

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 застосунків в Kubernetes (Azure Kubernetes Server, Digital Ocean
Kubernetes або локально в minikube)»

Виконав:

Студент групи ІА-11мн

Новиков Данило Михайлович

Перевірив:

доц. Волокіна Артем Миколайович

Київ 2022

Тема: 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
NAME          STATUS    ROLES          AGE    VERSION
minikube      Ready    control-plane, 61s    v1.22.3
master
```

```
C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ>kubectl get services
NAME          TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
kubernetes    ClusterIP   10.96.0.1     <none>         443/TCP    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:

```
! db-deploy.yaml > {} spec > [ ] ports > {} 0 > protocol
```

```
1  apiVersion: v1
2  kind: PersistentVolume
3  metadata:
4    name: lab4-pv
5    labels:
6      type: local
7      app: postgres
8  spec:
9    storageClassName: manual
10   capacity:
11     storage: 2Gi
12   accessModes:
13     - ReadWriteOnce
14   hostPath:
15     path: postgresdata
16
17 ---
18
19 apiVersion: v1
20 kind: PersistentVolumeClaim
21 metadata:
22   name: lab4-pvc
23   labels:
24     app: postgres
25 spec:
26   volumeName: lab4-pv
27   storageClassName: manual
28   accessModes:
29     - ReadWriteOnce
30   resources:
31     requests:
32       storage: 2Gi
33
34 ---
35
```

```
36 ---
37
38 apiVersion: apps/v1
39 kind: Deployment
40 metadata:
41   name: postgres
42 spec:
43   replicas: 1
44   selector:
45     matchLabels:
46       app: postgres
47   template:
48     metadata:
49       labels:
50         app: postgres
51     spec:
52       containers:
53         - name: postgres
54           image: postgres:13
55           imagePullPolicy: "IfNotPresent"
56           ports:
57             - containerPort: 5432
58           envFrom:
59             - configMapRef:
60               name: postgres-config
61           volumeMounts:
62             - mountPath: /var/lib/postgresql/data
63               name: postgreddb
64       volumes:
65         - name: postgreddb
66           persistentVolumeClaim:
67             claimName: lab4-pvc
68
69 ---
70
71 apiVersion: v1
72 kind: Service
73 metadata:
74   name: postgres
75 spec:
76   type: ClusterIP
77   selector:
78     app: postgres
79   ports:
80     - name: "postgres"
81       protocol: TCP
82       port: 5432
83       targetPort: 5432
84
```

6. Опишемо бекенд сервіс - створимо Deployment та Service:

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: lab4-webapp
5 spec:
6   selector:
7     matchLabels:
8       app: lab4-webapp
9   replicas: 1
10  template:
11    metadata:
12      labels:
13        app: lab4-webapp
14    spec:
15      containers:
16        - name: lab4-webapp
17          image: docker.io/joker759/jotterapi:latest
18          ports:
19            - containerPort: 5000
20              name: "http"
21          env:
22            - name: POSTGRES_DB
23              valueFrom:
24                configMapKeyRef:
25                  name: postgres-config
26                  key: POSTGRES_DB
27            - name: POSTGRES_USER
28              valueFrom:
29                configMapKeyRef:
30                  name: postgres-config
31                  key: POSTGRES_USER
32            - name: POSTGRES_PASSWORD
33              valueFrom:
34                configMapKeyRef:
35                  name: postgres-config
36                  key: POSTGRES_PASSWORD
37            - name: ConnectionStrings_JotterDbContext
38              value: User ID=${POSTGRES_USER};Password=${POSTGRES_PASSWORD};Host=${POSTGRES_SERVICE_HOST};Port=5432;Database=${POSTGRES_DB};Pooling=true;
39          resources:
40            limits:
41              memory: 2Gi
42              cpu: "1"
43
44  ---
45
```

[illegible]

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 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 Kubernetes 1.22.3.
  ▪ Want kubectll v1.22.3? Try 'minikube kubectll -- get pods -A'
🏁 Done! kubectll 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
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-pvc      Bound   lab4-pv  2Gi       RWO            manual        26s
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get services
NAME          TYPE        CLUSTER-IP  EXTERNAL-IP  PORT(S)          AGE
kubernetes    ClusterIP   10.96.0.1    <none>       443/TCP          30m
lab4-webapp    NodePort    10.108.252.50 <none>       5000:30103/TCP   32s
postgres      ClusterIP   10.101.99.58 <none>       5432/TCP         41s
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get pods
NAME                                READY  STATUS   RESTARTS  AGE
lab4-webapp-d65d7bf96-z6tr5        1/1    Running  0         39s
postgres-6b84868887-9cl62         1/1    Running  0         48s
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> kubectl get deployments
NAME          READY  UP-TO-DATE  AVAILABLE  AGE
lab4-webapp    1/1    1            1           56s
postgres      1/1    1            1           65s
```

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:

```
PS C:\Users\Danylo\Desktop\Sharaga 11 Term\PRZ> minikube service lab4-webapp
|-----|
| NAMESPACE | NAME      | TARGET PORT | URL                               |
|-----|
| default    | lab4-webapp | http/5000    | http://192.168.49.2:30103        |
|-----|
| NAMESPACE | NAME      | TARGET PORT | URL                               |
|-----|
| default    | lab4-webapp |             | http://127.0.0.1:55350          |
|-----|
Opening service default/lab4-webapp in default browser...
! Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

11. Перевіримо працездатність застосунку:

The screenshot shows the Swagger UI for a REST API. The interface is clean and modern, with a light blue and white color scheme. At the top, there's a header with the API name and version. Below that, there's a section for parameters, which is currently empty. The main part of the interface is the 'Responses' section, which shows the result of a GET request to the endpoint 'http://127.0.0.1:55350/test'. The response is a 200 status code, indicating a successful request. The response body is 'It works'. The response headers are also displayed, including 'content-type: text/plain; charset=utf-8', 'date: Mon, 02 Jun 2023 19:12:34 GMT', 'server: Kestrel', and 'transfer-encoding: chunked'. At the bottom, there's a 'Media type' dropdown menu set to 'text/plain' and an 'Example Value' field containing the string 'string'.

Висновок:

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