



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения

**ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №5**

по дисциплине

«Технологии виртуализации клиент-серверных приложений»

Выполнил студент группы ИКБО-01-22

Прокопчук Р.О.

Принял преподаватель кафедры ИиППО

Волков М.Ю.

Практические работы выполнены

«__»_____2025 г.

«Зачтено»

«__»_____2025 г.

Москва
2025

Теоретическое введение

Kubernetes — это система с открытым исходным кодом для развертывания, масштабирования и управления контейнеризованными приложениями.

Kubernetes, по сути, является не просто системой оркестрации. Технически оркестрация является выполнением определенного рабочего процесса: сначала сделай А, затем В, затем С.

Kubernetes же устраняет прямую необходимость в этом. В нем есть процессы управления, которые независимы и компонуемы. Главная задача процессов управления перевести текущее состояние к нужному следующему состоянию. Теперь нам неважно какой будет маршрут от А до С, что исключает централизованный контроль.

Благодаря этому система теперь более простая в использовании, мощная, надежная, а также устойчивая и расширяемая.

Kubernetes предоставляет:

— Быструю и автоматическую масштабируемость. При росте нагрузки можно быстро добавить необходимые узлы приложения, а также быстро их вывести, чтобы не тратить драгоценные ресурсы.

— Гибкий подход к эксплуатации. Мы можем быстро и легко построить структуру приложения, так как вся структура описывается в конфигурационных файлах — манифестах.

— Гибкий подход в управлении. Kubernetes не потребует перестройки инфраструктуры и прочего, если вы захотели провести тестирование, внедрить новый сервис или сделать деплой по методологии blue-green.

— Универсальность. С помощью манифестов легко переехать, если вы захотели поменять провайдера или переезжаете в свой собственный кластер.

— Низкий порог вхождения в использование. Kubernetes довольно легок в освоении манифестов, потому что большую часть работы он делает за вас.

Сам кластер K8S состоит из рабочих узлов. В узлах или нодах (Nodes, Worker nodes), помимо контейнеров компонентов самого кластера, размещаются контейнеры наших проектов и сервисов.

Minikube — это инструмент, позволяющий легко запускать Kubernetes на локальной машине. Для тестирования Kubernetes на локальной машине это является хорошим вариантом, потому что он запускает одноузловой кластер Kubernetes внутри виртуальной машины (VM) на компьютере пользователя.

Kubectl — это инструмент командной строки для управления кластерами Kubernetes. kubectl ищет файл config в директории \$HOME/. kube. Вы можете указать другие файлы kubeconfig, установив переменную окружения KUBECONFIG или флаг --kubeconfig.

Постановка задачи

Вам необходимо выполнить все указанные в задании пункты и отразить в отчете в формате снимков экрана.

Для начала работы необходимо установить и запустить minikube в соответствии с установленной ОС с официального сайта Kubernetes: <https://kubernetes.io/ru/docs/tasks/tools/install-minikube/>

Необходимо создать deployment при помощи файла deployment.yaml используя локальный docker образ с сервером:

- название deployment: Фамилия-НомерГруппы (ivanov-ikbo-99- 99)

- используемый образ: Фамилия-НомерГруппы-Образ (ivanovikbo-99-99-obraz).

Необходимо посмотреть информацию о Deployment при помощи команды: `kubectl get deployments`

Далее необходимо посмотреть информацию о поде при помощи команды: `kubectl get pods`

После этого нужно посмотреть события кластера при помощи команды: `kubectl get events`

Затем необходимо посмотреть `kubectl` конфигурацию при помощи команды: `kubectl config view`

Потом нужно сделать под с deployment Фамилия-НомерГруппы доступным для публичной сети Интернет с помощью команды `kubectl expose`:

- сервис должен быть виден вне кластера;

- порт: 8080. После чего необходимо посмотреть только что созданный сервис `kubectl get service`

Далее нужно запустить сервис `hello-node minikube service Ivanov-ikbo-99-99`

Затем требуется отобразить текущие поддерживаемые дополнения и включить дополнение, например `ingress: minikube addons enable ingress`

После этого нужно посмотреть Pod и Service, которые вы только что создали и отключить ingress.

После отключения необходимо включить dashboard.

Далее откройте во вкладке deployments созданный под и опишите в отчете отображаемые параметры.

После выполнения освободите ресурсы созданного вами кластера и остановите Minikube.

Ход работы

Ход выполнения работ представлен на рисунках 1-18.

```
@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube start --driver=docker
minikube v1.37.0 на Microsoft Windows 11 Iot Enterprise Ltsc 2024 10.0.26100.4061 Build 26100.4061
Используется драйвер docker на основе конфига пользователя
Using Docker Desktop driver with root privileges
Starting "minikube" primary control-plane node in "minikube" cluster
Pulling base image v0.0.48 ...
Creating docker container (CPUs=2, Memory=16300MB) ...
! Failing to connect to https://registry.k8s.io/ from inside the minikube container
💡 To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
Подготавливается Kubernetes v1.34.0 на Docker 28.4.0 ...
Configuring bridge CNI (Container Networking Interface) ...
Компоненты Kubernetes проверяются ...
  ▪ Используется образ gcr.io/k8s-minikube/storage-provisioner:v5
Включенные дополнения: storage-provisioner, default-storageclass

! C:\Program Files\ Docker\ Docker\resources\bin\kubectl.exe is version 1.31.4, which may have incompatibilities with Kubernetes 1.34.0.
  ▪ Want kubectl v1.34.0? Try 'minikube kubectl -- get pods -A'
Готово! kubectl настроен для использования кластера "minikube" и "default" по умолчанию
```

Рисунок 1 – Запуск minikube

```
@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# kubectl get nodes
NAME          STATUS    ROLES          AGE   VERSION
minikube      Ready     control-plane  60s   v1.34.0
```

Рисунок 2 – Проверка статуса minikube

```

const http : {...} = require('http');

const handleRequest = (req, res) :void => { Show usages new *
  console.log('Received request for ' + req.url);
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello from the Kubernetes practice server!\n');
};

const PORT : number = 8080;
const HOST : string = '0.0.0.0';

const server : Server<IncomingMessage, ServerResponse> = http.createServer(handleRequest);
server.listen(PORT, HOST, backlog: () :void => { new *
  console.log(`Server running at http://${HOST}:${PORT}/`);
});

```

Рисунок 3 – Файл server.js

```

FROM node:25-alpine
WORKDIR /app
COPY server.js /app/
EXPOSE 8080
CMD ["node", "server.js"]

```

Рисунок 4 – Dockerfile

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: prokopchuk-ikbo-01-22
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-kubernetes
  template:
    metadata:
      labels:
        app: hello-kubernetes
    spec:
      containers:
        - name: hello-kubernetes
          image: prokopchuk-ikbo-01-22-obraz:v1
          imagePullPolicy: IfNotPresent
          ports:
            - containerPort: 8080

```

Рисунок 5 – Файл deployment.yaml


```

@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# & minikube -p minikube docker-env --shell powershell | Invoke-Expression
@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# docker build -t prokopchuk-ikbo-01-22-obraz:v1 .
[+] Building 7.0s (9/9) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile              0.0s
=> => transferring dockerfile: 134B                               0.0s
=> [internal] load metadata for docker.io/library/node:25-alpine 2.4s
=> [auth] library/node:pull token for registry-1.docker.io       0.0s
=> [internal] load .dockerignore                                 0.0s
=> => transferring context: 2B                                     0.0s
=> [1/3] FROM docker.io/library/node:25-alpine@sha256:bce79c648e05d584ad9 4.3s
=> => resolve docker.io/library/node:25-alpine@sha256:bce79c648e05d584ad9 0.0s
=> => sha256:32719b16b76b0983064be14ced286bf3d9f9dc22cfd2 6.42kB / 6.42kB 0.0s
=> => sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc 3.80MB / 3.80MB 0.7s
=> => sha256:6d0bdc936f33e7a2e7faae4e44192a62c33e1c9f1f 54.99MB / 54.99MB 2.4s
=> => sha256:4350616372df2de9f6b15ad5b0bfc771662215b9192d 1.26MB / 1.26MB 1.3s
=> => sha256:bce79c648e05d584ad9ae2b45ed663ae6f07ebfa9e5f 3.87kB / 3.87kB 0.0s
=> => sha256:2e436d5d07062d2217a45fb43bc0363a1bfcab4965b6 1.72kB / 1.72kB 0.0s
=> => extracting sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db1 0.1s
=> => sha256:d9f0123823b504c41373c028b427fcf56cf338c454bf4291 443B / 443B 1.1s
=> => extracting sha256:6d0bdc936f33e7a2e7faae4e44192a62c33e1c9f1fe253e3f 1.7s
=> => extracting sha256:4350616372df2de9f6b15ad5b0bfc771662215b9192defaeb 0.0s
=> => extracting sha256:d9f0123823b504c41373c028b427fcf56cf338c454bf4291a 0.0s
=> [internal] load build context                                0.0s
=> => transferring context: 488B                                   0.0s
=> [2/3] WORKDIR /app                                          0.2s
=> [3/3] COPY server.js /app/                                  0.0s
=> exporting to image                                          0.0s
=> => exporting layers                                             0.0s
=> => writing image sha256:4dc9fcaad2d7329c982d9c885a8c1fdad827983081edfb 0.0s
=> => naming to docker.io/library/prokopchuk-ikbo-01-22-obraz:v1 0.0s

View build details: docker-desktop://dashboard/build/default/default/y7b4hy919xs6tb9iqnl4q3ajm

```

Рисунок 6 – Сборка Docker-образа сервиса

```

# kubectl apply -f deployment.yaml
deployment.apps/prokopchuk-ikbo-01-22 created
@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# kubectl get deployments
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
prokopchuk-ikbo-01-22  1/1      1              1             8s
@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# kubectl get pods -o wide
NAME                                READY    STATUS    RESTARTS    AGE    IP
NODE    NOMINATED NODE    READINESS GATES
prokopchuk-ikbo-01-22-6bf7fd679d-bklh7  1/1      Running    0            24s    10.24
4.0.3   minikube    <none>      <none>

```

Рисунок 7 – Создание deployment и просмотр информации о нем


```

▲ # kubectl get events
LAST SEEN   TYPE      REASON              OBJECT
MESSAGE
5m58s       Warning   PossibleMemoryBackedVolumesOnDisk  node/minikube
The tmpfs noswap option is not supported. Memory-backed volumes (e.g. secrets, emptyDirs, etc.) might be swapped to disk and should no longer be considered secure.
5m58s       Normal    Starting            node/minikube
Starting kubelet.
5m58s       Warning   CgroupV1            node/minikube
cgroup v1 support is in maintenance mode, please migrate to cgroup v2
5m58s       Normal    NodeAllocatableEnforced  node/minikube
Updated Node Allocatable limit across pods
5m58s       Normal    NodeHasSufficientMemory  node/minikube
Node minikube status is now: NodeHasSufficientMemory
5m58s       Normal    NodeHasNoDiskPressure    node/minikube
Node minikube status is now: NodeHasNoDiskPressure
5m58s       Normal    NodeHasSufficientPID      node/minikube
Node minikube status is now: NodeHasSufficientPID
5m54s       Normal    RegisteredNode          node/minikube
Node minikube event: Registered Node minikube in Controller
5m52s       Normal    Starting            node/minikube
90s        Normal    Scheduled            pod/prokopchuk-ikbo-01-22-6bf7fd679d-bklh7
Successfully assigned default/prokopchuk-ikbo-01-22-6bf7fd679d-bklh7 to minikube
90s        Normal    Pulled              pod/prokopchuk-ikbo-01-22-6bf7fd679d-bklh7
Container image "prokopchuk-ikbo-01-22-obraz:v1" already present on machine
90s        Normal    Created            pod/prokopchuk-ikbo-01-22-6bf7fd679d-bklh7
Created container: hello-kubernetes
90s        Normal    Started            pod/prokopchuk-ikbo-01-22-6bf7fd679d-bklh7
Started container hello-kubernetes
90s        Normal    SuccessfulCreate    replicaset/prokopchuk-ikbo-01-22-6bf7fd679d
Created pod: prokopchuk-ikbo-01-22-6bf7fd679d-bklh7
90s        Normal    ScalingReplicaSet    deployment/prokopchuk-ikbo-01-22
Scaled up replica set prokopchuk-ikbo-01-22-6bf7fd679d from 0 to 1

```

Рисунок 8 – Просмотр событий кластера

```

@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
▲ # kubectl expose deployment prokopchuk-ikbo-01-22 --type=NodePort --port=8080
service/prokopchuk-ikbo-01-22 exposed
@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
▲ # kubectl get services
NAME                                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)
AGE
kubernetes                          ClusterIP    10.96.0.1     <none>         443/TCP
6m36s
prokopchuk-ikbo-01-22              NodePort     10.108.37.4   <none>         8080:32402/TCP
8s

```

Рисунок 9 – Expose пода с deploment

```

Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube service prokopchuk-ikbo-01-22

```

NAMESPACE	NAME	TARGET PORT	URL
default	prokopchuk-ikbo-01-22	8080	http://192.168.49.2:32402

```

Starting tunnel for service prokopchuk-ikbo-01-22.

```

NAMESPACE	NAME	TARGET PORT	URL
default	prokopchuk-ikbo-01-22		http://127.0.0.1:49538

```

Starting tunnel for service prokopchuk-ikbo-01-22.
Opening service default/prokopchuk-ikbo-01-22 in default browser...
! Because you are using a Docker driver on windows, the terminal needs to be open to run it.

```

Рисунок 10 – Запуск сервиса

```

Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube addons list

```

ADDON	NAME	PROFILE	STATUS	MAINTAINER
ambassador		minikube	disabled	3rd party (Ambassador)
amd-gpu-device-plugin		minikube	disabled	3rd party (AMD)
auto-pause		minikube	disabled	minikube
cloud-spanner		minikube	disabled	Google
csi-hostpath-driver		minikube	disabled	Kubernetes
dashboard		minikube	enabled ✓	Kubernetes
default-storageclass		minikube	enabled ✓	Kubernetes
efk		minikube	disabled	3rd party (Elastic)
freshpod		minikube	disabled	Google
gcp-auth		minikube	disabled	Google
gvisor		minikube	disabled	minikube
headlamp		minikube	disabled	3rd party (kinvolk.io)
inacel		minikube	disabled	3rd party (InAccel [info@inacel.com])
ingress		minikube	disabled	Kubernetes
ingress-dns		minikube	disabled	minikube
inspektor-gadget		minikube	disabled	3rd party (inspektor-gadget.io)
istio		minikube	disabled	3rd party (Istio)
istio-provisioner		minikube	disabled	3rd party (Istio)
kong		minikube	disabled	3rd party (Kong HQ)
kubeflow		minikube	disabled	3rd party
kubetail		minikube	disabled	3rd party (kubetail.com)
kubevirt		minikube	disabled	3rd party (KubeVirt)
logviewer		minikube	disabled	3rd party (unknown)
metallb		minikube	disabled	3rd party (MetalLB)
metrics-server		minikube	disabled	Kubernetes
nvidia-device-plugin		minikube	disabled	3rd party (NVIDIA)
nvidia-driver-installer		minikube	disabled	3rd party (NVIDIA)
nvidia-gpu-device-plugin		minikube	disabled	3rd party (NVIDIA)

Рисунок 11 – Вывод списка дополнений


```

@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube addons enable ingress
  ingress is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.
  You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS
  After the addon is enabled, please run "minikube tunnel" and your ingress resources would be available at "127.0.0.1"
    ▪ Используется образ registry.k8s.io/ingress-nginx/controller:v1.13.2
    ▪ Используется образ registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.6.2
    ▪ Используется образ registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.6.2
  Verifying ingress addon...
  The 'ingress' addon is enabled

```

Рисунок 12 – Включение ingress

```

@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# kubectl get pods,svc -n kube-system

```

NAME	READY	STATUS	RESTARTS	AGE
pod/coredns-66bc5c9577-t2x5s	1/1	Running	0	17m
pod/etcd-minikube	1/1	Running	0	17m
pod/kube-apiserver-minikube	1/1	Running	0	17m
pod/kube-controller-manager-minikube	1/1	Running	0	17m
pod/kube-proxy-wcgzd	1/1	Running	0	17m
pod/kube-scheduler-minikube	1/1	Running	0	17m
pod/storage-provisioner	1/1	Running	0	17m

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
service/kube-dns	ClusterIP	10.96.0.10	<none>	53/UDP,53/TCP,9153/TCP

Рисунок 13 – Вывод подов

```

@ Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube addons disable ingress
  "Дополнение 'ingress' выключено"

```

Рисунок 14 – Выключение ingress

```

Remsely on ~/dev/mirea/csavt/practice-5-kubernetes main
# minikube addons enable dashboard
💡 dashboard is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS
  ▪ Используется образ docker.io/kubernetesui/dashboard:v2.7.0
  ▪ Используется образ 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

☀ The 'dashboard' addon is enabled
Remsely on ~/dev/mirea/csavt/practice-5-kubernetes main
# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
prokopchuk-ikbo-01-22-6bf7fd679d-bklh7  1/1     Running   0           8m12s
Remsely on ~/dev/mirea/csavt/practice-5-kubernetes main
# minikube dashboard
🤖 Verifying dashboard health ...
🚀 Launching proxy ...
🤖 Verifying proxy health ...
🌐 Opening http://127.0.0.1:49714/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...

```

Рисунок 15 – Запуск Dashboard

The screenshot shows the Kubernetes Dashboard interface. The breadcrumb navigation indicates the path: Workloads > Deployments > **prokopchuk-ikbo-01-22**. The left sidebar contains a navigation menu with categories like Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, Services, Config and Storage, and Cluster.

The main content area displays the following information for the deployment 'prokopchuk-ikbo-01-22':

- Metadata:**
 - Name: prokopchuk-ikbo-01-22
 - Namespace: default
 - Created: Oct 26, 2025
 - Age: 18 minutes ago
 - UID: 118c27f2-9deb-442a-9dd8-939734aa8030
 - Annotations: deployment.kubernetes.io/revision: 1, kubectl.kubernetes.io/last-applied-configuration
- Resource information:**
 - Strategy: RollingUpdate
 - Min ready seconds: 0
 - Revision history limit: 10
 - Selector: app: hello-kubernetes
- Rolling update strategy:**
 - Max surge: 25%
 - Max unavailable: 25%
- Pods status:** (Section header visible)

Рисунок 16 – Deployment в запущенном Dashboard (1/2)

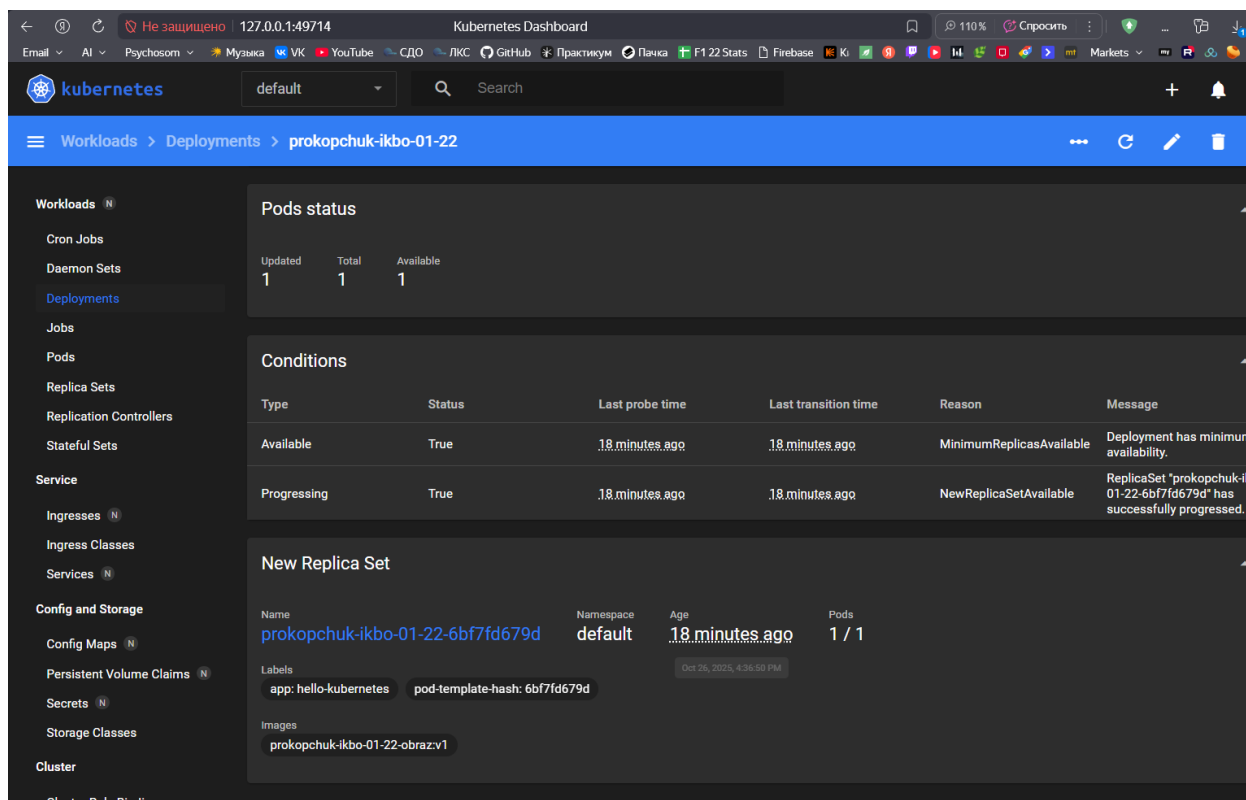


Рисунок 17 – Deployment в запущенном Dashboard (2/2)

Развертывание prokopchuk-ikbo-01-22 было создано в пространстве имен default и находится в активном состоянии. Уникальный идентификатор развертывания — 118c27f2-9deb-442a-9dd8-939734aa8030.

Для управления обновлениями приложения используется стратегия RollingUpdate, которая обеспечивает плавное обновление подов без простоя сервиса. Параметры стратегии настроены следующим образом: максимальное количество дополнительных подов при обновлении (Max surge) составляет 25%, а максимальное количество недоступных подов (Max unavailable) также ограничено 25%. Минимальное время готовности пода перед его использованием установлено в 0 секунд, а лимит истории ревизий составляет 10 версий.

Селектор приложения определен как app: hello-kubernetes, что позволяет развертыванию управлять подами с соответствующей меткой. В текущий момент развертывание содержит один реплика-сет с хешем шаблона 6bf7fd679d, использующий образ контейнера prokopchuk-ikbo-01-22-obraz:v1.

Статус подов показывает, что из одного требуемого пода один был обновлен и один доступен, что соответствует здоровому состоянию разворачивания. Условия разворачивания подтверждают достижение минимальной доступности и успешное создание нового реплика-сета. Время последней проверки и перехода в текущее состояние составляет 18 минут назад, что указывает на стабильную работу приложения.

```
Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# kubectl delete service prokopchuk-ikbo-01-22 --ignore-not-found
service "prokopchuk-ikbo-01-22" deleted
Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# kubectl delete deployment prokopchuk-ikbo-01-22 --ignore-not-found
deployment.apps "prokopchuk-ikbo-01-22" deleted
Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube addons disable dashboard
"Дополнение 'dashboard' выключено"
Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube stop
Узел "minikube" останавливается ...
Выключается "minikube" через SSH ...
Остановлено узлов: 1.
Remsely on ~ /dev/mirea/csavt/practice-5-kubernetes main
# minikube delete
Deleting "minikube" in docker ...
Deleting container "minikube" ...
Removing C:\Users\Remsely\.minikube\machines\minikube ...
Removed all traces of the "minikube" cluster.
```

Рисунок 18 – PUT-запрос для обновления студента

Вывод

В результате выполнения данной практической работы были изучены основы работы с Kubernetes и развернут deployment простого сервиса.

Ответы на вопросы к практической работе

1. Назовите виды контроллеров в Kubernetes.

ReplicaSet, Deployment, DaemonSet, StatefulSet, Job, CronJob.

Они управляют жизненным циклом подов: сколько их нужно, где запускать и когда перезапускать.

2. Как называется командная строка в Kubernetes?

Командная строка называется **kubectl** (произносится как *cube control* или *cube cuddle*).

3. Что такое под?

Под — это минимальная единица развертывания в Kubernetes, содержащая один или несколько контейнеров, которые разделяют общие ресурсы (сеть, хранилище и т.п.) и всегда запускаются вместе.

4. Назовите 2 типа ресурсов, из которых состоит кластер Kubernetes.

Master (control plane) — отвечает за управление кластером (API Server, Scheduler, Controller Manager, etcd).

Worker nodes — рабочие узлы, на которых запускаются приложения (Pods, Containers).

5. Чем Kubernetes отличается от Docker Swarm?

Kubernetes — более сложная и мощная система оркестрации, обеспечивающая автоматическое масштабирование, самовосстановление, продвинутую сетевую модель и декларативные манифесты.

Docker Swarm проще, но менее функционален, ориентирован на небольшие кластеры и быстрее в настройке.

Список источников информации

1. Установка Kubernetes с помощью Minikube — Текст: электронный [сайт]. — URL: <https://kubernetes.io/ru/docs/setup/learningenvironment/minikube/>
2. K8S для начинающих. Первая часть — Текст: электронный [сайт]. — URL: <https://habr.com/ru/post/589415/>
3. Kubernetes или с чего начать, чтобы понять что это и зачем он нужен — Текст: электронный [сайт]. — URL: <https://habr.com/ru/company/otus/blog/537162/>
4. Основы Kubernetes — Текст: электронный [сайт]. — URL: <https://habr.com/ru/post/258443/>