Тестовое задание для разработчика на TypeScript

Задача:

Необходимо разработать приложение для загрузки, обработки и отображения изображений. Приложение должно включать backend на NestJS с авторизацией через JWT, обработку изображений в фоновом режиме через очередь (BullMQ), использование библиотеки Sharp для сжатия изображений и gRPC для взаимодействия между сервисами. Всего должны быть 3 сервиса: Api Gateway(http взаимодействие с frontend), Auth Service, Image Processing Service)
Фронтенд должен быть реализован на React с TypeScript.

Требования к backend (NestJS)

1. Сервис авторизации (Auth service)

- Реализовать регистрацию и вход пользователя (email + пароль)
- Использовать JWT для аутентификации
- Защитить маршруты, требующие авторизации

2. Сервис загрузки изображений (Image Processing Service)

- Создать gRPC endpoint для загрузки изображений (проверка на авторизацию)
- Сохранять оригинальное изображение в хранилище (S3 Minio)
- Записывать информацию о загруженном изображении в базу данных (PostgreSQL)
- Поля в БД: id, оригинальное имя, путь к файлу, статус обработки, размеры, id пользователя, дата загрузки
- Создать gRPC endpoint для выдачи информации о последнем изображении пользователя

3. Обработка изображений (BullMQ + Sharp)

- Создать очередь для обработки загруженных изображений
- Использовать библиотеку Sharp для:
    - Конвертации в webp с качеством 80%
- Обновлять статус обработки в БД
- Сохранять обработанные изображения в S3

4. API Gateway

- Методы:
    - Загрузка изображения (stream)
    - Получение информации об изображении
    - Получение оптимизированной версии изображения

5. Docker:

- docker-compose

Требования к frontend (React + TypeScript)

1. Страница авторизации/регистрации

- Форма входа (email + пароль)
- Форма регистрации (email + пароль + подтверждение пароля)
- Обработка и отображение ошибок

2. Главная страница

- Кнопка загрузки нового изображения
- Отображение загруженного изображения:
    - Само изображение (если обработано)
    - Статус обработки
    - Дата загрузки

Также разрешается использование ui библиотек для проекта.

Технологический стек

Backend

- NestJS
- Typescript
- PostgreSQL
- TypeORM
- BullMQ + Redis

- Sharp
- gRPC
- JWT

Frontend

- React 18+
- TypeScript
- React Query

Дополнительные задания (по желанию)

1. Реализовать получение данных об обработанном изображении по Websockets

Рекомендации

Использовать готовые шаблоны и решения для ускорения разработки.
Опираться на TypeScript для типизации всего проекта.
Уделить внимание структурированию кода и поддержке его читаемости.
Проверять функциональность до отправки, обеспечивая корректное выполнение всех требований спецификации.

Ожидаемый результат:

Рабочая программа загруженная в github, инструкция по её запуску и тестированию, наличие docker-compose файла. Выполненные работы принимаются до 20 апреля включительно по ссылке https://forms.yandex.ru/u/67ebad8d493639670a048ddf/

Удачи!