Kubernetes Assignment -1

: - Create a simple web application (e.g., a "Hello, World!" application) and Dockerize it.

1: -Create web application hello-world

- package com.nagarro.controller;
-
- import org.springframework.web.bind.annotation.GetMapping;
- import org.springframework.web.bind.annotation.RestController;
-
- @RestController
- public class HelloController {
-
- @GetMapping("/hello")
- public String hello() {
- return "Hello, World!";
- }
  }

2: Create docker file.

```
# Use OpenJDK image to run the application
FROM openjdk:17
WORKDIR /app
COPY target/hello-world-0.0.1-SNAPSHOT.jar app.jar
ENTRYPOINT ["java", "-jar", "app.jar"]
```

3: Build the Docker Image:

-> docker build -t hello-world-app .

```
anil@IN-PG02P670:~/java-project/hello-world/hello-world$ docker build -t anil647/hello-world-app .
[+] Building 7.2s (8/8) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                        0.0s
 => => transferring dockerfile: 204B                                                                        0.0s
 => [internal] load metadata for docker.io/library/openjdk:17                                               6.9s
 => [internal] load .dockerignore                                                                           0.0s
```

4: Verify the Docker Image:

-> docker images

```
anil@IN-PG02P670:~/java-project/hello-world/hello-world$ docker images
REPOSITORY        TAG       IMAGE ID       CREATED        SIZE
hello-world-app   latest    e6607149431c   3 hours ago    491MB
kicbase/stable    v0.0.45   aeed0e1d4642   2 weeks ago    1.28GB
```

This command `docker push hello-world-app:01` uploads the Docker image `hello-world-app` with the tag `01` to a Docker registry .

Key Points:

: -Deploy the application to a Kubernetes cluster using kubectl.

Deploy the Application to a Kubernetes Cluster

1: Start minikube using this command

-> minikube start

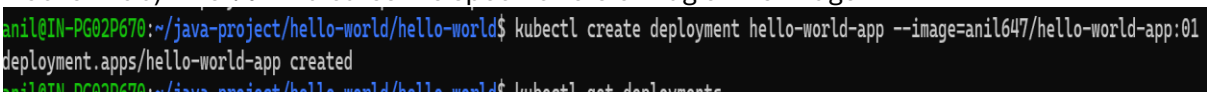
2: Check minikube status using this command:

-> minikube status

```
anil@IN-PG02P670:~/java-project/hello-world/hello-world$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

3: The command is used to create a deployment in a Kubernetes cluster:

- ➔ `kubectl create deployment hello-world-app --image=anil647/hello-world-app:01`
- ➔ The image `anil647/hello-world-app:01` is pulled from a container registry (e.g., Docker Hub). The `:01` indicates the specific version/tag of the image.



4: check list all the pods running in the current Kubernetes cluster:

-> kubectl get pods

```
anil@IN-PG02P670:~/java-project/hello-world/hello-world$ kubectl get pods
NAME                               READY   STATUS              RESTARTS   AGE
hello-world-app-6ffdc8bd6c-gq6v8   0/1     ContainerCreating   0          103s
```

5: The command is used to create a Kubernetes Service that exposes the hello-world-app deployment to external traffic

> `kubectl expose deployment hello-world-app --type=LoadBalancer --port=8085`

6: This command is used to lists all the services in the current namespace of the Kubernetes cluster. A service in Kubernetes provides a stable IP address and DNS name for a set of pods, enabling internal and external communication.

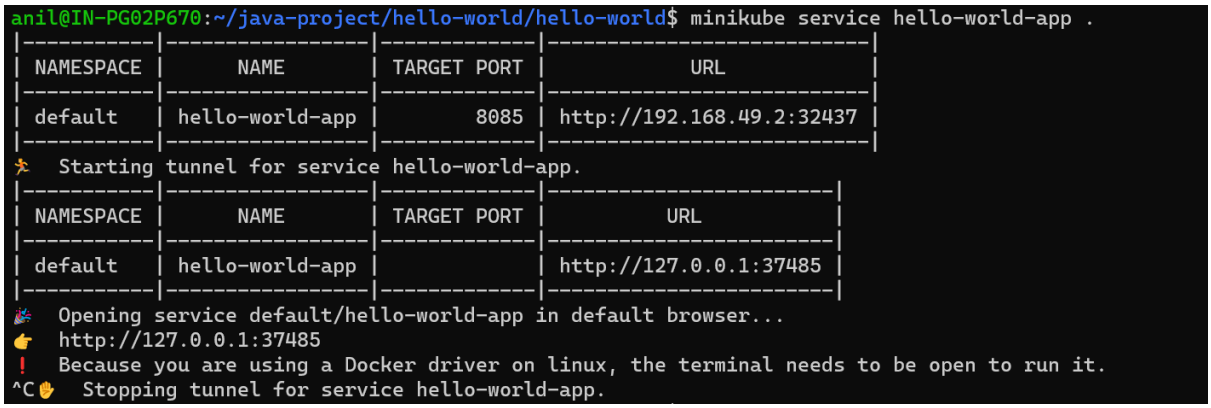
`->kubectl get services`

```
anil@IN-PG02P670:~/java-project/hello-world/hello-world$ kubectl get services
NAME              TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hello-world-app   LoadBalancer   10.106.252.17   <pending>     8085:32437/TCP   10s
kubernetes        ClusterIP      10.96.0.1       <none>        443/TCP          23h
```

-: Expose the application using a Kubernetes Service to access it externally.

7: This command is used in a Minikube environment to open a Kubernetes service in the default web browser or retrieve its details.

-> `minikube service hello-world-app`



-: Scale the application by increasing the number of replicas.

8: This command is used to adjust the number of replicas (or pods) for the `hello-world-app` deployment.

-> kubectl `scale deployment hello-world-app --replicas=4`