[**1. SECTION – 1 : What is Kubernetes** 1](#_Toc172561679)

[**2. SECTION – 2 : Kubernetes Setup** 3](#_Toc172561680)

[**3. SECTION – 3 : Create Kubernetes Pods** 5](#_Toc172561681)

[**4. SECTION – 4 : Create Kubernetes Service** 6](#_Toc172561682)

[**5. SECTION – 5 : Create Kubernetes Namespace** 8](#_Toc172561683)

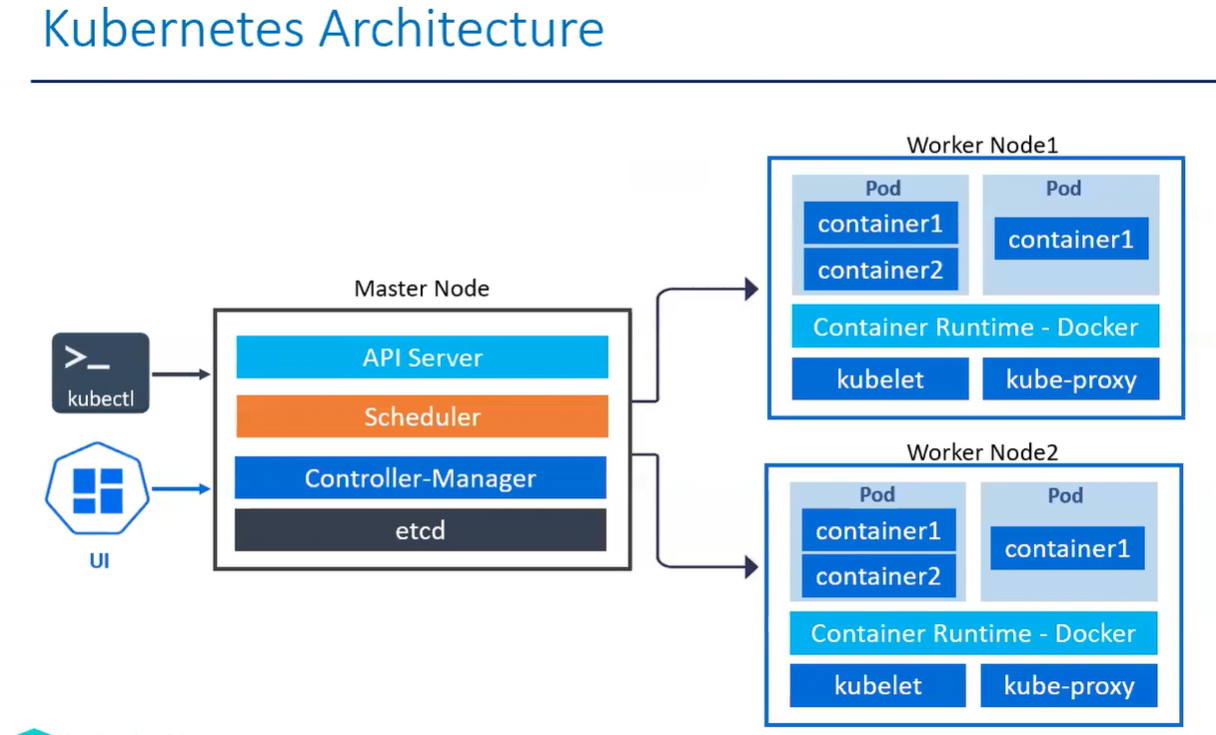
[**6. SECTION – 6 : Create Kubernetes Deployment** 11](#_Toc172561684)

[**7. SECTION – 7 : Create Docker Webapp and SQL Server for Kubernetes** 12](#_Toc172561685)

[**8. SECTION – 8 : Create Replica of Pods and Container (Self Healing and Load Balancing)** 17](#_Toc172561686)

# **1. SECTION – 1 : What is Kubernetes**

1. Kubernetes is an open-source container management (Orchestration) tool.



**Master Node** => Control the application deployment.

**Pod** => Pod is a logical unit, in there we run a container. Pod contain one or multiple running containers.

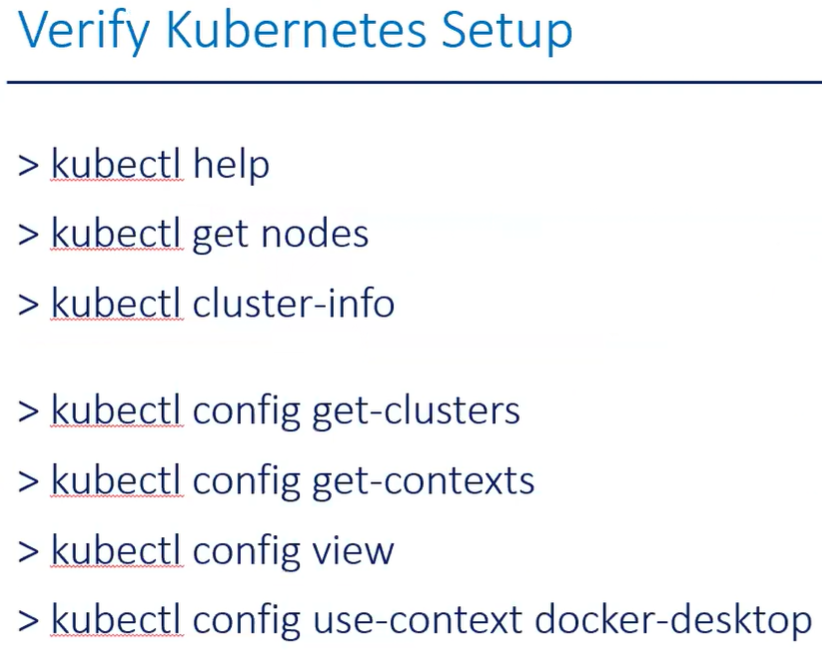
1. Enable the Kubernetes by using Docker Desktop. Open Docker Desktop 🡪 Settings 🡪 Select Kubernetes option from left side pane. 🡪 Enable Kubernetes 🡪 Apply & restart.

It install necessary images for Kubernetes.

A screenshot of a computer

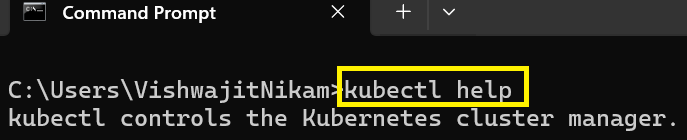
Description automatically generated

# **2. SECTION – 2 : Kubernetes Setup**



Right click on status bar where showing Kubernetes Context (docker-desktop). (Just form checking and confirmation)

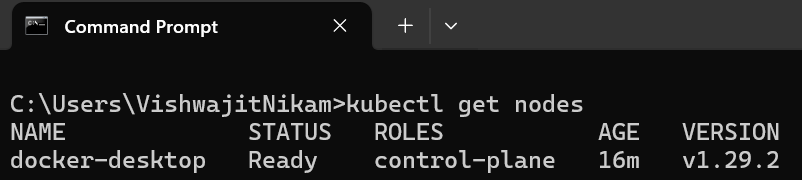
Open Command Prompt and type the following command.



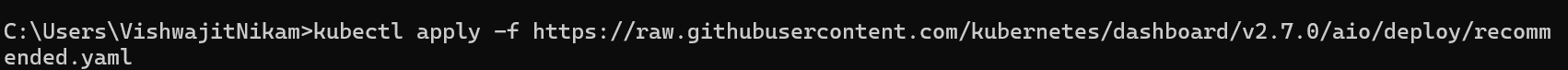
Below command gives the Kubernetes cluster information.

A screenshot of a computer

Description automatically generated



1. Setup Dashboard.
   1. Run the below command in command prompt (This command specify in pdf of scholar hat)



* 1. Run the K8s proxy command

A black background with white text

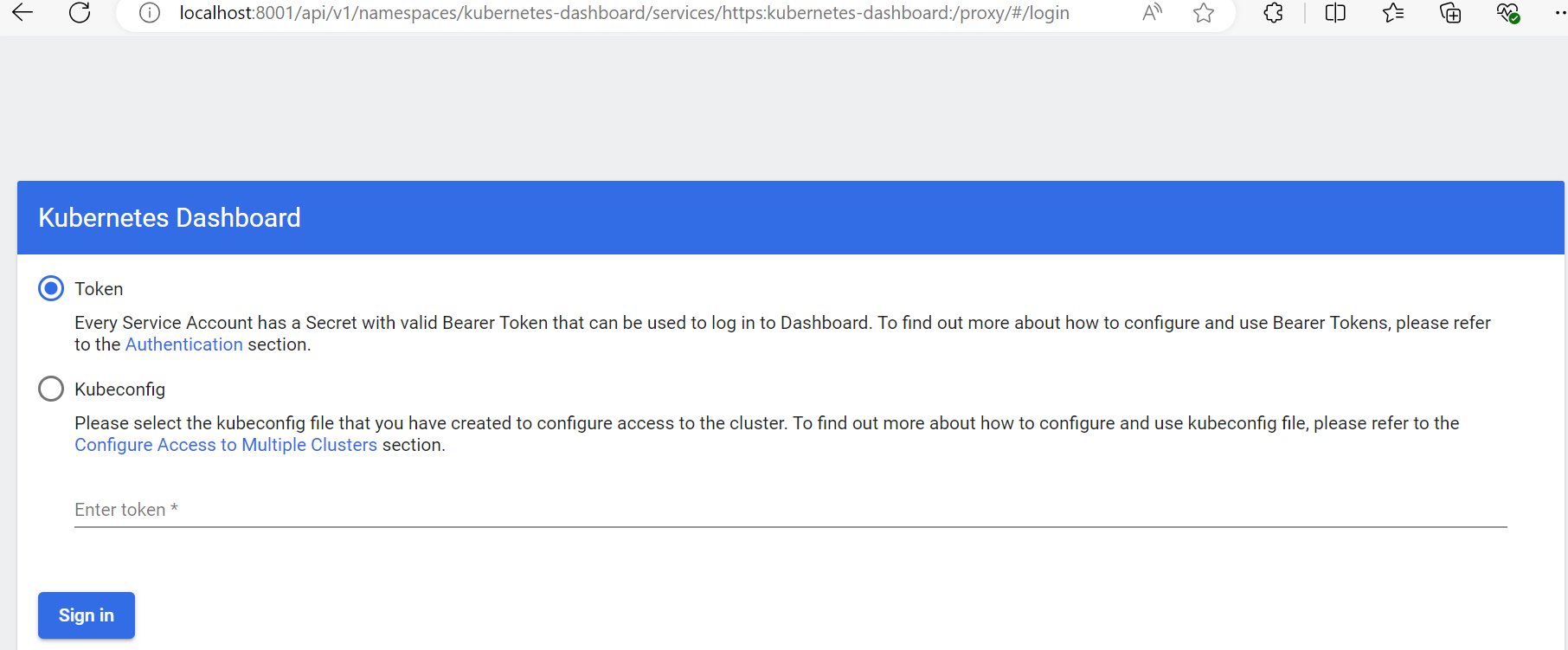
Description automatically generated

A screen shot of a computer code

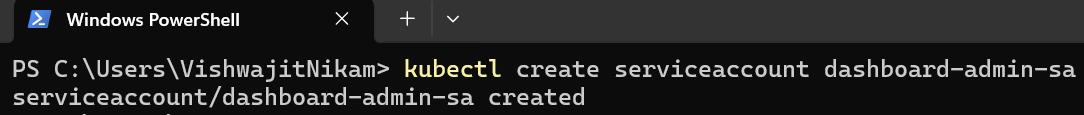
Description automatically generated

Open in the browser the following link





* 1. Create service account by using command prompt.

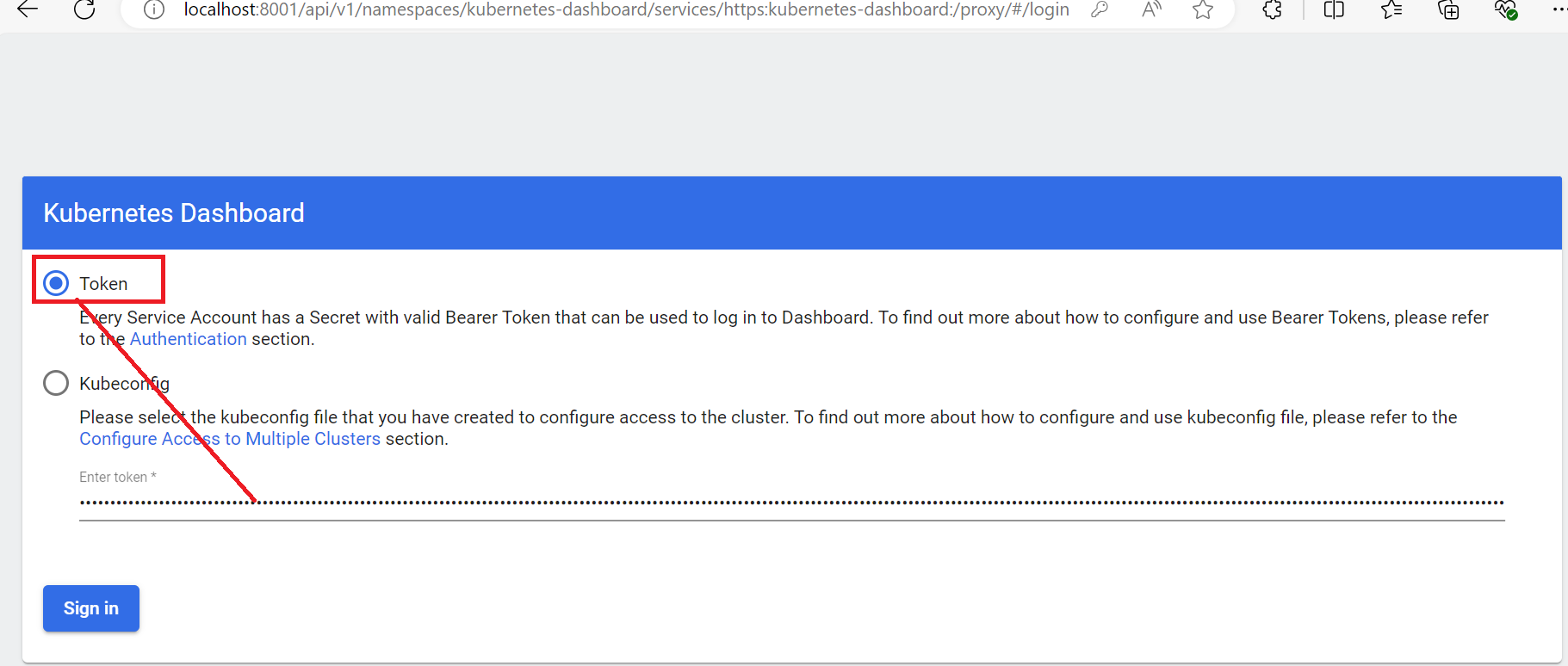




* 1. Generate the token and copy the generated token and paste into url (Screenshot of point 3.2) and sign it

A black screen with white text

Description automatically generated



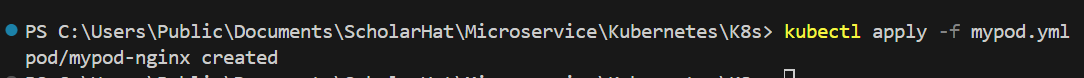
# **3. SECTION – 3 : Create Kubernetes Pods**

1. Run the nginx container using Kubernetes pod.
   1. - Open the Visual Studio Code editor and Create the folder under the file explorer.
   2. Install the Kubernetes snippets and kunernetes Templates extensions. (for yml file configuration purpose)

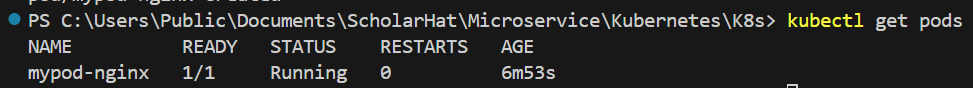
Below command we can manually run the container without using any .yml file



If want to run the pod (run container inside pod) using K8s .yml file then use the below command. (we generally run the container in K8s using .yml file)



To get the pods details using below command



A screenshot of a computer

Description automatically generated

# **4. SECTION – 4 : Create Kubernetes Service**

Pods can not access directly, as we need create the service. Service connect to pods to access the pods that is running inside K8s.

**Note :** Service isolate the pods, deployment**.**

A computer screen shot of a computer

Description automatically generated

10.2222.0.0 => Kubernetes Network

10.244.0.2 => Pod (logical unit. It isolate the container from other pods ) running inside a n/w 10.244.0.2.

30008 => Port to expose the pods through service outside the world. The range start from 30008 onwards, we can give any port number.

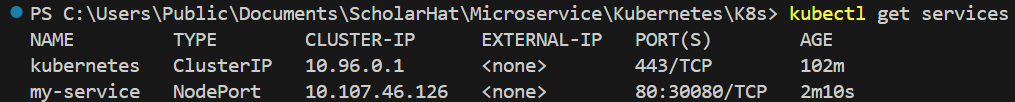
* 1. Create the service file and apply it using below command.



We can check created service in K8s dashboard as well as command also.

A screenshot of a computer

Description automatically generated

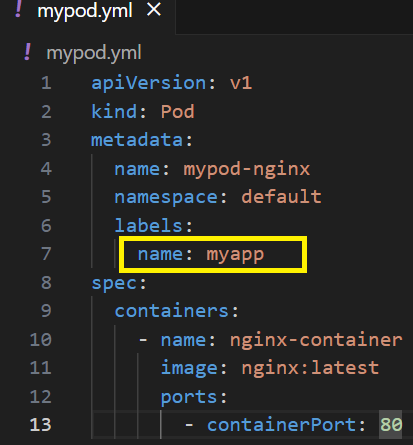
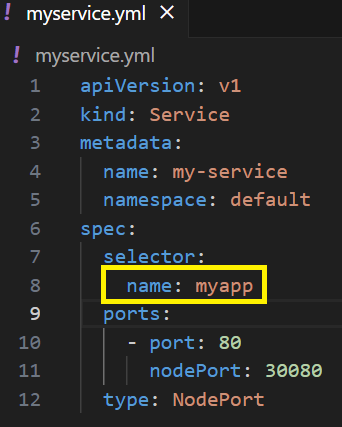


Now we can access the pods through port 30080 as below.

A screenshot of a computer

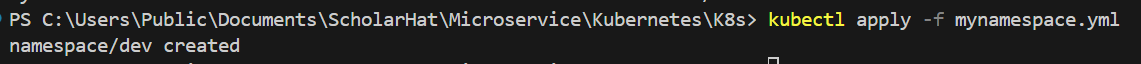
Description automatically generated

**Note : labels name from the mypod.yml must match with service selector name**.

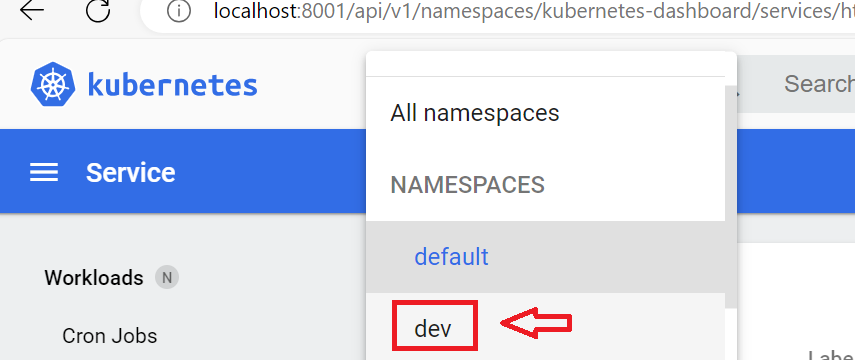
# **5. SECTION – 5 : Create Kubernetes Namespace**

1. Define the namespace.yml file and run the below command in vs code terminal and get the namespace as well



A screen shot of a computer

Description automatically generated



We can delete the namespace by using following command.



A black screen with white text

Description automatically generated

We can delete the service also by using following command.



Hence the port 30080 for nginx is not accessible because we delete the service.

A computer screen shot of a cloud

Description automatically generated

Now delete the pod also



A diagram of a load balancer

Description automatically generated

A computer screen shot of a computer

Description automatically generated

We always use the deployment type, not a separate service and separate port. We create a deployment yml file where we define the information about service and port to run.

If we change the namespace as **dev** in mypod.yml andmyservice.yml file  **(Which map to mynamespace.yml file with name as dev)**

Then the we apply the dev namespace and map that service under the dev namespace, then service will create under dev namespace and pod will run under the same dev namespace.

Order is 1. Apply the namespace first and then apply the service.

A screenshot of a computer screen

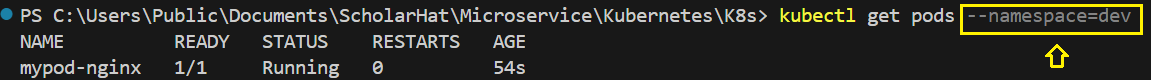
Description automatically generated A screenshot of a computer program

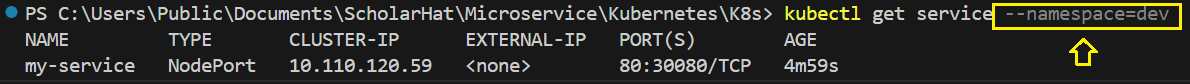
Description automatically generated A screen shot of a computer

Description automatically generated









# **6. SECTION – 6 : Create Kubernetes Deployment**

1. Create app-deployment.yaml file. It contains the combination of mypod.yaml and myservice.yaml file. i.e Clubbed two files content together by separate **---** (Refer all the files from the K8s folder)

apiVersion: v1

kind: Pod

metadata:

  name: myapp-pod

  namespace: dev

  labels:

    app: myapp

spec:

  containers:

    - name: myapp-container

      image: nginx:latest

      resources:

        limits:

          cpu: 200m

          memory: 500Mi

      ports:

        - containerPort: 80

          name: http

---

apiVersion: v1

kind: Service

metadata:

  name: myapp-service

  namespace: dev

spec:

  selector:

    app: myapp

  type: NodePort

  ports:

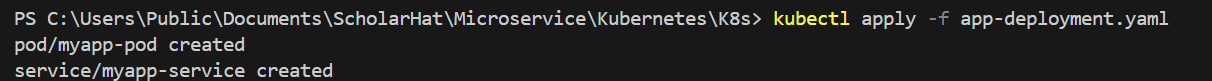
   - port: 80

     nodePort: 30080

When we run the below command for deployment. Then it creates the pod and service under dev namespace.

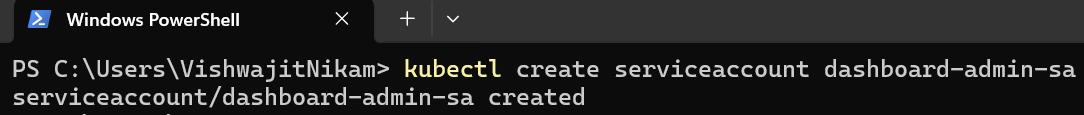
**Note :** When we run the below command then we don’t need to manually apply pod and service.

If we create the deployment for test environment,then just change the namespace for pod,service and deployment file and apply with - - namespace = <environment-name>



# **7. SECTION – 7 : Create Docker Webapp and SQL Server for Kubernetes**

1. Copy the application from Microservice > WebApp-Container into Microservice > Kubernetes and change the internal folder name from WebAppContainer to WebAppContainerK8S.
2. Open the solution in VS Code editor and delete docker compose and docker stack files.
3. Create the new file for asp.net core web application name as **app-deployment.yaml** file and specify the K8s configurations as per the file.
4. Create another new file for sql server name as sql-deployment.yaml file and specify the K8s configuration as per the file.
5. Here in both the files we mentioned the first part as pods configuration and second part as service configuration. Same as we die earlier for K8s deployment. (**Note: In both the files service selector app must match with pods labels. If name mismatch then it will gives error )**
6. Open the VS Code terminal and run the below command if it is not already run



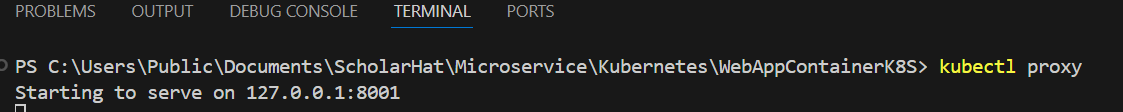


If token expired for the given timestamp (72Hrs) then again generate the token as below for K8s Dashboard.

A black screen with white text

Description automatically generated

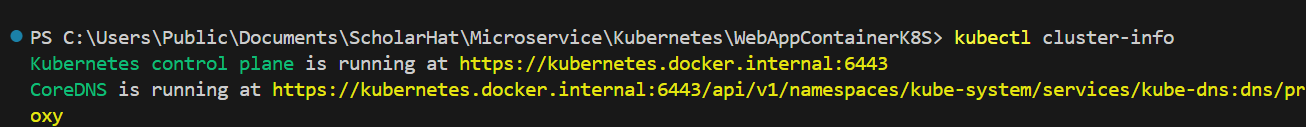
1. Run the below command to open the K8s Dashboard and enter the created token as below in the K8s Dashboard.

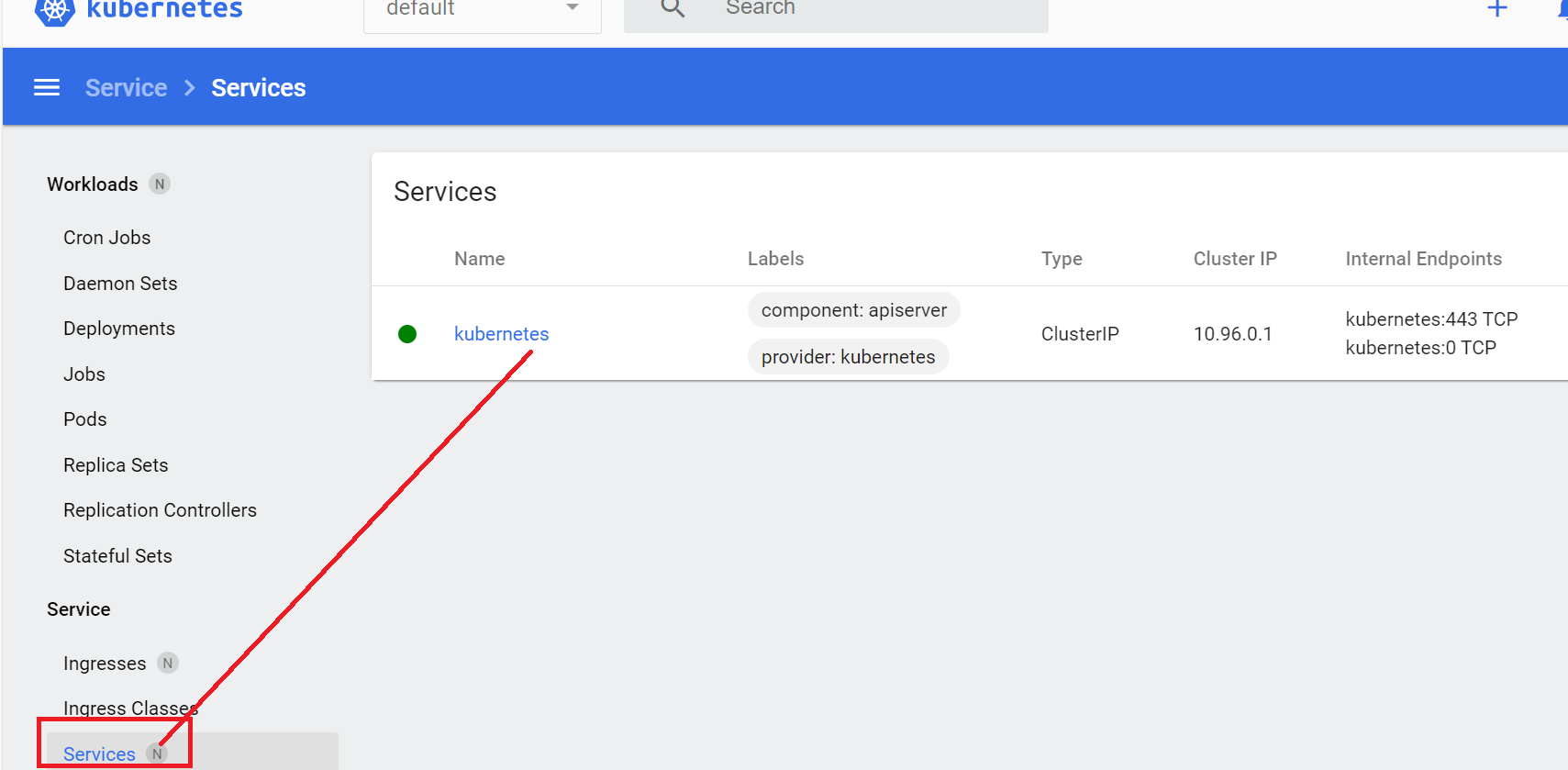


A screenshot of a computer

Description automatically generated

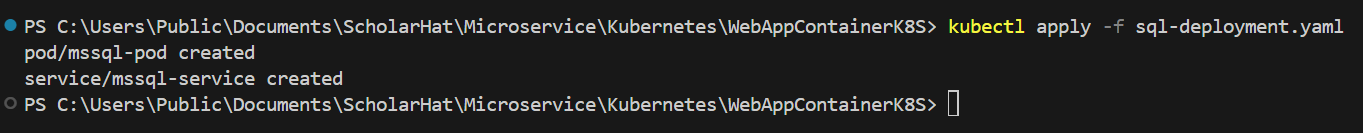
1. Run the below command, it will show the cluster-info

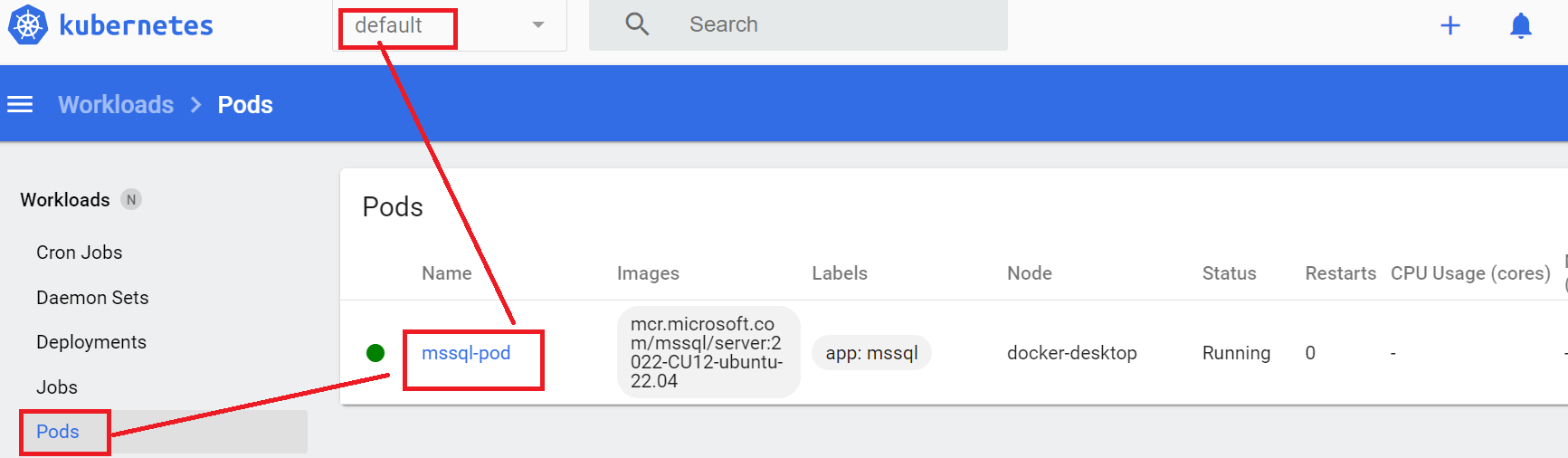


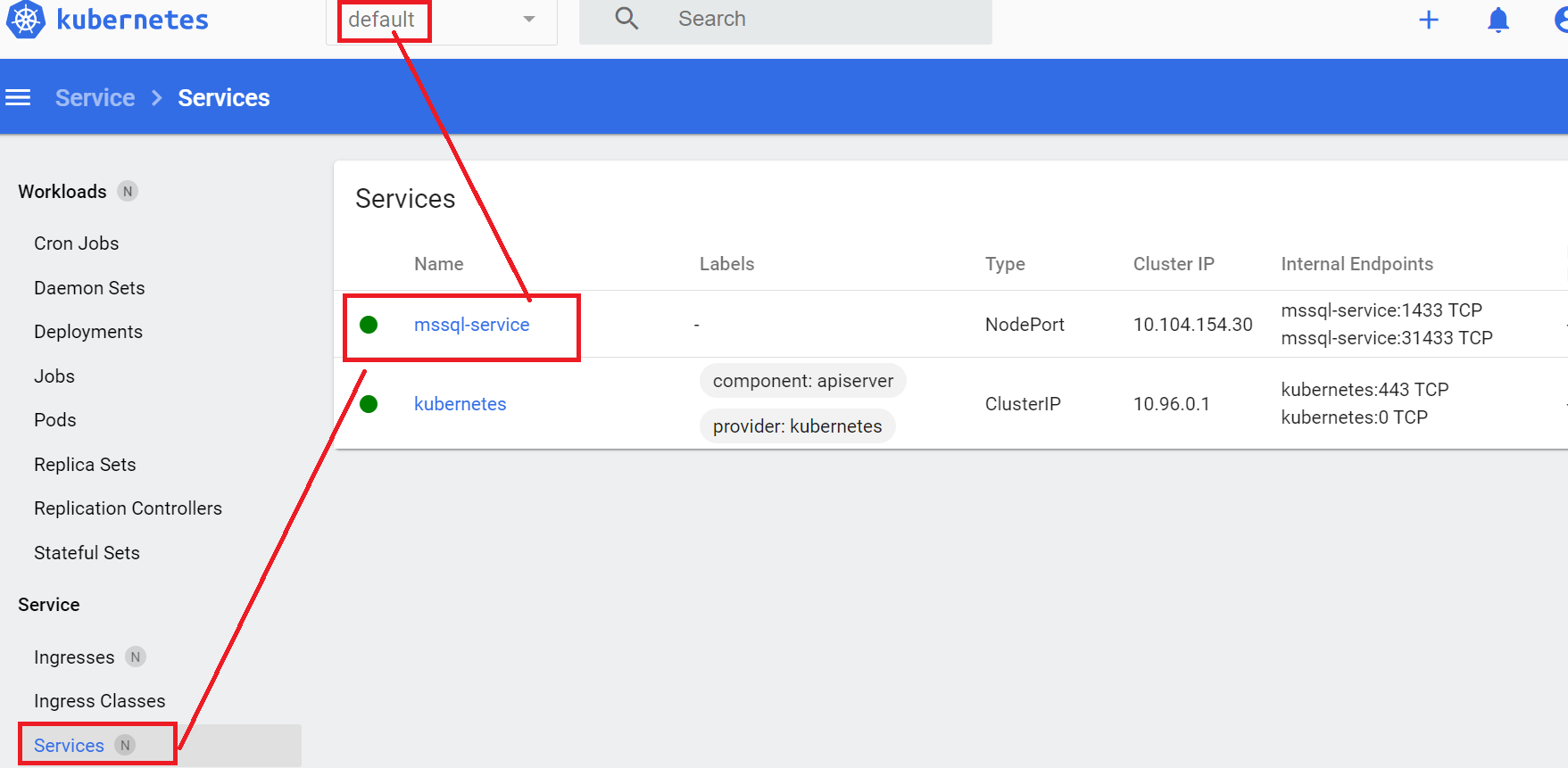


1. Run the following command in VS Code terminal. Its created the Pod and Service under default namespace (as we mentioned **namespace: default** for **sql-deployment.yaml** file).

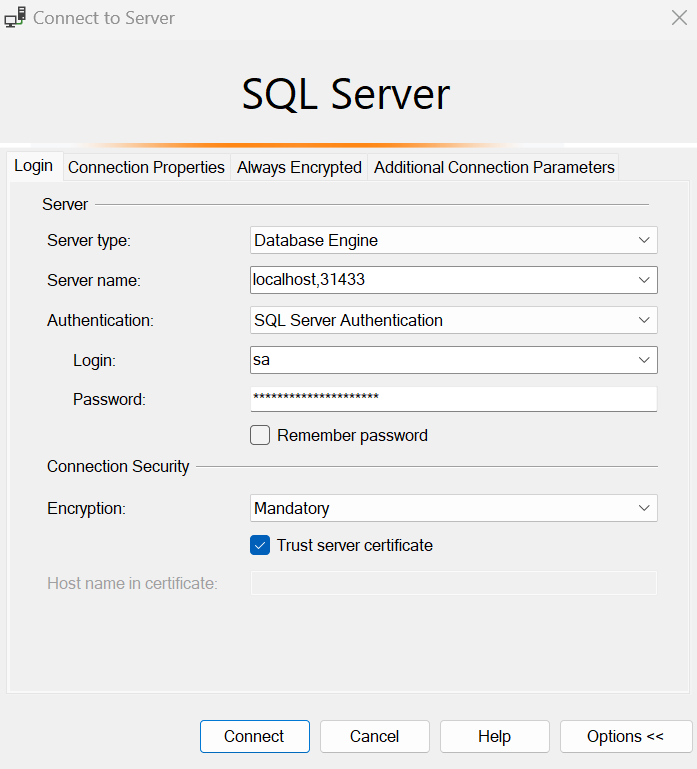
The created pod and service we can see in K8s Dashboard as below







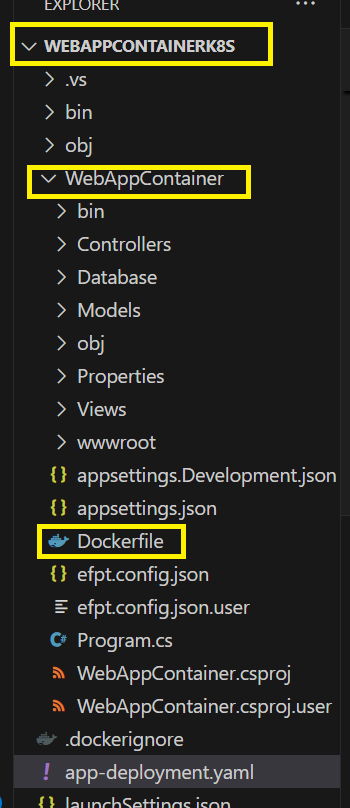
1. Open the SSMS and connect with SQL Server **nodePort:31433** which mentioned in sql-deployment.yaml file.
2. Login with the credentials – ( user id=**sa**; password=**yourStrong(!)Password** ) and connect.



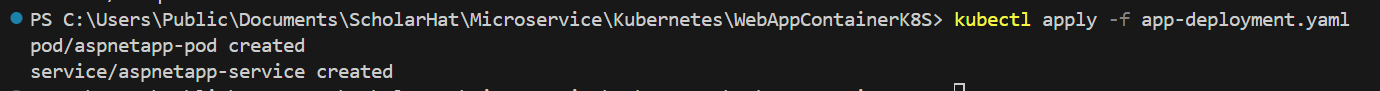
Execute the Sql schema script from the solution. ( WebAppContainer > Database > SQL-Schema > schema.sql )

1. Delete the old container and its image name webappcontainer from docker desktop. ( That Container and its image we created during docker-compose)
2. Create and build the new image **webappcontainer:v1** through VS Code Terminal as below





1. Run the below command in VS Code terminal for the **app-deployment.yaml** file **.**



1. It will create the Pod and service for web applications under default namespace. (as we specify default namespace for both app-deployment.yaml and sql-deployment.yaml file).

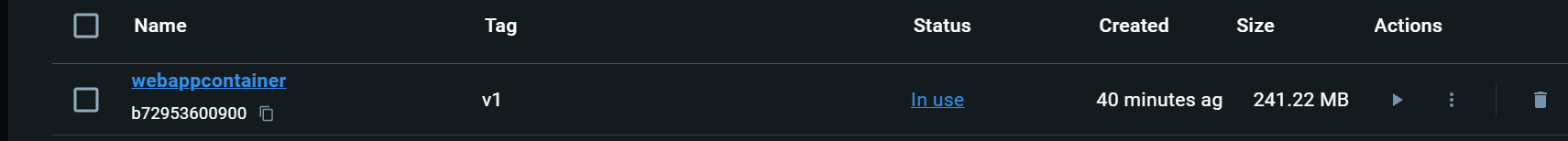
A screenshot of a computer

Description automatically generated

A screenshot of a search box

Description automatically generated

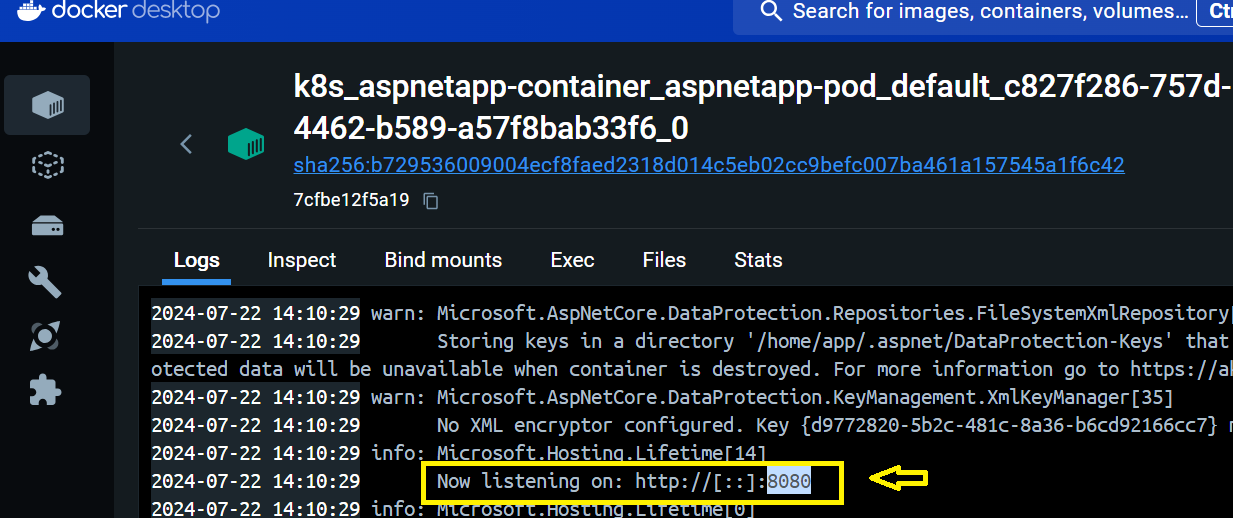
Image in Docker Dektop



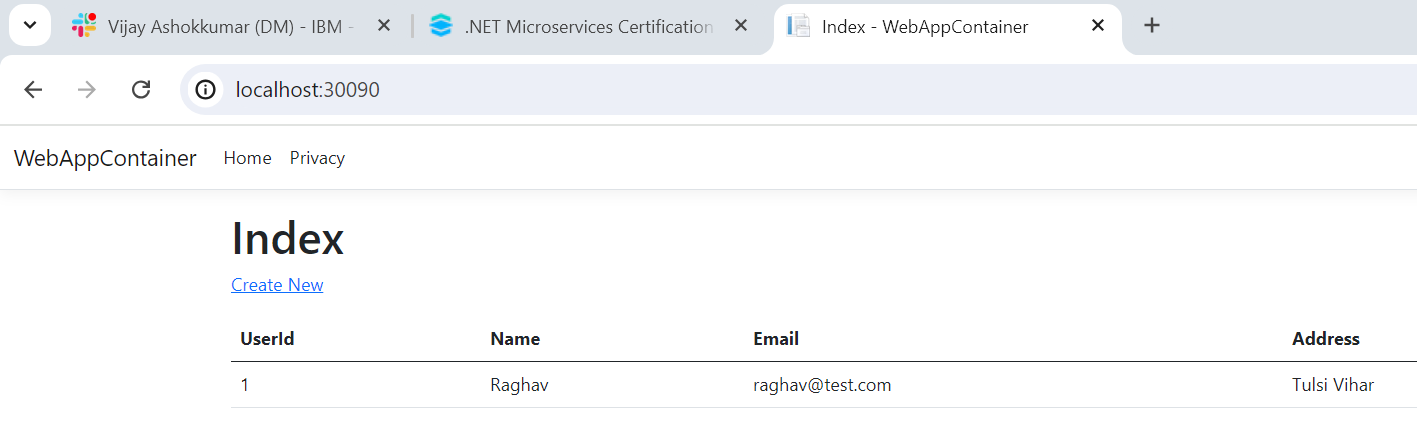
Container running in Docker Desktop for the above image is as below

A screenshot of a computer

Description automatically generated

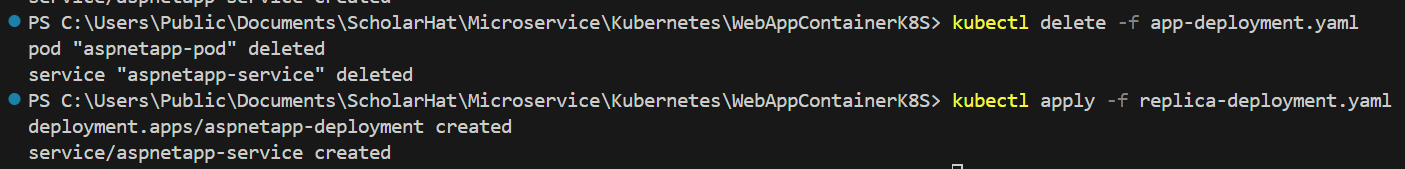


1. Now we able to access the port 30090 for web applications as below. ( WebApp and Sql Server we now running as a Kubernetes)



# **8. SECTION – 8 : Create Replica of Pods and Container (Self Healing and Load Balancing)**

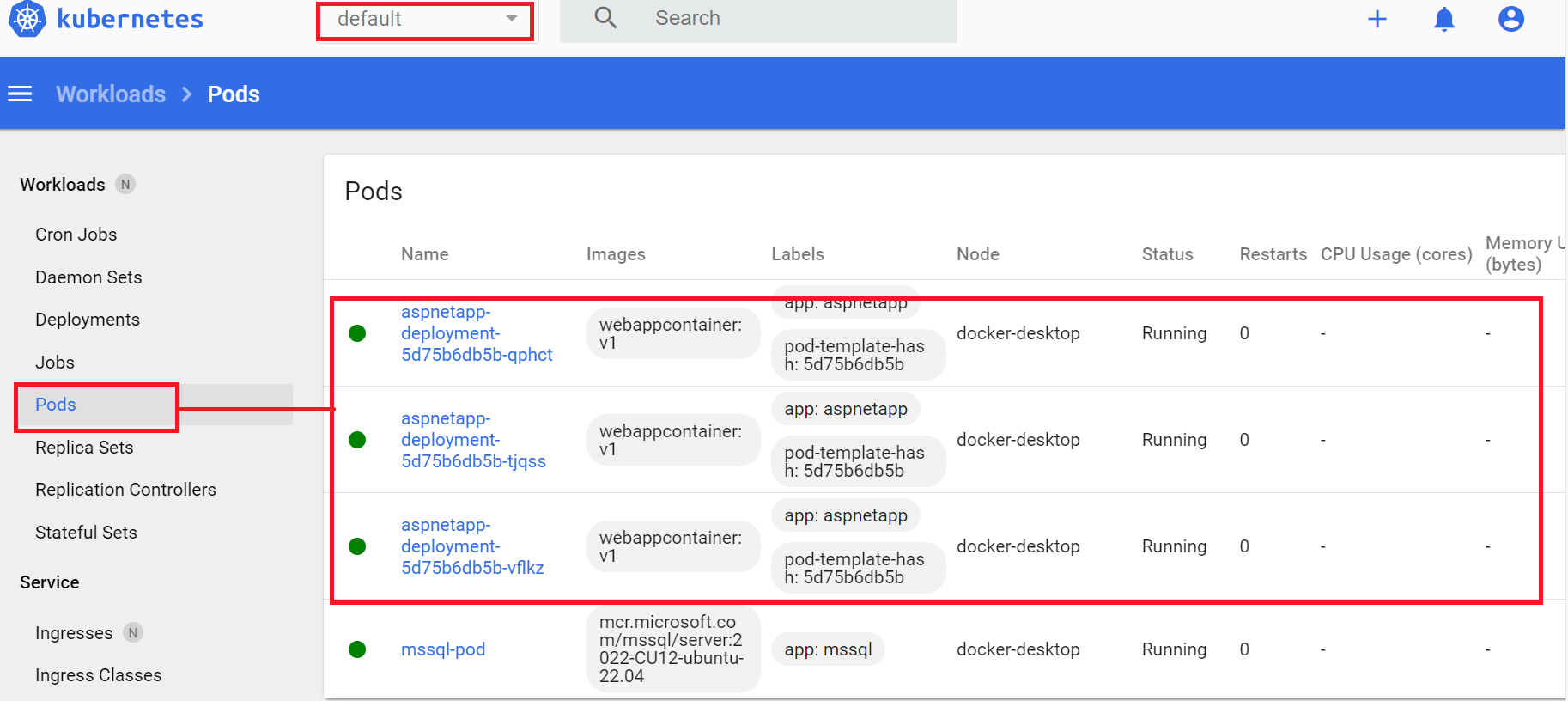
1. Self healing means Kubernetes maintain same number of pods. Eg.if 5 pods are running and one of them are fails due to some reason then it automatically create the pod for fail one and maintain the exact number of pods.
2. Create a file name **replica-deployment.yaml** for Web application.
3. Delete app-deployment.yaml file and apply replica-deployment from VS Code terminal through K8s command.



1. replica-deployment.yaml file create 3 Pods in Docker Desktop and K8s Dashboard (Pods and Replica Sets option of left side pane) ( As we mentioned 3 replica in our replica-deployment.yaml file)

A screenshot of a computer

Description automatically generated



I