Task To Deploy an Openmrs and Game-of-life webapps:

Procedure :

1. Create an images of openmrs and game-of-life using Docker.
2. After creating the images push the images to your docker hub repository.
3. Now, write a manifest to deploy the images in Kubernetes cluster.
4. After the deployment of the manifest file check the Kubernetes service and find the External IP from the service and past the link and see wether the service is deployed or not .

Pre-installation :

Before you start the procedure your system should contain the following software pre-installed:

1. Aws cli installed and configured.
2. Terraform installed and configured.
3. Kubectl installed and configured.
4. Docker installed and you have to login to your docker hub account.

Building docker images:

openmrs image:

Steps:

1. Write a Dockerfile .
2. Take an ubuntu image from the docker hub.
3. Install the git,java-11,maven,and tar gzip softwares in the ubuntu image.
4. The git is to clone the repository of the openmrs files, maven is to build a package of the openmrs and tar gzip is to unzip the tomcat tar file.
5. Pass the command to build the package 'mvn clean package' after the package is build the package is stored in the webapp/target/openmrs.war file.
6. After copy the openmrs.war file to the /opt/tomcat/apache-tomcat-8.5.84/webapp
7. Now, pass the command ["catalina.sh","run"].
8. After writing the Dockefile now run the command "docker image build -t <image-name> :<tag>.".
9. After the image is built run a container to check whether the image is working the 'tag' the image and push it to your docker hub repository.

game-of-life:

Steps:

1. Write a Dockerfile.
2. Take an ubuntu image from the docker hub.
3. Install the git,java-11,maven,and tar gzip softwares in the ubuntu image.
4. The git is to clone the repository of the game-of-life files, maven is to build a package of the gameof-life and tar gzip is to unzip the tomcat tar file.

5. Pass the command to build the package ‘mvn clean package’ after the package is build the package is stored in the /gameoflife-web/target/gameoflife.war file.
6. After copy the gameoflife.war file to the /opt/tomcat/apache-tomcat-8.5.84/webapp
7. Now, pass the command [“catalina.sh”,”run”].
8. After writing the Dockefile now run the command “docker image build -t <image-name> :<tag>.”.
9. After the image is built run a container to check whether the image is working the ‘tag’ the image and push it to your docker hub repository.

writing the manifest file :

The manifest file consists a deployment, service, ingress yaml files written in the same file.

The yaml file written is :

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: openmrstest
spec:
  minReadySeconds: 10
  replicas: 1
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      name: openmrs
      labels:
       app: web
    spec:
     containers:
       - name: openmrs
         image: vamsibakka/openmrs:2.0
```

```
        ports:
         - containerPort: 8080
           protocol: TCP
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gameoflifetest
spec:
  minReadySeconds: 10
  replicas: 1
  selector:
    matchLabels:
      app1: web2
  template:
    metadata:
      name: gameoflife
      labels:
       app1: web2
    spec:
     containers:
       - name: gameoflife
         image: vamsibakka/gameoflife:1.0
         ports:
          - containerPort: 8080
            protocol: TCP
---
apiVersion: v1
kind: Service
```

```
metadata:
  name: service-openmrs
spec:
  type: LoadBalancer
  selector:
    app: web
  ports:
    - port: 35000
      targetPort: 8080
      #nodePort: 30000
      protocol: TCP
---
apiVersion: v1
kind: Service
metadata:
  name: service-gameoflife
spec:
  type: LoadBalancer
  selector:
    app1: web2
  ports:
    - port: 35001
      targetPort: 8080
      #nodePort: 30000
      protocol: TCP
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
```

```
  name: ingress-test
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/listen-ports: '[{"HTTPS":443}, {"HTTP":80}]'
spec:
  rules:
  - http:
    paths:
    - path: /openmrs
      pathType: Prefix
      backend:
        service:
          name: service-openmrs
          port:
            number: 8080
  - http:
    paths:
    - path: /gameoflife
      pathType: Prefix
      backend:
        service:
          name: service-gameoflife
          port:
            number: 8080
```

---

output :

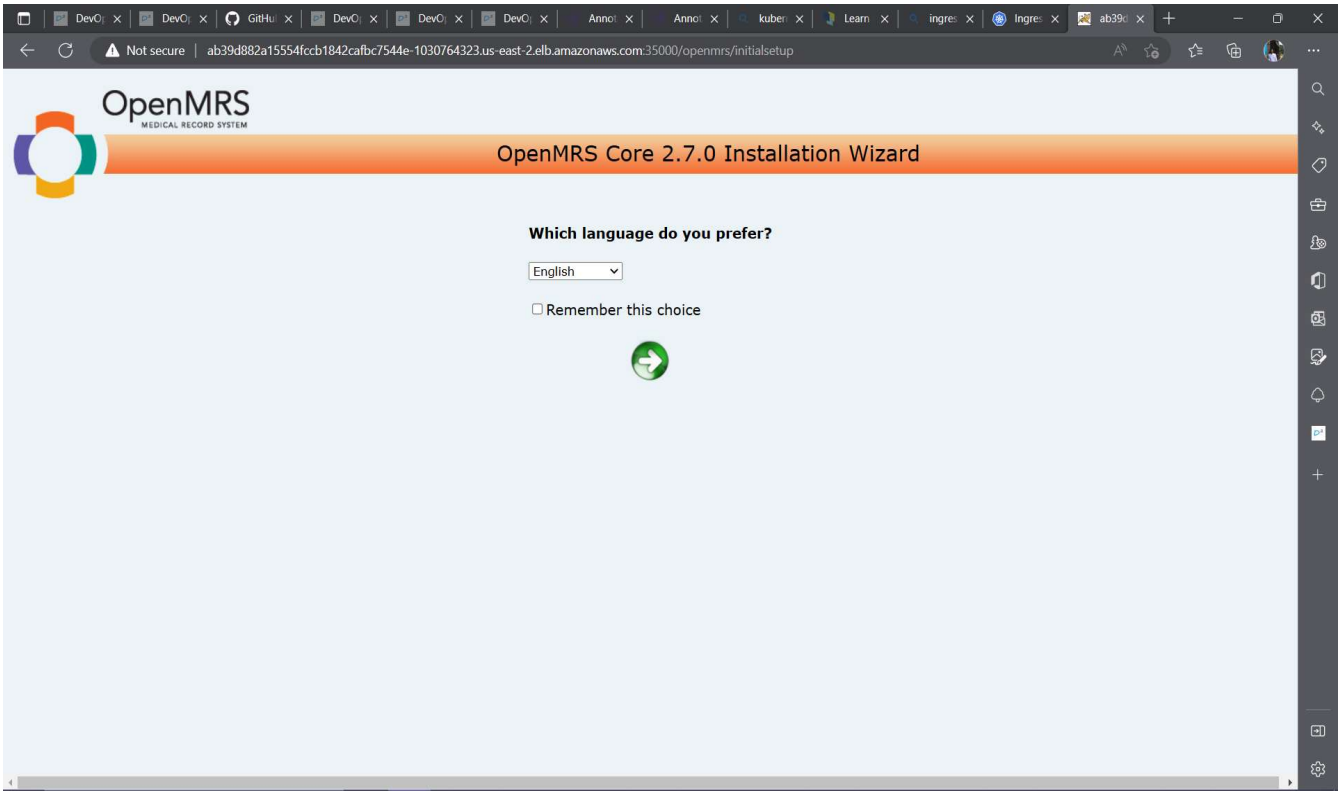

Fig: openmrs output

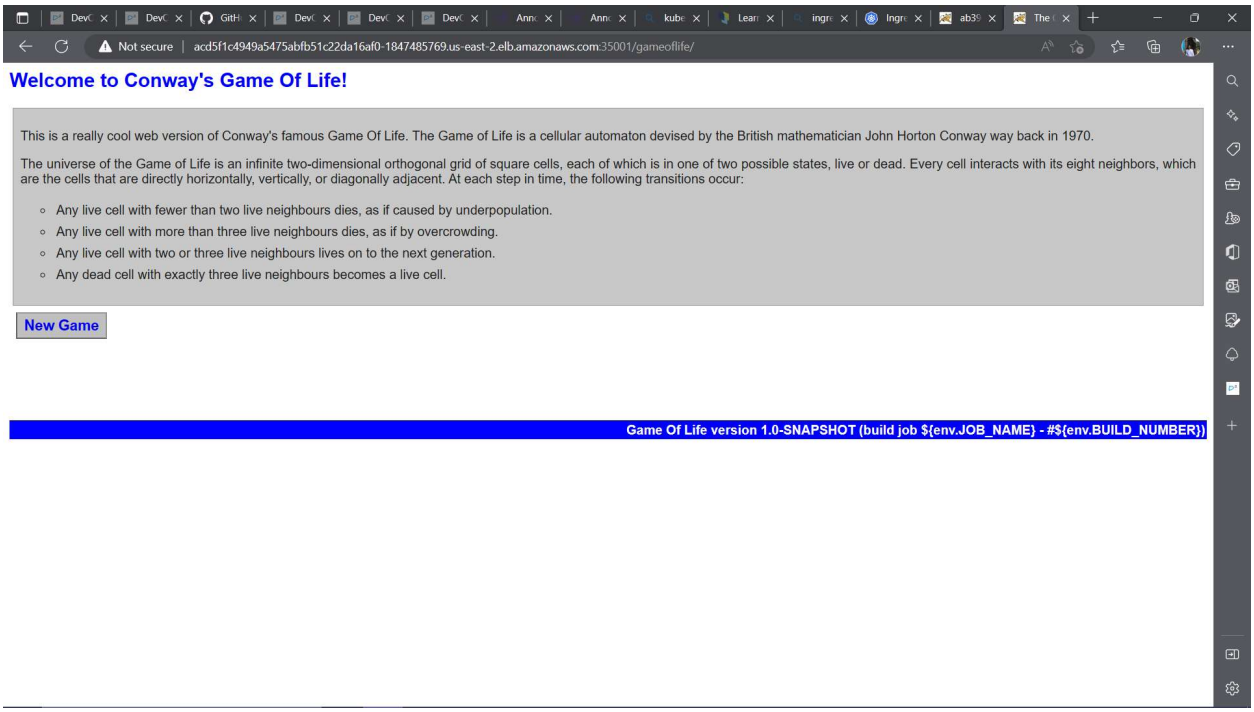

Fig : Game-of-life output