Тестовое задание для Junior Python разработчика

Задание: Создать API-сервис для управления списком задач ("To-Do List") с использованием FastAPI и PostgreSQL. Результат необходимо продемонстрировать в виде:

1. Ссылки на развернутое приложение (например, в Яндекс.Облаке, Heroku или любом другом доступном сервере).
2. Репозитория на GitHub с исходным кодом проекта.

Функциональные требования:

1. **Эндпоинты**:
    - `POST /tasks/`: создание задачи.
        - Входные параметры: название (`title`), описание (`description`), статус (`status`, значения: `todo`, `in_progress`, `done`).
    - `GET /tasks/`: получение списка всех задач с возможностью фильтрации по статусу.
    - `GET /tasks/{task_id}/`: получение информации о конкретной задаче.
    - `PUT /tasks/{task_id}/`: обновление задачи по ID.
    - `DELETE /tasks/{task_id}/`: удаление задачи по ID.
2. **База данных**:
    - Использовать PostgreSQL.
    - Схема БД - на усмотрение исполнителя
3. **Docker**:
    - Создать `Dockerfile` для приложения.
    - Создать `docker-compose.yml` для запуска FastAPI и PostgreSQL.
4. **Документация API**:
    - Использовать встроенную документацию FastAPI (Swagger).

Нефункциональные требования:

1. **Тесты**:
    - Написать 2-3 теста с использованием `pytest` для проверки работы API.
        - Проверка создания задачи.
        - Проверка получения задачи по ID.
        - Проверка фильтрации задач по статусу.
2. **Развертывание**:
    - Развернуть приложение на сервере. Можно использовать бесплатный грант Яндекс.Облака или любой другой хостинг.
    - Предоставить ссылку на работающий API.

3. **Инструкция**:
    - Подготовить файл `README.md`, который должен содержать:
        - Описание проекта.
        - Инструкции по локальному запуску.
        - Инструкции по развертыванию на сервере.
        - Примеры запросов к API с использованием `curl` или Postman.

Оценочные критерии:

1. Умение работать с FastAPI и PostgreSQL.
2. Знание Docker и навыки настройки контейнеризации.
3. Способность писать базовые тесты с использованием `pytest`.
4. Умение развернуть проект на сервере.
5. Чистота, структурированность и читаемость кода.
6. Наличие понятной документации.

Дополнительное задание:

- Реализовать визуальную часть(например на VueJS)

Формат сдачи:

1. Ссылка на развернутое приложение.
2. Ссылка на репозиторий GitHub с проектом.