Docker Compose на практике

08 апреля 2026 г.

Развитие приложения: новое требование

Текущее состояние нашего приложения:

- nginx работает в контейнере и раздает html
- Настроен автоматический деплой через CI/CD pipeline
- Приложение деплоится как докер контейнер

Новое требование:

- Добавить вызов backend API для работы с динамическими данными.
- Request per seconds (RPS) > 10

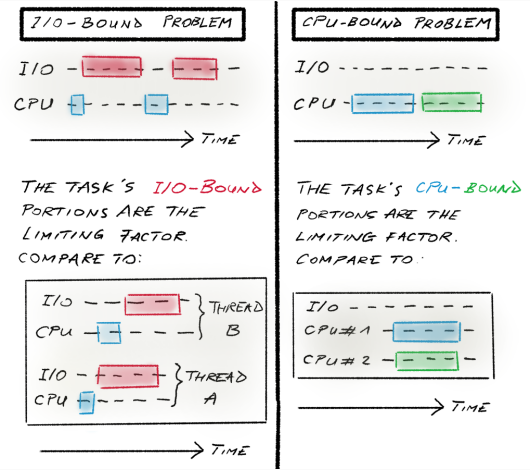
RPS

Requests per second — количество запросов, обрабатываемых сервером за одну секунду.

$$RPS_{\max} \approx \frac{\overbrace{N_{\text{backends}} \cdot T}^{\text{число backend-реплик} \times \text{потоки на backend}}}{\underbrace{t_{\text{delay}}}_{\text{задержка на один запрос (сек)}}},$$

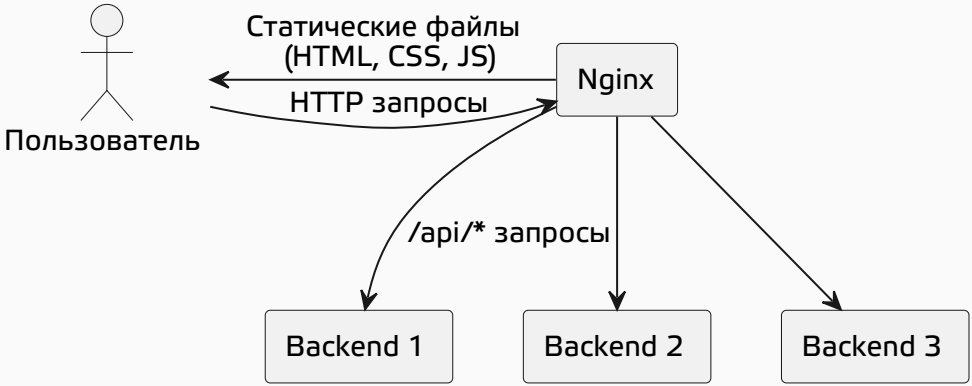
только для IO bound задач

IO & CPU bound задачи



https://alexandra-zaharia.github.io/posts/concurrency-crash-course-part-1/

Целевая архитектура



Демонстрация: Docker Compose

Что посмотрим:

1. Переиспользуемый GitHub Action, ссылка на коммит
2. Сборка и деплой backend приложения
3. Env файлы и docker compose (документация)
4. Горизонтальное масштабирование (документация)
5. wrk для нагрузочного тестирования

Демо-репозиторий:
https://github.com/prafdin/devops-demo-website/tree/docker-compose-demo

Ссылки по теме

- https://github.com/docker/awesome-compose/tree/master