Тестовое задание

Время выполнения задания оценивается до 12 человеко-часов (для специалиста уровня Junior).

Результат выполнения задания должен быть выложен в публичный репозиторий github и помимо кода проекта содержать подробные инструкции по сборке и запуску.

Задача 1

1. С помощью Docker (предпочтительно - docker-compose) развернуть образ с любой опенсорсной СУБД (предпочтительно - PostgreSQL). Предоставить все необходимые скрипты и конфигурационные (docker/compose) файлы для развертывания СУБД, а также инструкции для подключения к ней. Необходимо обеспечить сохранность данных при рестарте контейнера (то есть - использовать volume-ы для хранения файлов СУБД на хост-машине.
2. Реализовать на Python3 веб сервис (с помощью FastAPI или Flask, например), выполняющий следующие функции:
    2.1. В сервисе должно быть реализован POST REST метод, принимающий на вход запросы с содержимым вида {"questions_num": integer}.
    2.2. После получения запроса сервис, в свою очередь, запрашивает с публичного API (англоязычные вопросы для викторин) https://jservice.io/api/random?count=1 указанное в полученном запросе количество вопросов.
    2.3. Далее, полученные ответы должны сохраняться в базе данных из п. 1, причем сохранена должна быть как минимум следующая информация (название колонок и типы данный можете выбрать сами, также можете добавлять свои колонки): 1. ID вопроса, 2. Текст вопроса, 3. Текст ответа, 4. - Дата создания вопроса. В случае, если в БД имеется такой же вопрос, к публичному API с викторинами должны выполняться дополнительные запросы до тех пор, пока не будет получен уникальный вопрос для викторины.
    2.4. Ответом на запрос из п.2.а должен быть предыдущей сохранённый вопрос для викторины. В случае его отсутствия - пустой объект.
3. В репозитории с заданием должны быть предоставлены инструкции по сборке докер-образа с сервисом из п. 2., его настройке и запуску. А также пример запроса к POST API сервиса.
4. Желательно, если при выполнении задания вы будете использовать docker-compose, SQLAalchemy, пользоваться аннотацией типов.

Задача 2

Необходимо реализовать веб-сервис, выполняющий следующие функции:

1. Создание пользователя;
2. Для каждого пользователя - сохранение аудиозаписи в формате wav, преобразование её в формат mp3 и запись в базу данных и предоставление ссылки для скачивания аудиозаписи.

Детализация задачи:

1. С помощью Docker (предпочтительно - docker-compose) развернуть образ с любой опенсорсной СУБД (предпочтительно - PostgreSQL). Предоставить все необходимые скрипты и конфигурационные (docker/compose) файлы для развертывания СУБД, а также инструкции для подключения к ней. Необходимо обеспечить сохранность данных при рестарте контейнера (то есть - использовать volume-ы для хранения файлов СУБД на хост-машине.
2. Реализовать веб-сервис со следующими REST методами:
   - 2.1. Создание пользователя, POST:
     - 2.1.1. Принимает на вход запросы с именем пользователя;
     - 2.1.2. Создаёт в базе данных пользователя заданным именем, так же генерирует уникальный идентификатор пользователя и UUID токен доступа (в виде строки) для данного пользователя;
     - 2.1.3. Возвращает сгенерированные идентификатор пользователя и токен.
   - 2.2. Добавление аудиозаписи, POST:
     - 2.2.1. Принимает на вход запросы, содержащие уникальный идентификатор пользователя, токен доступа и аудиозапись в формате wav;
     - 2.2.2. Преобразует аудиозапись в формат mp3, генерирует для неё уникальный UUID идентификатор и сохраняет их в базе данных;
     - 2.2.3. Возвращает URL для скачивания записи вида http://host:port/record?id=id_записи&user=id_пользователя.
   - 2.3. Доступ к аудиозаписи, GET:
     - 2.3.1. Предоставляет возможность скачать аудиозапись по ссылке из п 2.2.3.
3. Для всех сервисов метода должна быть предусмотрена предусмотрена обработка различных ошибок, возникающих при выполнении запроса, с возвращением соответствующего HTTP статуса.
4. Модель данных (таблицы, поля) для каждого из заданий можно выбрать по своему усмотрению.
5. В репозитории с заданием должны быть предоставлены инструкции по сборке докер-образа с сервисами из пп. 2. и 3., их настройке и запуску. А также пример запросов к методам сервиса.
6. Желательно, если при выполнении задания вы будете использовать docker-compose, SQLAlchemy, пользоваться аннотацией типов.