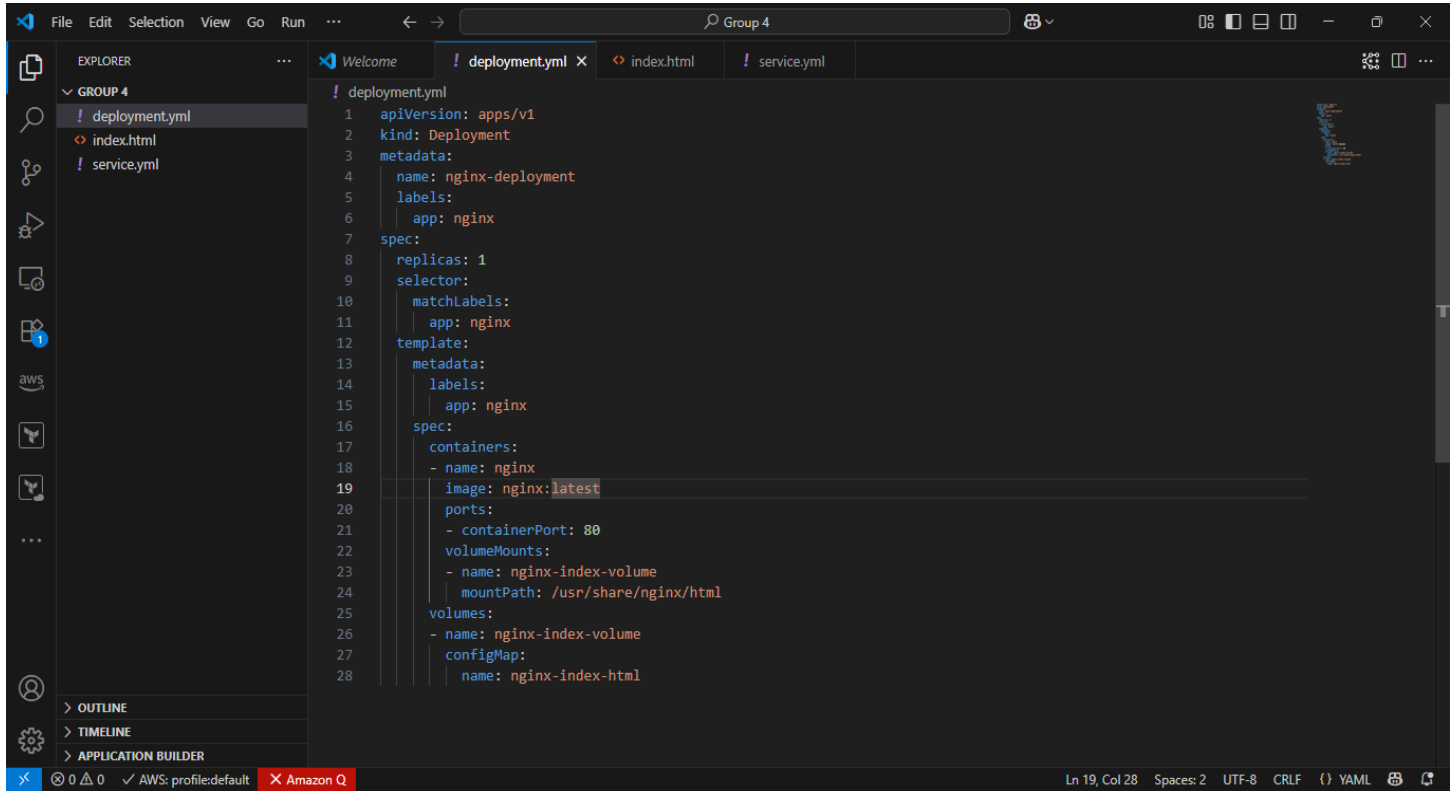
- Nginx is lightweight and easy to deploy.
- It supports customization. You can easily replace the default page with your own index.html to demonstrate custom deployments.

TASK 2: SETTING UP ENVIRONMENT



- For this task, we started minikube by running ***minikube start*** and confirmed whether kubectl was working the command ***kubectl version.***

TASK 3: CREATING A DEPLOYMENT MANIFEST



Before diving into file creation, we first set up a dedicated project folder on our desktop named ***GROUP 4***. This provided a clean workspace and easy access, which we then opened using Visual Studio Code.

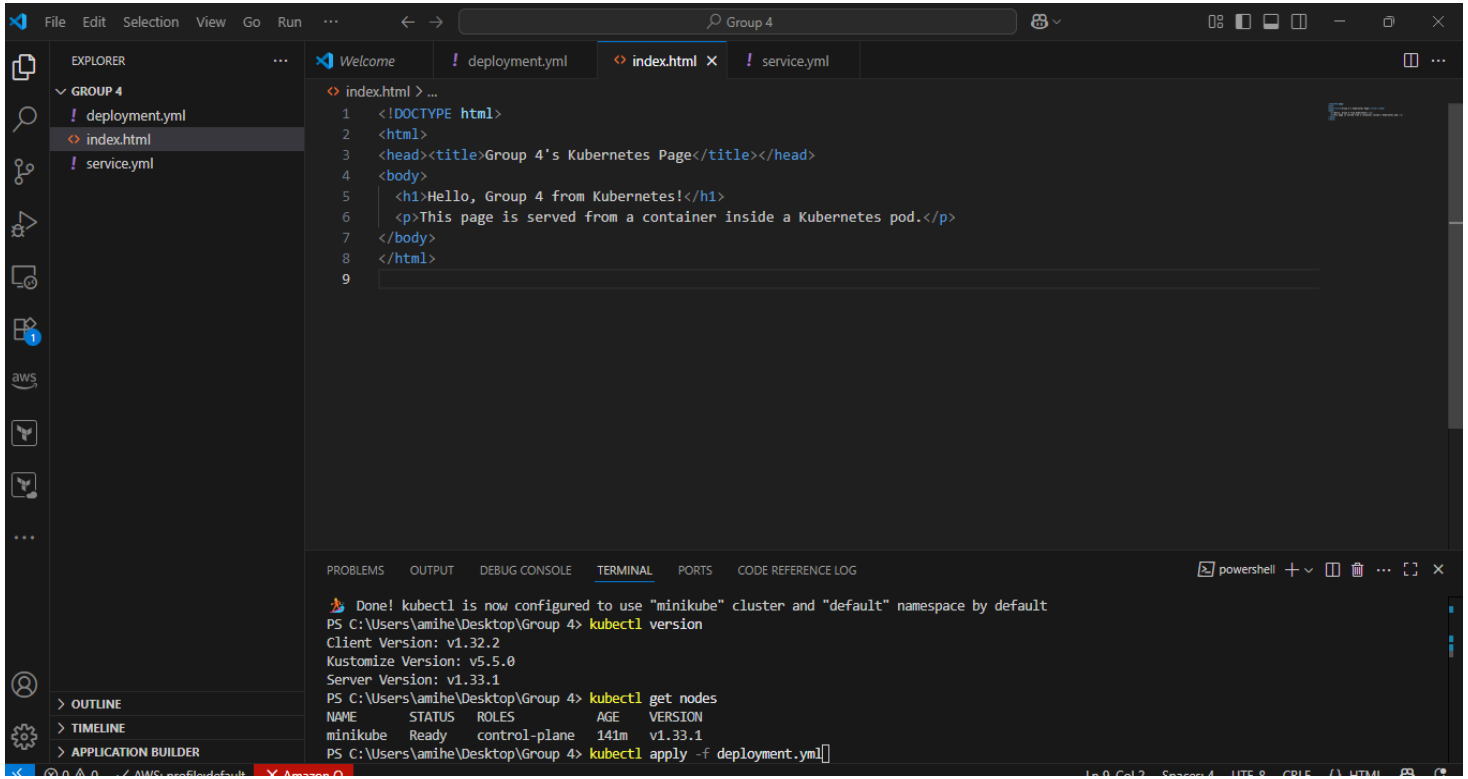
- ❑ Process:

We created file name ***deployment.yml***, that defines how Kubernetes should deploy our app including the container image, replicas, and ports.

- ❑ Purpose:

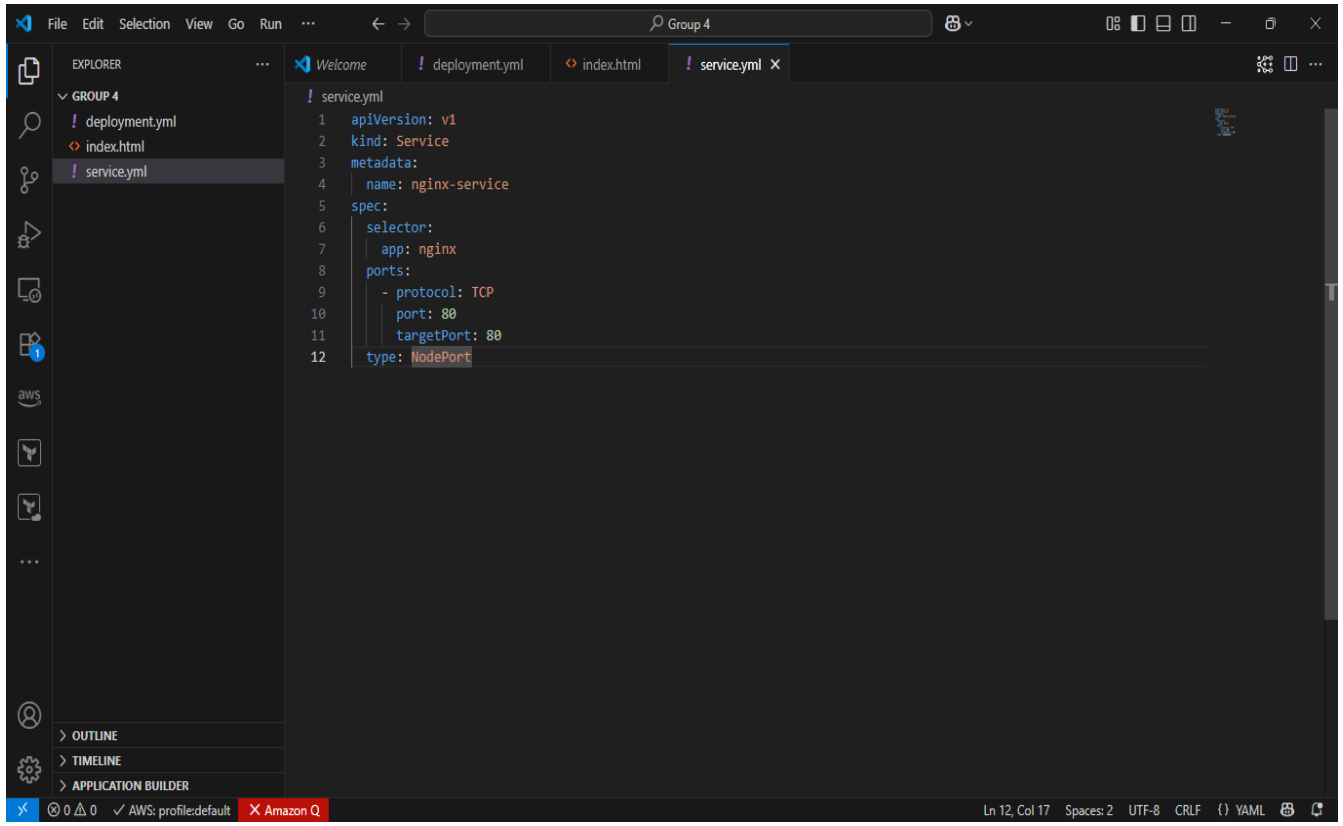
The deployment keeps our app running, even if a pod crashes.

TASK 3 CONT.: CREATING AN HTML FILE



- We also created an HTML file to personalize the default Nginx welcome page and customization by giving it a custom message: ***Hello Group 4 from Kubernetes.***

TASK 4: EXPOSING THE APPLICATION



For this we did the following:

- ❑ Process:

  We created file and named it ***Service.yml*** to make our app accessible outside the cluster using a NodePort

- ❑ Purpose:

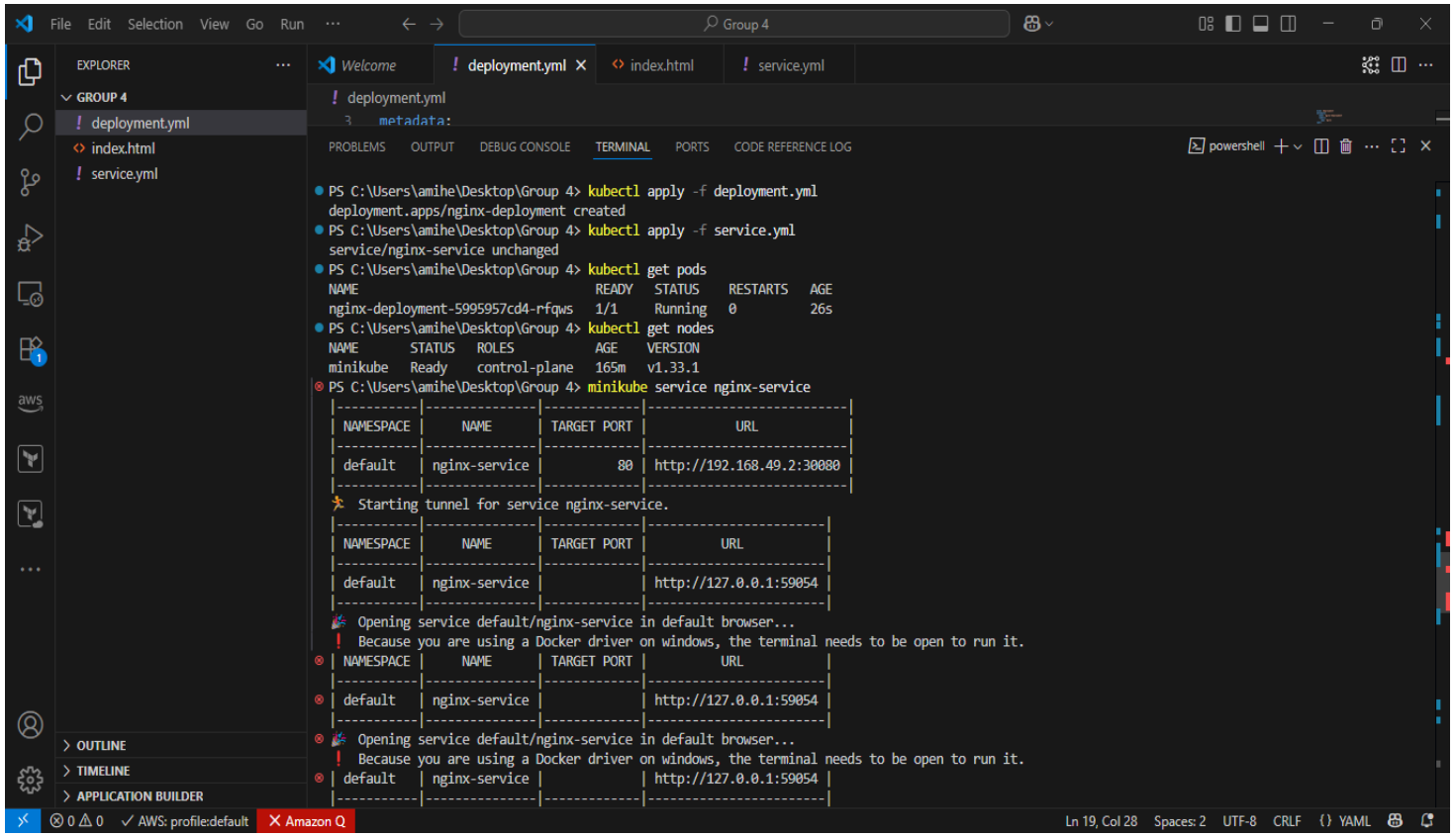
  Kubernetes apps are not accessible by default. You expose them so users can reach them.

TASK 4 CONT.

```
PS C:\Users\amihe\Desktop\Group 4> kubectl apply -f deployment.yml
deployment.apps/nginx-deployment created
PS C:\Users\amihe\Desktop\Group 4> kubectl apply -f service.yml
service/nginx-service unchanged
PS C:\Users\amihe\Desktop\Group 4> kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5995957cd4-pzb9w   1/1     Running   0          23s
```

- Applied the deployment using the Kubernetes manifest file: ***deployment.yml*** with the command ***kubectl apply –f deployment.yml***
- Applied the service using the Kubernetes manifest file: ***service.yml***with the command ***kubectl apply –f service.yml***. The reason for the service unchanged is that we had run the command earlier.
- Verified pod status and confirmed successful deployment with ***kubectl get pods.***
- Created 1 replica of the custom Nginx container serving a styled HTML page.

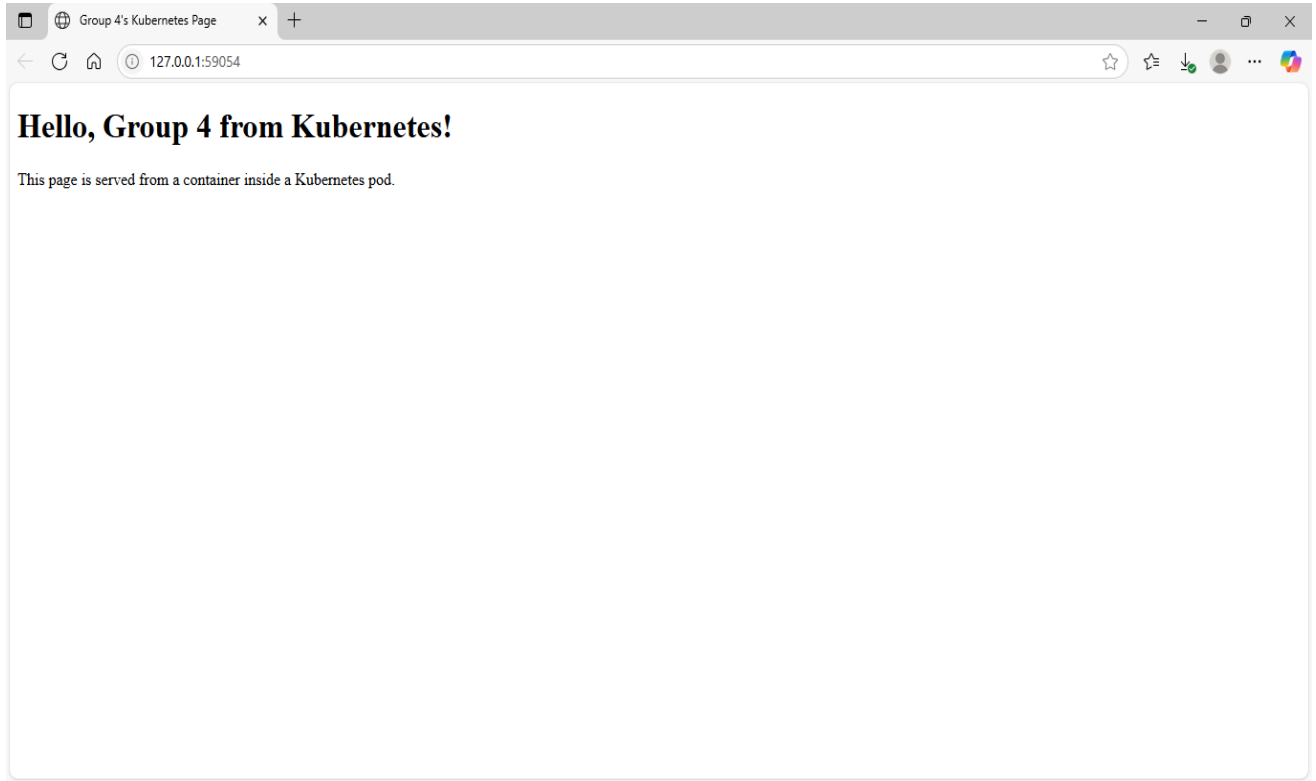
TASK 5: DEPLOY AND TEST



- For deploying and testing , we used the command ***minikube service nginx-service*** to test our app in the browser.

- **Purpose:**
  This step confirms our app is working and accessible.

TASK 5 CONT.: OUTCOME OF DEPLOYMENT



- After executing ***minikube service nginx-service***, Minikube automatically launched the associated web service URL in our default browser, rendering our custom landing page as expected.

TASK 6 : SCALING AND UPDATE EXERCISE

```
PS C:\Users\amihe\Desktop\Group 4> kubectl scale deployment/nginx-deployment --replicas=5
deployment.apps/nginx-deployment scaled
deployment.apps/nginx-deployment scaled
PS C:\Users\amihe\Desktop\Group 4> kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
nginx-deployment-5995957cd4-27q66   0/1     ContainerCreating   0          12s
nginx-deployment-5995957cd4-cxf8p   1/1     Running             0          12s
nginx-deployment-5995957cd4-lcc6q   1/1     Running             0          12s
nginx-deployment-5995957cd4-pbk2k   1/1     Running             0          12s
nginx-deployment-5995957cd4-rfqws   1/1     Running             0          9m42s
```

- For scaling we used the command:
  kubectl scale deployment/nginx-deployment --replicas=5

- Purpose:
  Adjusts the number of running pod replicas for the nginx-deployment

- Outcome:
  The application now runs 5 instances, enhancing availability and load distribution.

TASK 6 CONT.:EFFECT OF SCALING ON THE APPLICATION

- **Improved availability:** More pods = higher uptime.
- **Load balancing:** Traffic is distributed across all replicas.
- **Faster response times:** Increased capacity to handle requests.
- **Resource usage:** Higher resource consumption depending on the replica count.

TASK 6 CONT.: SCALING AND UPDATE EXERCISE

```
PS C:\Users\amihe\Desktop\Group 4> kubectl set image deployment/nginx-deployment nginx=nginx:1.23
nginx-deployment-5995957cd4-27q66   0/1     ContainerCreating   0          12s
nginx-deployment-5995957cd4-cxf8p   1/1     Running             0          12s
nginx-deployment-5995957cd4-lcc6q   1/1     Running             0          12s
nginx-deployment-5995957cd4-pbk2k   1/1     Running             0          12s
nginx-deployment-5995957cd4-rfqws   1/1     Running             0          9m42s
nginx-deployment-5995957cd4-27q66   0/1     ContainerCreating   0          12s
nginx-deployment-5995957cd4-cxf8p   1/1     Running             0          12s
nginx-deployment-5995957cd4-lcc6q   1/1     Running             0          12s
nginx-deployment-5995957cd4-pbk2k   1/1     Running             0          12s
nginx-deployment-5995957cd4-27q66   0/1     ContainerCreating   0          12s
nginx-deployment-5995957cd4-cxf8p   1/1     Running             0          12s
nginx-deployment-5995957cd4-27q66   0/1     ContainerCreating   0          12s
nginx-deployment-5995957cd4-27q66   0/1     ContainerCreating   0          12s
nginx-deployment-5995957cd4-cxf8p   1/1     Running             0          12s
nginx-deployment-5995957cd4-lcc6q   1/1     Running             0          12s
nginx-deployment-5995957cd4-cxf8p   1/1     Running             0          12s
nginx-deployment-5995957cd4-lcc6q   1/1     Running             0          12s
nginx-deployment-5995957cd4-lcc6q   1/1     Running             0          12s
nginx-deployment-5995957cd4-pbk2k   1/1     Running             0          12s
nginx-deployment-5995957cd4-rfqws   1/1     Running             0          9m42s
```

For rolling update, we did the following:

- ❑ Updated Kubernetes Deployment without downtime using ***kubectl set image.***
- ❑ Swapped container image tag to the latest stable version for improved performance.
- ❑ Pods replaced incrementally, ensuring uninterrupted service throughout the rollout.
- ❑ Verified rollout status using ***kubectl rollout*** and confirmed success

ROLLING OUT UPDATE (OPTIONAL)

```
PS C:\Users\amihe\Desktop\Group 4> kubectl rollout status deployment/nginx-deployment
deployment "nginx-deployment" successfully rolled out
```

- Monitored the rollout of the updated deployment to ensure all pods transitioned successfully.
- Confirmed that the new container image was applied without interruption to service availability.
- Used ***kubectl rollout status*** to track progress and validate a smooth rolling update completion.

ISSUES AND TROUBLESHOOTING

- Issue: Pod not creating properly.
- Fix: We restarted Minikube and re-applied all YAML configurations.

CONCLUSION

In conclusion, we were able to do the following :

- Successfully deployed and exposed application.
- Understood scaling and rolling updates.
- Practiced writing and applying YAML configs.

THANK YOU