Лебедев Олег

oleglebedev222@gmail.com | github | kaggle | leetcode | lintcode

Образование

НИЯУ МИФИ Москва

Прикладная математика и информатика. Институт лазерных и плазменных технологий *3 курс*

Проекты

High-Performance Article API (C++ Crow + PostgreSQL + Redis) | *C++, Crow, PostgreSQL, Redis (Valkey), wrk*

- Разработал веб-сервис на C++ с использованием фреймворка Crow для обработки GET-запросов, извлекающих случайную статью с комментариями из PostgreSQL
- Реализовал двусвязную модель хранения: таблица ‘articles‘ (статьи) и ‘comments‘ (связанные комментарии), отношение один-ко-многим
- Внедрил Redis (Valkey) как слой кеширования: при первом обращении ответ извлекается из базы, сериализуется в JSON и сохраняется в Redis, последующие обращения используют кеш
- Настроил логирование времени ответа (до и после внедрения Redis) и сохранил результаты в файл для последующего анализа производительности
- Провёл нагрузочное тестирование сервиса с помощью инструмента `wrk`, сравнив производительность с кешем и без
- Построил график времени ответа по каждому запросу, на котором наглядно видно, как с течением времени Redis-кеш постепенно наполняется, а задержка резко снижается

Stream Processing Microservices Pipeline | *Python, Faust (Kafka), FastAPI, PostgreSQL, Docker, Prometheus, Grafana*

- Спроектировал и реализовал event-driven конвейер микросервисов для обработки потоковых данных валютных пар на Faust/Kafka
- Написал сервис *data_requester* на Faust, который опрашивает FastAPI-demo-сервис и публикует сообщения в топик `src-data` через aiohttp
- Реализовал *data_processor* для нормализации и разделения входных сообщений, публикуя результаты в топик `processed-data`
- Разработал *data_aggregator* с Faust Tables (RocksDB) для расчёта скользящего среднего по последним 10 значениям и управления состоянием через changelog-топик
- Настроил *db_loader* для асинхронной записи сырых и агрегированных данных в PostgreSQL с помощью SQLAlchemy Async и asyncpg
- Создал API-шлюз на FastAPI для REST-доступа к хранимым данным, интегрировал Prometheus-metrics и построил дашборд Grafana
- Контейнеризовал весь стек (Kafka, Zookeeper, Postgres, Prometheus, Grafana и все микросервисы) с Docker Compose для лёгкого развертывания и масштабирования

Real-Time Chat Application | *Django, Channels, WebSocket, SQLite, Python, HTML/CSS*

- Реализовал систему обмена сообщениями в реальном времени через WebSocket с использованием Django Channels (асинхронные consumers)
- Спроектировал структуру БД на SQLite с моделями Room, Message и User, настроил CRUD-операции через Django ORM
- Разработал интерфейс чата с динамическим обновлением через JavaScript (без перезагрузки страницы)
- Реализовал формы аутентификации (регистрация/вход/выход) с использованием встроенной Django-логики

Car Sharing Database Design | *PostgreSQL, Database Design*

- Спроектировал концептуальную, логическую и физическую модели базы данных для сервиса каршеринга
- Реализовал физическую модель в PostgreSQL: создание таблиц (пользователи, автомобили, договоры, страховки, повреждения), установка связей и ограничений целостности
- Обеспечил соблюдение бизнес-правил: уникальность автомобилей по VIN/номеру, проверка дат договора, ограничение на несколько активных аренд у пользователя

Технические навыки

Языки программирования: Python, C/C++, Java, SQL (Postgres)
Фреймворки: Django, pyTelegramBotAPI, FastAPI, Crow, PyTest
Инструменты разработчика: Docker, Git, Google Collab, VS Code, IntelliJ, CLion, pgAdmin
Библиотеки: Requests, pandas, NumPy, Matplotlib, tkinter, pyzbar, psycopg2, cv2, torch, transformers, sklearn