|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №8**

по дисциплине «Технологии виртуализации клиент-серверных приложений»

**Студент группы** ИКБО-20-21 Мухаметшин А. Р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель практической работы**  Волков М.Ю.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Москва 2024

**ОГЛАВЛЕНИЕ**

[Цель работы 3](#_heading=h.30j0zll)

[Ход работы 5](#_heading=h.1fob9te)

[Выводы 15](#_heading=h.3znysh7)

[Ответы на вопросы к практической работе 16](#_heading=h.2et92p0)

[Список использованной литературы 18](#_heading=h.tyjcwt)

# Цель работы

Разработать Backend или часть Backend системы в соответствии с выбранной для ВКР темой с использованием Spring/Spring Boot. Темы в группах не должны повторяться. Система должна иметь сервис авторизации с хранением токенов в RedisDB и минимум 2 сервиса, которые должны общаться между собой с помощью шины сообщений с использованием Kafka. Хранение данных должно производиться с помощью СУБД PostgreSQL. В качестве API-шлюза к сервисам должен выступать KrakenD. Необходимо логировать все обращения к сервисам с помощью Graylog. В логах должны указываться HTTP-метод, URL, IP-адрес отпрsавителя. Также необходимо собирать метрики с баз данных с помощью связки Prometheus + Grafana, а все транзакции подвергать трассировке с помощью Jaeger. В случае невозможности реализации системы с использованием предложенного стека технологий необходимо обосновать это и предложить решение на собственном стеке. Для каждого сервиса необходимы:

⎯ Установка в Minikube при помощи Helm-чартов;

⎯ Настройка лимитов;

⎯ Настройка Liveness, Readiness and Startup Probes;

⎯ Настройка перезапуска при падении;

⎯ Настройка использования ресурсов пода на 60% и использование Autoscaling.

Для СУБД:

⎯ Использование persistent volume;

⎯ Конфигурация макс количества соединений;

⎯ Создание пользователя для считывания данных мониторингом;

⎯ Настройка мониторинга пода с БД при помощи Prometheus и Grafana;

⎯ Должен соблюдаться паттерн “1 БД на 1 сервис”.

В отчете вам необходимо отобразить все этапы конфигурации системы, показать конфигурационные файлы Minikibe, Helm Charts, KrakenD, отобразить работу системы с учетом авторизации, показать процесс в сервисах Kafka, Jaeger, Prometheus, Grafana, Graylog, отобразить сохранение данных в persistent volume при отключении пода с БД.

# Ход работы

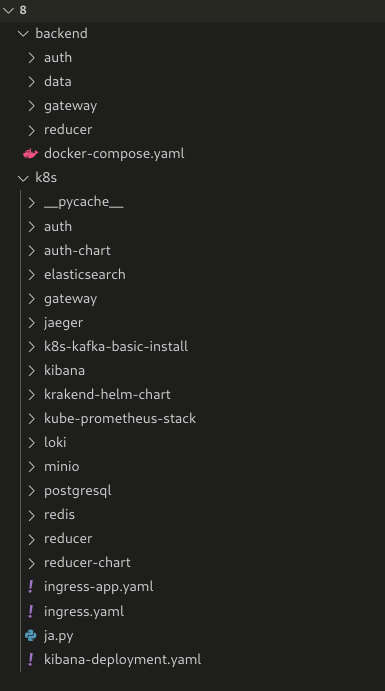


Рисунок 1 – Структура проекта

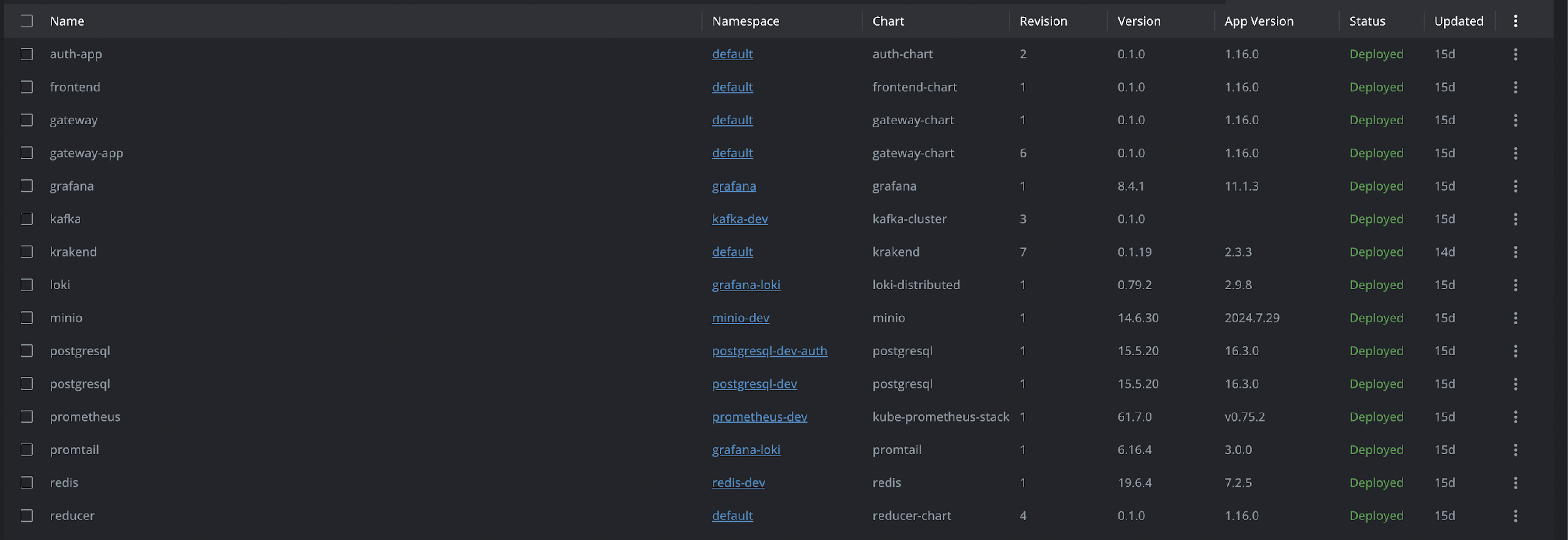


Рисунок 2 – Список используемых helm charts в кластере

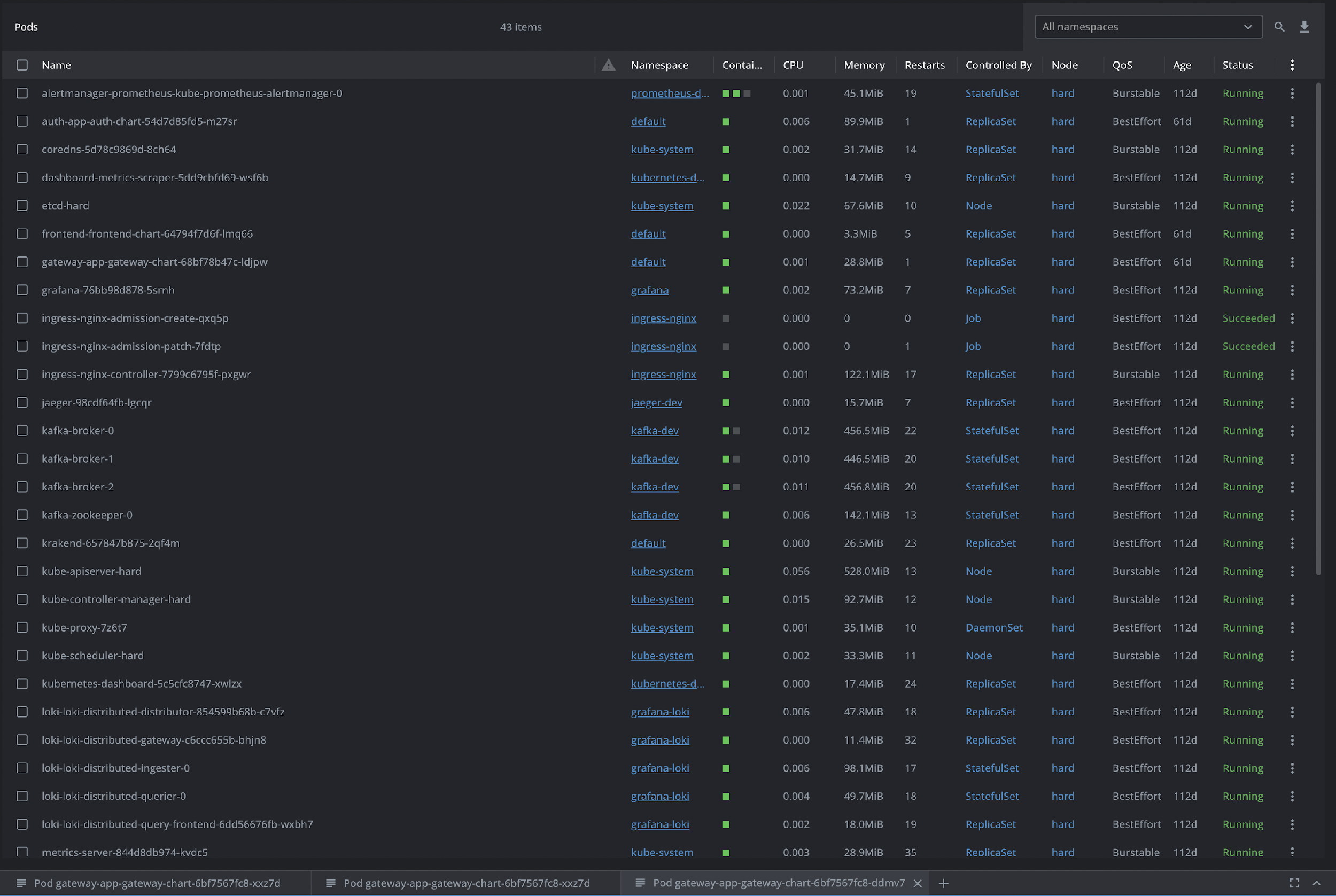
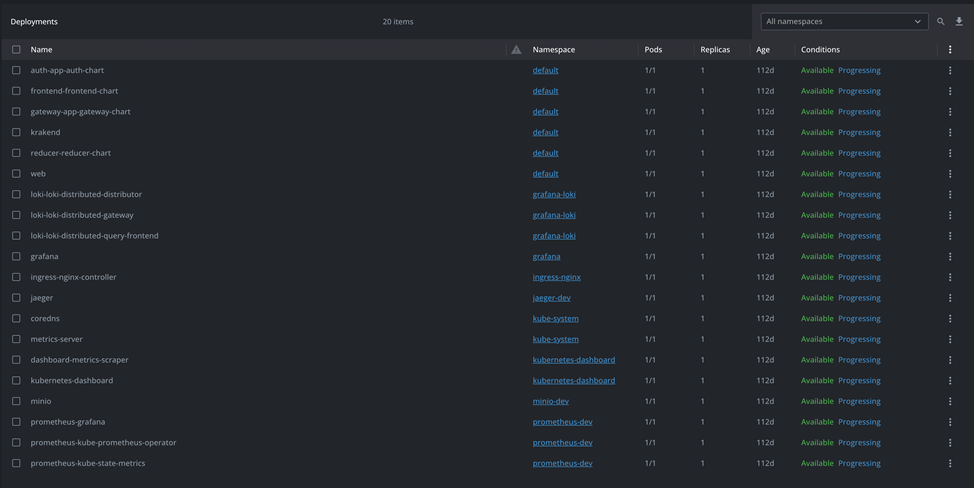


Рисунок 3 – Список развернутых pods



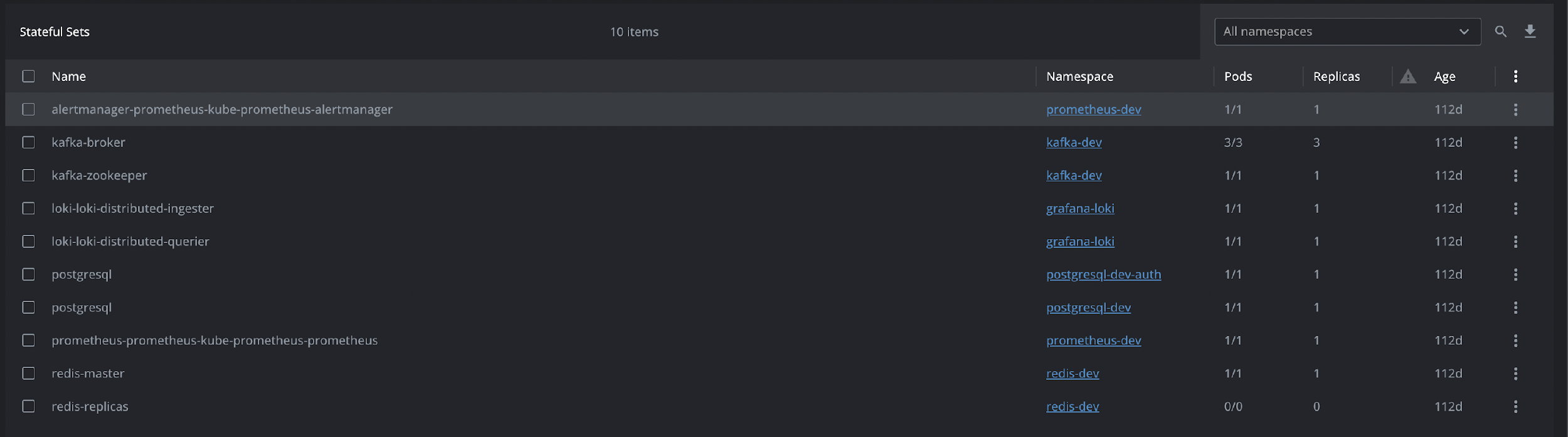
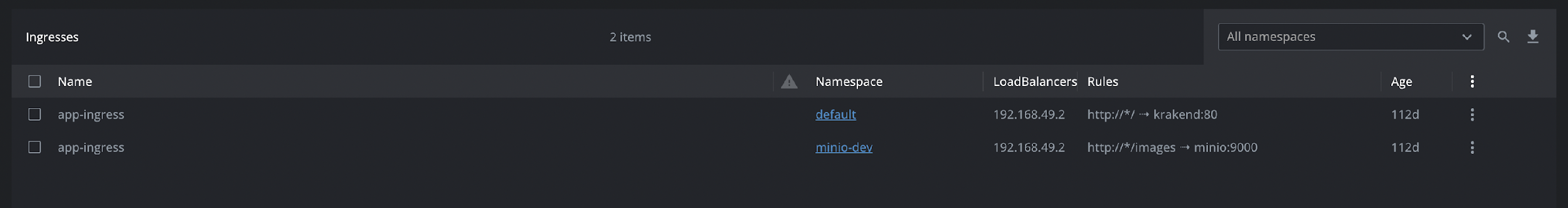


Рисунок 4 – Список развернутых deployments, statefulsets

 Рисунок 5 – Список развернутых ingress

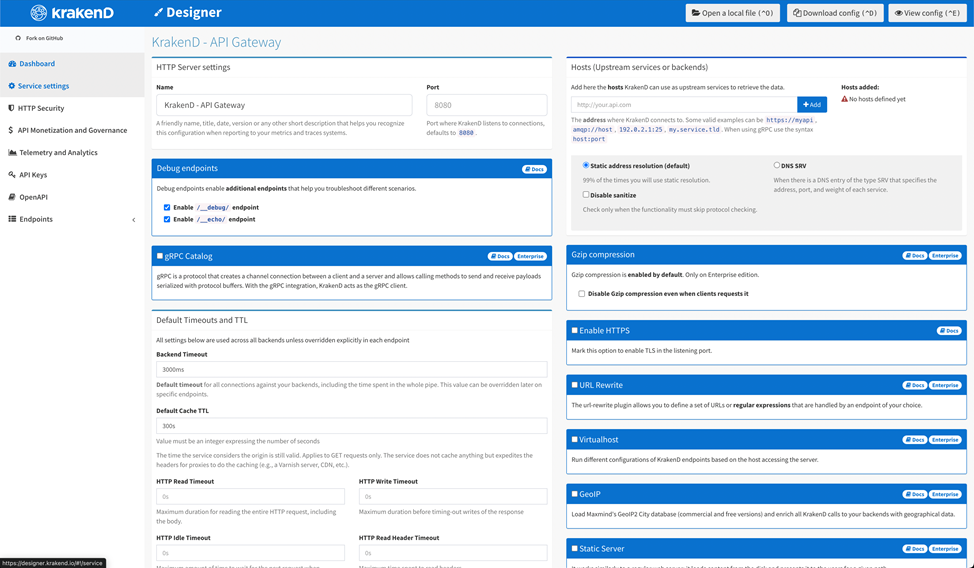


Рисунок 6 – Пример настройки krakend.json



Рисунок 7 – Пример dashboard Grafana для отображения логов, собранных через Loki

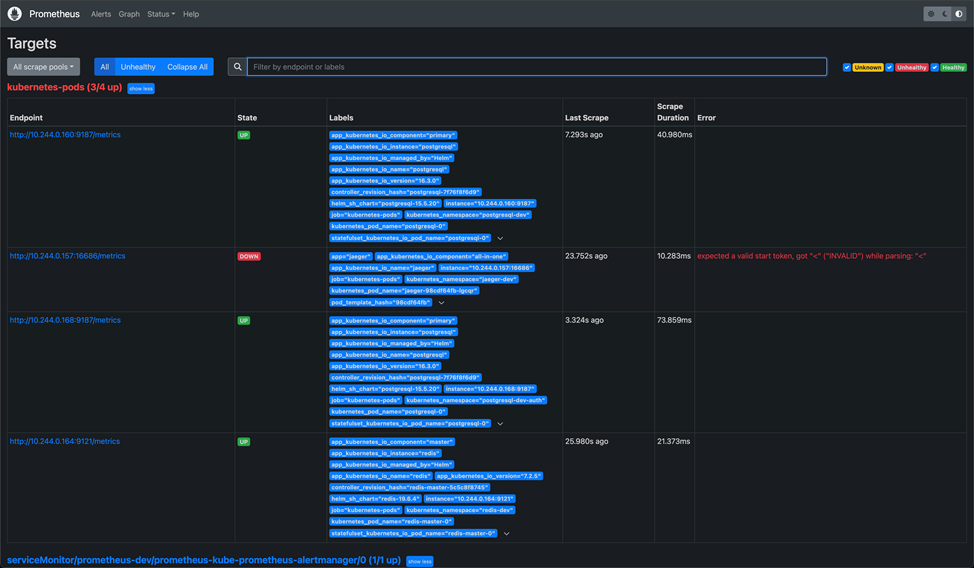


Рисунок 8 – Развернутый Prometheus server

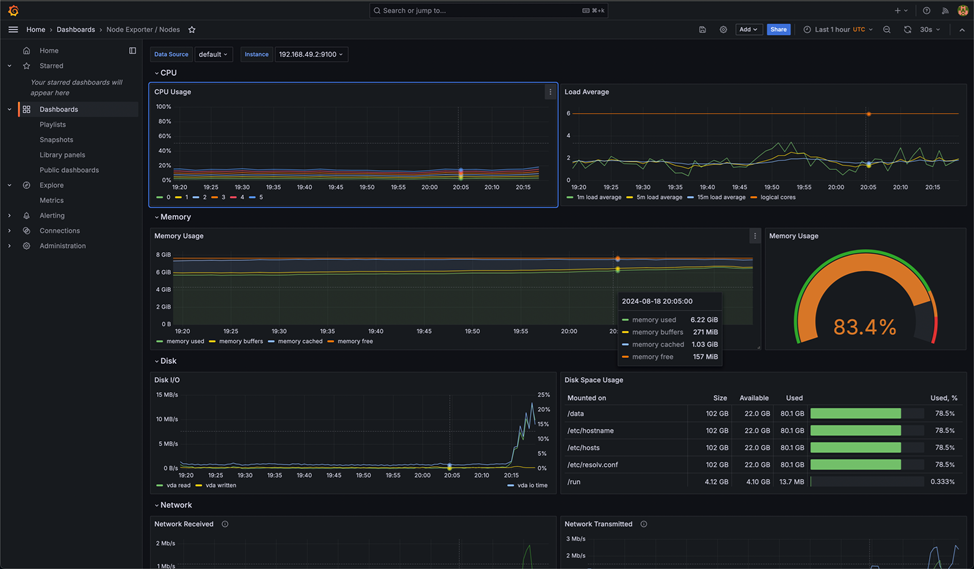


Рисунок 9 – Пример dashboard Grafana, отображающий метрики nodes

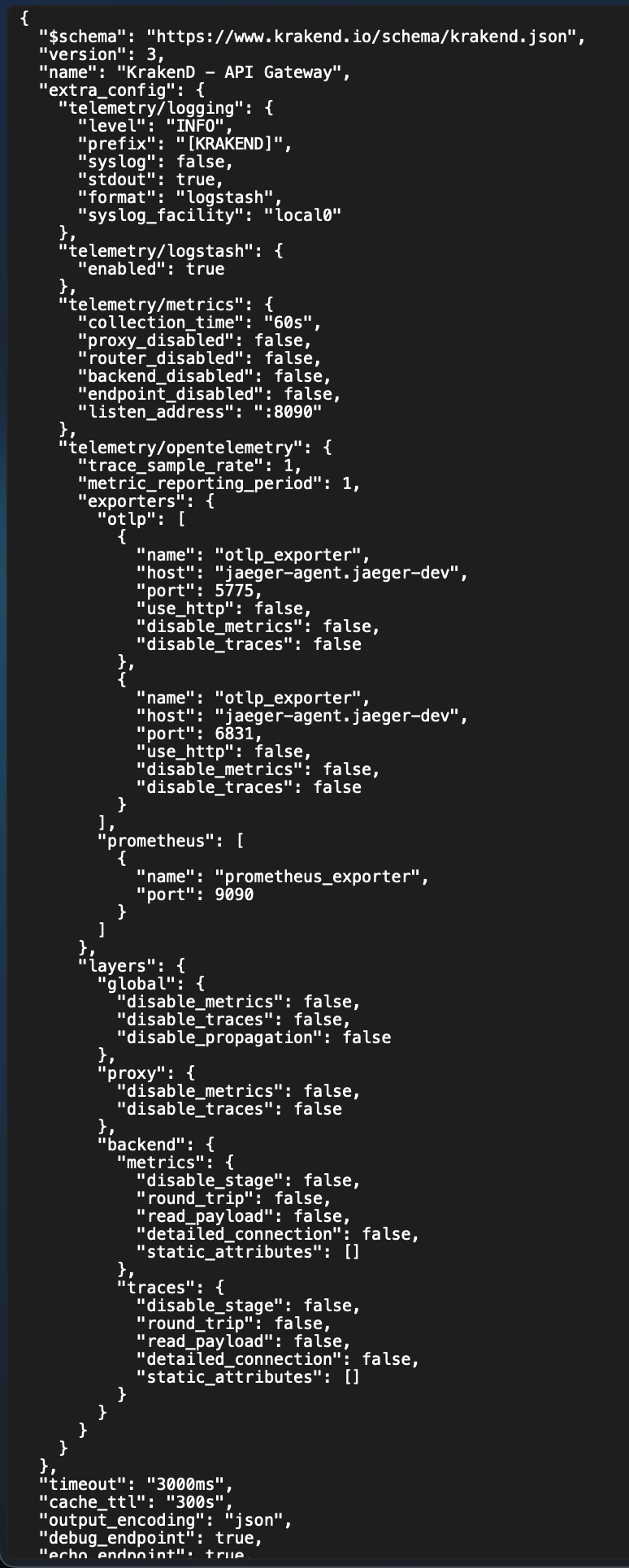


Рисунок 10 – Конфигурационный файл krakend



Рисунок 11 – Пример работы Jaeger

Изображение выглядит как программное обеспечение, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок 12 – Регистрация

Изображение выглядит как Мультимедийное программное обеспечение, программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок 13 – Логин

Изображение выглядит как программное обеспечение, текст, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок 14 – Валидация токена

# Выводы

В ходе выполнения практической работы были освоены ключевые навыки работы с системой Kubernetes. Это включает в себя развертывание типового приложения в кластере Kubernetes, а также изучение основных понятий, таких как сервисы, деплойменты, масштабирование. Было получено глубокое понимание принципов работы Kubernetes и его преимуществ в контексте развертывания и управления масштабируемыми приложениями.

# Ответы на вопросы к практической работе

1. **Назовите плюсы и минусы использования helm. Какие форматы конфигурационных файлов он поддерживает? Какие существуют аналоги?**

Использование Helm в контейнеризированных приложениях имеет свои плюсы и минусы. К плюсам можно отнести стандартизацию и упрощение управления приложениями, а также возможность использовать шаблоны для конфигурации. Однако есть и минусы, такие как усложнение процесса разработки и необходимость изучения Helm для новых пользователей. Helm поддерживает форматы конфигурационных файлов в виде YAML и JSON, а его аналогами являются такие инструменты, как Kustomize и Terraform.

1. **Почему для микросервисного взаимодействия используется шина сообщений? Какие аналоги для микросервисного взаимодействия существуют? Каковы их преимущества и недостатки?**

В микросервисной архитектуре шина сообщений применяется для асинхронного взаимодействия между сервисами, что позволяет улучшить устойчивость и масштабируемость приложения. Аналогами шины сообщений могут быть REST API и gRPC. Причем каждый из этих подходов имеет свои преимущества: шина сообщений хорошо справляется с высокой нагрузкой и разрозненностью сервисов, тогда как REST и gRPC обеспечивают более прямое взаимодействие, но могут быть более уязвимыми к сбоям.

1. **Для чего используется persistent volume в БД? Какие преимущества и недостатки у паттерна “1 БД на 1 сервис”?**

Persistent Volume для баз данных используется для обеспечения сохранности данных даже при перезапуске контейнеров, что критично для многих приложений. Однако у паттерна «1 БД на 1 сервис» есть свои плюсы, такие как упрощение управления и возможность настройки базы данных под конкретные нужды сервиса. Но могут возникнуть и недостатки, например, сложности с согласованием данных между сервисами и потенциальное увеличение накладных расходов от большого числа баз данных.

1. **Почему для чтения данных из БД системами мониторинга создается отдельный пользователь? К чему может привести использование аккаунта с лишними правами?**

Специального пользователя для систем мониторинга создают для ограничения прав доступа, что минимизирует риски. Использование аккаунта с избыточными правами может привести к случайным изменениям или утечкам данных. Опасно подвергать риску критически важные данные и операции, особенно если мониторинг не нуждается в полном доступе ко всем аспектам базы.

1. **Какой функционал предоставляет KrakenD? Какие существуют аналоги?**

KrakenD предоставляет функционал API Gateway, позволяя агрегировать запросы, управлять маршрутизацией и другими аспектами взаимодействия с микросервисами. Его аналогами являются такие инструменты, как Kong и NGINX. Каждый из них имеет свои особенности: Kong предлагает множество плагинов для интеграции, а NGINX славится высокой производительностью и гибкостью настройки, что может стать важным фактором выбора в зависимости от требований проекта.

# Список использованной литературы

1. Virtualization and Containers: An Overview [Электронный ресурс]. – URL: https://www.example.com/virtualization-containers-overview (дата обращения: 04.08.2024)
2. Docker Documentation. [Электронный ресурс]. – URL: https://docs.docker.com/ (дата обращения: 04.08.2024)
3. VMware Official Website. [Электронный ресурс]. – URL: https://www.vmware.com/ (дата обращения: 04.08.2024)
4. "Virtual Machines Versus Containers: What's the Diff?" by John Smith. Virtualization Journal, vol. 25, no. 3, 2019, pp. 45-58.
5. "Understanding the Performance Trade-Offs in Virtualization" by Jane Doe. Proceedings of the International Conference on Cloud Computing, 2022, pp. 112-120.
6. 50 вопросов по Docker, которые задают на собеседованиях, и ответы на них | Хабр. — Текст: электронный [сайт]. — URL: https://habr.com/ru/company/southbridge/blog/528206/
7. Docker Documentation | Docker Documentation — Текст: электронный [сайт]. — URL: https://docs.docker.com/
8. Что такое режим Docker Swarm и когда его использовать? — CloudSavvy ИТ | Cpab. — Текст: электронный [сайт]. — URL: https://cpab.ru/chtotakoe-rezhim-docker-swarm-i-kogda-ego-ispo4lzovat-cloudsavvy-it/
9. Dockerfile reference | Docker Documentation — Текст: электронный [сайт]. — URL: https://docs.docker.com/engine/reference/builder/