ЗАДАНИЕ № 4

1) Напишите deployment для запуска сервера базы данных Postgresql. Приложение должно запускаться из образа postgres:10.13 Должен быть описан порт: 5432 TCP. В деплойменте должна быть одна реплика, при этом при обновлении образа НЕ ДОЛЖНО одновременно работать несколько реплик (то есть сначала должна удаляться старая реплика и только после этого подниматься новая). Это можно сделать или с помощью maxSurge/maxUnavailable или указав стратегию деплоя Recreate. В базе данных при запуске должен автоматически создаваться пользователь testuser с паролем testpassword. А также база testdatabase. Для этого нужно указать переменные окружения POSTGRES_PASSWORD, POSTGRES_USER, POSTGRES_DB в деплойменте. При этом значение переменной POSTGRES_PASSWORD должно браться из секрета. Так же нужно указать переменную PGDATA со значением /var/lib/postgresql/data/pgdata (См. документацию к образу https://hub.docker.com/_/postgres раздел PGDATA). База данных должна хранить данные в PVC с размером диска в 10Gi, замонтированном в pod по пути /var/lib/postgresql/data.
Для проверки работоспособности базы данных:
Узнайте IP пода postgresql
kubectl get pod -o wide
Запустите рядом тестовый под
kubectl run -t -i --rm --image postgres:10.13 test bash
Внутри тестового пода выполните команду для подключения к БД
psql -h <postgresql pod IP из п.1> -U testuser testdatabase
Введите пароль - testpassword
Все в том же тестовом поде, после подключения к инстансу БД выполните команду для создания таблицы
CREATE TABLE testtable (testcolumn VARCHAR (50));
Проверьте что таблица создалась. Для этого все в том же тестовом поде выполните команду
\dt. Выйдите из тестового пода. Попробуйте удалить под с postgresql. После его пересоздания повторите все с п.1, кроме п.4. Проверьте что созданная ранее таблица никуда не делась.

РЕШЕНИЕ:

1)
kubectl apply -f pv1.yaml
kubectl apply -f pvc1.yaml
kubectl apply -f secure.yaml
kubectl apply -f deploy_psql.yaml

```
NAME   STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
pvc1   Bound    pv1      1Gi        RWO            local          68s
```

```
root@debian1:~# kubectl describe pods
Name:         postgres-78f5f5d6f4-7s9js
Namespace:    default
Priority:     0
Node:         kub1-default1-0/10.0.0.4
Start Time:   Mon, 09 May 2022 15:23:05 +0000
Labels:       app=postgres
              pod-template-hash=78f5f5d6f4
Annotations:  cni.projectcalico.org/containerID: fae3e49402dac
              cni.projectcalico.org/podIP: 10.100.140.192/32
              cni.projectcalico.org/podIPs: 10.100.140.192/32
              kubernetes.io/limit-ranger: LimitRanger plugin s
Status:       Running
IP:           10.100.140.192
IPs:
  IP:           10.100.140.192
Controlled By:  ReplicaSet/postgres-78f5f5d6f4
Containers:
  postgres:
    Container ID:   cri-o://c6887b62a40ed362844babee866a9052fe
    Image:          postgres:10.13
    Image ID:       docker.io/library/postgres@sha256:0bed71d0
    Port:           5432/TCP
    Host Port:      0/TCP
    State:          Running
```

Убедимся, что наш PVC примаплен:

```
root@debian1:~# kubectl describe pvc pvc1
Name:          pvc1
Namespace:     default
StorageClass:  local
Status:        Bound
Volume:        pv1
Labels:        app=postgres
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Used By:       postgres-78f5f5d6f4-7s9js
Events:        <none>
```

Запустим тестовый под и подключимся к БД на ранее созданном поде, создадим таблицу:

```
root@debian1:~# kubectl get pod -o wide
NAME                        READY   STATUS    RESTARTS   AGE     IP
postgres-78f5f5d6f4-7s9js   1/1     Running   0          5m21s   10.100.140.192
root@debian1:~# kubectl run -t -i --rm --image postgres:10.13 test bash
If you don't see a command prompt, try pressing enter.
root@test:/#
root@test:/#
root@test:/# psql -h 10.100.140.192 -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)
```

Пересоздадим под и убедимся, что табл. на месте:

```
root@debian1:~# kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
postgres-78f5f5d6f4-7s9js   1/1     Running   0          13m
root@debian1:~# kubectl delete pods postgres-78f5f5d6f4-7s9js
pod "postgres-78f5f5d6f4-7s9js" deleted
root@debian1:~# kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
postgres-78f5f5d6f4-nczxt   1/1     Running   0          8s
```

```
root@debian1:~# kubectl get pod
NAME                        READY   STATUS    RESTARTS   AGE
postgres-78f5f5d6f4-nczxt   1/1     Running   0          22m
root@debian1:~# kubectl exec -t -i postgres-78f5f5d6f4-nczxt bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in
root@postgres-78f5f5d6f4-nczxt:/# psql -U testuser testdatabase
psql (10.13 (Debian 10.13-1.pgdg90+1))
Type "help" for help.

testdatabase=# \dt
          List of relations
 Schema |   Name    | Type  |  Owner
--------+-----------+-------+----------
 public | testtable | table | testuser
(1 row)
```