1. Микросервисная архитектура и контейнеризация

1.1. Урок 4. Хранение данных и ресурсы

https://github.com/adterskov/geekbrains-conteinerization/tree/master/homework/4.resources-and-persistence

2. Деплой

Создаем пространство имен и переключаемся на него:

```
kubectl create ns postgresql
kubectl config set-context --current --namespace=postgresql
```

Смотрим, какие хранилища нам доступны:

```
PS C:\Repos\GB-DevOps-Conteinerization\HW04> kubectl get storageclasses.storage.k8s.io
NAME                     PROVISIONER                RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
csi-ceph-hdd-gz1         cinder.csi.openstack.org   Delete          Immediate           true                   8d
csi-ceph-hdd-gz1-retain  cinder.csi.openstack.org   Retain          Immediate           true                   8d
csi-ceph-hdd-ms1         cinder.csi.openstack.org   Delete          Immediate           true                   8d
csi-ceph-hdd-ms1-retain  cinder.csi.openstack.org   Retain          Immediate           true                   8d
csi-ceph-ssd-gz1         cinder.csi.openstack.org   Delete          Immediate           true                   8d
csi-ceph-ssd-gz1-retain  cinder.csi.openstack.org   Retain          Immediate           true                   8d
csi-ceph-ssd-ms1         cinder.csi.openstack.org   Delete          Immediate           true                   8d
csi-ceph-ssd-ms1-retain  cinder.csi.openstack.org   Retain          Immediate           true                   8d
csi-high-iops-gz1        cinder.csi.openstack.org   Delete          Immediate           true                   8d
csi-high-iops-gz1-retain cinder.csi.openstack.org   Retain          Immediate           true                   8d
csi-high-iops-ms1        cinder.csi.openstack.org   Delete          Immediate           true                   8d
csi-high-iops-ms1-retain cinder.csi.openstack.org   Retain          Immediate           true                   8d
```

Выбираем SSD на той же площадке, что и кубер **csi-ceph-ssd-gz1**, создаем манифест **PersistentVolumeClaim.yaml** и применяем его к куберу:

```yaml
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: postgresql-storage
  namespace: postgresql
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  storageClassName: "csi-ceph-ssd-gz1"
```

```
kubectl apply -f PersistentVolumeClaim.yaml
```

Готово:

```
PS C:\Repos\GB-DevOps-Conteinerization\HW04> kubectl get pvc
NAME                STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS       AGE
postgresql-storage  Bound    pvc-56219a68-06ab-4677-8d5d-0045eeb6af5a   10Gi       RWX            csi-ceph-ssd-ms1   62s
```

Создадим секрет для хранения пароля от БД:

```
kubectl create secret generic postgresql-secret --from-literal=PASS=testpassword
```

Проверим содержимое secret'a:

```
PS C:\Repos\GB-DevOps-Conteinerization\HW04> kubectl get secret postgresql-secret -oyaml
apiVersion: v1
data:
  PASS: dGVzdHBhc3N3b3Jk
kind: Secret
metadata:
  creationTimestamp: "2023-10-09T09:11:01Z"
  name: postgresql-secret
  namespace: postgresql
  resourceVersion: "37190"
  uid: 83733f4d-d65f-43c8-8aa1-caec881309ba
type: Opaque
```

Создадим деплоймент для базы:

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: postgresql-db
  namespace: postgresql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: postgresql-db
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: postgresql-db
    spec:
      initContainers:
      - image: busybox
        name: mount-permissions-fix
        command: ["sh", "-c", "chmod 777 /var/lib/postgresql/data"]
        volumeMounts:
        - name: data
          mountPath: /var/lib/postgresql/data
      containers:
      - image: postgres:10.13
        name: postgres
        env:
        - name: POSTGRES_USER
          value: "testuser"
        - name: POSTGRES_DB
          value: "testdatabase"
        - name: PGDATA
          value: "/var/lib/postgresql/data/pgdata"
        - name: POSTGRES_PASSWORD
          valueFrom:
            secretKeyRef:
              name: postgresql-secret
              key: PASS
        ports:
        - containerPort: 5432
          protocol: TCP
        resources:
          requests:
            cpu: 100m
            memory: 1Gi
          limits:
            cpu: 100m
            memory: 1Gi
```

```
        volumeMounts:
        - name: data
          mountPath: /var/lib/postgresql/data
      volumes:
      - name: data
        persistentVolumeClaim:
          claimName: postgresql-storageent
```

Развернем в кубере деплоймент с базой:

```
kubectl apply -f Deployment.yaml
```

Посмотрим PODы:

```
PS C:\Repos\GB-DevOps-Conteinerization\HW04> kubectl get po
NAME                             READY   STATUS    RESTARTS   AGE
postgresql-db-848ccf5799-zjmmr   1/1     Running   0          65s
```

3. Проверка

Для проверки работоспособности базы данных:

1. Узнайте IP пода postgresql: kubectl get pod -o wide: `10.100.237.70`
2. Запустите рядом тестовый под: `kubectl run -t -i --rm --image postgres:10.13 test bash`
3. Внутри тестового пода выполните команду для подключения к БД: `psql -h 10.100.237.70 -U testuser testdatabase` Введите пароль - `testpassword`
4. Все в том же тестовом поде, после подключения к инстансу БД выполните команду для создания таблицы: `CREATE TABLE testtable (testcolumn VARCHAR (50) );`
5. Проверьте что таблица создалась. Для этого все в том же тестовом поде выполните команду `\dt`

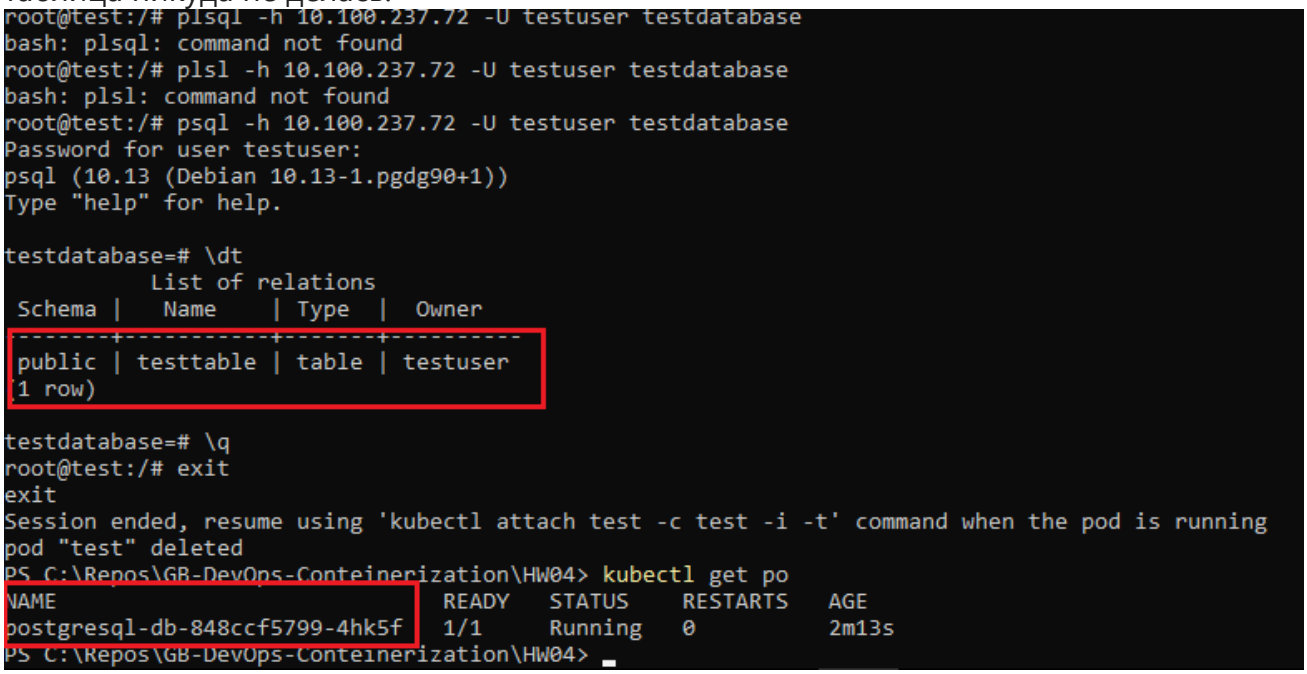
```
root@test:/# psql -h 10.100.237.70 -U testuser testdatabase
Password for user testuser:
psql (10.13 (Debian 10.13-1.pgdg90+1))
Type "help" for help.

testdatabase=# CREATE TABLE testtable (testcolumn VARCHAR (50) );
CREATE TABLE
testdatabase=# \dt
          List of relations
 Schema |   Name    | Type  |  Owner
--------+-----------+-------+----------
 public | testtable | table | testuser
(1 row)

testdatabase=#
```

6. Выйдите из тестового пода. Попробуйте удалить под с postgresql: `kubectl delete po postgresql-db-848ccf5799-zjmmr`

7. После его пересоздания повторите все с п.1, кроме п.4 Проверьте что созданная ранее таблица никуда не делась.



Видим, что новый под создался, что таблица присутствует, данные сохраняются.