Домашнее задание для к уроку 5 - Сетевые абстракции Kubernetes

- Разверните в кластере сервер базы данных Postgresql. Из предыдущего задания.
- Добавьте к нему service с портом 5432 и именем database.
- В этом же неймспэйсе создайте deployment с образом redmine:4.1.1

Для запуска нужно передать переменные окружения:

REDMINE_DB_POSTGRES = database
REDMINE_DB_USERNAME = <postgres_user>
REDMINE_DB_PASSWORD = <postgres_password> (значение должно браться из секрета)
REDMINE_DB_DATABASE = <postgres_database>
REDMINE_SECRET_KEY_BASE = supersecretkey (значение должно браться из секрета)

> Обратите внимание что имя пользователя, пароль и база данных должны соответствовать значениям которые указаны в переменных окружения деплоймента postgresql
> В деплойменте приложения должен быть описан порт 3000

- Создайте serivce для приложения с портом 3000
- Создайте ingress для приложения, так чтобы запросы с любым доменом на белый IP вашего сервиса nginx-ingress-controller (тот что в нэймспэйсе ingress-nginx с типом LoadBalancer) шли на приложение
- Проверьте что при обращении из браузера на белый IP вы видите открывшееся приложение Redmine (https://www.redmine.org/)

Листинг манифестов:

cat deployment_postgres2.yaml

```
---
 apiVersion: apps/v1
 kind: Deployment
 metadata:
   name: deployment-postgres2
 spec:
   replicas: 1
   selector:
     matchLabels:
       app: postgres2
   strategy:
      type: Recreate
   template:
     metadata:
       labels:
         app: postgres2
     spec:
       containers:
```

```
      - image: postgres:12.4-alpine
        name: postgres2
        env:
         - name: POSTGRES_DB
           value: testdatabase
         - name: POSTGRES_USER
           value: testuser
         - name: POSTGRES_PASSWORD
           valueFrom:
             secretKeyRef:
               name: postgres
               key: POSTGRES_PASSWORD
         - name: PGDATA
           value: "/var/lib/postgresql/data/pgdata"
        volumeMounts:
                - mountPath: "/var/lib/postgresql/data"
                  name: ps-pgdata
      volumes:
      - name: ps-pgdata
        persistentVolumeClaim:
          claimName: ps-pgdata
```

cat service_postgres.yaml

```
---
apiVersion: v1
kind: Service
metadata:
  name: postgres-service
spec:
  selector:
    app: postgres2
  ports:
  - protocol: TCP
    port: 5432
  type: ClusterIP
```

cat deployment_redmine.yaml

```
---
 apiVersion: apps/v1
 kind: Deployment
 metadata:
   name: deployment-redmine
 spec:
   replicas: 1
   selector:
     matchLabels:
       app: redmine
   strategy:
```

```
    type: Recreate
  template:
    metadata:
      labels:
        app: redmine
    spec:
      containers:
      - image: redmine:4.1.1
        name: redmine
        env:
         - name: REDMINE_SECRET_KEY_BASE
           valueFrom:
             secretKeyRef:
               name: redmine
               key: REDMINE_SECRET_KEY_BASE
         - name: REDMINE_DB_DATABASE
           value: testdatabase
         - name: REDMINE_DB_USERNAME
           value: testuser
         - name: REDMINE_DB_PASSWORD
           valueFrom:
             secretKeyRef:
               name: postgres
               key: POSTGRES_PASSWORD
        ports:
        - containerPort: 3000
        resources:
         requests:
           cpu: 100m
           memory: 100Mi
         limits:
           cpu: 700m
           memory: 700Mi
```

Создадим сервис с типом LoadBalancer для доп теста другого внешнего IP.
cat service_redmine.yaml

```
---
apiVersion: v1
kind: Service
metadata:
  name: redmine-service
spec:
  selector:
    app: redmine
  ports:
  - protocol: TCP
    port: 80
    targetPort: 3000
  type: LoadBalancer
```

cat ingress_redmine.yaml

```
---
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: redmine-ingress
spec:
  rules:
  - http:
      paths:
      - backend:
          serviceName: redmine-service
          servicePort: 80
```

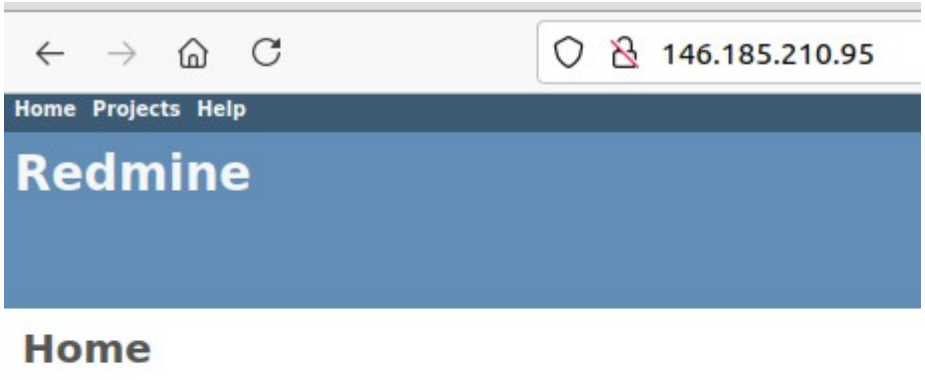
Смотрим POD,SVC,ING.

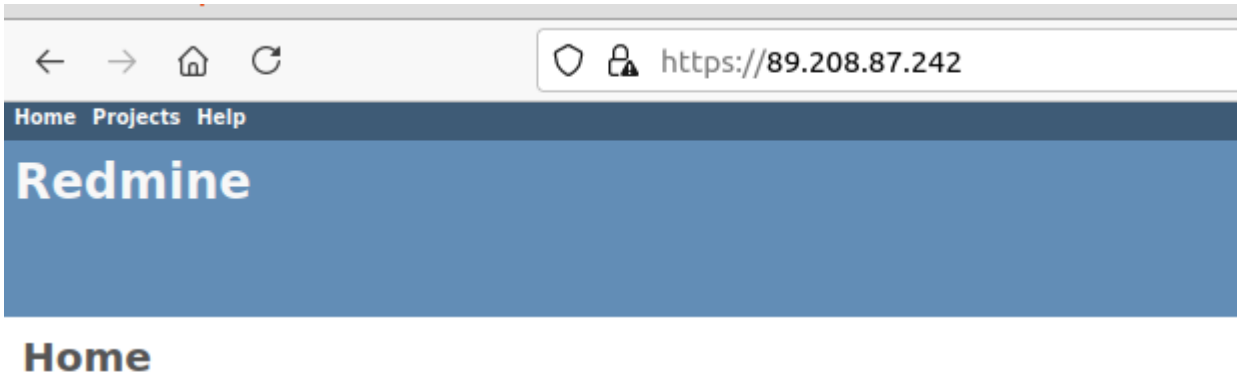
```
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW5$ k get po -owide
NAME                                  READY   STATUS    RESTARTS   AGE    IP              NODE                                        NOMINATED NODE   READINESS GATES
deployment-postgres2-86797f58f9-k9bp4 1/1     Running   0          132m   10.100.180.191  kubernetes-cluster-5268-default-group-0     <none>           <none>
deployment-redmine-5df77b555c-95p75   1/1     Running   0          10m    10.100.180.130  kubernetes-cluster-5268-default-group-0     <none>           <none>
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW5$ k get svc
NAME               TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
kubernetes         ClusterIP      10.254.0.1      <none>           443/TCP        6d8h
postgres-service   ClusterIP      10.254.92.16    <none>           5432/TCP       78m
redmine-service    LoadBalancer   10.254.35.196   146.185.210.95   80:30607/TCP   20m
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW5$ k get ing
NAME              CLASS    HOSTS   ADDRESS   PORTS   AGE
redmine-ingress   <none>   *                 80      21m
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW5$
```

Два внешних ip адреса

```
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW5$ k get svc -A
NAMESPACE       NAME                            TYPE           CLUSTER-IP       EXTERNAL-IP      PORT(S)                      AGE
default         kubernetes                      ClusterIP      10.254.0.1       <none>           443/TCP                      6d8h
default         postgres-service                ClusterIP      10.254.92.16     <none>           5432/TCP                     80m
default         redmine-service                 LoadBalancer   10.254.35.196    146.185.210.95   80:30607/TCP                 22m
ingress-nginx   nginx-ingress-controller        LoadBalancer   10.254.128.123   89.208.87.242    80:30080/TCP,443:30443/TCP   6d8h
ingress-nginx   nginx-ingress-controller-metrics ClusterIP     10.254.11.207    <none>           9913/TCP                     6d8h
ingress-nginx   nginx-ingress-default-backend   ClusterIP      10.254.244.136   <none>           80/TCP                       6d8h
kube-system     calico-node                     ClusterIP      None             <none>           9091/TCP                     6d8h
kube-system     csi-cinder-controller-service   ClusterIP      10.254.209.85    <none>           12345/TCP                    6d8h
kube-system     dashboard-metrics-scraper       ClusterIP      10.254.174.238   <none>           8000/TCP                     6d8h
kube-system     kube-dns                        ClusterIP      10.254.0.10      <none>           53/UDP,53/TCP,9153/TCP       6d8h
kube-system     kubernetes-dashboard            ClusterIP      10.254.179.232   <none>           443/TCP                      6d8h
kube-system     metrics-server                  ClusterIP      10.254.209.248   <none>           443/TCP                      6d8h
magnum-tiller   tiller-deploy                   ClusterIP      10.254.126.244   <none>           44134/TCP                    6d8h
```

Тестируем подключение по двум ip




https://89.208.87.242
Home Projects Help
Redmine
Home