Project Document: Event Management System

1. Project Overview

The **Event Management System** is a web-based platform for managing events, RSVPs, and user accounts.

Key Features:

- **Admin Functionality:**
  - Create, update, and delete events.
  - View RSVP lists for events.
- **User Functionality:**
  - View upcoming events.
  - RSVP to events.
- **Authentication:**
  - JWT-based authentication for secure access.
  - Role-based access (Admin/User).

This project uses a modern tech stack:

- **Frontend:** React (styled for a clean and crisp user experience).
- **Backend:** Node.js with Express.js.
- **Database:** MariaDB.

---

2. Project Setup

2.1 Backend Setup

1. Open the backend folder in visual studio code, open up a terminal, and Install dependencies:

```
npm install
```

2. Open up the .env folder in the backend code and update the DB_PASSWORD and DB_USER

DB_USER=root

DB_PASSWORD=<your_db_password>

3. Import the database:

- Open a normal terminal where the create_database.sql file exists
- Run the SQL script (create_database.sql) using MariaDB/MySQL:

```
mysql -u <username> -p < create_database.sql
```

if for any reason the command doesn't work the .sql file simply contains SQL commands that you can run in the mariadb terminal

4. Start the server:

```
npm run start
```

The backend will run on http://localhost:5000.

2.2 Frontend Setup

1. Open the frontend folder in a **new vscode window (keep the backend service up)**
2. Install dependencies:

```
npm install
```

The frontend will run on http://localhost:3000.

---

3. Project Deliverables

3.1 Postman Workspace

- A Postman collection that tests all backend APIs.
- The collection should includes:
  - User registration and login.
  - Admin-only APIs for event management.
  - User APIs for viewing events and RSVPing.
- **Export both the environment variables and the collection.**

3.2 Mocha + Selenium Test Folder

- A folder named tests that contains:
  - **Mocha + Selenium Tests:** Automated end-to-end tests for the frontend, including:

- User login and navigation.
- Admin event creation and deletion.
- RSVP functionality.

3.3 SQL Script

- Any SQL instructions you had to use to setup your test cases should be placed inside a pdf file along with an explanation for why you needed to use them.
- As an example: to properly test all functionality you need to setup both a normal user and an admin user on the site, so you will need to manually run a query to create an admin user. Put the query as well as why you needed to run it in the file.

3.4 Submit your JMeter Test Plan file (.jmx) along with the HTML report generated after executing the test.

Test Plans:

1. **Load Test:**
    - Design a test to simulate 10 concurrent users for a duration of 2 minutes.
    - Each user should sequentially:
        - Send a request to the Login endpoint.
        - Retrieve all events using the Get All Events endpoint.
        - RSVP to an event using the appropriate endpoint.
2. **Stress Test:**
    - Create a test that gradually increases the number of users from 0 to 15.
    - Increase users in increments of 5 (i.e., 0 → 5 → 10 → 15).
    - This test will help evaluate how the system performs under increasing load.

Thank You
Edges For Training Team