Project Documentation: Campus Event Management System

Introduction

The Campus Event Management System is a web-based platform designed to facilitate event planning, management, and participation within a campus community. The system allows administrators to create and manage events, while users can view, RSVP, and interact with events based on their preferences. The system also integrates a calendar view to visualize events.

System Features

1. User Features

* Profile Management
  + Users can view their profile details, including username, email, and event preferences.
  + Event preferences are displayed as a comma-separated list of categories.
* Event Interaction
  + Users can view upcoming events and RSVP.
  + An event calendar displays events in a monthly view.
* Event Filtering
  + Users can filter events based on their preferences for easier discovery.

2. Admin Features

* Event Creation
  + Admins can create events by providing details like title, date, time, description, and category.
* Event Management
  + Admins can view all created events and modify or delete them as necessary.

3. Calendar Integration

* The system integrates a calendar view using FullCalendar.js to display events by date.
* Users and admins can see all events in a visually intuitive format.

System Design

Frontend

The frontend is developed using:

* HTML5: For structuring web pages.
* CSS3: For styling and responsiveness.
* JavaScript: For dynamic interactions and DOM manipulation.

Backend

The backend is developed using:

* Node.js: For server-side programming.
* Express.js: For routing and handling HTTP requests.
* MongoDB: For data storage and management.

Backend Routes

* User Registration
  + Endpoint: /register
  + Validates role and hashes passwords before saving to the database.
* User Login
  + Endpoint: /login
  + Verifies user credentials and generates a JWT token.

# Project Documentation: Campus Event Management System ## \*\*Introduction\*\* The Campus Event Management System is a web-based platform designed to facilitate event planning, management, and participation within a campus community. The system allows administrators to create and manage events, while users can view, RSVP, and interact with events based on their preferences. The system also integrates a calendar view to visualize events. --- ## \*\*System Features\*\* ### \*\*1. User Features:\*\* - \*\*Profile Management:\*\* - Users can view their profile details, including username, email, and event preferences. - Event preferences are displayed as a comma-separated list of categories. - \*\*Event Interaction:\*\* - Users can view upcoming events and RSVP. - An event calendar displays events in a monthly view. - \*\*Event Filtering:\*\* - Users can filter events based on their preferences for easier discovery. ### \*\*2. Admin Features:\*\* - \*\*Event Creation:\*\* - Admins can create events by providing details like title, date, time, description, and category. - \*\*Event Management:\*\* - Admins can view all created events and modify or delete them as necessary. ### \*\*3. Calendar Integration:\*\* - The system integrates a calendar view using FullCalendar.js to display events by date. - Users and admins can see all events in a visually intuitive format. --- ## \*\*System Design\*\* ### \*\*Frontend:\*\* The frontend is developed using: - \*\*HTML5\*\*: For structuring web pages. - \*\*CSS3\*\*: For styling and responsiveness. - \*\*JavaScript\*\*: For dynamic interactions and DOM manipulation. ### \*\*Backend:\*\* The backend is developed using: - \*\*Node.js\*\*: For server-side programming. - \*\*Express.js\*\*: For routing and handling HTTP requests. - \*\*MongoDB\*\*: For data storage and management. --- ## \*\*Technical Details\*\* ### \*\*1. User Profile Implementation:\*\* #### HTML Structure: The profile page includes placeholders for user details: ```html <h2>Welcome, <span id="user-name"></span></h2> <p>Email: <span id="user-email"></span></p> <p>Event Preferences: <span id="user-preferences"></span></p> ``` #### JavaScript: ```javascript document.addEventListener('DOMContentLoaded', function() { const token = localStorage.getItem('token'); if (token) { const decoded = jwt\_decode(token); document.getElementById('user-name').textContent = decoded.username; document.getElementById('user-email').textContent = decoded.email; document.getElementById('user-preferences').textContent = decoded.preferences.join(', '); } else { window.location.href = '/login.html'; } }); ``` ### \*\*2. Calendar Integration:\*\* #### HTML Structure: ```html <div id="calendar"></div> ``` #### JavaScript: ```javascript document.addEventListener('DOMContentLoaded', function() { const calendarEl = document.getElementById('calendar'); const calendar = new FullCalendar.Calendar(calendarEl, { plugins: ['dayGrid'], initialView: 'dayGridMonth', events: [ { title: 'Sample Event', date: '2024-12-20' } ] }); calendar.render(); }); ``` ### \*\*3. Backend Routes:\*\* - \*\*User Registration:\*\* - Endpoint: `/register` - Validates role and hashes passwords before saving to the database. - \*\*User Login:\*\* - Endpoint: `/login` - Verifies user credentials and generates a JWT token. --- ## \*\*Project Files Structure\*\* ```plaintext root/ |-- public/ | |-- styles.css | |-- scripts/ | |-- main.js |-- views/ | |-- index.html | |-- user-profile.html | |-- admin.html | |-- calendar.html |-- routes/ | |-- userRoutes.js |-- models/ | |-- User.js |-- app.js ``` --- ## \*\*Setup Instructions\*\* ### \*\*1. Prerequisites:\*\* - Node.js installed - MongoDB database configured ### \*\*2. Installation Steps:\*\* 1. Clone the repository: ```bash git clone <repository\_url> cd <project\_folder> ``` 2. Install dependencies: ```bash npm install ``` 3. Start the server: ```bash node app.js ``` 4. Access the application at `http://localhost:3000/`. --- ## \*\*Future Enhancements\*\* 1. Add email notifications for RSVP'd events. 2. Implement a search feature for events. 3. Provide support for recurring events in the calendar. --- ## \*\*Conclusion\*\* The Campus Event Management System simplifies event planning and participation within a campus environment. With an intuitive user interface and robust backend, it streamlines the process of managing and attending events. Convert this into something I can paste in a word document

Here's your project documentation in a format that's perfect for pasting into a Word document:

**# Project Documentation: Campus Event Management System**

**Introduction**

The Campus Event Management System is a web-based platform designed to facilitate event planning, management, and participation within a campus community. The system allows administrators to create and manage events, while users can view, RSVP, and interact with events based on their preferences. The system also integrates a calendar view to visualize events.

**System Features**

**1. User Features**

* **Profile Management**
  + Users can view their profile details, including username, email, and event preferences.
  + Event preferences are displayed as a comma-separated list of categories.
* **Event Interaction**
  + Users can view upcoming events and RSVP.
  + An event calendar displays events in a monthly view.
* **Event Filtering**
  + Users can filter events based on their preferences for easier discovery.

**2. Admin Features**

* **Event Creation**
  + Admins can create events by providing details like title, date, time, description, and category.
* **Event Management**
  + Admins can view all created events and modify or delete them as necessary.

**3. Calendar Integration**

* The system integrates a calendar view using FullCalendar.js to display events by date.
* Users and admins can see all events in a visually intuitive format.

**System Design**

**Frontend**

The frontend is developed using:

* **HTML5**: For structuring web pages.
* **CSS3**: For styling and responsiveness.
* **JavaScript**: For dynamic interactions and DOM manipulation.

**Backend**

The backend is developed using:

* **Node.js**: For server-side programming.
* **Express.js**: For routing and handling HTTP requests.
* **MongoDB**: For data storage and management.

**Technical Details**

**1. User Profile Implementation**

**HTML Structure**

The profile page includes placeholders for user details:

html

<h2>Welcome, <span id="user-name"></span></h2>

<p>Email: <span id="user-email"></span></p>

<p>Event Preferences: <span id="user-preferences"></span></p>

**JavaScript**

javascript

document.addEventListener('DOMContentLoaded', function() {

const token = localStorage.getItem('token');

if (token) {

const decoded = jwt\_decode(token);

document.getElementById('user-name').textContent = decoded.username;

document.getElementById('user-email').textContent = decoded.email;

document.getElementById('user-preferences').textContent = decoded.preferences.join(', ');

} else {

window.location.href = '/login.html';

}

});

**2. Calendar Integration**

**HTML Structure**

html

<div id="calendar"></div>

**JavaScript**

javascript

document.addEventListener('DOMContentLoaded', function() {

const calendarEl = document.getElementById('calendar');

const calendar = new FullCalendar.Calendar(calendarEl, {

plugins: ['dayGrid'],

initialView: 'dayGridMonth',

events: [

{ title: 'Sample Event', date: '2024-12-20' }

]

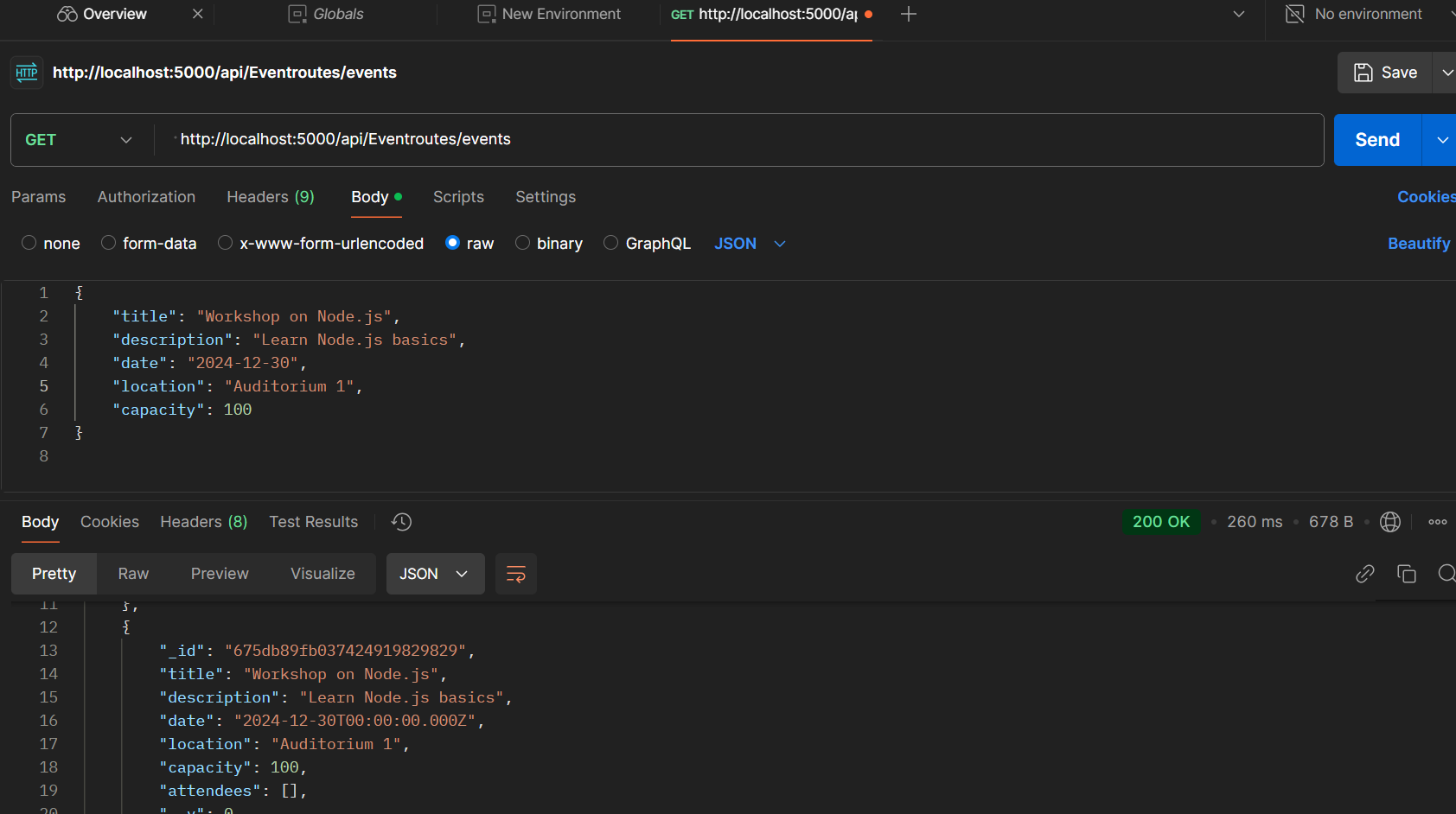
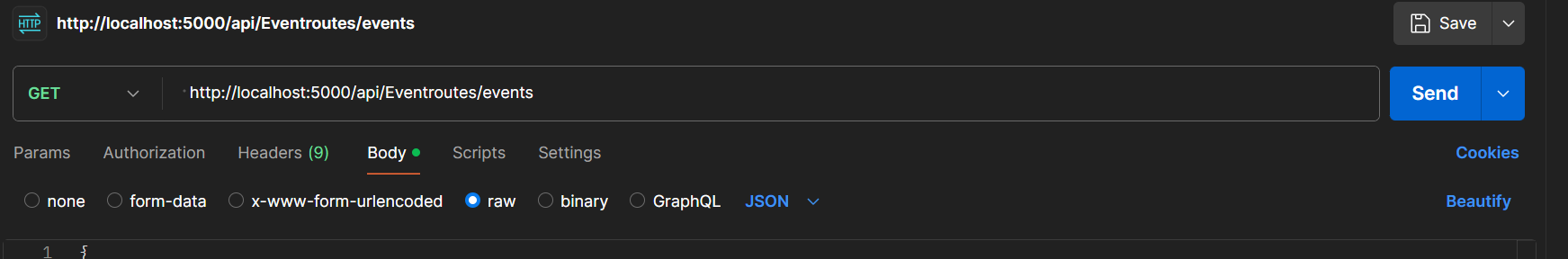
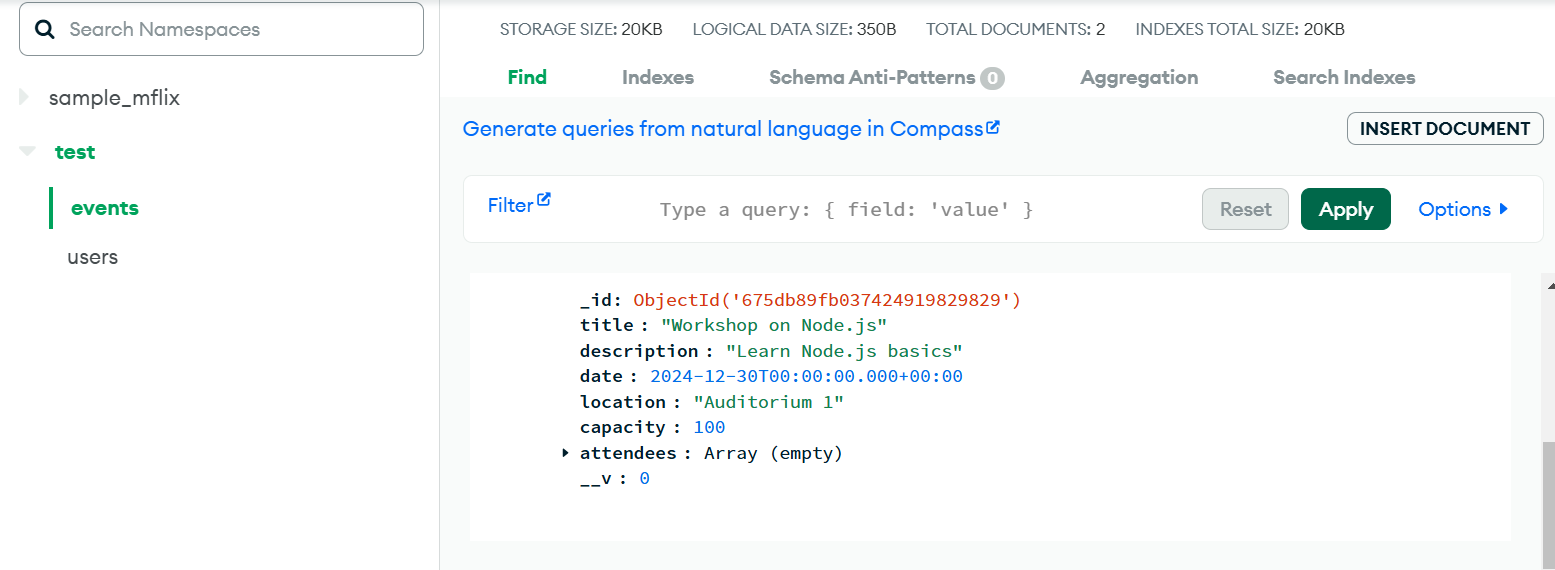
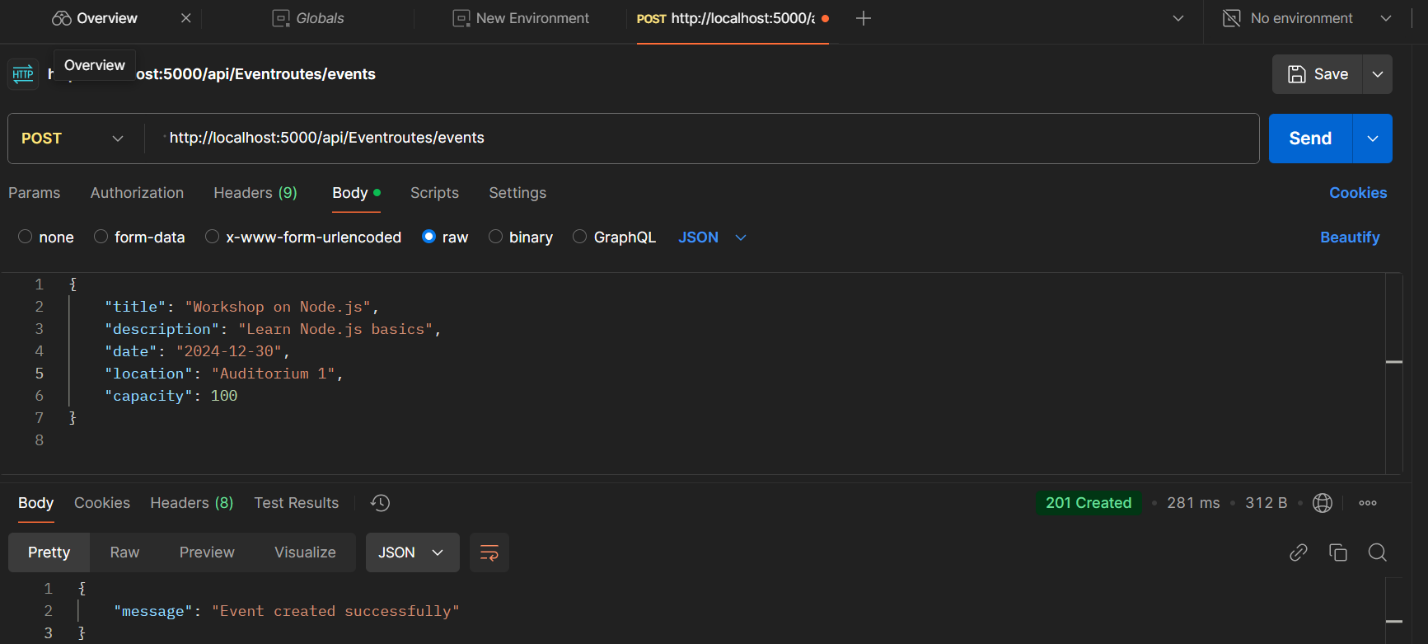
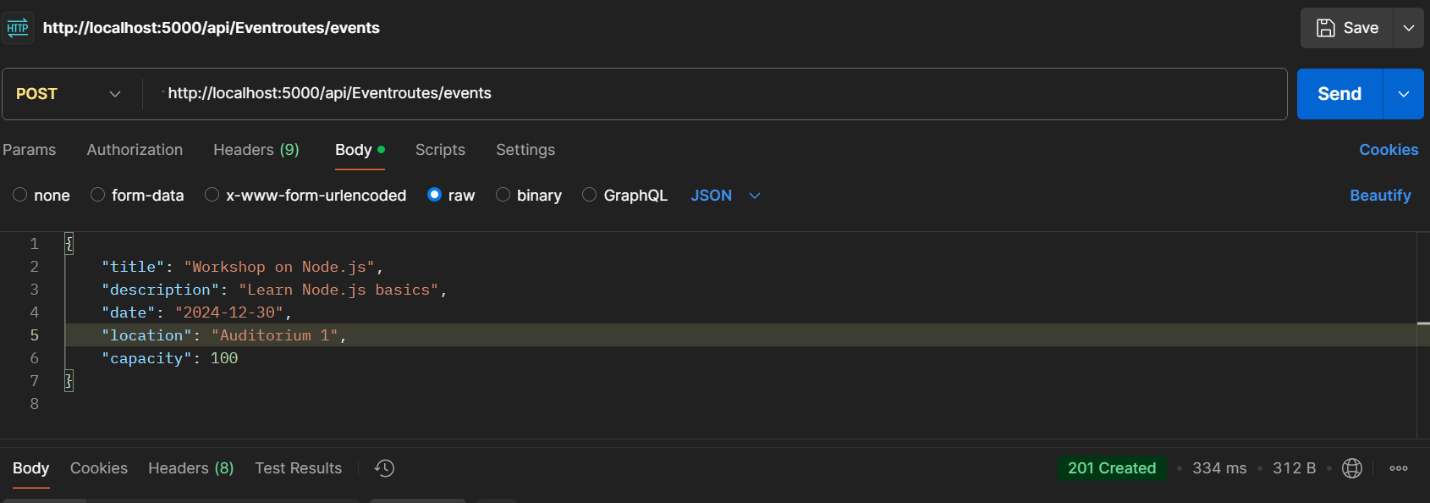
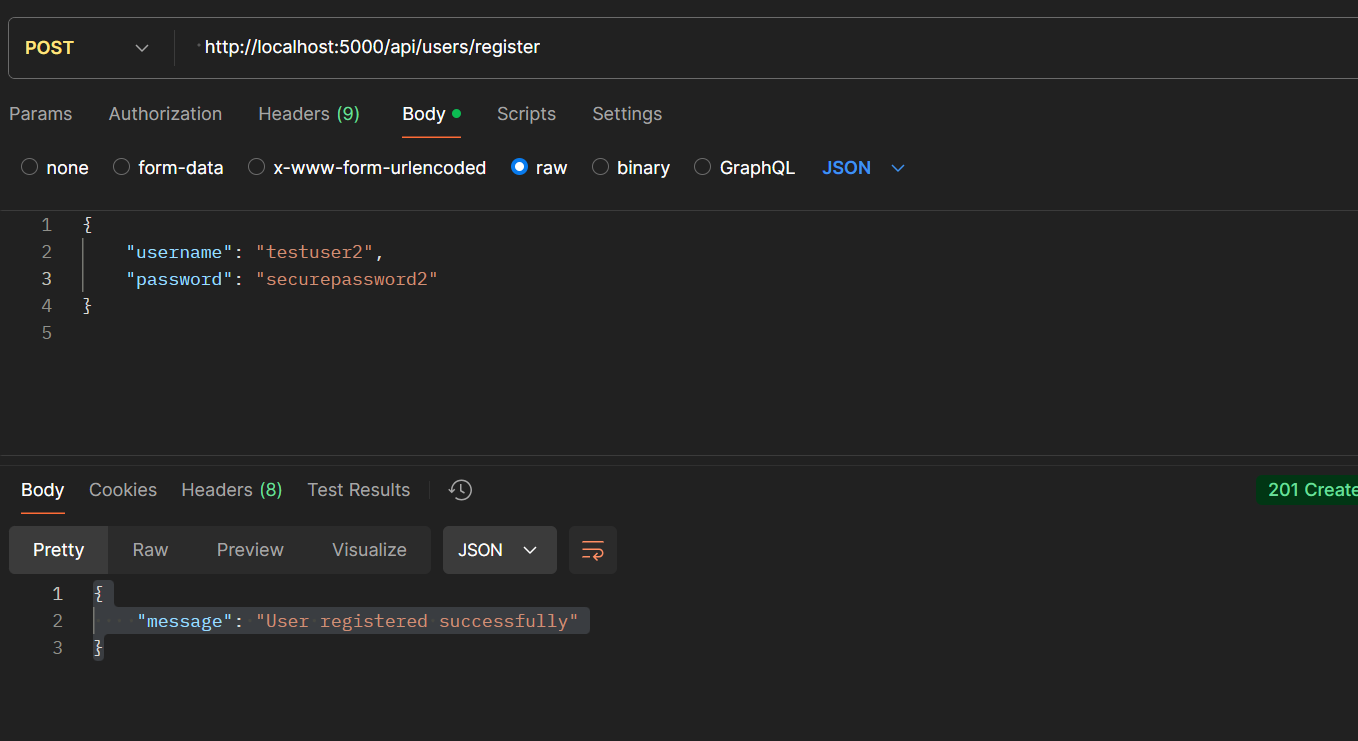
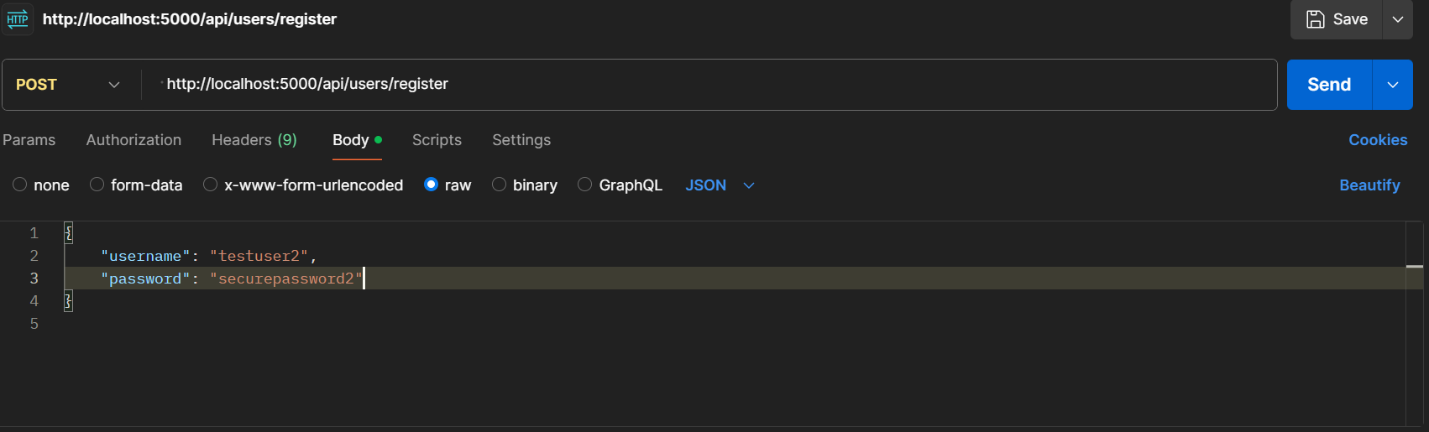
});

calendar.render();

});

**3. Backend Routes**

* **User Registration**
  + Endpoint: /register
  + Validates role and hashes passwords before saving to the database.
* **User Login**
  + Endpoint: /login
  + Verifies user credentials and generates a JWT token.

Screenshots Taken  


**Future Enhancements**

1. Add email notifications for RSVP'd events.
2. Implement a search feature for events.
3. Provide support for recurring events in the calendar.

**Conclusion**

The Campus Event Management System simplifies event planning and participation within a campus environment. With an intuitive user interface and robust backend, it streamlines the process of managing and attending events.