|  |
| --- |
|  |
| 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 №4** | |
| **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**

Docker Compose — это инструментальное средство, входящее в состав Docker. Оно предназначено для решения задач, связанных с развёртыванием проектов. Docker Compose позволяет управлять набором контейнеров, каждый из которых представляет собой один сервис проекта. Управление включает в себя сборку, запуск с учетом зависимостей и конфигурацию. Конфигурация Docker Compose описывается в файле docker-compose.yml, лежащем в корне проекта.

Для получения дополнительной информации о файле создания вы можете изучить официальную документацию Docker.

Помимо сервисов на серверах необходимы системы мониторинга, чтобы осуществлять контроль над работоспособностью системы. Для этого используются различные виды систем мониторинга:

⎯ Инфраструктурный мониторинг (Например Zabbix, Nagios)

⎯ Мониторинг ошибок программных платформ (Sentry)

⎯ Мониторинг производительности приложений (Prometheus)

⎯ Мониторинг безопасности систем (Nesus, OpenVas)

⎯ Сбор системных журналов (GrayLog)

Zabbix — это универсальный инструмент мониторинга, способный отслеживать динамику работы серверов и сетевого оборудования, быстро реагировать на внештатные ситуации и предупреждать возможные проблемы с нагрузкой. Система мониторинга Zabbix может собирать статистику в указанной рабочей среде и действовать в определенных случаях заданным образом.

У Zabbix есть 4 основных инструмента, с помощью которых можно мониторить определенную рабочую среду и собирать о ней полный пакет данных для оптимизации работы.

⎯ Сервер — ядро, хранящее в себе все данные системы, включая статистические, оперативные и конфигурацию. Дистанционно управляет сетевыми сервисами, оповещает администратора о существующих проблемах с оборудованием, находящимся под наблюдением.

⎯ Прокси — сервис, собирающий данные о доступности и производительности устройств, который работает от имени сервера. Все собранные данные сохраняются в буфер и загружаются на сервер. Нужен для распределения нагрузки на сервер. Благодаря этому процессу можно уменьшить нагрузку на процессор и жесткий диск. Для работы прокси Zabbix отдельно нужна база данных.

⎯ Агент — программа (демон), которая активно мониторит и собирает статистику работы локальных ресурсов (накопители, оперативная память, процессор и др.) и приложений.

⎯ Веб-интерфейс — является частью сервера системы и требует для работы веб-сервер. Часто запускается на том же физическом узле, что и Zabbix.

Graylog — это платформа, которая позволяет легко управлять записями структурированных и неструктурированных данных. вместе с отладкой приложений. Он основан на Elasticsearch, MongoDB и Scala.

Он имеет главный сервер, который принимает данные от своих клиентов, установленных на разных серверах, и веб-интерфейс, который отображает данные и позволяет работать с записями, добавленными основным сервером.

Graylog эффективен при работе с необработанными строками (например, с системным журналом) - инструмент анализирует их на нужные нам структурированные данные.

Основное преимущество Graylog заключается в том, что он предоставляет единый идеальный экземпляр сбора журналов для всей системы.

Prometheus — система мониторинга. Основные преимущества — предоставление возможности создания гибких запросов к данным и хранение значений метрик в базе данных временных рядов, возможность автоматизации при администрировании. Разработана фондом облачных вычислений (Cloud Native Computing Foundation или CNCF).

Для получения метрик с удаленных узлов используется метод pull (сервер сам забирает данные). На узлы для сбора информации устанавливаются экспортеры (exporter) — пакеты, получающие данные для операционной системы или конкретного сервиса. Существует большое количество уже написанных экспортеров для различных приложений. Также метрики могут собираться с помощью механизма push — для этого используется компонент pushgateway, который должен быть установлен дополнительно.

Довольно часто Prometheus настраивают в связке с Grafana, которая позволяет визуализировать показания наших метрик. В Grafana для этого есть уже настроенный источник, таким образом, настройка выполняется из коробки.

Grafana — универсальная обертка для работы с аналитическими данными, которые хранятся в разных источниках. Она сама ничего не хранит и не собирает, а является лишь универсальным клиентом для систем хранения метрик.

**Постановка задачи**

Вам необходимо создать Spring Boot сервис и произвести мониторинг его работы. В работе должны быть отражены все пункты создания dockercompose файла, проверена работоспособность сервиса и систем мониторинга. Произведена нагрузка на систему с целью проверки работоспособности. Все пункты должны быть отражены в отчете в формате снимков экрана.

Требования к системе:

1. Наличие сервера с CRUD набором для взаимодействия с базой данных и выгрузкой данных с GrayLog по эндпоинту
   1. CRUD набор должен содержать эндпоинты для добавления, обновления, чтения и удаления записей
   2. Количество используемых моделей данных должно быть не менее 3, а также содержать как минимум 1 связь 1-N, 1 связь NN.
   3. При обращении к эндпоинту логов он должен выгружать данные в формате csv из GrayLog
2. PostgreSQL в качестве СУБД с которой будет производиться взаимодействие/сниматься метрики и логи
3. Zabbix для мониторинга базовой работоспособности сервера
4. Prometheus для сбора данных с PostgreSQL, Grafana для вывода этих данных
5. GrayLog для сбора данных с PostgreSQL
6. Adminer для управления базой данных PostgreSQL.

**Ход работы**

Было разработано приложение, которое хранит информацию о студентах, преподавателях и курсах. Его структура представлена на рисунке 1.

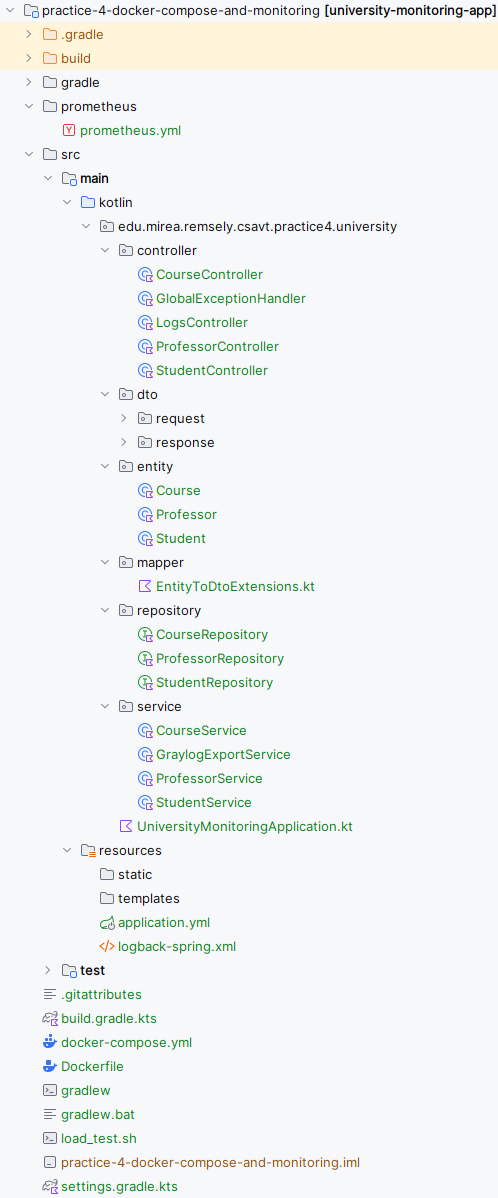
****

Рисунок 1 – Структура проекта

Файл с зависимостями приложения представлен на рисунке 2.

****

Рисунок 2 – Класс сущности Course

Код основных сущностей приложения представлен на русунках 3-5.

****

Рисунок 3 – Класс сущности Course

****

Рисунок 4 – Репозиторий сущности Professor

****

Рисунок 5 – Класс сущности Student

Конфигурация Prometheus представлена на рисунке 6.

****

Рисунок 6 – Сервис для получения логотипа из ресурсов

Конфигурация docker-compose представлена на рисунках 7-11.

****

Рисунок 7 – Конфигурация Spring Boot приложения

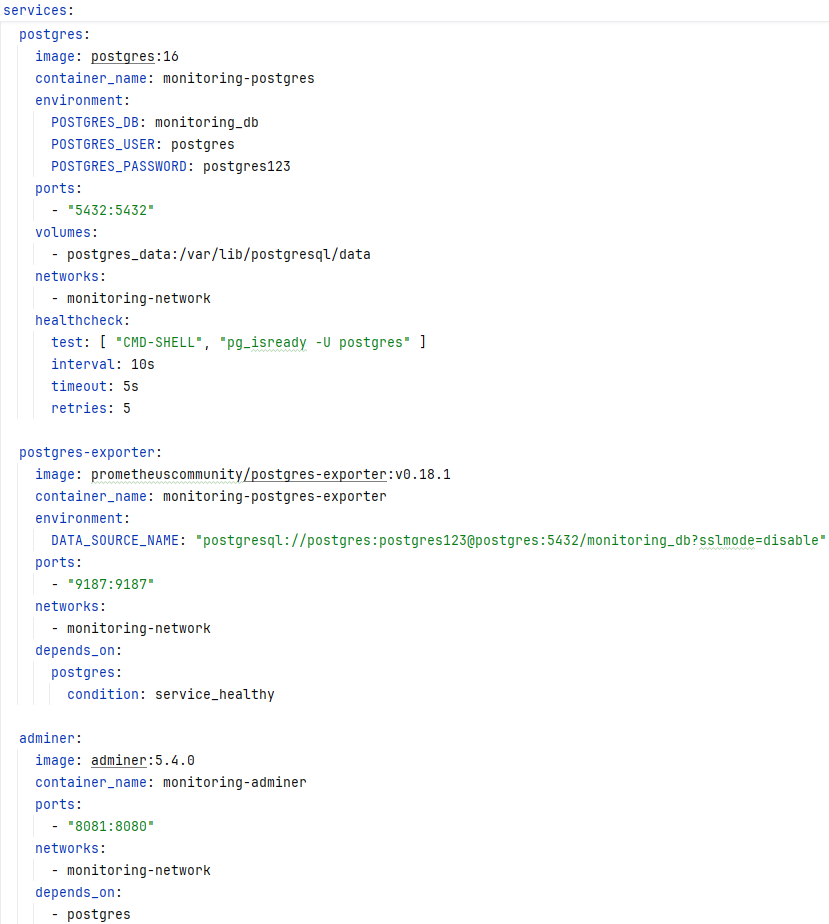
****

Рисунок 8 – Конфигурация СУБД PostgreSQL и сервисов ее мониторинга

****

Рисунок 9 – Конфигурация Prometheus или Grafana

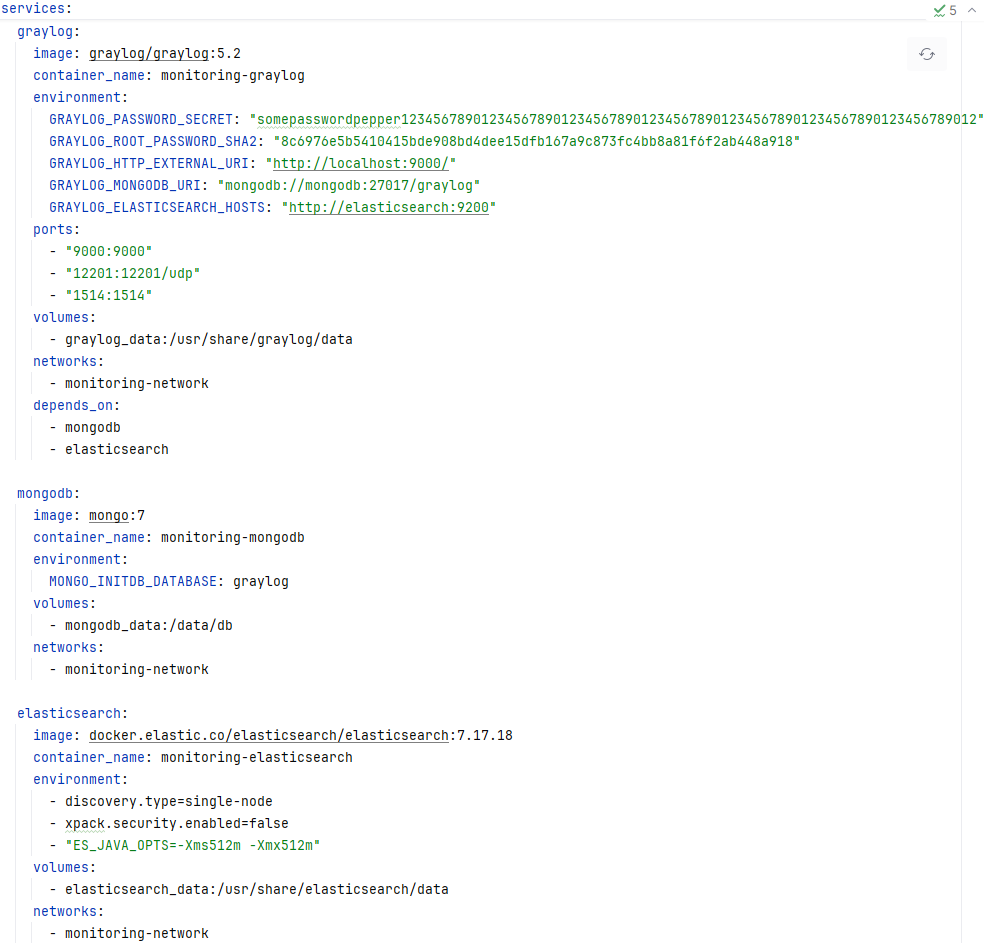
****

Рисунок 10 – Конфигурация GrayLog и его зависимостей

****

Рисунок 11 – Конфигурация Zabbix и связанных контейнеров

Запущенные контейнеры представлены на рисунке 12.

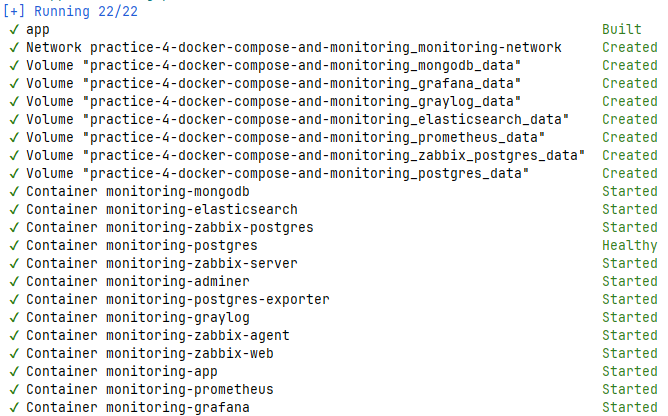
****

Рисунок 12 – Запущенные контейнеры

Примеры POST-запросов к сервису представлены на рисунках 13-15.

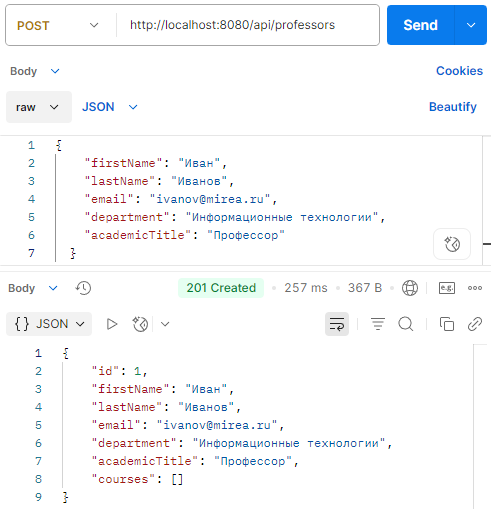
****

Рисунок 13 – POST-запрос для добавления преподавателя

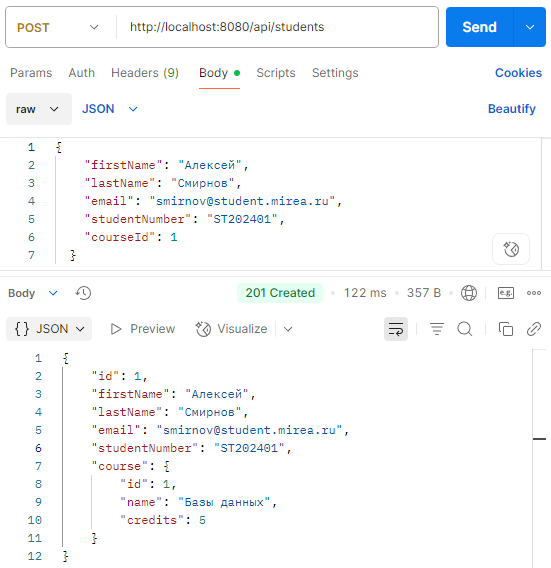
****

Рисунок 14 – POST-запрос для добавления студента

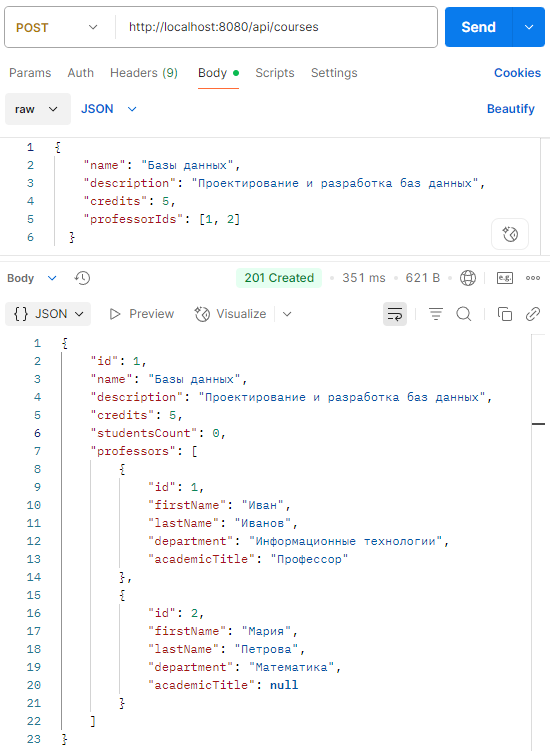
****

Рисунок 15 – POST-запрос для добавления курса

Примеры GET-запросов к сервису представлены на рисунках 16-17.

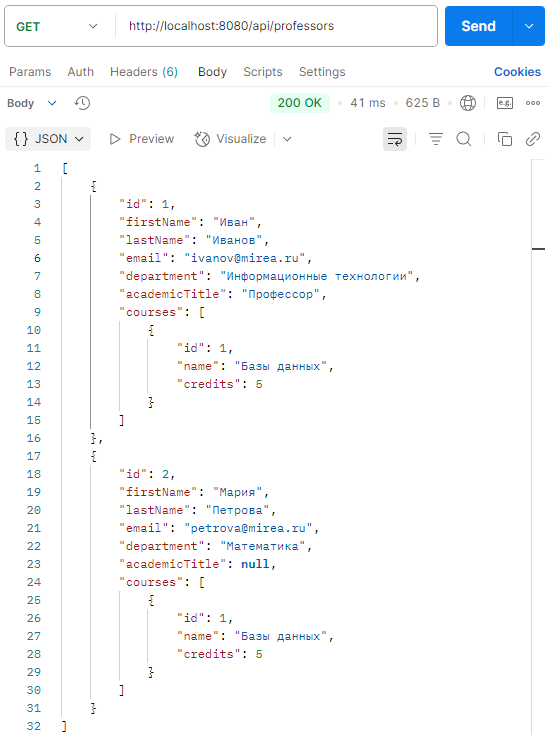
****

Рисунок 16 – GET-запрос для получения преподавателей

****

Рисунок 17 – GET-запрос для получения студентов курса

Пример PUT-запроса к сервису представлен на рисунке 18.

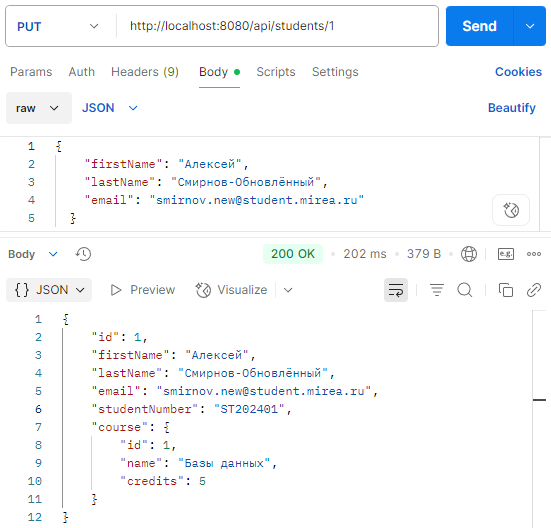
****

Рисунок 18 – PUT-запрос для обновления студента

Пример DELETE-запроса к сервису представлен на рисунке 19 (GET-запрос для подтверждения удаления представлен на рисунке 20).

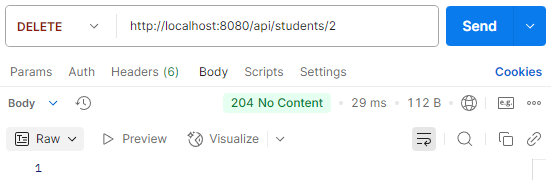
****

Рисунок 19 – DELETE-запрос для удаления студента

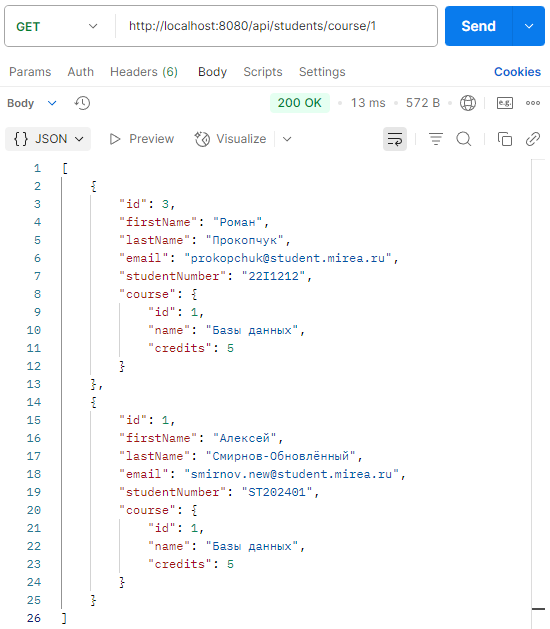
****

Рисунок 20 – Подтверждающий удаление GET-запрос

Запрос к Actuator представлен на рисунке 21.

Информация о БД в Adminer представлен на рисунке 22.

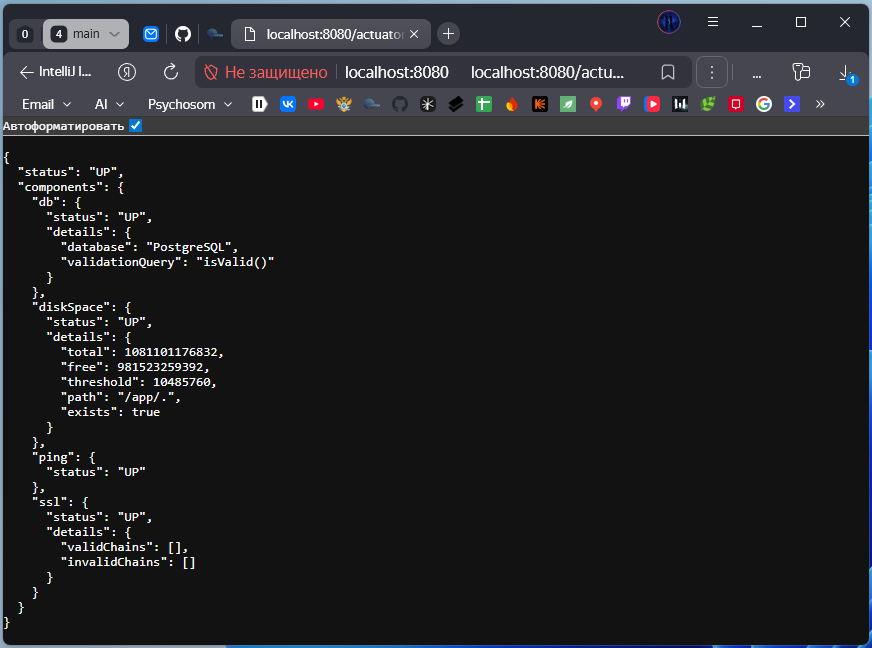
****

Рисунок 21 – Запрос к Actuator

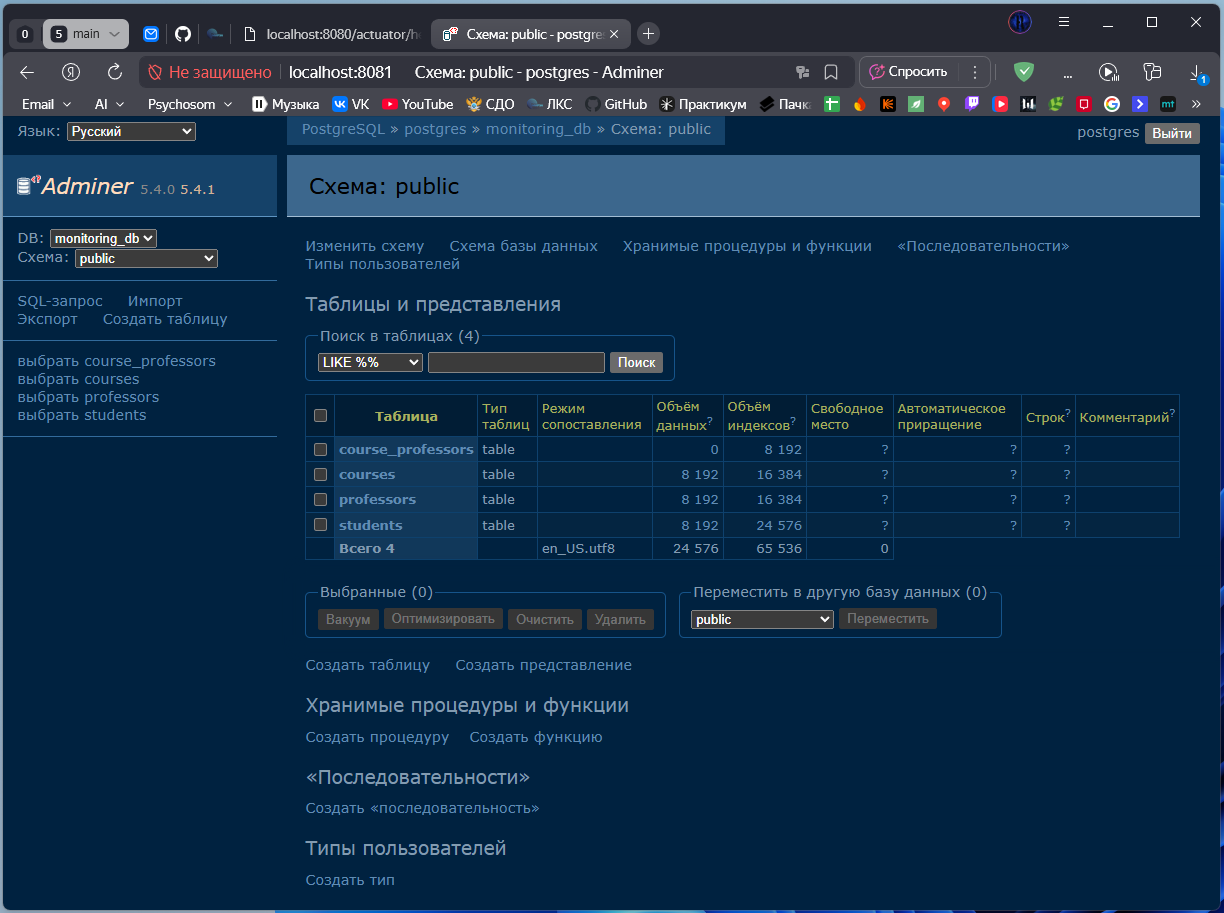
****

Рисунок 22 – Информация о БД в Adminer

Информация о метриках в Prometheus предтавлена на рисунке 23.

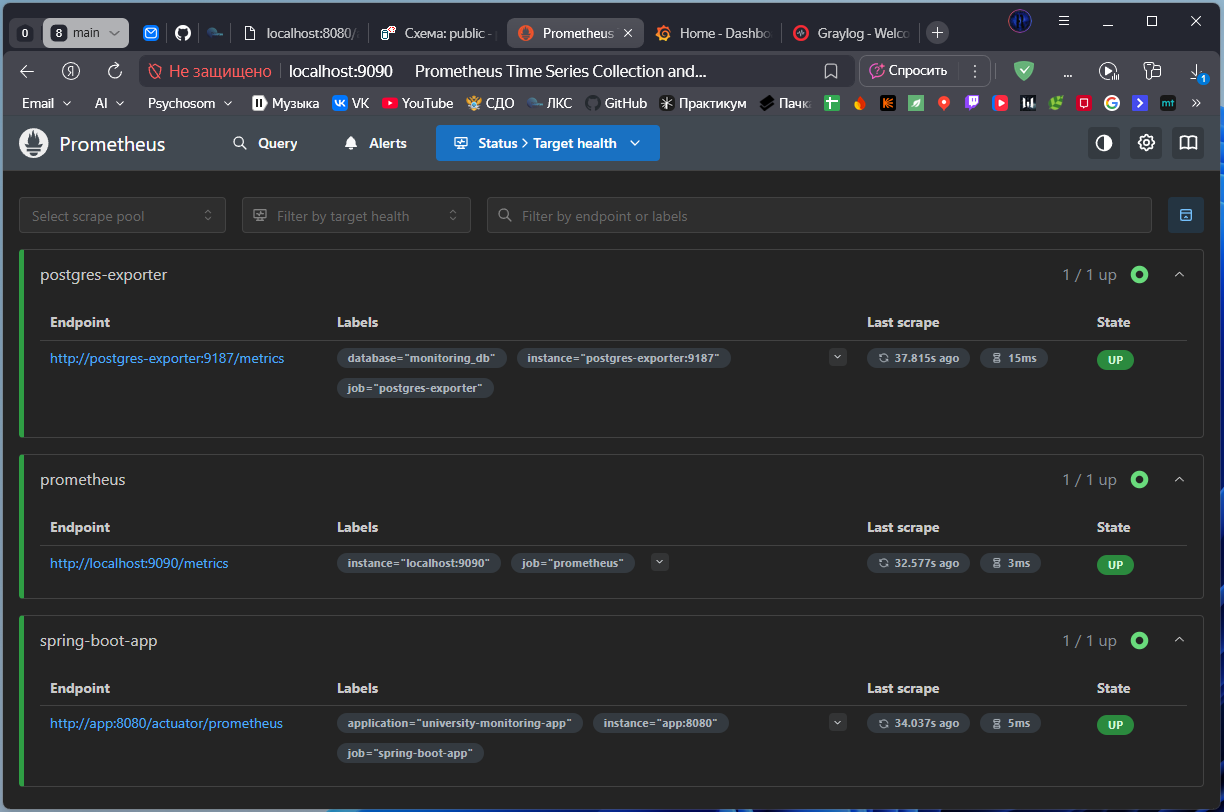
****

Рисунок 23 – Информация о метриках в Prometheus

Пример потока метрик представлен на рисунке 24.

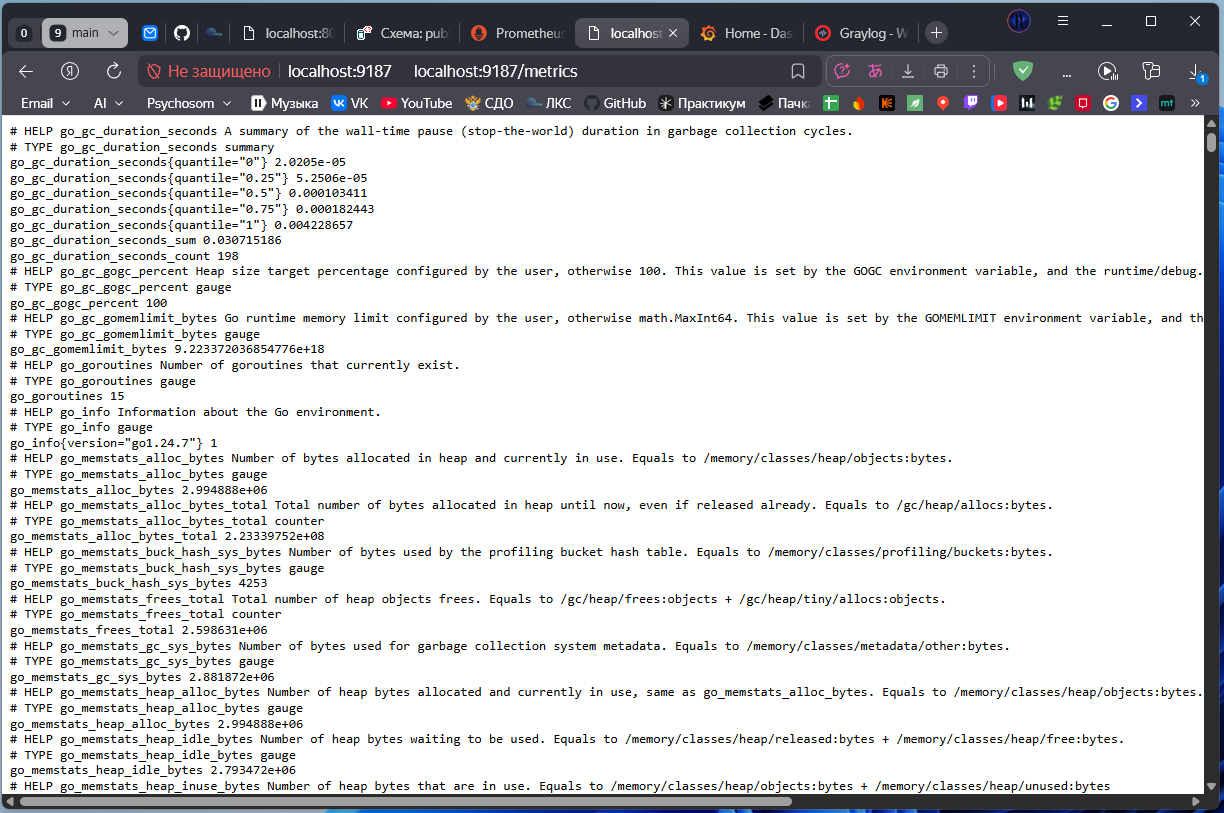
****

Рисунок 24 – Пример потока метрик

Импортирование дашборда для PostgreSQL представлено на рисунке 25.

Итоговый список дашбордов представлен на рисунке 26.

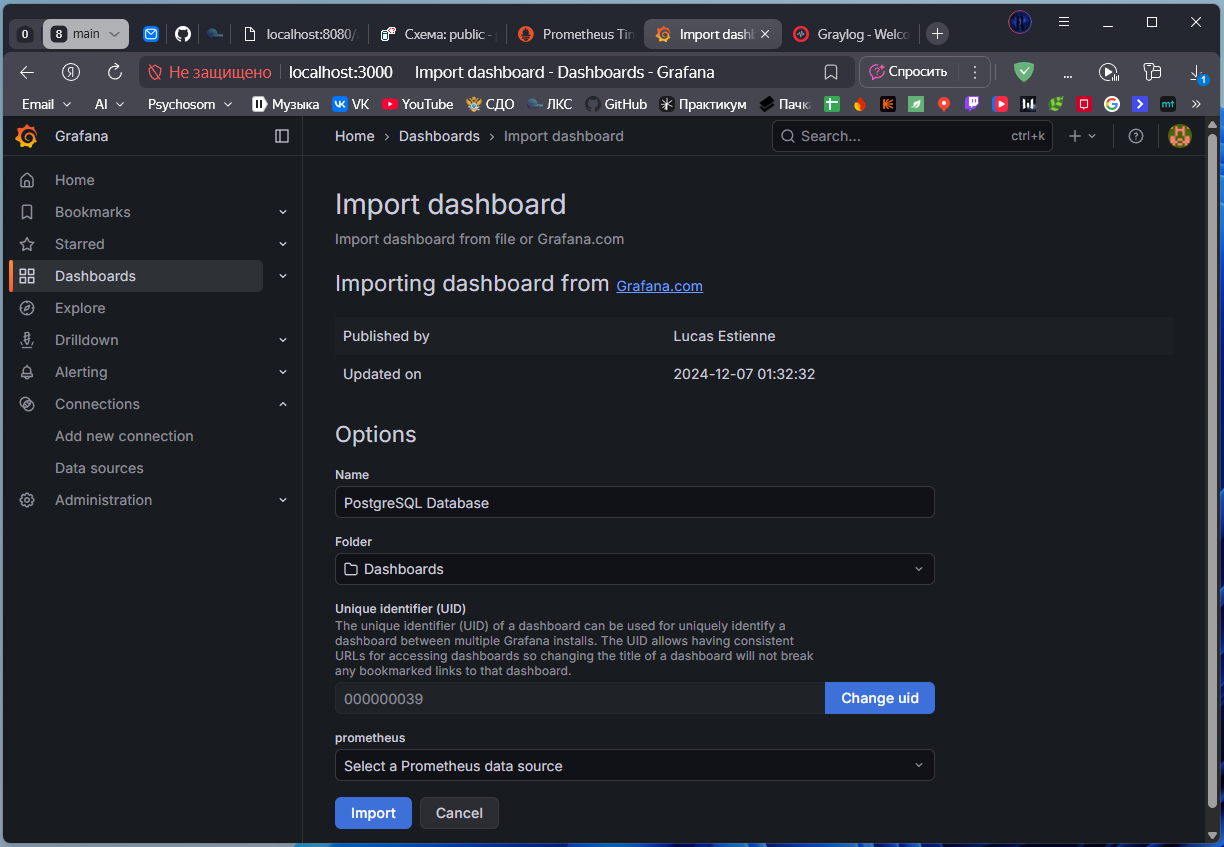
****

Рисунок 25 – Импорт дашборда PostgreSQL

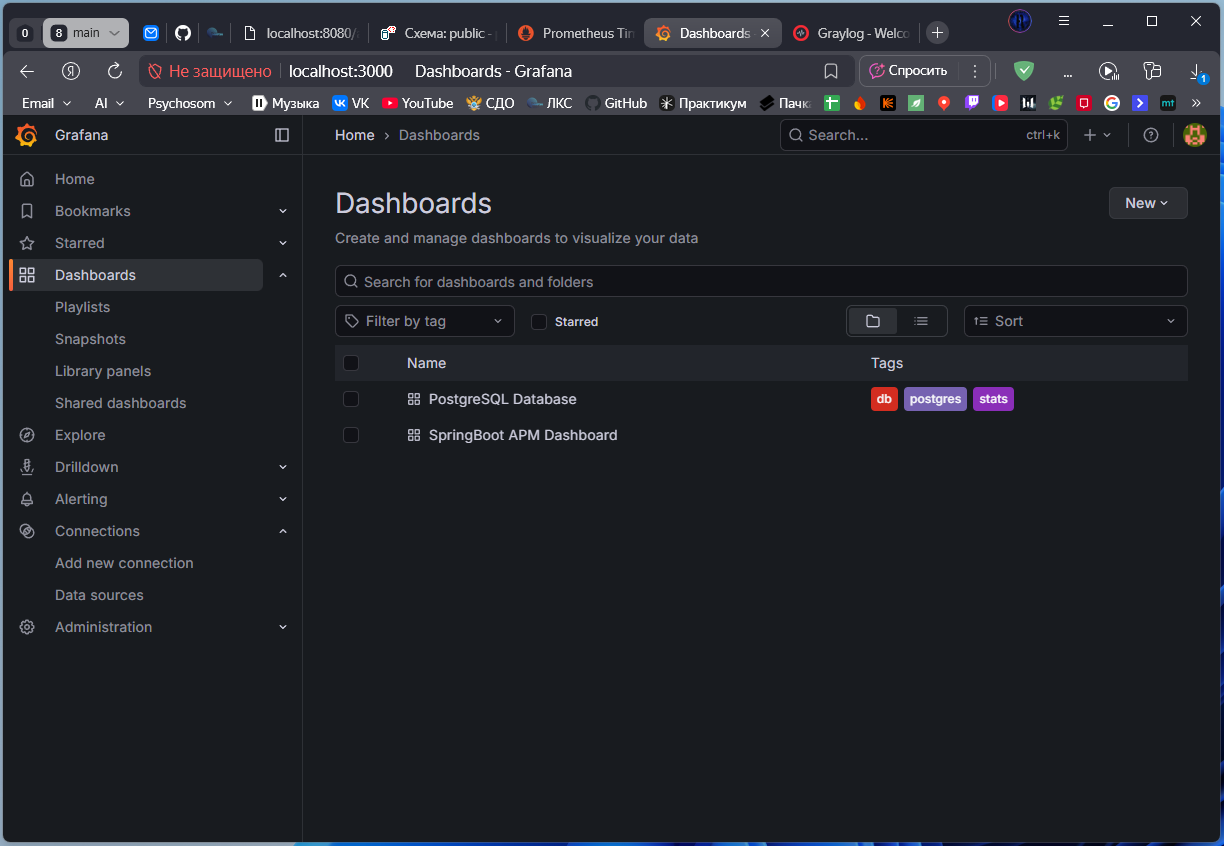
****

Рисунок 26 – Итоговый список дашбордов

Дашборд с метриками PostgreSQL представлен на рисунке 27, дашборд с метриками Spring Boot приложения – на рисунке 28.

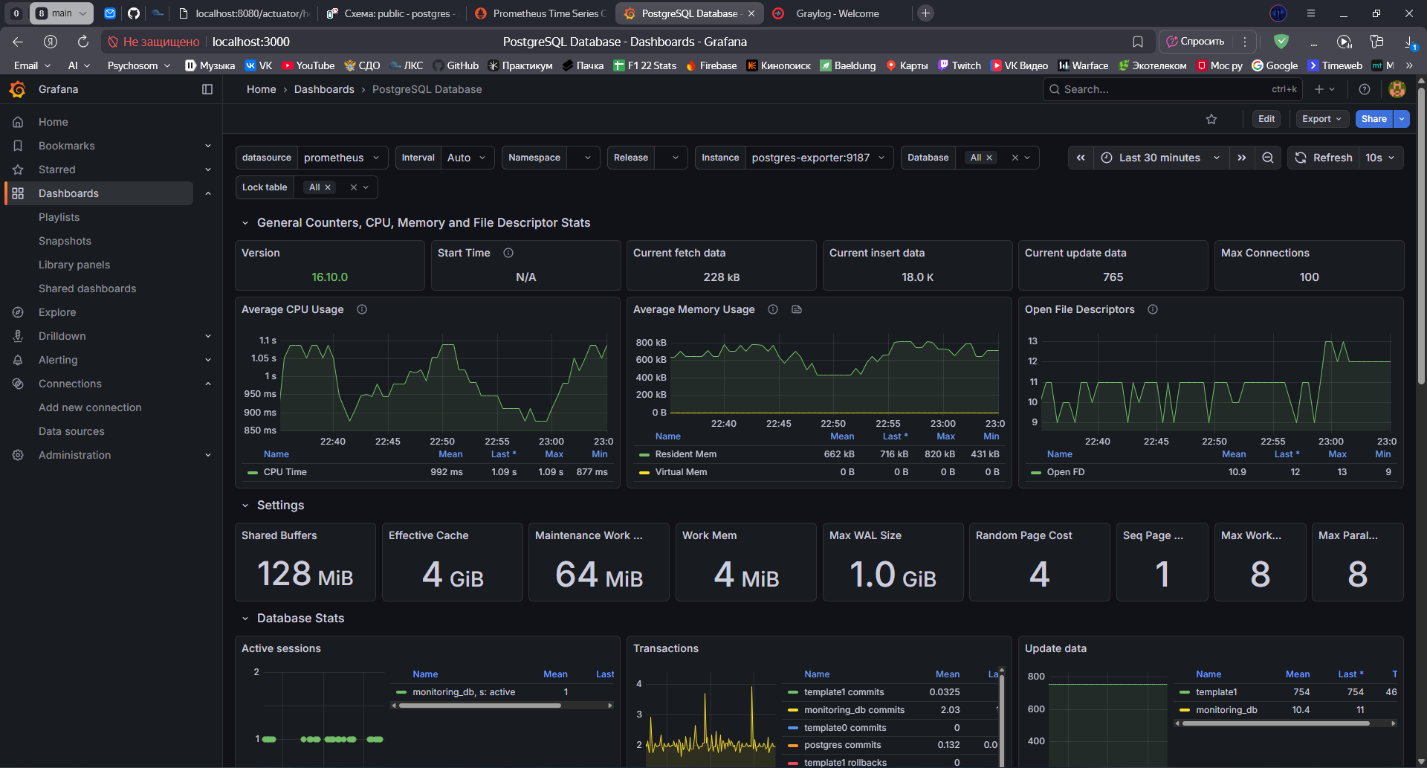


Рисунок 27 – Дашборд с метриками PostgreSQL

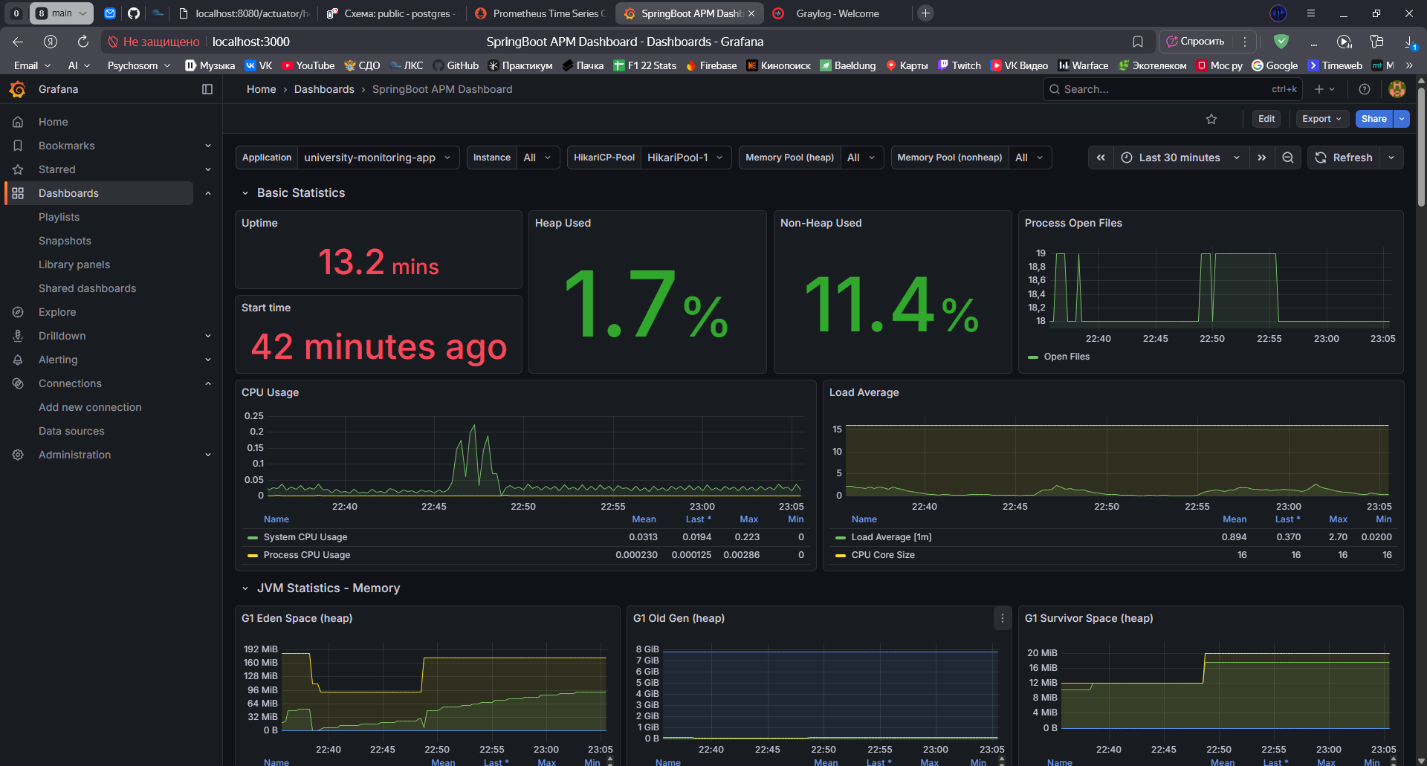
****

Рисунок 28 – Дашборд с метриками PostgreSQL

Логи в интерфейсе GrayLog представлены на рисунке 29, логи, полученные в формате CSV – на рисунке 30.

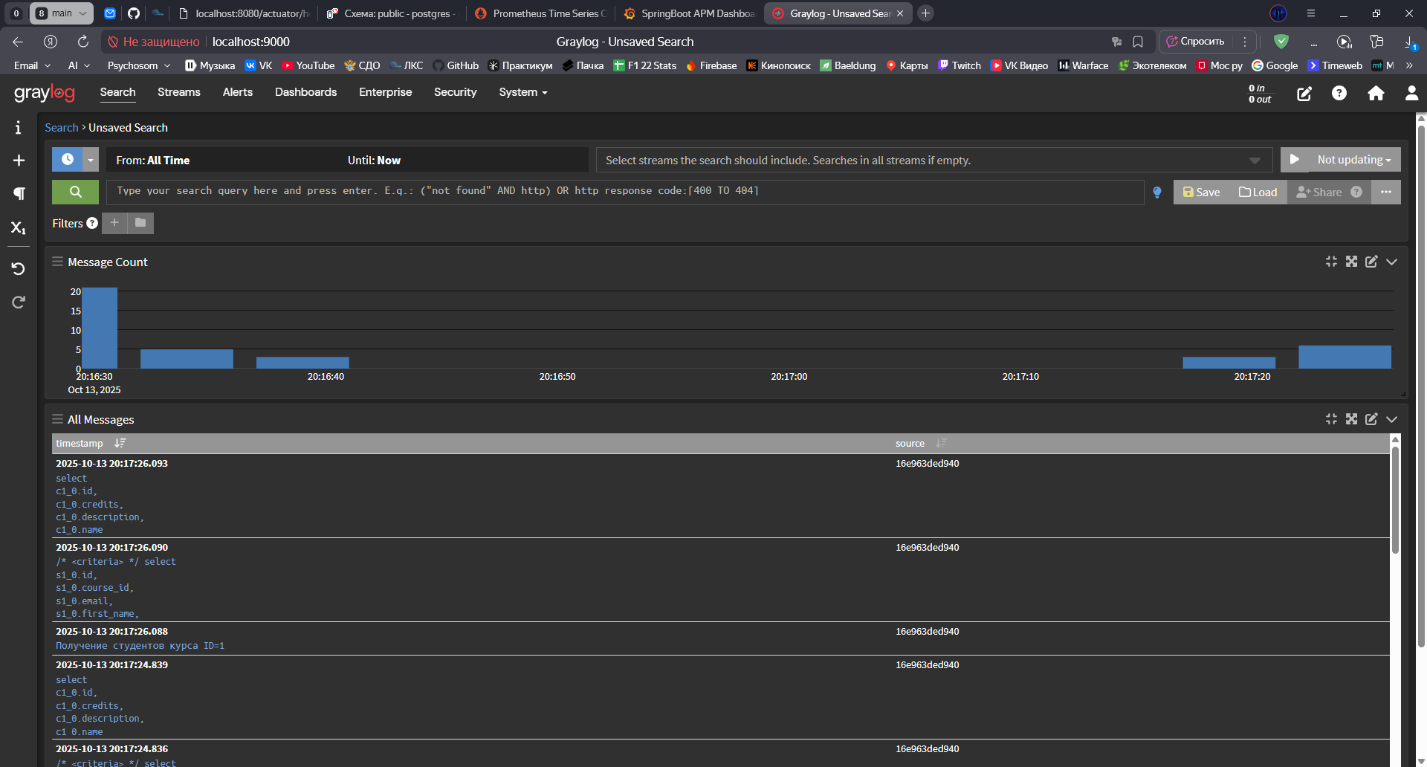
****

Рисунок 29 – Логи приложения в интерфейсе GrayLog

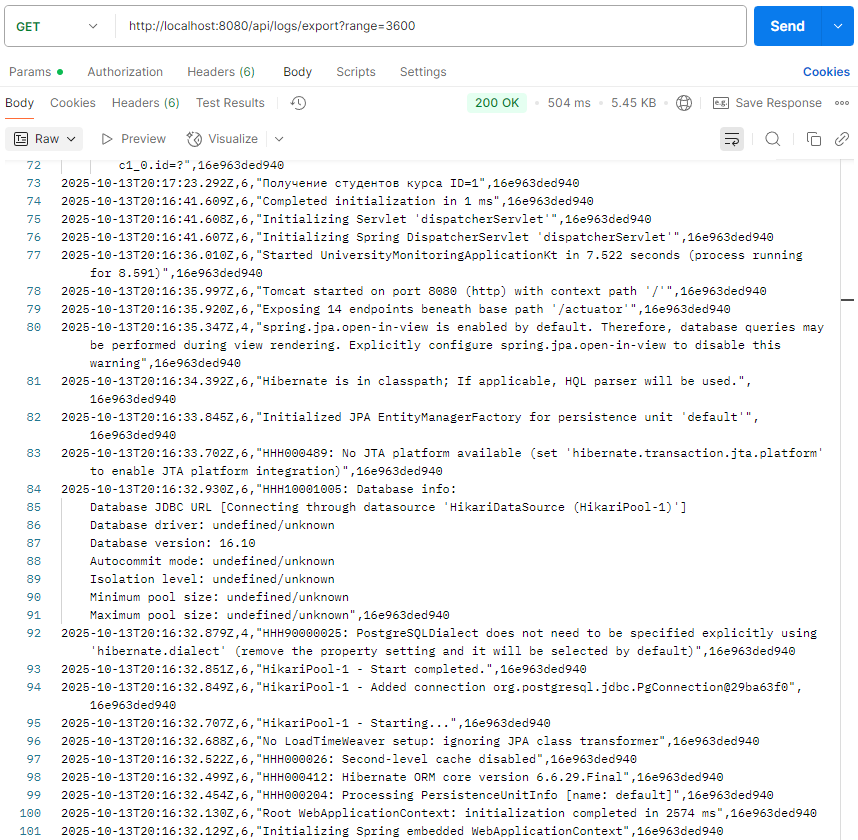
****

Рисунок 30 – Логи, полученные в формате CSV

Информация о приложении в Zabbix представлена на рисунке 31.

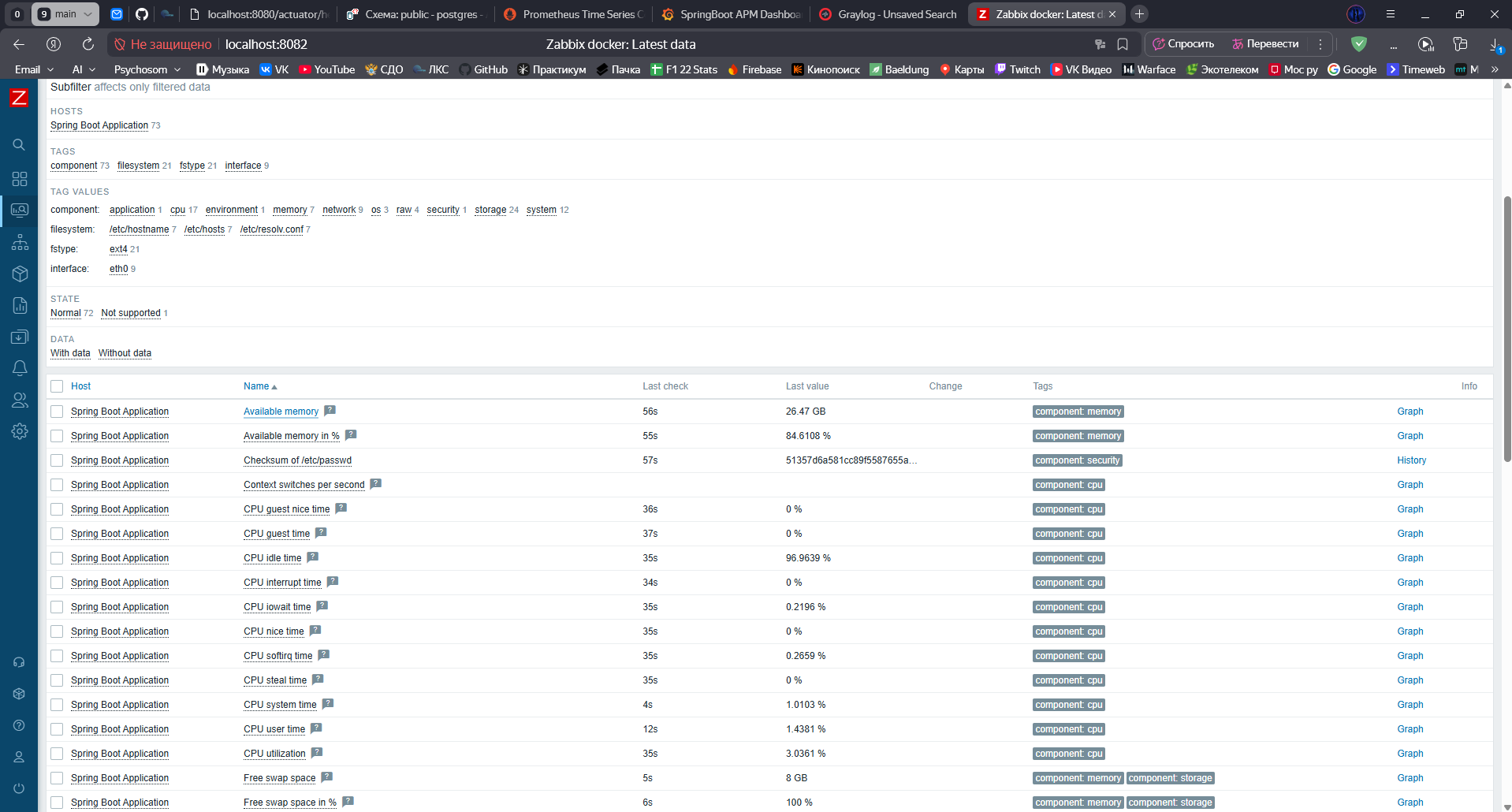
****

Рисунок 31 – Информация о приложении в интерфейсе Zabbix

Скрипт для подачи нагрузки на приложение представлен на рисунке 32.

****

Рисунок 32 – Скрипт для тестирования нагрузки

Запуск скрипта для нагрузочного тестирования представлен на рисунке 33.

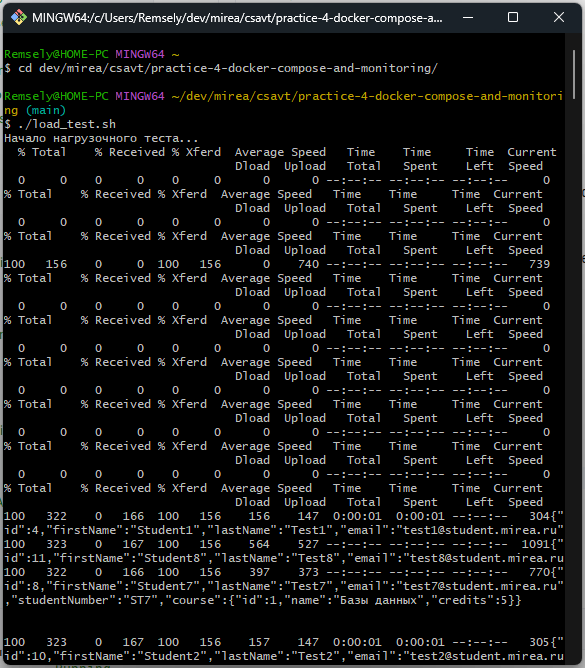
****

Рисунок 33 – Запуск скрипта для нагрузочного тестирования

Метрики приложения после нагрузочного тестирования представлены на рисунке 34.

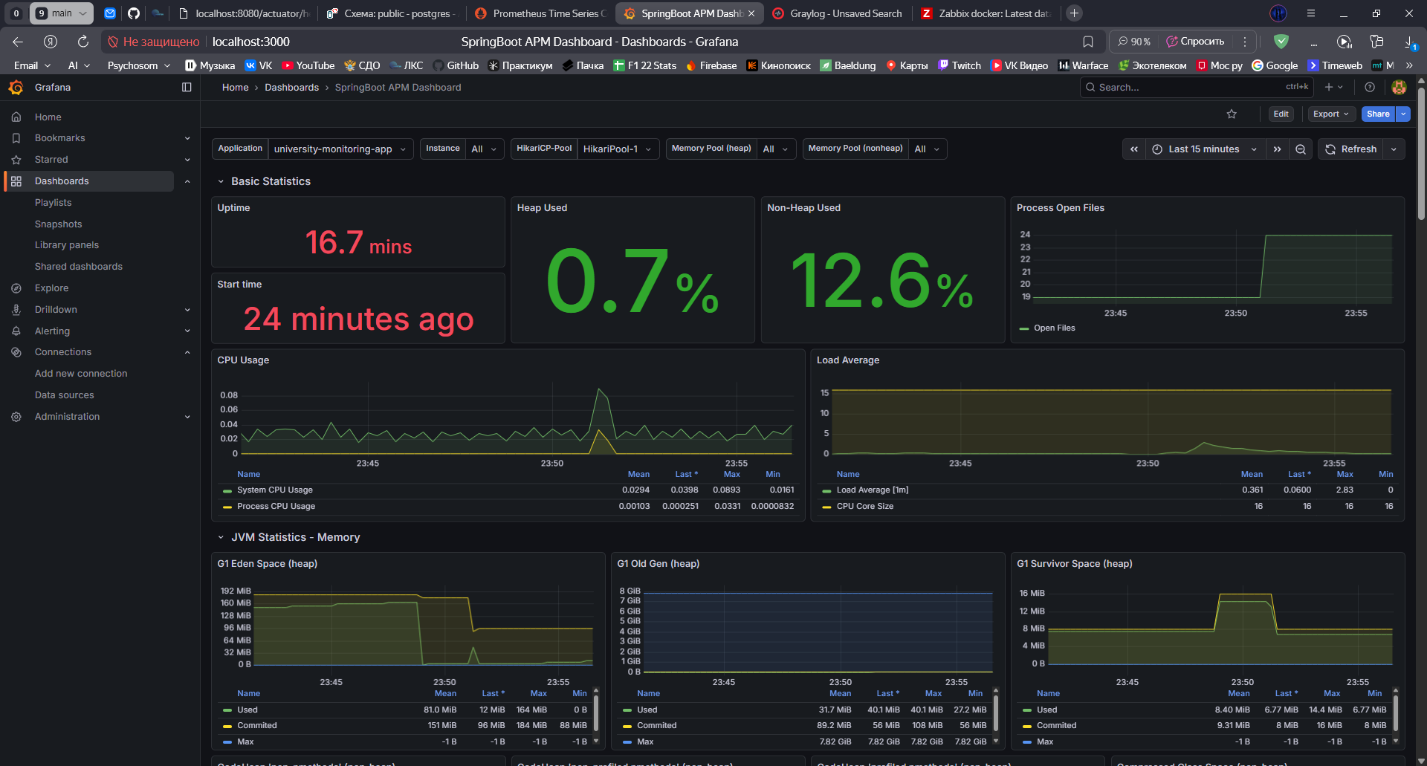
****

Рисунок 34 – Метрики приложения после нагрузочного тестирования

**Вывод**

В результате выполнения данной практической работы было разработано Spring Boot CRUD-приложение с окружением из инструментов мониторинга GrayLog, Adminer, Zabbix, Prometheus, Grafana.

**Ответы на вопросы к практической работе**

1. **Назовите основные различия между Prometheus и Zabbix.**

Prometheus использует модель pull, хранит метрики во временных рядах и ориентирован на cloud-native среду. Zabbix применяет агентов и push-модель, хранит данные в реляционной БД и подходит для классического инфраструктурного мониторинга.

1. **Как можно запустить две базы PostgreSQL в одном docker-compose файле, чтобы таблицы не пересекались?**

Нужно создать два сервиса PostgreSQL с разными именами, портами и томами. Каждый контейнер будет иметь свою БД и каталог данных, например: postgres1:5432, postgres2:5433, разные volumes и переменные окружения.

1. **Назовите виды мониторинга систем.**

Инфраструктурный, прикладной (APM), сетевой, лог-мониторинг, мониторинг безопасности, пользовательский и бизнес-мониторинг. Также выделяют три основы наблюдаемости: метрики, логи и трейсы.

1. **При помощи чего можно передавать конфигурационные переменные в контейнер?**

Через переменные окружения (environment или env\_file), файлы конфигураций в volumes, Docker Secrets и Config, а также аргументы командной строки.

1. **Назовите основные различия Docker Swarm и Docker Compose.**

Docker Compose используется для локального запуска нескольких контейнеров. Docker Swarm — для кластерного развёртывания, масштабирования и балансировки нагрузки между нодами.

1. **Назовите пример задачи, которую невозможно выполнить, используя один лишь docker-compose без Dockerfile.**

Сборка собственного образа из исходников (например, компиляция приложения или установка зависимостей) требует Dockerfile; Compose может только запускать уже готовые образы.

**Список источников информации**

* 1. 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/>
  2. Docker Documentation | Docker Documentation — Текст: электронный [сайт]. — URL: — Текст: электронный [сайт]. — URL: <https://docs.docker.com/>
  3. Zabbix Documentation — Текст: электронный [сайт]. — URL: <https://www.zabbix.com/manuals>
  4. Prometheus Documentation — Текст: электронный [сайт]. — URL: <https://prometheus.io/docs/introduction/overview/>
  5. Grafana Documentation — Текст: электронный [сайт]. — URL: <https://grafana.com/docs/>
  6. GrayLog Documentation — Текст: электронный [сайт]. — URL: <https://docs.graylog.org/>