Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» Факультет інформатики та обчислювальної техніки Кафедра автоматики та управління в технічних системах

Лабораторна робота №4 з дисципліни «Проектування розподілених систем» за темою «Розгортання застосунок в Kubernetes (Azure Kubernetes Server, Digital Ocean Kubernetes або локально в minikube)»

Виконав:

Студент групи IA-11мн Новиков Данило Михайлович

Перевірив:

доц. Волокина Артем Миколайович

Tema: Розгортання застосунок в Kubernetes (Azure Kubernetes Server, Digital Ocean Kubernetes або локально в minikube).

Завдання:

- Розгорнути Kubernetes as a Service або локально minikube;
- Розгорнути застосунок і всі допоміжні сервіси в Kubernetes.

Хід роботи

Репозиторій: https://github.com/JokerFunny/PRZ.

1. Перевіримо чи включена віртуалізація на машині:

C:\Windows\System32>systeminfo

Hyper-V Requirements:

A hypervisor has been detected. Features required for Hyper-V will not be displayed.

2. Віртуалізація включена. Kubectl/Minikube встановлено рік тому, перевіримо що вони ще живі:

```
C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ>kubectl version --client
WARNING: This version information is deprecated and will be replaced with the output from kubectl version --short. Use
--output=yaml|json to get the full version.
Client Version: version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.2", GitCommit:"5835544ca568b757a8ecae5c153f317e573
6700e", GitTreeState:"clean", BuildDate:"2022-09-21T14:33:49Z", GoVersion:"go1.19.1", Compiler:"gc", Platform:"windows/a
md64"}
Kustomize Version: v4.5.7
```

```
C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ>minikube start

* minikube v1.24.0 on Microsoft Windows 11 Enterprise 10.0.22621 Build 22621

* Using the docker driver based on existing profile

* Starting control plane node minikube in cluster minikube

* Pulling base image ...

* docker "minikube" container is missing, will recreate.

* Creating docker container (CPUs=2, Memory=8100MB) ...

* Preparing Kubernetes v1.22.3 on Docker 20.10.8 ...

- Generating certificates and keys ...

- Booting up control plane ...

- Configuring RBAC rules ...

* Verifying Kubernetes components...

- Using image gcr.io/k8s-minikube/storage-provisioner:v5

* Enabled addons: storage-provisioner, default-storageclass

! C:\Program Files\Docker\Docker\resources\bin\kubectl.exe is version 1.25.2, which may have incompatibilites with Kuber netes 1.22.3.

- Want kubectl v1.22.3? Try 'minikube kubectl -- get pods -A'

* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ>minikube status

minikube

type: Control Plane

host: Running kubelet: Running apiserver: Running

kubeconfig: Configured

3. Перевіримо ноди та сервіси kubectl:

```
C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ>kubectl get nodes
                    ROLES
                                            AGE
NAME
           STATUS
                                                  VERSION
minikube
           Ready
                    control-plane, master
                                            61s
                                                   v1.22.3
C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ>kubectl get services
            TYPE
                        CLUSTER-IP
                                     EXTERNAL-IP
                                                   PORT(S)
            ClusterIP
                        10.96.0.1
                                                   443/TCP
kubernetes
                                     <none>
                                                             74s
```

4. Перейдемо до деплою застосунку. Створимо конфіг для бази даних:

```
! db-conf.yaml > {} metadata

1    apiVersion: v1
2    kind: ConfigMap
3    metadata:
4    name: postgres-config
5    labels:
6    app: postgres
7    data:
8    POSTGRES_DB: jotter
9    POSTGRES_USER: pgadmin
10    POSTGRES_PASSWORD: pgadmin
11
```

5. Задеплоємо базу даних - створимо Volume, VolumeClaims, Deployment та Service:

```
apiVersion: apps/v1
kind: Deployment

metadata:

mane: postgres
spec:

replicas: 1
selector:

matchlabels:
 app: postgres

template:

metadata:

labels:
 app: postgres

spec:

containers:

- name: postgres:

image: postgres:13
imagerullPolicy: "IfNotPresent"
ports:

- contigNapRef:
 name: postgres-config
volumeNounts:
 - mountPath: /var/lib/postgresql/data
name: postgredb

volumes:

- name: postgredb

volumes:

- name: postgredb

volumes:

- name: postgredb

volumes:

- mane: postgredb

volumes:

- mane: postgredb

volumes:

- name: postgres

- ports: 5432

targetPort: 5432
```

6. Опишемо бекенд сервіс - створимо Deployment та Service:

```
1 apiVersion: ppp://si
2 kinds Deployment
3 metadata:
4 mass: labs-websupp
5 spec;
5 spec;
6 selector:
7 matchiabels:
8 app: labs-websupp
9 replicas: 1
10 template:
11 template:
12 app: labs-websupp
13 labsis:
14 spec;
15 containers:
16 mass: labs-websupp
17 image: docker.lo/joker/S0/jotterspi:latest
18 ports:
19 image: docker.lo/joker/S0/jotterspi:latest
19 ports:
19 containers:
10 mass: labs-websupp
17 image: docker.lo/joker/S0/jotterspi:latest
18 ports:
19 containers:
10 containers:
10 mass: postgres-config
10 mass: PostGRES_DB
10 mass: PostGRES_DB
11 mass: postgres-config
12 key: PostGRES_DB
12 mass: postgres-config
13 key: PostGRES_DB
14 mass: postgres-config
15 mass: postgres-config
16 mass: postgres-config
17 mass: postgres-config
18 key: PostGRES_DB
19 mass: postgres-config
19 mass: postgres-config
10 mass: postgres-config
11 mass: postgres-config
12 mass: postgres-config
13 mass: postgres-config
14 mass: postgres-config
15 mass: postgres-config
16 mass: postgres-config
17 mass: postgres-config
18 mass: postgres-config
19 mass: postgres-config
10 mass: postgres-config
10 mass: postgres-config
11 mass: postgres-config
12 mass: postgres-config
13 mass: postgres-config
14 mass: postgres-config
15 mass: postgres-config
16 mass: postgres-config
17 mass: configuencynef:
18 mass: postgres-config
18 mass: postgres-config
19 mass: postgres-config
19 mass: postgres-config
10 mass: postgres-config
11 mass: postgres-config
12 mass: postgres-config
13 mass: postgres-config
14 mass: postgres-config
15 mass: postgres-config
16 mass: postgres-config
17 mass: postgres-config
18 mass: postgres-config
18 mass: postgres-config
18 mass: postgres-config
19 mass: postgres-config
10 mass: postgres-config
11 mass: postgres-config
12 mass: postgres-config
14 mass: postgres-config
15
```

```
42 | | | | | cpu: "1"

43

44 ---

45

46 apiVersion: v1

47 kind: Service

48 metadata:

49 | name: lab4-webapp

50 spec:

51 | selector:

52 | app: lab4-webapp

53 ports:

54 | - name: "http"

| protocol: TCP

| port: 5000

57 | targetPort: 5000

58 | nodePort: 30103

59 | type: NodePort

60
```

7. Запустимо minikube:

```
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> minikube start --driver=docker
minikube v1.24.0 on Microsoft Windows 11 Enterprise 10.0.22621 Build 22621
Using the docker driver based on existing profile
Starting control plane node minikube in cluster minikube
Pulling base image ...
Updating the running docker "minikube" container ...
Preparing Kubernetes v1.22.3 on Docker 20.10.8 ...
Verifying Kubernetes v1.22.3 on Docker 20.10.8 ...
Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: storage-provisioner, default-storageclass

C:\Program Files\Docker\Docker\Docker\resources\bin\kubectl.exe is version 1.25.2, which may have incompatibilites with Kubernetes 1.22.3.
Want kubectl v1.22.3? Try 'minikube kubectl -- get pods -A'
Donel kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

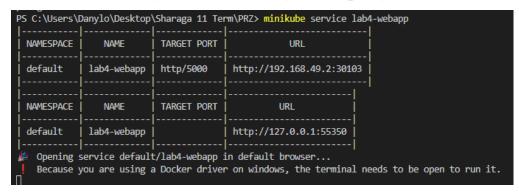
8. Задеплоїмо db-conf.yaml + db-deploy.yaml + app-deploy.yaml:

```
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl apply -f .\db-conf.yaml configmap/postgres-config created
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl apply -f .\db-deploy.yaml persistentvolume/lab4-pv created persistentvolumeclaim/lab4-pvc created deployment.apps/postgres created service/postgres created
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl apply -f .\app-deploy.yaml deployment.apps/lab4-webapp created service/lab4-webapp created
```

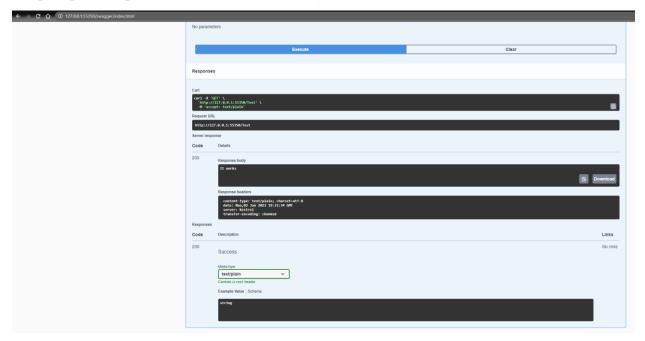
9. Перевіримо, що все коректно працює:

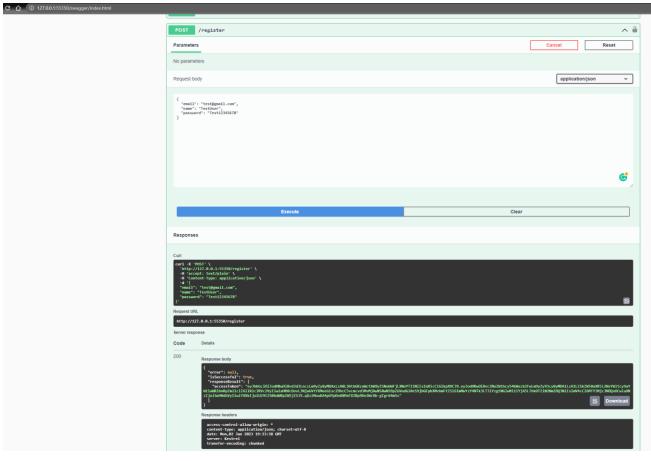
```
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get pvc
NAME
         STATUS VOLUME
                           CAPACITY ACCESS MODES STORAGECLASS
                                                                  AGE
                  lab4-pv
                            2Gi
lab4-pvc Bound
                                      RWO
                                                    manual
                                                                  26s
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get services
            TYPE
                      CLUSTER-IP
                                    EXTERNAL-IP PORT(S)
                                                                    AGE
                      10.96.0.1
kubernetes
          ClusterIP
                                      <none>
                                                    443/TCP
                                                                    30m
lab4-webapp NodePort
                        10.108.252.50 <none>
                                                    5000:30103/TCP
                                                                    32s
            ClusterIP 10.101.99.58 <none>
                                                    5432/TCP
postgres
                                                                    415
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get pods
                            READY
                                   STATUS
lab4-webapp-d65d7bf96-z6tr5
                           1/1
                                   Running
                                            0
                                                       39s
                                   Running
postgres-6b84868887-9c162
                          1/1
                                            0
                                                       485
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get deployments
             READY
                    UP-TO-DATE
                              AVAILABLE
                                           AGE
lab4-webapp
             1/1
                                           56s
postgres
             1/1
```

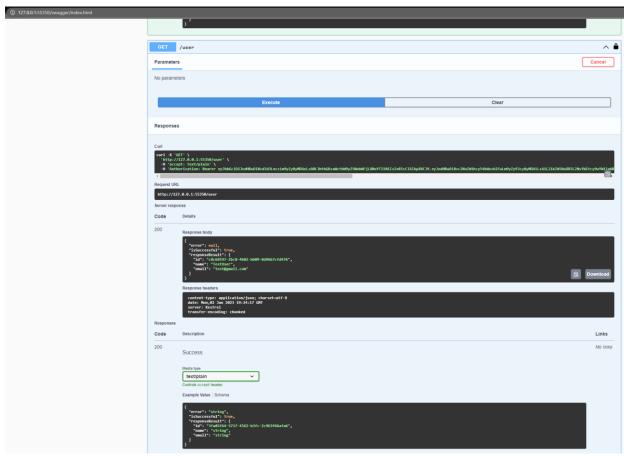
10. Служба NodePort — це найпростіший спосіб отримати зовнішній трафік безпосередньо до служби. NodePort відкриває певний порт, і будь-який трафік, який надсилається на цей порт, перенаправляється до служби. Відкриємо порт назовні, за яким будемо мати доступ до сервісу:



11. Перевіримо працездатність застосунку:







Висновок:

В результаті виконання лабораторної роботи було розгорнуто .NET застосунок + PostgreSQL локально за допомогою minikube.