**Техническое задание:**

«Реализовать REST API, вычисляющее частоту встречи символов по заданной строке. Результат должен быть отсортирован по убыванию количества вхождений символа в заданную строку.

Пример входной строки: “aaaaabcccc”

Пример выходного результата: “a”: 5, “c”: 4, “b”: 1»

**Возможности приложения:**

Регистрация:

Пользователь переходит на /api/singup и отправляет методом POST json:

{

    " email ": " его почта ",

    " password ": " его пароль ",

    " name ": " его имя "

}

Сервер проверяет, зарегистрирован ли такой пользователь в БД по почте. Если существует, выбрасывает отказ с просьбой авторизоваться по /api/login. Если не существует, то вносит пользователя в БД и возвращает ему уникальный token, сгенерированный UUID библиотекой, в headres (постман поддерживает просмотр и отправку). По умолчанию новому пользователю присваивается статус DEFAULT, которое можно поменять через БД вручную через поле “role” на ADMIN, чтобы открыть дополнительные возможности приложения. Дальше пользователь получает сообщение об успешной регистрации.

Авторизация:

Пользователь переходит на /api/login и отправляет методом POST json:

{

    " email ": " его почта ",

    " password ": " его пароль "

}

Сервер проверят, зарегистрирован ли такой пользователь в БД по почте. Если существует, возвращает ему его уникальный token в headres (постман поддерживает просмотр и отправку) и приветственное сообщение с именем, указанным при регистрации, об успешной авторизации. Если не существует, то выбрасывает отказ с просьбой зарегистрироваться по /api/signup.

Отправка сообщения в БД:

Пользователь переходит на /api/messages и отправляет методом POST json:

{

    "text": "текст сообщения"

}

В headers он передает «Authorization : его уникальный токен», чтобы БД понимала, кто пишет сообщение.

Сервер записывает в БД сообщение со временем Timestamp и возвращает ответ об успешной отправке сообщения.

Получение всех сообщений из БД:

Пользователь переходит на /api/messages и отправляет методом GET запрос на получение всех сообщений, которые хранятся в БД.

Сервер выводит все сообщения пользователю в теле ответа с их уникальными ID.

Удаление сообщения из БД:

Пользователь переходит на /api/messages/id, где id – это уникальный ID конкретного сообщения и отправляет методом DELETE запрос на удаление данного сообщения из БД.

В headers он пользователь передает «Authorization : его уникальный токен», чтобы БД понимала, есть ли права для удаления данного сообщения.

**ВАЖНО!** Пользователь со статусом DEFAULT может удалить из БД только созданное им сообщение. Пользователь со статусом ADMIN может удалять любое сообщение.

Сервер проверяет права доступа по пользователя, удаляет сообщение или не удаляет в зависимости от совпадения токена и выводит сообщение об статусе операции.

Получение списка пользователей:

Пользователь переходит на /api/users и отправляет методом GET запрос на получение всех пользователей, которые зарегистрированы в БД.

В headers он пользователь передает «Authorization : его уникальный токен», чтобы БД понимала, есть ли права для получения списка пользователей.

**ВАЖНО!** Доступ к этой функции есть только у пользователей со статусом ADMIN.

Сервер проверяет права доступа по пользователя, возвращает список пользователей или не возвращает в зависимости от совпадения токена и выводит сообщение об статусе операции.

**Компоненты приложения:**

Структура базы данных:

|  |  |  |
| --- | --- | --- |
| OnteToMany Bi-directional |  |  |
|  |  |  |
| users |  | messages |
| id (autoincrement) |  | id (autoincrement) |
| token (unique generated value by UUID, PK) |  | user\_token (FK) |
| email |  | text\_of\_msg |
| password |  | timestamp\_of\_msg |
| name |  |  |
| role (default "DEFAULT") |  |  |