Министерство науки и высшего образования Российской Федерации

ФГАОУ ВО «Уральский федеральный университет   
имени первого Президента России Б.Н. Ельцина»

Институт новых материалов и технологий

Кафедра «Теплофизика и информатика в металлургии»

**Разработка контейнерного приложения Java и настройка конвейера CI/CD с помощью GitHub Actions**

**ОТЧЕТ**

**по домашней работе**

**по дисциплине «Основы методологии Development Operation»**

Направление 09.03.02 «Информационные системы и технологии» (уровень бакалавриата)

Образовательная программа  
09.03.02/33.02 «Информационные системы и технологии» (СУОС)

Студент

группы НМТ-413901 И.В.Машуков

Преподаватель:

профессор, д.т.н. В.В.Лавров

Екатеринбург

2024

СОДЕРЖАНИЕ

[1 ЗАДАНИЕ 3](#_Toc189881237)

[2 РАЗРАБОТКА ПРИЛОЖЕНИЯ 3](#_Toc189881238)

[2.1 Разработка основного алгоритма 3](#_Toc189881239)

[2.2 Разработка контейнера 4](#_Toc189881240)

[2.3 Настройка конвейера 7](#_Toc189881241)

# **1 ЗАДАНИЕ**

Необходимо разработать контейнерное приложения Java и настроить конвейер CI/CD с помощью GitHub Actions. Приложение будет выполнять простые функции: сложение, вычитание, умножение и деление.

# **2 РАЗРАБОТКА ПРИЛОЖЕНИЯ**

## **2.1 Разработка основного алгоритма**

С помощью maven создан конфигурационный файл,, который содержит информацию о проекте (Рисунок 2.1).

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.1 – файл pom.xml

Разработаем основной файл, который реализует веб-сервер, обрабатывающий HTTP-запросы (Рисунок 2.2).

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.2 – файл DevCalcApplication.java

Запустим приложение на локальном адресе, произведем вычитание 40 и 30 и их сложение (Рисунок 2.3).

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.3 – демонстрация функций приложения

Таким образом, было запущенно наше приложение локально, и оно работает.

## **2.2 Разработка контейнера**

Создадим образ и контейнер (Рисунки 2.4 и 2.5).

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.4 – создание образа

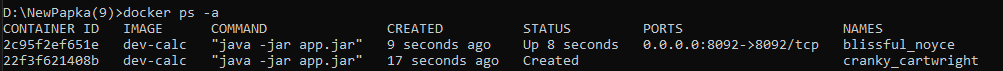


Рисунок 2.5 – работающий контейнер

Проверим доступность с устройства на котором расположен контейнер (Рисунок 2.6).

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.6 – проверка веб-приложения

Проверим доступность с разных устройств (Рисунок 2.7).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.7 – проверка веб-приложения с другого устройства (сложение)

## **2.3 Настройка конвейера**

Настроим конвейер CI/CD с помощью GitHub Actions. Для этого выгрузим репозиторий и перейдем во вкладку “Actions” и выберем Docker image (Рисунок 2.8).

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.8 – Docker image

Автоматические сгенерируется код. Я немного его дополнил, для работы с maven (Рисунок 2.9).

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.9 – код файла docker-image.yml

Запустим Action (Рисунки 2.10 и 2.11).

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.10 – успешное выполнение

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.11 – успешное выполнение

Таким образом был настроен автоматический сбор контейнера с помощью GitHub

Ссылка на github: <https://github.com/MashukovI/DevCalc>

# 3 ЗАКЛЮЧЕНИЕ

Разработка контейнерного приложения на Java и настройка CI/CD-конвейера с помощью GitHub Actions позволяют автоматизировать процессы сборки, тестирования и развертывания приложения, что значительно ускоряет delivery и повышает надежность продукта.

Использование Docker обеспечивает единую среду выполнения приложения на всех этапах разработки, устраняя проблему "у меня работает на локальной машине". GitHub Actions предоставляет удобный инструмент для настройки CI/CD без необходимости использования сторонних сервисов, позволяя интегрировать тестирование, сборку образов и деплой в единый рабочий процесс.

В результате, настроенный пайплайн сокращает время между внесением изменений в код и их доставкой в production, минимизирует человеческий фактор и повышает стабильность приложения. Это делает процесс разработки более эффективным и соответствует современным DevOps-практикам.