Тестовое задание: Загрузка и обработка файлов

Цель:

Разработать Django REST API, который позволяет загружать файлы на сервер, а затем асинхронно обрабатывать их с использованием Celery.

Требования:

1. Создать Django проект и приложение.
2. Использовать Django REST Framework для создания API.
3. Реализовать модель File, которая будет представлять загруженные файлы. Модель должна содержать поля:
    - file: поле типа FileField, используемое для загрузки файла.
    - uploaded_at: поле типа DateTimeField, содержащее дату и время загрузки файла.
    - processed: поле типа BooleanField, указывающее, был ли файл обработан.
4. Реализовать сериализатор для модели File.
5. Создать API эндпоинт upload/, который будет принимать POST-запросы для загрузки файлов. При загрузке файла необходимо создать объект модели File, сохранить файл на сервере и запустить асинхронную задачу для обработки файла с использованием Celery. В ответ на успешную загрузку файла вернуть статус 201 и сериализованные данные файла.
6. Реализовать Celery задачу для обработки файла. Задача должна быть запущена асинхронно и изменять поле processed модели File на True после обработки файла.
7. Реализовать API эндпоинт files/, который будет возвращать список всех файлов с их данными, включая статус обработки.

Дополнительные требования:

1. Использовать Docker для развертывания проекта.
2. Реализовать механизм для обработки различных типов файлов (например, изображений, текстовых файлов и т.д.).
3. Предусмотреть обработку ошибок и возвращение соответствующих кодов статуса и сообщений об ошибках.

Примечания:

- При выполнении задания рекомендуется использовать официальную документацию Django, DRF, Celery и Docker.
- Вы можете использовать любые дополнительные библиотеки, если считаете нужным.

Усложения:

- Тесты (постарайтесь достичь покрытия в 70% и больше)
- Опишите, как изменится архитектура, если мы ожидаем большую нагрузку
- Попробуйте оценить, какую нагрузку в RPS сможет выдержать ваш сервис