DevOps challenge

To implement python web application, I have used Flask frame work, which is simple, light weight and provides useful tools and features for creating web application in python.

Python Version: Python 3.11.6

Editor: VS code

Followed below best practices

- Designed endpoints with proper names
- Used comments in the code when needed
- Code is stable, not an error prone and as I am using flask, performance also good.
- Containerized the application
- Well tested before running the application

Continuous Integration(CI)

- Continuous Integration is the DevOps best practice. Integrating code changes into main branch, test and build the changes when developer commit the code .
- Wrote the below tests for the web application, which ensure quality of the code
- **https://github.com/chandrasehkhar/devops_home_test/tree/main/tests**
- Containerized the application using Docker. It has all the dependencies that an application might need to run on any host. below is the Dockerfile
- **https://github.com/chandrasehkhar/devops_home_test/blob/main/Dockerfile**
- Image scanning is an important security step in containerized applications, we can detect vulnerabilities. For that used **trivy image scanning tool.**
- Trivy is opensource vulnerability scanner with complete database of security vulnerabilities and super easy to use.
- https://github.com/chandrasehkhar/devops_home_test/blob/main/trivy-image-scan.sh
- To automate the process of build, test and scanning of the image used jenkins tool. Which is an open source and efficient tool.
- https://github.com/chandrasehkhar/devops_home_test/blob/main/Jenkinsfile
- Setup the webhooks to trigger the job automatically , when the code merged into master.

Kubernetes Deployment

- Kubernetes is an opensource orchestration tool, which is used for automating deployment, scaling and management of containerized applications.
- Our Application(container) is running in the pods, which we exposed using service
- Here, I am using deployment which is useful to scale in and scale out of replica pods. Always runs desired number of pods. We can rollout to previous versions easily.

- I have deployed the application with 2 pods currently, we can increase pods using kubectl command or by modifying in deployment.
- I have created a service, which exposes my application to outside the cluster and listens to user input.
- It is highly available solution, if one node goes down in the cluster, Kubernetes moves our application to other node for availability of application.

https://github.com/chandrasehkhar/devops_home_test/blob/main/deployment.yaml

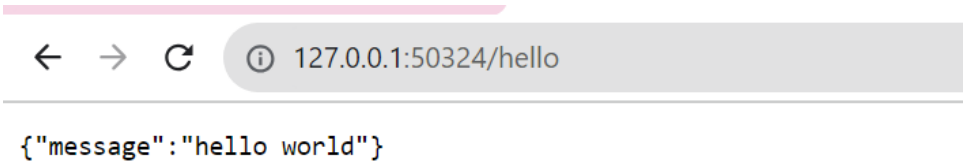
We can deploy our application on kubernetes cluster using below command. And check pods are running or not.

kubectl apply -f deployment.yaml

```
✋ Stopping tunnel for service flask-app-service.
PS C:\Users\lavan\devops_home_test> kubectl get pods
NAME                                    READY   STATUS    RESTARTS   AGE
flask-app-deployment-c7454bd58-pwhtn    1/1     Running   0          2m53s
flask-app-deployment-c7454bd58-sh6t4    1/1     Running   0          2m53s
PS C:\Users\lavan\devops_home_test>
```

```
PS C:\Users\lavan\devops_home_test> kubectl get svc
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
flask-app-service   LoadBalancer   10.105.140.64   <pending>     6000:30002/TCP   34h
kubernetes          ClusterIP      10.96.0.1       <none>        443/TCP          2d4h
PS C:\Users\lavan\devops_home_test>
```

Access the application from the browser with /hello and /health endpoints



{"message":"hello world"}

127.0.0.1:50324/health

{"health":"Success"}

127.0.0.1:50575/health

{"health":"unhealthy"}