|  |  |  |
| --- | --- | --- |
|  | Изображение выглядит как зарисовка, рисунок, символ, корона  Автоматически созданное описание |  |
| 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 №4**

по дисциплине «Технологии виртуализации клиент-серверных приложений»

**Тема практической работы: «Мониторинг работы сервисов с использованием Docker Compose»**

**Студент группы** ИКБО-01-21 Денисов А.В.

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

(подпись студента)

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

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

(подпись руководителя)

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

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

Москва 2024

**СОДЕРЖАНИЕ**

[ЦЕЛЬ РАБОТЫ 3](#_Toc180234959)

[Сервер Spring Boot 4](#_Toc180234960)

[PostgreSQL 5](#_Toc180234961)

[Zabbix 6](#_Toc180234962)

[Prometheus и Grafana 7](#_Toc180234963)

[Graylog 9](#_Toc180234964)

[Adminer 10](#_Toc180234965)

[ВЫВОД 12](#_Toc180234966)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 13](#_Toc180234967)

ЦЕЛЬ РАБОТЫ

Освоить навыки создания и настройки многоконтейнерного приложения с использованием Docker Compose, включающего Spring Boot сервис и различные системы мониторинга. Научиться разворачивать комплексную инфраструктуру, состоящую из веб-приложения, базы данных и инструментов мониторинга, таких как Zabbix, Prometheus, Grafana и GrayLog. Получить практический опыт в настройке взаимодействия между различными компонентами системы и реализации механизмов сбора и анализа данных о производительности приложения.

ХОД РАБОТЫ

## **Сервер Spring Boot**

Создадим Spring Boot приложение с CRUD набором для взаимодействия с базой данных и выгрузкой данных с Graylog по эндпоинту. Исходный код контроллера с выгрузкой логов приведен на рисунке 1. Dockerfile и часть docker-compose.yml для развертывания приложения приведены на рисунках 2–3.

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

Автоматически созданное описание

Рисунок 1 – Исходный код контроллера GraylogController

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2 – Dockerfile приложения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3 – Часть файла docker-compose.yml для запуска приложения

## **PostgreSQL**

Развернем СУБД PostgreSQL, с которой в дальнейшем будут сниматься метрики.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 4 – Часть файла docker-compose.yml для запуска PostgreSQL

## **Zabbix**

Развернем Zabbix для мониторинга базовой работоспособности сервера. Для этого потребуются сам сервер Zabbix, web-интерфейс и агент для сбора метрик. Конфигурация показана на рисунке 5.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 5 – Часть файла docker-compose.yml для запуска Zabbix

Далее настроим сбор метрик с сервера (рисунок 6) и откроем дашбоарды (рисунок 7).

Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание

Рисунок 6 – Настройка сбора метрик с хоста

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Рисунок 7 – Дашбоарды с хоста

## **Prometheus и Grafana**

Развернем Prometheus для сбора данных с PostgreSQL и Grafana для отображения этих данных.

Для сбора данных с PostgreSQL в формате Prometheus необходимо развернуть postgres-exporter, что показано на рисунке 8.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 8 – Часть файла docker-compose.yml для запуска postgres-exporter

Затем зададим создадим файл конфигурации prometheus.yml (рисунок 9).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 9 – Конфигурация Prometheus

Развернем Prometheus и Grafana в Docker (рисунок 10).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 10 – Часть файла docker-compose.yml для запуска Prometheus и Grafana

Просмотрим метрики базы данных (рисунок 11).

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

Автоматически созданное описание

Рисунок 11 – Просмотр метрик в Grafana

## **Graylog**

Развернем Graylog для сбора данных с PostgreSQL. Для этого необходимо собирать логи с СУБД и передавать их в Graylog. Для этих целей развернем контейнер pg-graylogger (рисунок 12).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 12 – Часть файла docker-compose.yml для отправки логов в Graylog

Далее развернем сам Graylog вместе с его зависимостями: MongoDB и Elasticsearch (рисунок 13).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 13 – Часть файла docker-compose.yml для запуска Graylog

Настроим input в Graylog (рисунок 14). Видим, что нам поступает в среднем 57 сообщений в секунду.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 14 – Созданный input в Graylog

Проверим логи (рисунок 15).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 15 – Логи СУБД PostgreSQL в Graylog

## **Adminer**

Развернем Adminer для управления PostgreSQL (рисунок 16) и откроем в нем нашу базу данных (рисунок 17).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 16 – Часть файла docker-compose.yml для запуска Adminer

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 17 – Демонстрация работы Adminer

ВЫВОД

В ходе выполнения практической работы мы успешно создали и развернули комплексную систему на базе Docker Compose, включающую Spring Boot сервис, PostgreSQL базу данных и набор инструментов для мониторинга и анализа производительности. Мы научились настраивать взаимодействие между различными контейнерами, реализовали CRUD операции для работы с базой данных, а также настроили сбор и визуализацию метрик с использованием Zabbix, Prometheus и Grafana.

Особое внимание было уделено настройке системы логирования с использованием GrayLog, что позволило нам эффективно отслеживать и анализировать события в нашем приложении. Мы также получили практический опыт в использовании Adminer для управления базой данных PostgreSQL в контейнеризированной среде.

Эта работа значительно расширила наше понимание принципов построения и мониторинга микросервисных архитектур, а также предоставила ценный опыт в области DevOps практик, таких как непрерывный мониторинг и управление логами в распределенных системах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Docker Documentation | Docker Documentation – Текст: электронный [сайт]. – URL: <https://docs.docker.com/>

2. Zabbix Documentation – Текст: электронный [сайт]. – URL: <https://www.zabbix.com/manuals>

3. Prometheus Documentation – Текст: электронный [сайт]. – URL: <https://prometheus.io/docs/introduction/overview/>

4. Grafana Documentation – Текст: электронный [сайт]. – URL: <https://grafana.com/docs/>

5. GrayLog Documentation – Текст: электронный [сайт]. – URL: <https://docs.graylog.org/>

6. Spring Boot Documentation | Spring – Текст: электронный [сайт]. – URL: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>

7. PostgreSQL Documentation – Текст: электронный [сайт]. – URL: <https://www.postgresql.org/docs/>

8. Docker Compose Documentation | Docker Documentation – Текст: электронный [сайт]. – URL: <https://docs.docker.com/compose/>

9. Adminer Documentation – Текст: электронный [сайт]. – URL: <https://www.adminer.org/en/>

10. 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/>