Разработка и внедрение CI/CD пайплайна для автоматизации развертывания веб-приложения по подсчету калорий

Выполнил:

студент 4 курса

09.03.01 Информатика и вычислительная техника, Технологии разработки программного обеспечения

Мельникова Алена Сергеевна

Руководитель:

к.п.н., Власов Дмитрий Викторович

Актуальность

Актуальность темы обусловлена необходимостью ускорения релизов, уменьшения человеческого фактора при деплое и обеспечения стабильной и воспроизводимой среды. Особенно важным это становится в проектах с микросервисной архитектурой, где компоненты могут разрабатываться и обновляться независимо.

Предмет

Предмет исследования — методы и инструменты построения CI/CD пайплайнов для микросервисных веб-приложений с использованием Docker и GitHub Actions.

В работе применяются:

1. методы анализа и синтеза архитектуры приложений;
2. практика контейнеризации компонентов с помощью Docker;
3. использование инструментов CI/CD (GitHub Actions);
4. настройка микросервисной инфраструктуры и среды исполнения;
5. эмуляция производственного процесса релизов.

Цель

Целью дипломной работы является разработка и внедрение CI/CD-пайплайна для автоматизации развертывания веб-приложения по подсчёту калорий с разделением на dev- и prod-окружения.

Задачи

Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработать MVP веб-приложения, включающего frontend и backend.
2. Упаковать компоненты в Docker-контейнеры.
3. Настроить систему docker-compose для локального и серверного запуска.
4. Настроить отдельные окружения для разработки (dev) и продакшна (prod).
5. Реализовать CI/CD пайплайн через GitHub Actions.
6. Обеспечить автоматическую доставку на dev-стенд и ручной деплой на prod.
7. Расширить функционал приложения и протестировать устойчивость пайплайна.

Инструменты и технологии

Для реализации DevOps-подхода и CI/CD в рамках проекта использовались следующие технологии:

- Docker — для контейнеризации приложений;
- Docker Compose — для управления многоконтейнерной инфраструктурой;
- GitHub Actions — для автоматизации CI/CD;
- Nginx — для проксирования запросов;
- Python/Django — для серверной части API;
- React — для построения интерфейса пользователя.

Каждая из технологий обеспечивает изоляцию, масштабируемость и переносимость компонентов.

Результат

В процессе разработки:

1. Реализовано веб-приложение с использованием Django (backend) и React (frontend);
2. Организован REST API для взаимодействия между клиентом и сервером;
3. Выполнена контейнеризация всех компонентов с помощью Docker;
4. Настроен автоматический деплой на dev-сервер при пуше в основную ветку;
5. Обеспечен ручной запуск пайплайна для прод-сервера через workflow_dispatch;
6. Реализована изоляция окружений, защита переменных и логирование.

Демонстрация работы продукта