Департамент образования города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

«Интеграция и развертывание программного обеспечения с помощью контейнеров»

Практическое задание

Тема: «Основы работы с Kubernetes»

Выполнила:

Студентка группы АДЭУ-211

Кравцова Алёна Евгеньевна

Руководитель:

Босенко Т.М

Москва

2025

Цель работы: Получить практические навыки работы с кластером Kubernetes, включая развертывание базовых компонентов, настройку мониторинга и работу с service mesh.

Задачи:

– Изучить основные концепции Kubernetes через практические вопросы;

– Научиться анализировать и применять манифесты Kubernetes.

Задание 1: Теоретические основы Kubernetes

Ответить на 3 случайных вопроса из репозитория (https://github.com/bregman-arie/devops-exercises/blob/master/topics/kubernetes/README.md#kubernetes-questions):

1. Когда и почему НЕ следует использовать Kubernetes?

Нет смысла использовать Kubernetes для небольшого проекта, где минимальная инфраструктура. Kubernetes потребляет много ресурсов, иногда проще использовать PaaS-решения или управляемые контейнерные сервисы. Также нет смысла прибегать к данным технологиям если в организации нет обученных специалистов (не менее 20 инженеров), которые разбираются в данной теме. Также не имеет смысла разворачивать Kubernetes, если у организации нет большой базы пользователей или если эти лиды непостоянны.

2. Что такое кластер Kubernetes?

Кластер Kubernetes (K8s) — это группа узлов (серверов), которые работают вместе для управления и оркестрации контейнеризированных приложений.

3. Какой тип сервиса используется по умолчанию?

В Kubernetes по умолчанию используется сервис ClusterIP. ClusterIP — это тип сервиса, который создает виртуальный IP-адрес, доступный только

внутри кластера. Такой сервис нельзя вызвать напрямую извне (например, из интернета).

Ответить на 3 случайных вопроса из репозитория (https://github.com/bregman-arie/devops-exercises/blob/master/topics/kubernetes/CKA.md):

1. Запустите команду для просмотра всех узлов кластера

Чтобы просмотреть все узлы кластера Kubernetes, необходимо использовать следующую команду: kubectl get nodes. Также данная команда выведет информацию о статусе узлов, их ролях и версии Kubernetes.

2. Как проверить, действителен ли манифест?

Чтобы проверить, действителен ли манифест Kubernetes, необходимо использовать команду kubectl apply --dry-run=client -f <имя_файла>. Проверяет синтаксис и структуру YAML-файла без отправки в кластер.

3. Что означает эта ошибка ImagePullBackOff?

Ошибка ImagePullBackOff в Kubernetes означает, что узел (Node) не может загрузить образ контейнера из указанного контейнерного реестра.

Задание 2: Развертывание локального кластера на Kubernetes с использованием MiniKube

2.1. Установка MiniKube

Minikube — это инструмент, позволяющий легко запускать Kubernetes на локальной машине иными словами упрощенная реализация полноценного Kubernetes-кластера. Установим MiniKube с помощью команд, представленных на рисунке 1 и рисунке 2.

```
mgpu@mgpu-VirtualBox:~$ curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  119M  100  119M    0     0  8100k      0  0:00:15  0:00:15 --:--:-- 9820k
```

Рис. 1 – Загрузка необходимых файлов

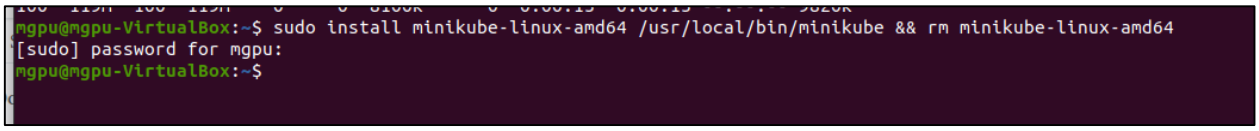

Рис. 2 – Установка MiniKube

2.2. Установите kubectl

kubectl — это командный инструмент для управления кластерами Kubernetes. Загрузим последнюю версию с помощью команды (Рис. 3).

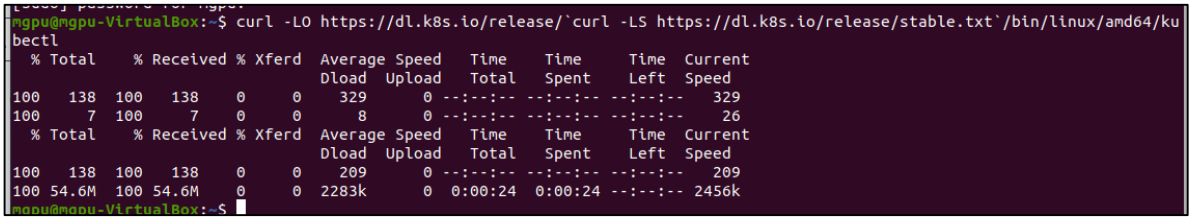

Рис. 3 – Загрузка необходимых файлов для kubectl

Далее необходимо сделать бинарный файл kubectl исполняемым и переместить бинарный файл в директорию из переменного окружения PATH (Рис. 4).

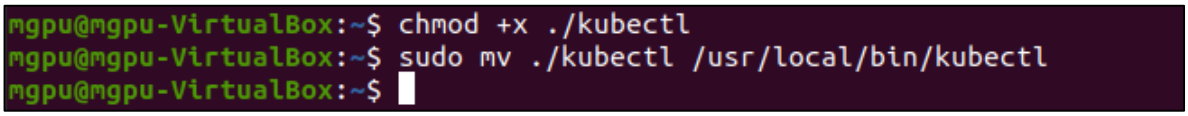

Рис. 4 – Настройка прав и перенос файла

Проверим версию (Рис. 5).

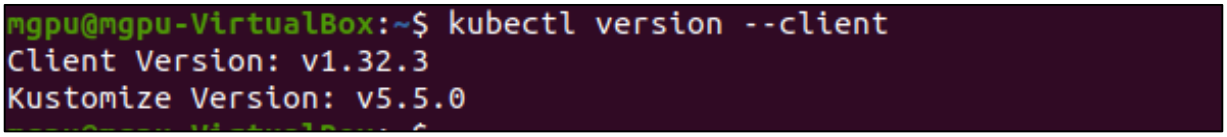

Рис. 5 – Проверка версии

Установим kubectl (Рис. 6).

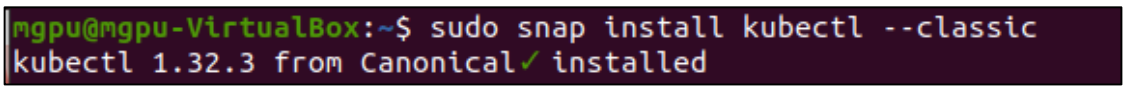

Рис. 6 – Установка kubectl

Перед началом работы необходимо добавить пользователя в группу Docker (Рис. 7).

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/lab4_1$ sudo usermod -aG docker $USER && newgrp docker
[sudo] password for mgpu:
```

Рис. 7 – Добавление пользователя в группу docker

Задание 2.3. Убедитесь, что kubectl работает и произведите осмотрите кластера.

Результат представлен на Рисунках 9, 10, 11, 12.

Задание 2.4. Установите графический интерфейс Dashboard.

Для установки необходимо загрузить helm, а далее выполнить команды в соответствии с инструкциями на официальном сайте (Рис. 8).

```
mgpu@mgpu-VirtualBox:~$ sudo snap install helm --classic
helm 3.17.2 from Snapcrafters✪ installed
mgpu@mgpu-VirtualBox:~$ helm repo add kubernetes-dashboard https://kubernetes.github.io/dashboard/
"kubernetes-dashboard" has been added to your repositories
mgpu@mgpu-VirtualBox:~$ helm upgrade --install kubernetes-dashboard kubernetes-dashboard/kubernetes-dashboard --create-namespace --n
amespace kubernetes-dashboard
Release "kubernetes-dashboard" does not exist. Installing it now.
NAME: kubernetes-dashboard
LAST DEPLOYED: Sat Mar 29 14:35:14 2025
NAMESPACE: kubernetes-dashboard
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
*********************************************************************************
*** PLEASE BE PATIENT: Kubernetes Dashboard may need a few minutes to get up and become ready ***
*********************************************************************************

Congratulations! You have just installed Kubernetes Dashboard in your cluster.

To access Dashboard run:
  kubectl -n kubernetes-dashboard port-forward svc/kubernetes-dashboard-kong-proxy 8443:443

NOTE: In case port-forward command does not work, make sure that kong service name is correct.
      Check the services in Kubernetes Dashboard namespace using:
        kubectl -n kubernetes-dashboard get svc

Dashboard will be available at:
  https://localhost:8443
mgpu@mgpu-VirtualBox:~$
```

Рис. 8 – Установка графического интерфейса

Далее запустим Dashboard (Рис. 9).

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/lab4_1$ minikube dashboard
🔌  Enabling dashboard ...
    ▪ Using image docker.io/kubernetesui/dashboard:v2.7.0
    ▪ Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
💡  Some dashboard features require the metrics-server addon. To enable all features please run:

        minikube addons enable metrics-server

🤔  Verifying dashboard health ...
🚀  Launching proxy ...
🤔  Verifying proxy health ...
🎉  Opening http://127.0.0.1:41983/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your
 default browser...
```

Рис. 9 – Запуск Dashboard

После чего откроется локальный адрес с дашбордами, что свидетельствует об успешной загрузке и запуске (Рис. 10).

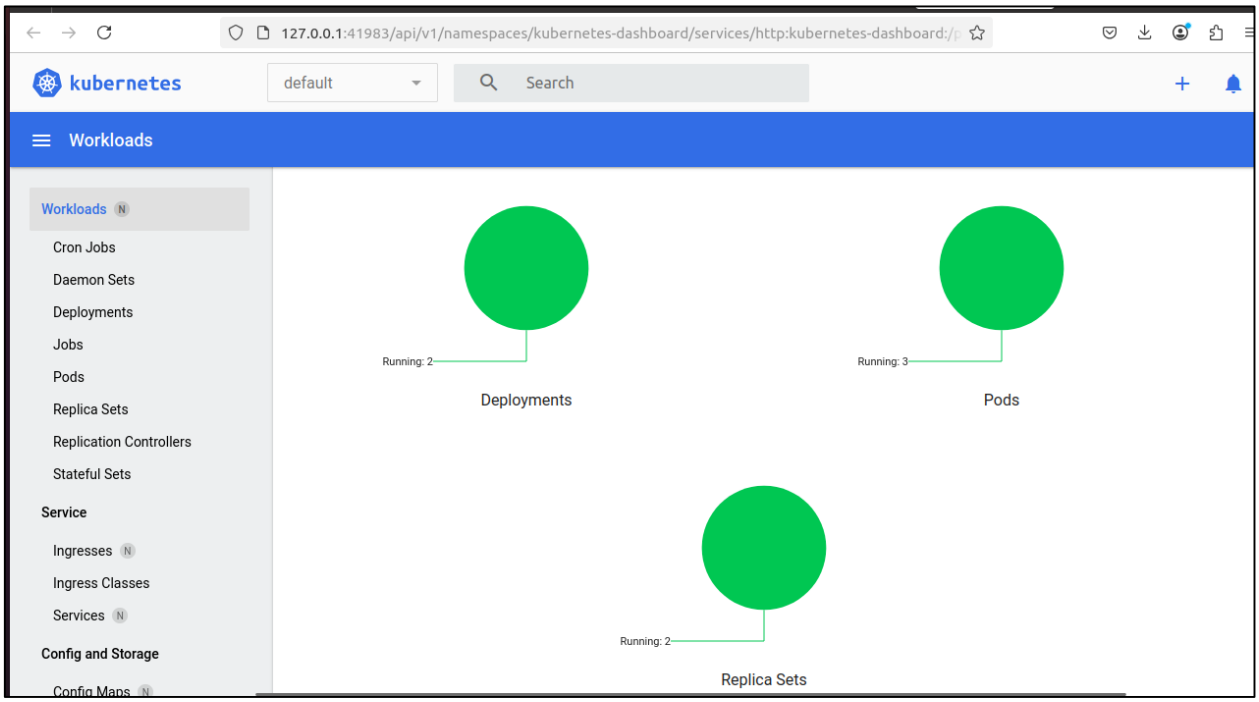

Рис. 10 – kubernetes Dashboard

Групповое задание.

Далее инициализируем Minikube-кластер (Рис. 11).

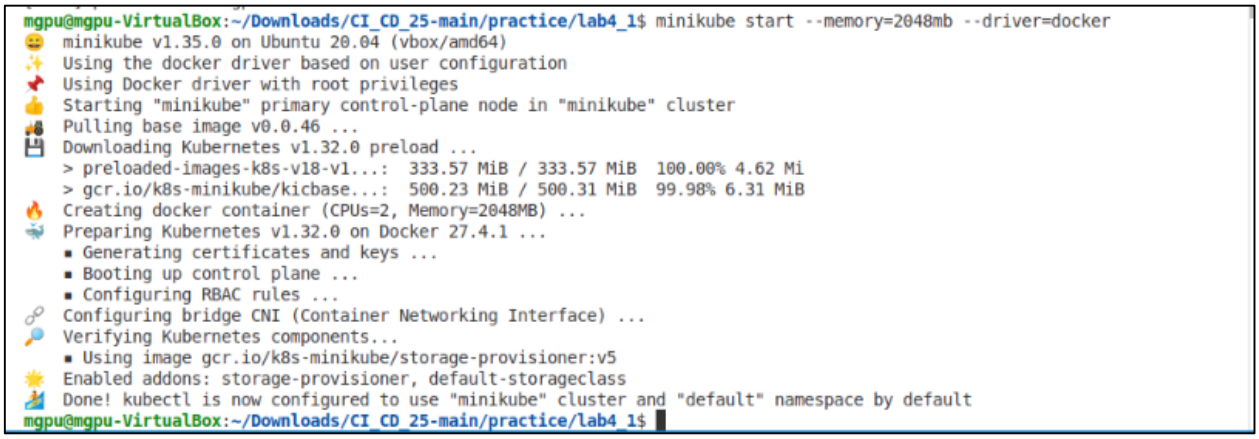

Рис. 11 – Инициализация Minikube

Далее проверим, что kubectl работает и выполним команду *kubectl get node*. Эта команда показывает список узлов в Kubernetes-кластере. Узлы — это машины (физические или виртуальные), на которых запускаются поды (Pods) и другие ресурсы Kubernetes (Рис. 12).

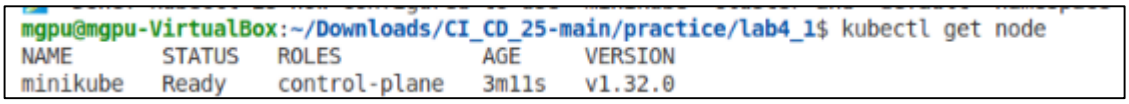

Рис. 12 – Список узлов

kubectl get po получает список подов в default namespace (пространстве имён по умолчанию). Поскольку подов в этом пространстве нет, выводится сообщение: «No resources found in default namespace.» (Рис. 13).

Рис. 13 – Список подов в пространстве имен

kubectl get po –A проверяет поды во всех пространствах имен (Рис. 14).

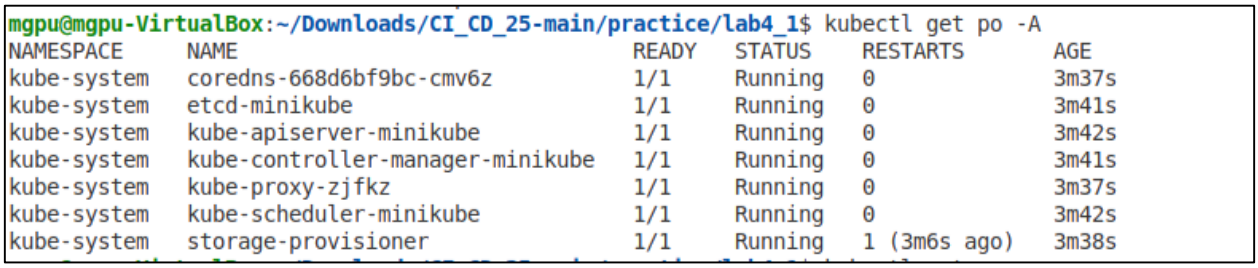

Рис. 14 – Список подов во всех пространствах

kubectl get svc показывает список сервисов, их IP-адреса и порты. Сервис kubernetes имеет тип ClusterIP, IP-адрес 10.96.0.1 и работает на порту 443 (Рис. 15).

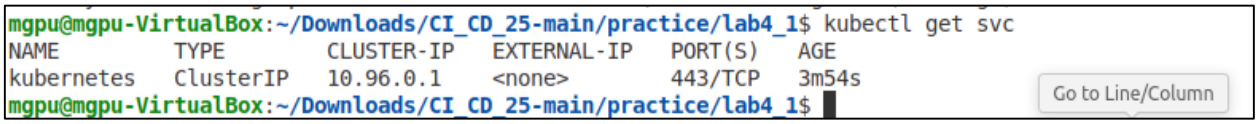

Рис. 15 – Список сервисов и информация о них

Далее используем команду для настройки окружения командной строки Ubuntu для работы с Docker, который управляется Minikube (Рис. 16).

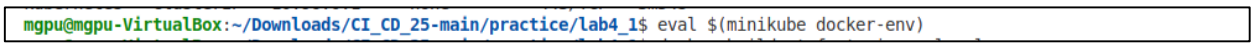

Рис. 16 – eval $(minikube docker-env)

Далее соберем docker образ Fast Api (Рис. 17).

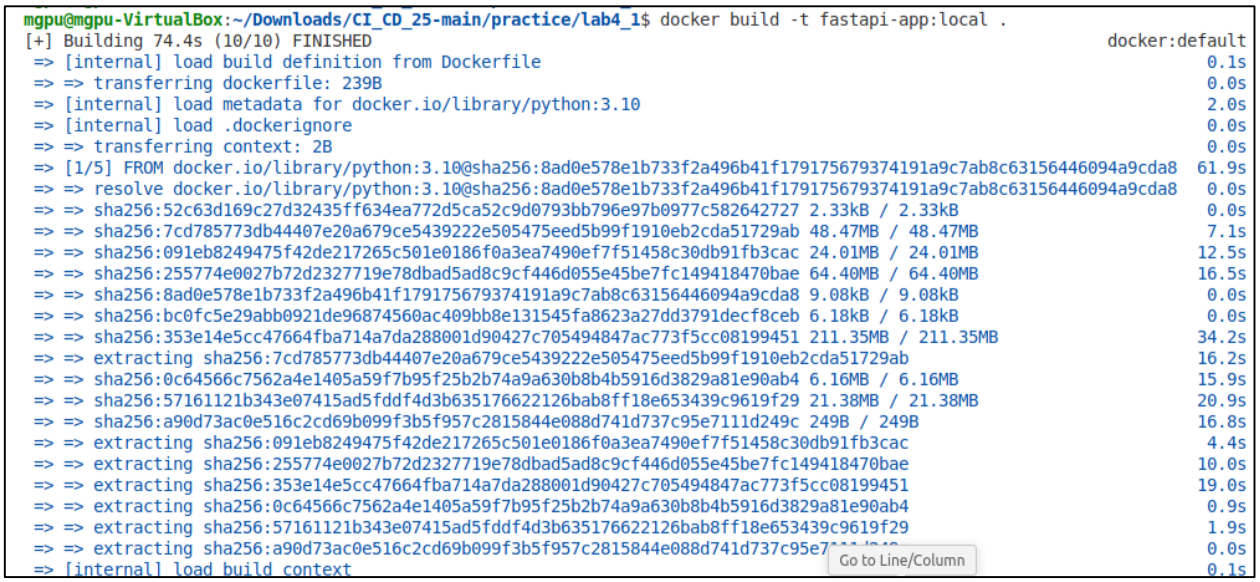

Рис. 17 – Сборка образа

Далее развернем необходимые ресурсы: конфигурация, secret.yml, deployment и сервисы (для fastapi и redis) (Рис. 18).

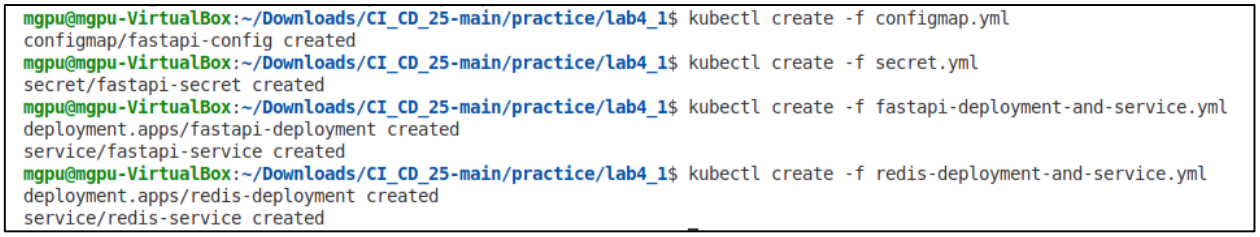

Рис. 18 – Создание ресурсов

В результате видим, что созданные ресурсы успешно запущены в созданном окружении (Рис. 19).

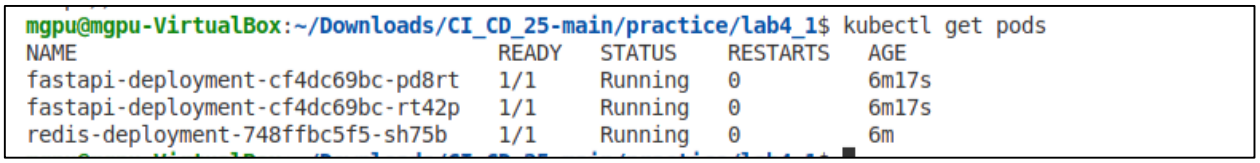

Рис. 19 –kubectl get pods

Далее проверим доступность в вебе. Необходимо узнать на каком адресе запущен FastaApi (Рис. 20).

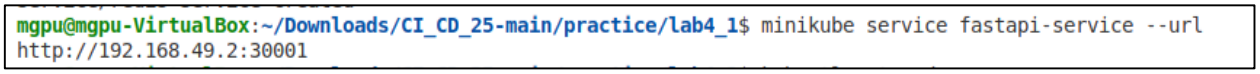

Рис. 20 – Адрес FastApi

Перейдем по данному адресу, FastApi запущенн и открыт (Рис. 21, Рис. 22).

Рис. 21 – FastApi

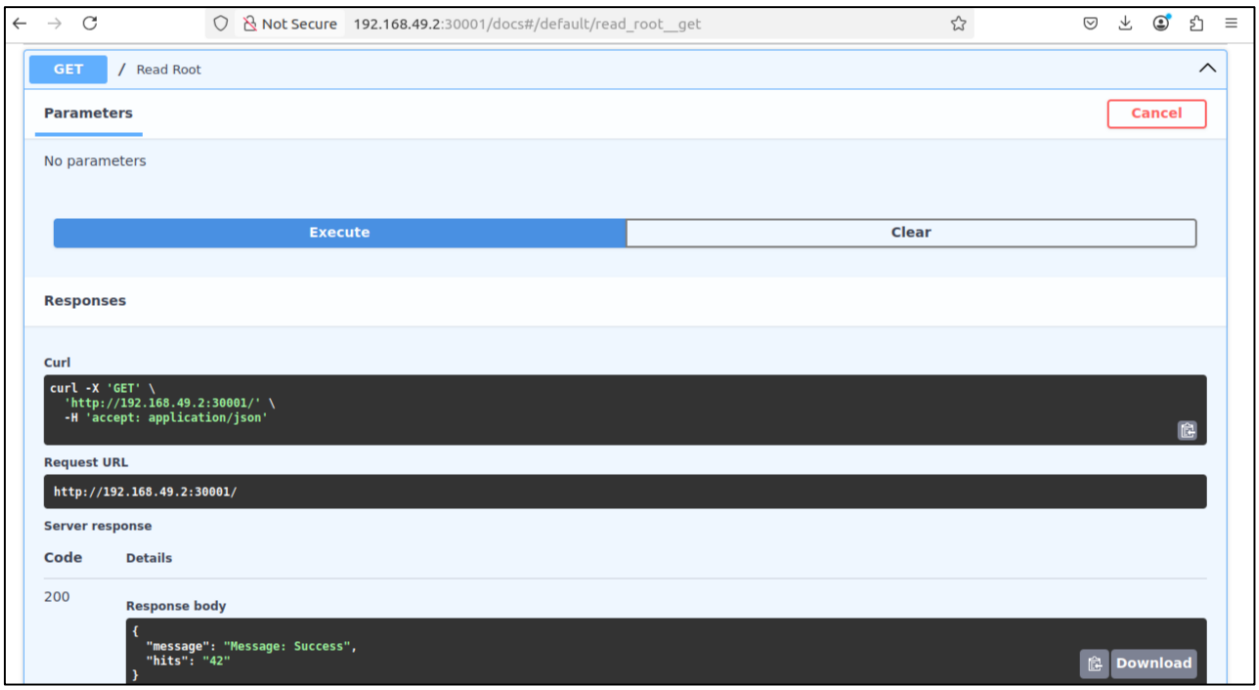

Рис. 22 – FastApi

С помощью команды, представленной на Рисунке 23, можно просмотреть детальную информацию о поде fastapi-deployment-cf4dc69bc-pd8rt. Исходя и полученного вывода можно прийти к заключению, что под успешно запущен и работает. Все контейнеры:

- Init-контейнер (init-myservice) корректно завершился.
- Основной контейнер (fastapi) работает стабильно.

- Liveness Probe настроен на проверку состояния FastAPI через HTTP-запросы.

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/lab4_1$ kubectl describe pod fastapi-deployment-cf4dc69bc-pd8rt
Name:             fastapi-deployment-cf4dc69bc-pd8rt
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Sat, 29 Mar 2025 14:03:46 +0300
Labels:           app=fastapi
                  pod-template-hash=cf4dc69bc
Annotations:      <none>
Status:           Running
IP:               10.244.0.3
IPs:
  IP:           10.244.0.3
Controlled By:  ReplicaSet/fastapi-deployment-cf4dc69bc
Init Containers:
  init-myservice:
    Container ID:  docker://4dc4aa06bf665f97439b940424b3f2facf21c40b1407b6d4851eabcb94c02673
    Image:         busybox
    Image ID:      docker-pullable://busybox@sha256:37f7b378a29ceb4c551b1b5582e27747b855bbfaa73fa11914fe0df028dc581f
    Port:          <none>
    Host Port:     <none>
    Command:
      sh
      -c
      echo 'This is the init container'
    State:          Terminated
      Reason:       Completed
      Exit Code:    0
```

Рис. 23 –Детальная информация о поде

Вывод команды *kubectl config view* показывает конфигурацию Kubernetes-контекста, используемого для взаимодействия с кластером (Рис. 24). Например, просмотреть действующее пространство имен, сведения о кластере и контекстах.

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/lab4_1$ kubectl config view
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/mgpu/.minikube/ca.crt
    extensions:
    - extension:
        last-update: Sat, 29 Mar 2025 13:41:42 MSK
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: cluster_info
    server: https://192.168.49.2:8443
  name: minikube
contexts:
- context:
    cluster: minikube
    extensions:
    - extension:
        last-update: Sat, 29 Mar 2025 13:41:42 MSK
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: context_info
    namespace: default
    user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
```

Рис. 24 – Конфигурация контекста

Индивидуальное задание. Вариант 6.

Вариант 6. Kubernetes. Часть 1 (elasticsearch)	Запустите Kubernetes локально (k3s или minikube). Проверьте работу системных контейнеров и пришлите скриншот команды: kubectl get po -n kube-system.	Имеется YAML с деплоем для **elasticsearch**. Измените файл: – Отключите базовую аутентификацию; – Фиксируйте образ на версии **7.10.0**; – Добавьте Service для доступа к кластеру. Приложите итоговый YAML.	Напишите команды kubectl для контейнера: – Выполнить внутри контейнера команду ps aux; – Просмотреть логи за 5 минут; – Удалить pod; – Пробросить порт для отладки.	Доп. задание*: Создайте YAML для: – ConfigMap с настройками для **elasticsearch**; – Deployment, использующий ConfigMap; – Ingress, направляющий запросы по пути /search на сервис.

1. Запустите Kubernetes локально (k3s или minikube). Проверьте работу системных контейнеров и пришлите скриншот команды: kubectl get po -n kube-system.

Запуск Kubernetes был произведен в групповом задании (см. выше). Команда kubectl get po -n kube-system показывает список подов (pods) в пространстве имён kube-system (Рис. 25). Исходя из вывода все основные компоненты Kubernetes работают исправно (STATUS: Running).

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/lab4_1$ kubectl get po -n kube-system
NAME                               READY   STATUS    RESTARTS      AGE
coredns-668d6bf9bc-cmv6z           1/1     Running   0             64m
etcd-minikube                      1/1     Running   0             64m
kube-apiserver-minikube            1/1     Running   0             64m
kube-controller-manager-minikube   1/1     Running   0             64m
kube-proxy-zjfkz                   1/1     Running   0             64m
kube-scheduler-minikube            1/1     Running   0             64m
storage-provisioner                1/1     Running   1 (63m ago)   64m
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/lab4_1$
```

Рис. 25 – Список подов

2. Имеется YAML с деплоем для elasticsearch. Измените файл: отключите базовую аутентификацию, фиксируйте образ на версии 7.10.0, добавьте Service для доступа к кластеру. Приложите итоговый YAML.

Для выполнения данного задания создадим отдельную директорию, в которой будет единый yaml файл со всеми настройками: Service для доступа к кластеру и дополнительные YAML-файлы для ConfigMap, Deployment и Ingress (с учётом доп. задания).

```
---
apiVersion: v1
kind: Secret
metadata:
  name: elasticsearch-secret
  namespace: default
type: Opaque
data:
  elastic-password: "cGFzc3dvcmQ=" # base64 от "password"
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: elasticsearch-config
  namespace: default
data:
  elasticsearch.yml: |
    cluster.name: "elasticsearch-cluster"
    network.host: 0.0.0.0
    discovery.type: single-node
    xpack.security.enabled: false #отключение базовой аутентификации
---
apiVersion: apps/v1
kind: Deployment # Deployment
metadata:
  name: elasticsearch
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: elasticsearch
  template:
    metadata:
      labels:
```

```
        app: elasticsearch
    spec:
      containers:
        - name: elasticsearch
          image:              docker.elastic.co/elasticsearch/elasticsearch:7.10.0
#фиксирование версии
          env:
            - name: discovery.type
              value: "single-node"
            - name: ELASTIC_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: elasticsearch-secret
                  key: elastic-password
          volumeMounts:
            - name: config-volume
              mountPath: /usr/share/elasticsearch/config/elasticsearch.yml
              subPath: elasticsearch.yml
          ports:
            - containerPort: 9200
      volumes:
        - name: config-volume
          configMap:
            name: elasticsearch-config
---
apiVersion: v1
kind: Service
metadata:
  name: elasticsearch-service
  namespace: default
spec:
  selector:
    app: elasticsearch
  ports:
    - protocol: TCP
      port: 9200
      targetPort: 9200
---
apiVersion: networking.k8s.io/v1
kind: Ingress #маршрутизатор Ingress
metadata:
  name: elasticsearch-ingress
  namespace: default
spec:
  rules:
    - host: minikube.local
```

```
      http:
        paths:
          - path: /search
            pathType: Prefix
            backend:
              service:
                name: elasticsearch-service
                port:
                  number: 9200
```

Рис. 26 - elascticsearch.yaml

Далее запустим minikube (Рис. 27) и применим созданный yaml файл (Рис. 28).

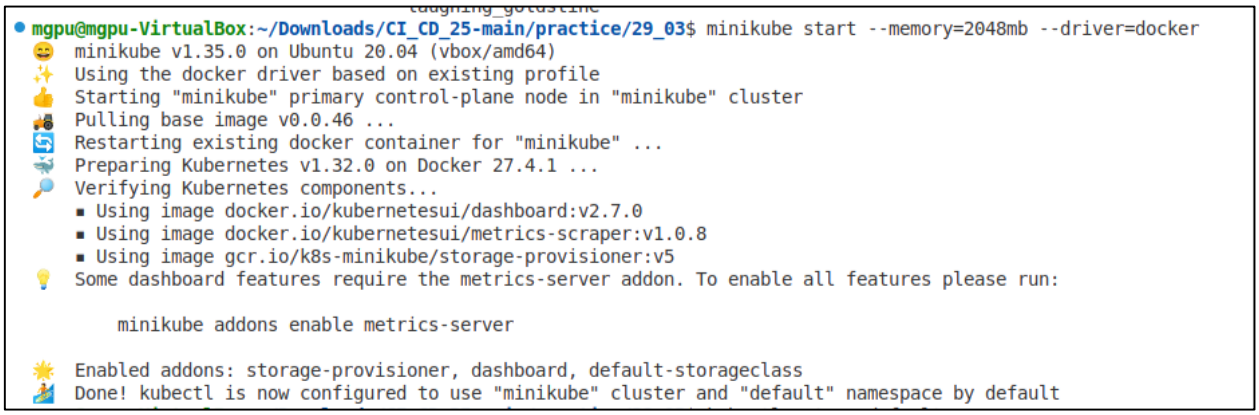

Рис. 27 – Запуск minikube

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/29_03$ kubectl apply -f elasticsearch.yaml
secret/elasticsearch-secret unchanged
configmap/elasticsearch-config unchanged
deployment.apps/elasticsearch created
service/elasticsearch-service unchanged
ingress.networking.k8s.io/elasticsearch-ingress unchanged
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/29_03$ kubectl get pods -n default
NAME                                READY   STATUS              RESTARTS       AGE
elasticsearch-d7647f7fd-25dwl       0/1     ContainerCreating   0              81s
fastapi-deployment-cf4dc69bc-pd8rt  1/1     Running             9 (9m4s ago)   3d9h
fastapi-deployment-cf4dc69bc-rt42p  1/1     Running             9 (9m4s ago)   3d9h
redis-deployment-748ffbc5f5-sh75b   1/1     Running             8 (9m5s ago)   3d9h
```

Рис. 28 – Применение манифеста

Проверим все поды, в итоге elasticsearch запущен (Рис. 29).

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/29_03$ kubectl get pods -n default
NAME                                READY   STATUS    RESTARTS      AGE
elasticsearch-d7647f7fd-25dwl       1/1     Running   0             2m35s
fastapi-deployment-cf4dc69bc-pd8rt  1/1     Running   9 (10m ago)   3d9h
fastapi-deployment-cf4dc69bc-rt42p  1/1     Running   9 (10m ago)   3d9h
redis-deployment-748ffbc5f5-sh75b   1/1     Running   8 (10m ago)   3d9h
```

Рис. 29 – Список подов в текущем пространстве

Следующий шаг – необходимо пробросить порты с помощью команды, представленной на Рисунке 30 для дальнейшего доступа.

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/29_03$ kubectl port-forward svc/elasticsearch-service 9200
:9200
Forwarding from 127.0.0.1:9200 -> 9200
Forwarding from [::1]:9200 -> 9200
Handling connection for 9200
Handling connection for 9200
Handling connection for 9200
```

Рис. 30 – Пробрасывание порта

Теперь можно проверять доступ к localhost:9200. Это можно сделать посредством команды (Рис. 31) или открыв localhost:9200 в браузере (Рис. 32). В результате elasticsearch успешно поднят.

```
mgpu@mgpu-VirtualBox:~/Downloads/CI_CD_25-main/practice/29_03$ curl localhost:9200
{
  "name" : "elasticsearch-fc8d59584-5qx28",
  "cluster_name" : "elasticsearch-cluster",
  "cluster_uuid" : "jScIRT07Sia5_JU4MlqShw",
  "version" : {
    "number" : "7.10.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "51e9d6f22758d0374a0f3f5c6e8f3a7997850f96",
    "build_date" : "2020-11-09T21:30:33.964949Z",
    "build_snapshot" : false,
    "lucene_version" : "8.7.0",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
```

Рис. 31 – Проверка доступности в терминале

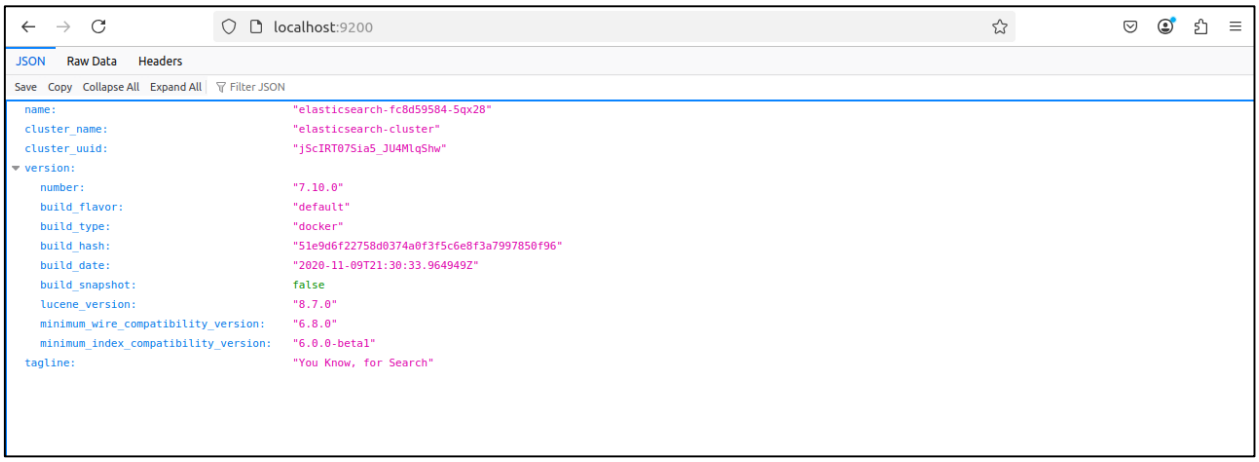

Рис. 32 – Проверка доступности в браузере

3. Напишите команды kubectl для контейнера:

- Выполнить внутри контейнера команду ps aux (Рис. 33).

Данная команда отображает все запущенные процессы в контейнере. В качестве интересующего пода выберем elasticsearch и просмотрим его процессы.

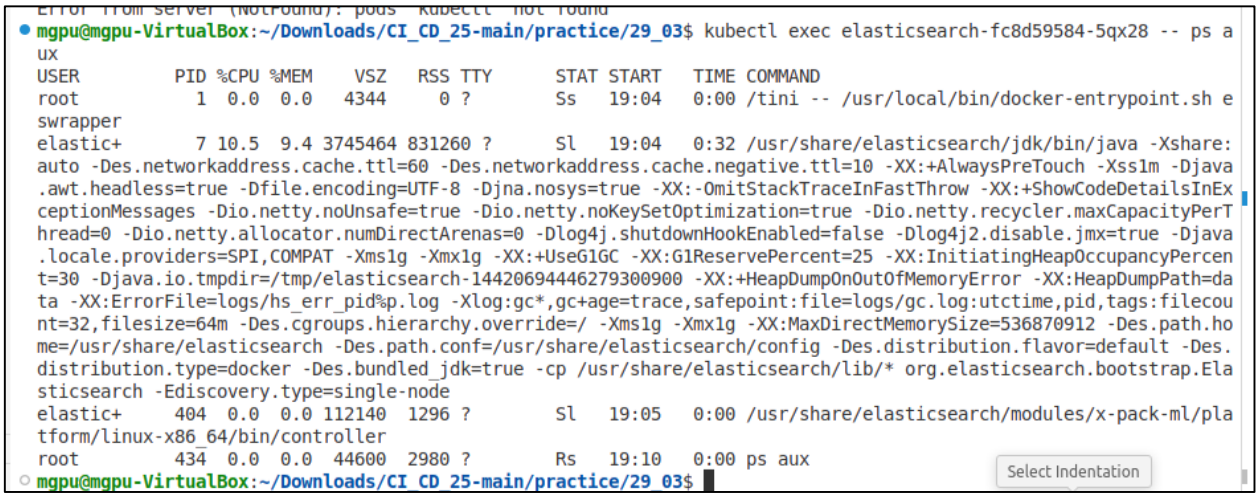

Рис. 33 – ps aux

- Просмотреть логи за 5 минут (Рис. 34).

Вызвать логи за определенный период времени можно с помощью команды `kubectl logs <имя пода> --since=<временной период>`. Исходя из рисунка 34 видно, что в последние 5 минут не было никаких логов.

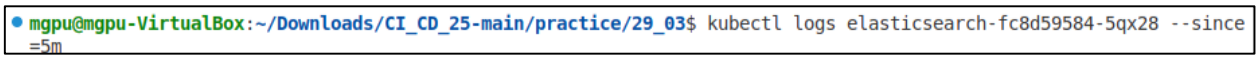

Рис. 34 – Логи за 5 минут

Попробуем установить временной интервал больше, например, 10 минут, чтобы убедиться в работоспособности команды. Команда выполнена успешно (Рис. 35).

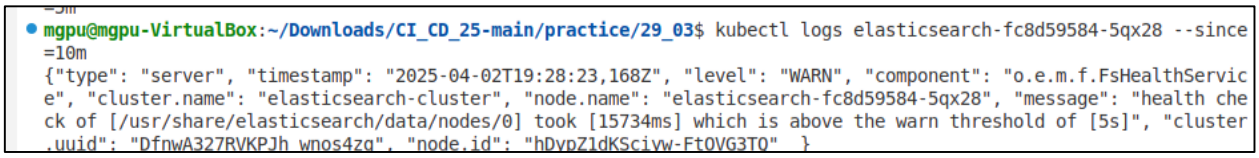

Рис. 35 – Логи за 10 минут

- Удалить pod.

Удалим под (Рис. 36), однако если Pod управляется Deployment, он автоматически пересоздастся после удаления (Рис. 36).

Рис. 36 – Удаление Pod

Чтобы удалить Pod и предотвратить его пересоздание, нужно удалить Deployment с помощью команды `kubectl delete deployment elasticsearch` (Рис. 37).

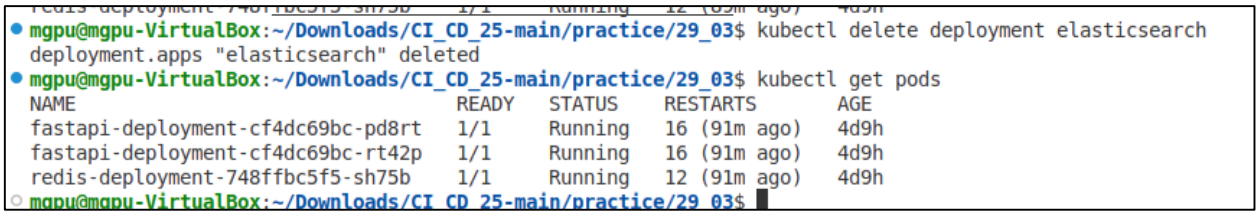

Рис. 37 –Удаление Deployment

- Пробросить порт для отладки – Выполнено в предыдущем задании (см. Рис. 30)

4. Доп. задание*: Создайте YAML для: ConfigMap с настройками для elasticsearch; Deployment, использующий ConfigMap; Ingress, направляющий запросы по пути /search на сервис.

ConfigMap, Deployment и Ingress их конфигурация представлена в листинге elasricsearch.yaml (см. Рис. 26).

Также Ingress требует дополнительного запуска для маршрутизации. Включим Ingress (Рис. 38).

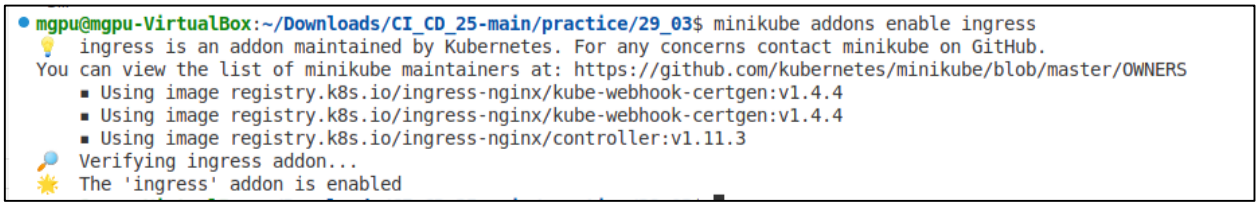

Рис. 38 – Запуск Ingress

Далее узнаем ip minikube, для того, чтобы добавить его в /etc/hosts (Рис. 39).

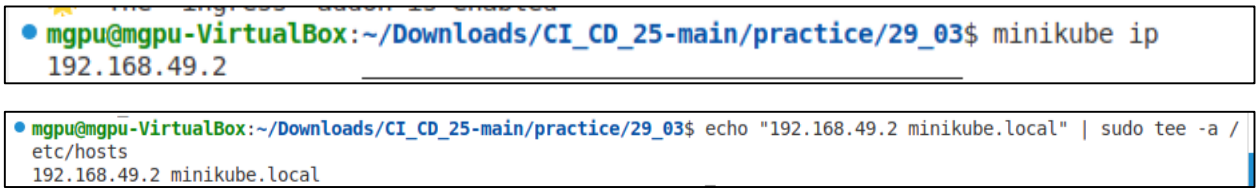

Рис. 39 – Добавление ip

После Elasticsearch будет доступен через http://minikube.local/search (Рис. 40).

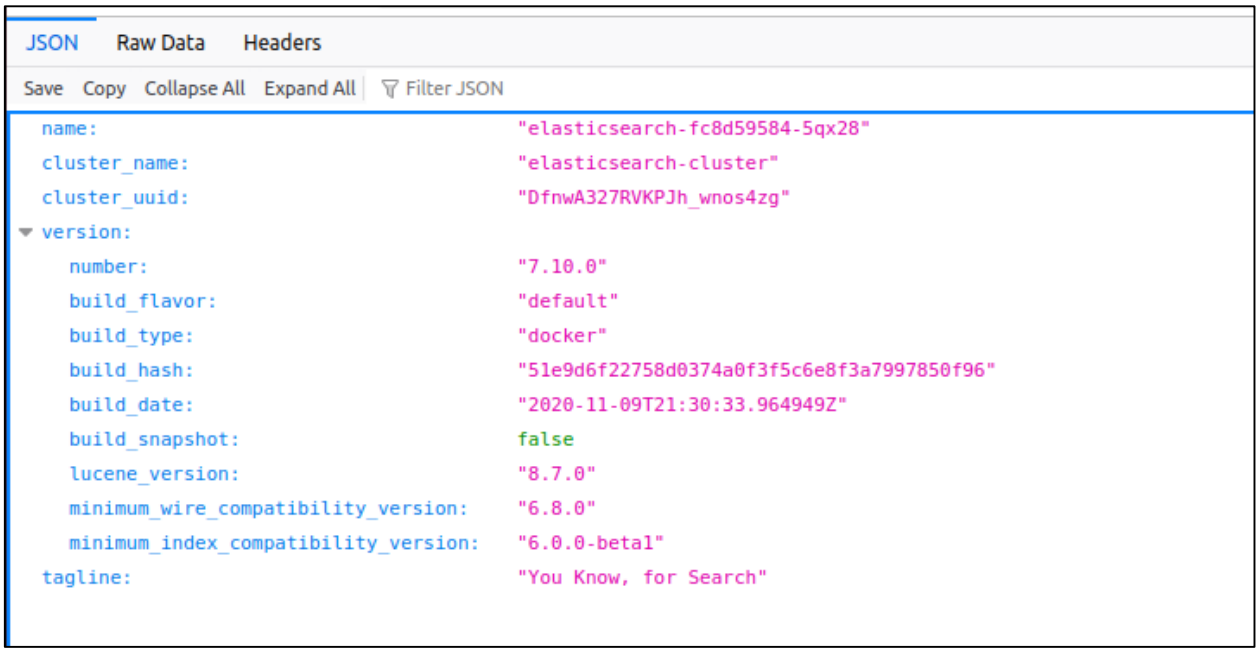

Рис. 40 – Организация доступа к Elasticsearch

Вывод: таким образах, в рамках работы был получен опыт установки и запуска minikube. Также приобретены знания в части базовых команд kubectl. Был настроен yaml файл. Все задания были выполнены.