EventConnect

System Requirement Specification

PREPARED FOR

Dr.Antonios

PREPARED BY

Diamon Goffe

Summer 2024

EventConnect

123 Goffe STREET
New Haven, CT

Table of Contents

1.0 Project Name & Contribution

Project Name: Event Connect

Link to GitHub repo: https://github.com/DiamonGoff/CSC400.git

Diamon Goffe: Backend Development, API integration, Frontend Development, Registration implementation, Front end Development an UI/UX design

2.0 Background and Project Conception

Event Connect is designed to streamline the planning and attending of birthday parties. The application provides tools for organizers to search and book venues, manage guest lists, and send invitations. Attendees can find travel and lodging options near the event venue and interact with each other via social media integration. The goal is to ensure a seamless experience for both organizers and attendees, making birthday party planning more efficient and enjoyable.

Core functionalities of EventConnect include:

Create an account/profile: Users can create a personal account and profile to access all the features of Event Connect.

Search for venues: Organizers can search for suitable venues for their birthday parties the application.

Manage guest lists and send invitations: Organizers can create and manage guest lists, and send out invitations to attendees.

Find travel and lodging options: Attendees can search for travel and lodging options near the event venue to facilitate their attendance.

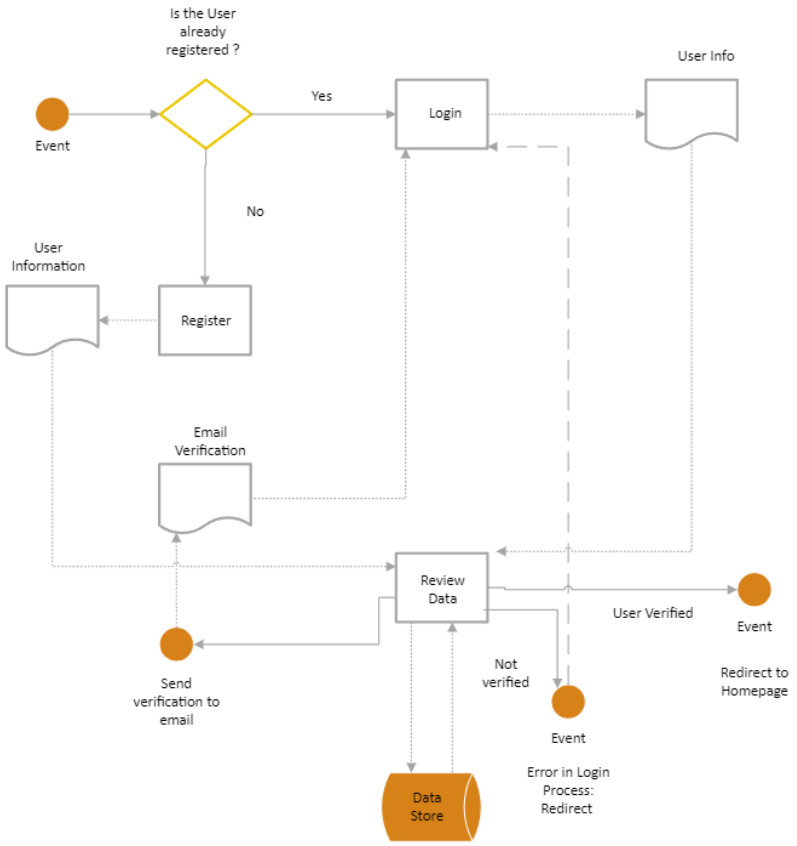
Share and interact on social media platforms: Attendees can share event details and interact with each other via integrated social media platforms.

Event Connect streamlines the birthday party planning process from start to finish, ensuring a seamless and enjoyable experience for both organizers and attendees.

3.0 Business Workflow

This section introduces diagrams simulating Business Process Model and Notation (BPMN) to articulate intricate workflows within the EventConnect platform. These diagrams serve to elucidate the systematic flow of activities and the interconnection of various use cases, thereby providing insight into the operational dynamics of EventConnect.

3.1 User Registration and Login Process]



This diagram depicts the steps involved in the user registration and login process for EventConnect.

4.0 User Stories

As an organizer, I want to create an account and sign into the application so I can begin to plan birthday gatherings.

As an organizer, I want to create a profile with personal details like my name and contact information so that the app can store and manage my events effectively.

As an organizer, I want to search for venues based on location, capacity, and budget using the Google Maps API so that I can find the ideal spot for the birthday celebration.

As an organizer, I want to input and manage event details such as date, time, and guest list so that all the information is centralized and easy to access.

As an organizer, I want to send invitations and updates to attendees through the app so that everyone is informed about the event details.

As an attendee, I want to search for transportation and lodging options near the event venue using the Google Places API so that I can easily make travel arrangements.

As an attendee, I want to share the birthday event details via a link so that my friends and family can easily access the event information.

As an attendee, I want to receive notifications and updates from the organizer so that I stay informed about any changes or important information regarding the event.

As a user, I want to access the event page to view all the details about the birthday gathering, including venue, date, time, and guest list.

As a user, I want to confirm my attendance to the birthday event so that organizers know who will be attending.

As a user, I want to view hotel suggestions near the event venue so that I can conveniently book my stay.

As an Admin, I want to manage content related to birthday events, like venues and service providers, so that the information provided to users remains

relevant and current.

5.0 User Cases

The use case diagram for EventConnect visually communicates the various interactions that users, organizers, and attendees can have with the EventConnect platform. It outlines the functionalities available to each actor, such as registering for an account, managing event details, searching for venues, managing guest lists, sending invitations, managing RSVPs, exploring gift ideas, interacting on social media, and sending notifications. Each use case connected to the user, organizer, or attendee represents an action they can perform directly through the platform's interface.

5.1 Use Case Descriptions

This section provides a comprehensive step-by-step depiction of each use case represented in the use case diagram. Actors are identified by name: User, Organizer, Attendee, System, Database, Email Service, and Google Maps API. The System references actions performed by the EventConnect application itself. The descriptions have been updated to reflect newly implemented features, such as task management, venue management, hotel suggestions, and RSVP list management.

5.1.1 User Registration

This section provides a comprehensive step-by-step depiction of each use case represented in the use case diagram. Actors are identified by name: User, Organizer, Attendee, System, Database, Email Service, and Google Maps API. The System references actions performed by the EventConnect application itself.

Option 1: Successful Registration

1. The User submits a request to register via the registration form.
2. The System checks with the Database for username availability.
3. The Database confirms the username is unique.
4. The System sends a validation email to the User.
5. The System confirms registration and informs the User of successful account creation.

Option 2: Unsuccessful Registration Due to Duplicate Username

1. The User attempts to register with a chosen username.
2. The System checks with the Database for username availability.
3. The Database indicates the username already exists.
4. The System informs the User of the duplicate username and requests a different one.

Option 3: Unsuccessful Email Validation

1. The User attempts to validate their email with a provided code.
2. The System checks the code and finds it incorrect.
3. The System notifies the User of the invalid code and prompts for re-entry or resends a new code.

The registration process has been refined to include better error handling, especially around email validation and username availability, ensuring users experience a smooth sign-up process.

5.1.2 User Login

Option 1: Successful Login

1. The User enters their credentials to log in.
2. The System validates the credentials with the Database.
3. The Database confirms the credentials are correct.
4. The System grants access and welcomes the User.

Option 2: Unsuccessful Login Due to Incorrect Credentials

1. The User enters their credentials.
2. The System attempts to validate the credentials with the Database.
3. The Database finds the credentials do not match any user account.
4. The System denies access and alerts the User to incorrect credentials.

I've implemented JWT-based authentication, providing users with secure, continuous access throughout their session on the app.

5.1.3 Create Event

Option 1: Successful Event Creation

1. The Organizer requests to create an event and submits the event details.
2. The System saves the event details in the Database.
3. The System successfully creates the event and notifies the Organizer.

Option 2: Unsuccessful Event Creation Due to Missing Details

1. The Organizer submits event details.
2. The System checks the submission and identifies missing details.
3. The System requests the Organizer to provide the missing details to complete the event creation.

The event creation process now allows organizers to store comprehensive event details, including venue information, guest lists, and any special requirements. With Google Maps integration, organizers can now precisely geolocate venues, enhancing the planning process.

5.1.4 Search for Venues

Option 1: Successful Venue Search

1. The Organizer searches for venues using the venue search form.
2. The System queries the Google Maps API to retrieve venue information based on location, capacity, and amenities.
3. The System displays a list of available venues to the Organizer.
4. The Organizer selects a preferred venue and saves this selection within the app for further planning.
5. The System confirms the venue selection and stores it in the Database.

Option 2: Unsuccessful Venue Search Due to Location Unavailability

1. The Organizer searches for venues using the venue search form.
2. The System queries the Google Maps API to retrieve venue information.
3. The System displays a list of available venues to the Organizer.
4. The Organizer searches for a specific location but finds no suitable venues available.
5. The System notifies the Organizer that no venues match the criteria and suggests adjusting the search parameters or trying a different location.

5.1.5 Manage Guest List

Option 1: Successful Guest List Management

1. The Organizer adds, edits, or removes guests from the guest list.
2. The System updates the guest list in the Database.
3. The System displays the updated guest list to the Organizer.

Option 2: Unsuccessful Guest List Management Due to System Error

1. The Organizer adds, edits, or removes guests from the guest list.
2. The System attempts to update the guest list in the Database but encounters an error.
3. The System notifies the Organizer of the error and requests to try again later.

Organizers can now assign specific roles or groups to guests, allowing for more personalized invitations and communication strategies.

5.1.6 Send Invitations

Option 1: Successful Invitation Sending

1. The Organizer selects guests to invite and sends invitations.
2. The System uses the Email Service to send email invitations to the selected guests.
3. The System confirms that the invitations have been sent successfully.

Option 2: Unsuccessful Invitation Sending Due to Invalid Email Addresses

1. The Organizer selects guests to invite and sends invitations.
2. The System uses the Email Service to send email invitations to the selected guests.
3. The Email Service identifies invalid email addresses.
4. The System notifies the Organizer of the invalid email addresses and suggests correcting them.

The invitation-sending process has been improved with better error handling for invalid email addresses and a more efficient batch processing system for sending out large volumes of invites.

5.1.7 Manage RSVPs

Option 1: Successful RSVP Management

1. The Attendee submits an RSVP.
2. The System updates the RSVP status in the Database.
3. The System notifies the Organizer of the updated RSVP status.
4. The System displays the updated RSVP status to the Attendee.

Option 2: Unsuccessful RSVP Submission Due to System Error

1. The Attendee submits an RSVP.
2. The System attempts to update the RSVP status in the Database but encounters an error.
3. The System notifies the Attendee of the error and requests to try again later.

RSVP management has been expanded with new features, allowing organizers to view, filter, and update guest responses easily. Attendees now have a straightforward interface for submitting and managing their RSVPs

5.1.8 Explore Gift Ideas

Option 1: Successful Gift Purchase

1. The Attendee browses the gift registry and selects a gift.
2. The Attendee proceeds to purchase the selected gift.
3. The System processes the purchase and updates the registry in the Database.
4. The System confirms the purchase to the Attendee.

Option 2: Unsuccessful Gift Purchase Due to Inventory Issues

1. The Attendee browses the gift registry and selects a gift.
2. The Attendee proceeds to purchase the selected gift.
3. The System attempts to process the purchase but finds the gift is out of stock.
4. The System notifies the Attendee of the inventory issue and suggests choosing another gift.

Attendees can now browse gift suggestions tailored to the event and their preferences, complete with real-time inventory checks and a seamless purchase process.

5.1.9 Social Media Interaction

Option 1: Successful Event Link Sharing

1. The Attendee decides to share the event with others via social media.
2. The System generates a shareable link to the event.
3. The Attendee successfully posts the link on their chosen social media platform.
4. The System confirms the successful sharing of the event link.

Option 2: Unsuccessful Event Link Sharing Due to System Error

1. The Attendee attempts to share the event link via social media.
2. The System encounters an error while generating the shareable link or during the sharing process.
3. The System notifies the Attendee of the error and suggests trying again later.

5.1.10 Send Notifications

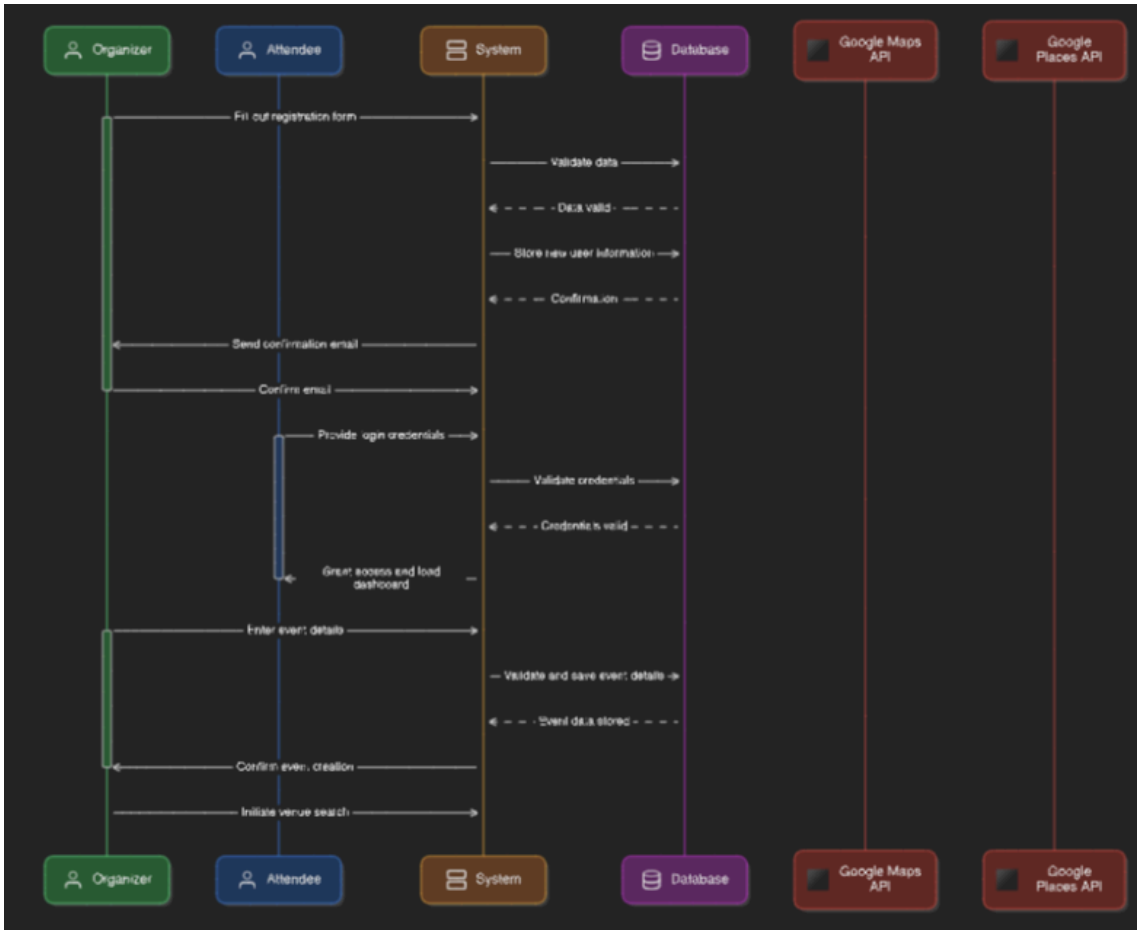
Option 1: Successful Notification Sending

1. The System detects an event update or reminder trigger.
2. The System composes a notification message.
3. The System sends the notification to the relevant users.
4. The Users receive the notification.

Option 2: Unsuccessful Notification Sending Due to System Error

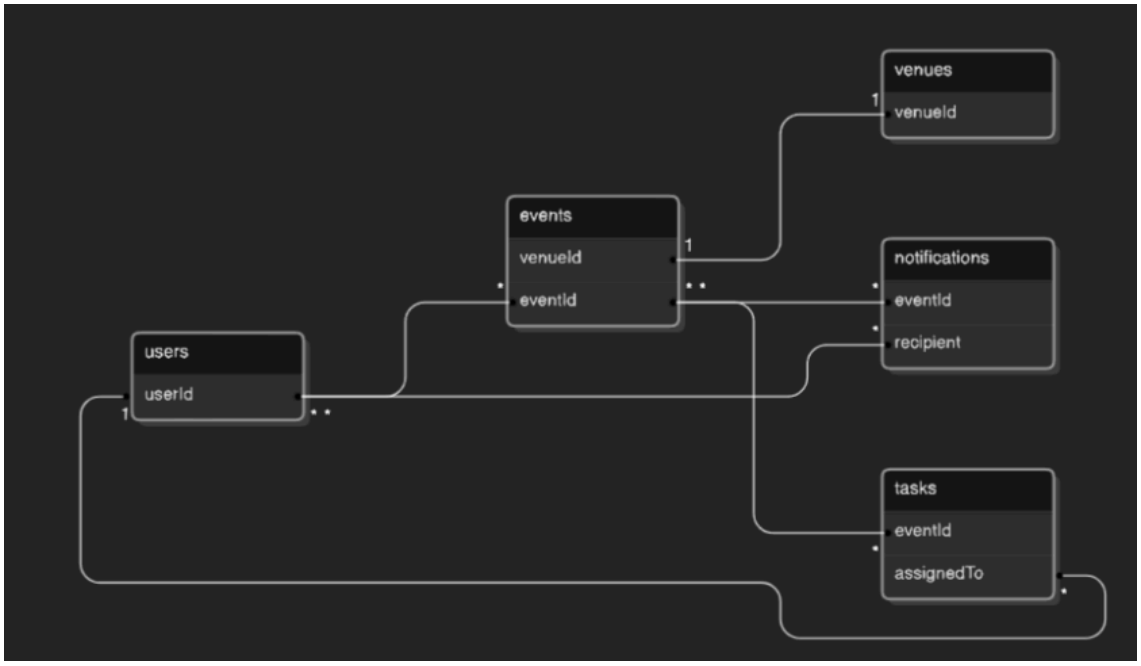
1. The System detects an event update or reminder trigger.
2. The System attempts to compose and send a notification message but encounters an error.
3. The System logs the error and retries sending the notification.

6.0 Sequence Diagrams



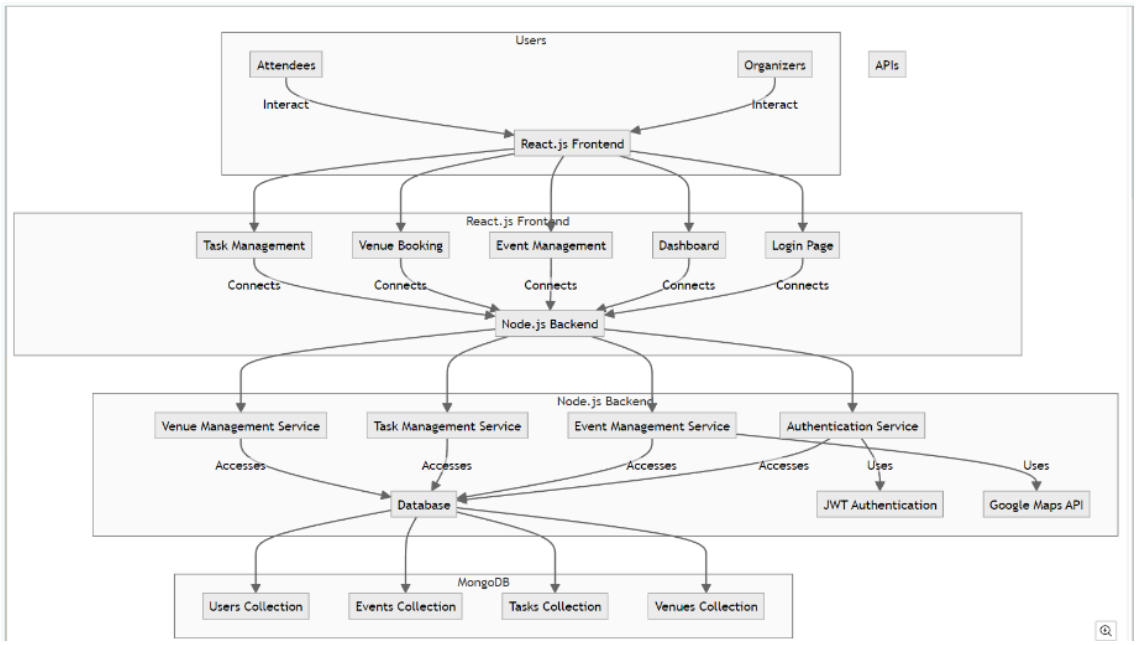
The sequence diagram for EventConnect illustrates the interactions between various entities, including Organizer, User, Profile, Event, Venue, GuestList, and Invitation. It outlines the processes involved in creating an account, updating profile details, planning an event, searching venues, managing guest lists, and sending invitations. Each interaction is detailed, showing the flow of data and actions taken by different entities within the system to ensure a seamless event planning and management experience.

7.0 Class Diagram



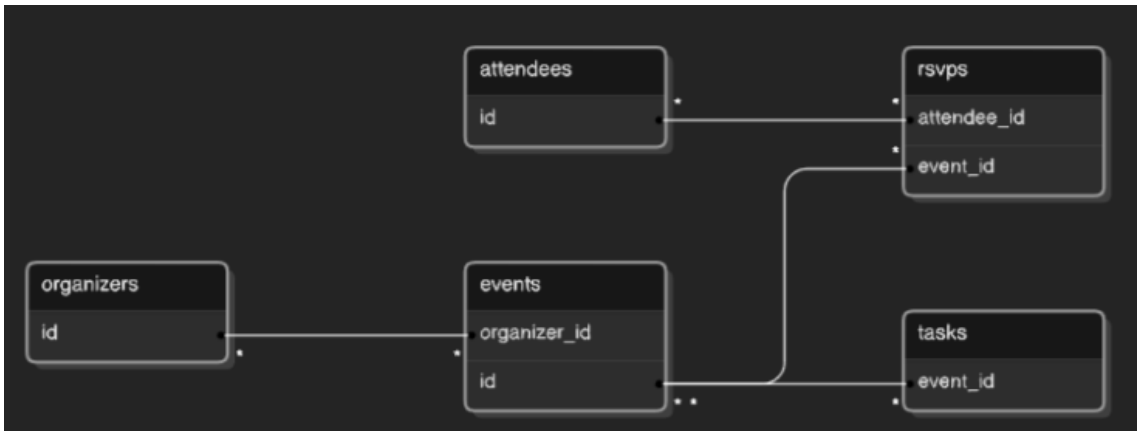
The class diagram for EventConnect illustrates the structure of the system by showcasing its various classes and the relationships between them. Key entities such as User, Organizer, Attendee, Event, Venue GuestList, Invitation, TravelOption, LodgingOption, SocialMedia, Admin, and Content are represented, highlighting their attributes and methods. This diagram provides a clear overview of how data is organized and managed within the EventConnect application, detailing the responsibilities of each class and the interactions between them.

8.0 Entity Relationship Diagrams



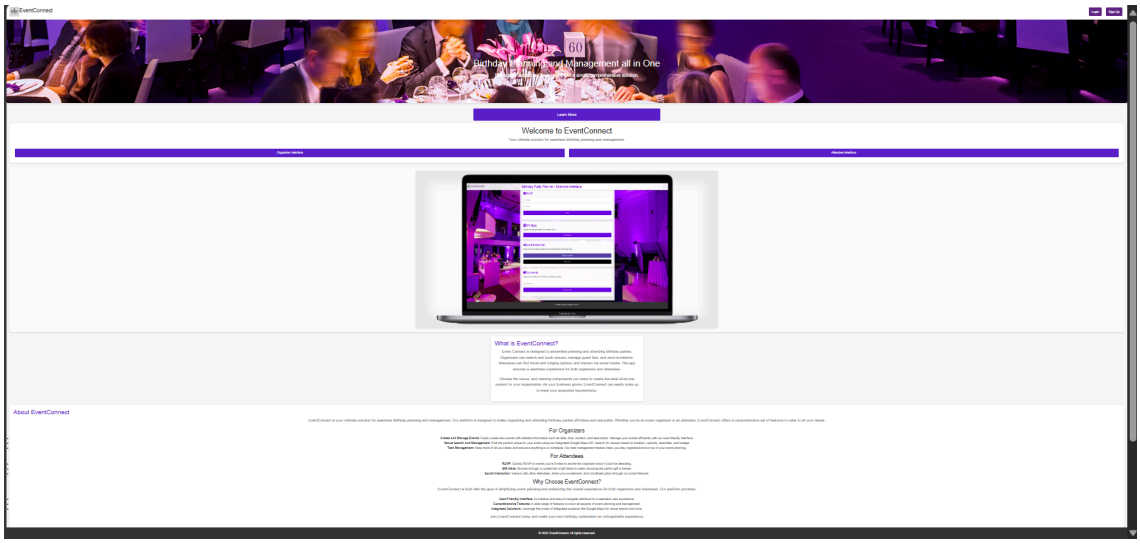
The Entity Relationship Diagram (ERD) for the EventConnect system provides a visual representation of the data structure and relationships within the application. It showcases various entities such as Users, Organizers, Attendees, Events, Venues, GuestLists, Invitations, TravelOptions, LodgingOptions, SocialMedia, Admins, and Content. Each entity is detailed with its attributes, highlighting how they interact with each other to support the core functionalities of EventConnect, such as event planning, venue guest management, and social media integration.

9.0 Use Case Diagram

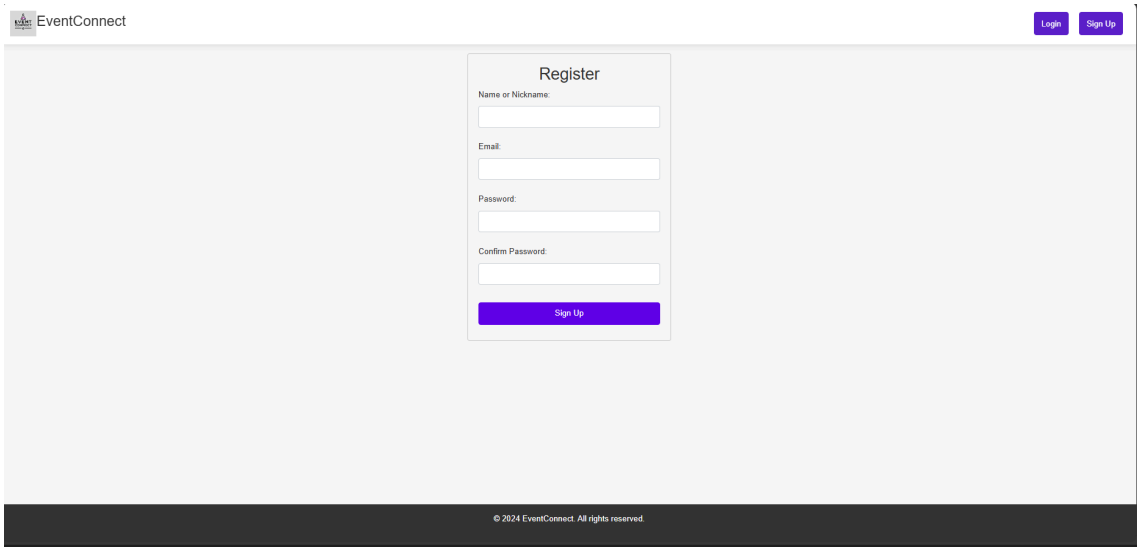


The use case diagram for EventConnect illustrates the various interactions that different actors—Admin, Attendee, Organizer, and User—have with the system. It highlights the functionalities available to each actor, such as account management, event planning, venue searching, guest list management, sending invitations, social media interactions, and content management. Each use case represents a specific action that can be performed by the actors, providing a clear visual overview of the system's capabilities and user interactions.

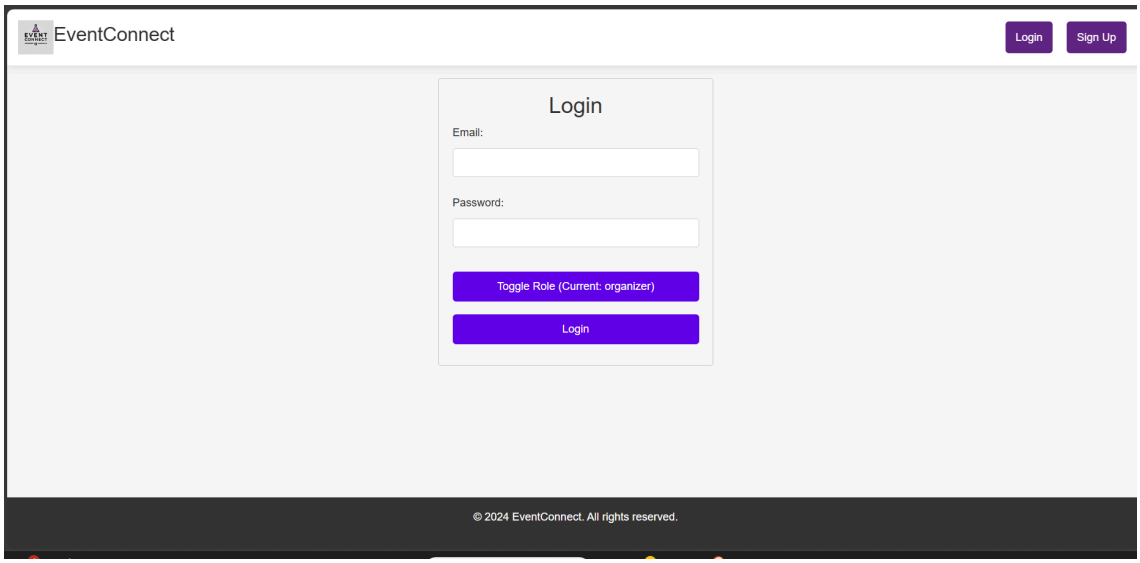
10.0 User Interface Design



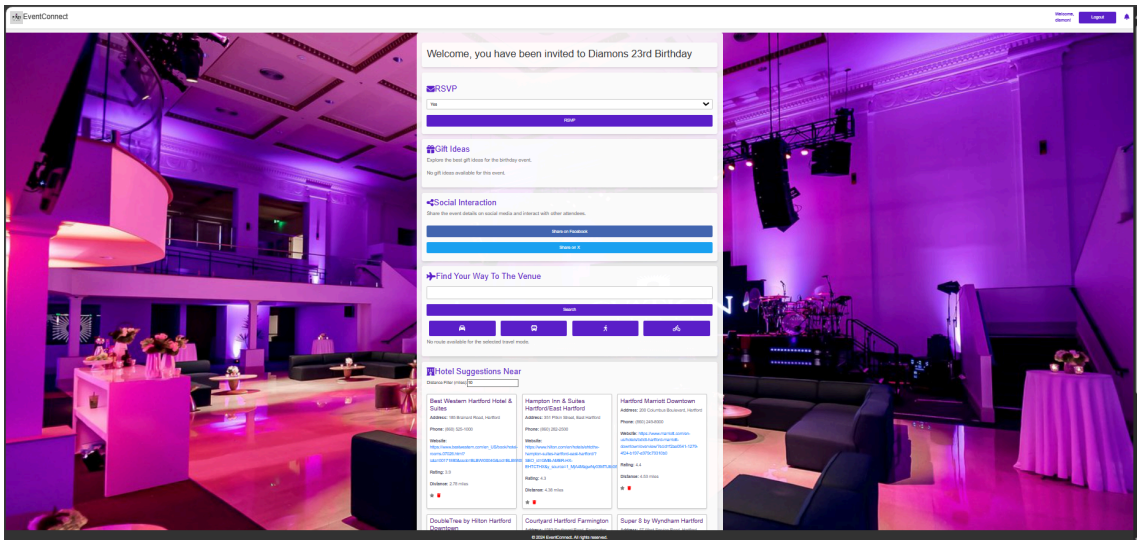
This image shows the landing page of the EventConnect application. It features a cohesive purple theme with sections for a Hero banner, a welcome message, interface navigation, and an informational segment about EventConnect. The Hero banner includes a "Learn More" button centered beneath a background image of a formal event setting. Below this, the welcome section provides quick links to the Organizer and Attendee interfaces. An image of the Attendee Interface is displayed prominently, followed by a section explaining the purpose and benefits of EventConnect. The design is clean, modern, and user-friendly.



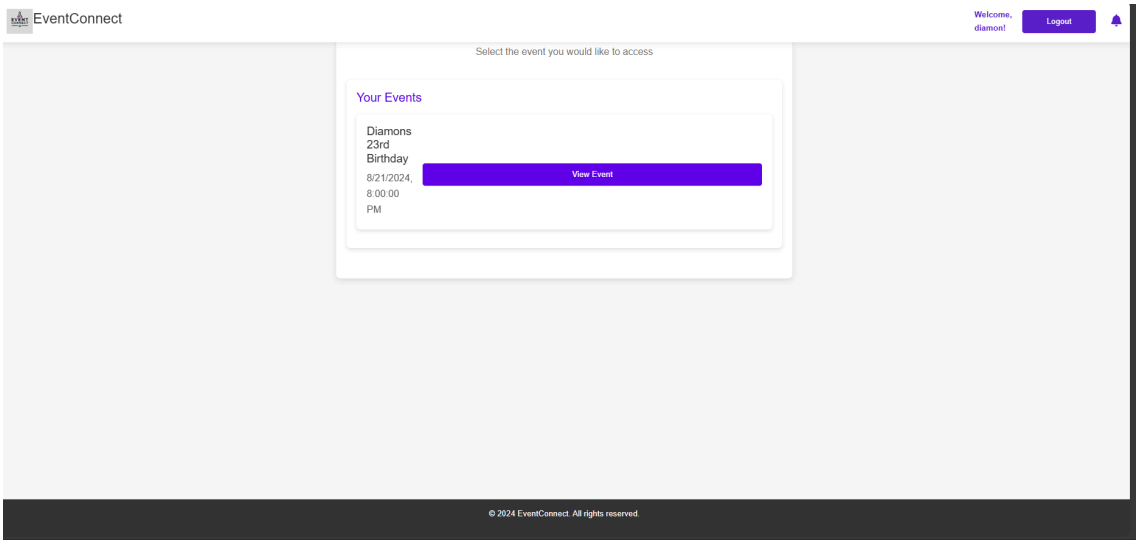
The Register page provides a clean and user-friendly interface for new users to create an account on EventConnect. The page features input fields for the user's name or nickname, email, and password, along with a password confirmation field to ensure accuracy.The page also includes navigation options for logging in, making it easy for returning users to access their accounts.



The Login Page gives users secure entry point to access their EventConnect accounts. Users are required to input their email and password to log in. The interface also features a toggle option for users to switch between organizer and attendee roles, ensuring they access the correct interface for their needs. The design also offer clear navigation options to either log in or sign up for a new account.

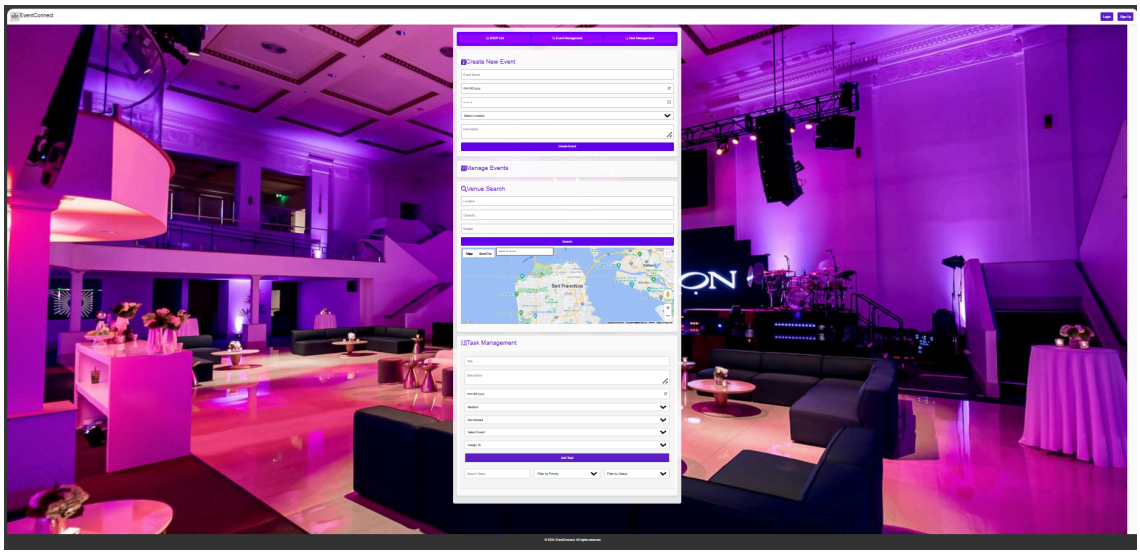


This image shows the Attendee Interface of the EventConnect application overlaid on a background image of a stylish event venue. The interface includes sections for RSVP, Gift Ideas, Social Interaction with the social media share buttons,navigation options each with relevant buttons for user interaction. The layout is designed to be intuitive and user-friendly



The Event List page provides users with a concise overview of their upcoming events. Users can view and select events from a list, with the event name and date displayed prominently. The design is straightforward, ensuring that users can quickly access and

manage their events. Each event entry includes a "View Event" button, leading the user to detailed information and management options for the selected event.



This image shows the Event Organizer Interface page of the EventConnect application. It includes sections for creating new events, managing existing events, searching venues, and task management. The interface features a modern, purple-lit venue background, with all interactive buttons styled in purple for a cohesive look. The layout is user-friendly and designed to facilitate efficient event management.

11.0 EventConnect User Guide

1. Creating an Account

1. **Visit the Registration Page:**
    - Navigate to the EventConnect homepage and click on the "Sign Up" button located at the top right corner of the page.
2. **Fill in Your Details:**
    - Enter your preferred name or nickname in the "Name or Nickname" field.
    - Provide a valid email address in the "Email" field.
    - Create a secure password and confirm it in the "Password" and "Confirm Password" fields.
3. **Complete the Registration:**
    - Click the "Sign Up" button.
    - A confirmation email will be sent to the provided email address. Follow the instructions in the email to verify your account.

4. **Log In:**
    - After verifying your email, return to the homepage and log in using your email and password.

2. Creating an Event

1. **Access the Event Management Page:**
    - Once logged in, navigate to the dashboard and click on the "Event Management" tab.
2. **Create a New Event:**
    - Click the "Create New Event" button.
    - Fill in the event details, including the event name, date, time, and location.
    - Use the Google Maps search feature to select a venue based on your preferences.
    - Provide a brief description of the event, and if needed, specify any special requirements.
3. **Save the Event:**
    - After entering all the details, click the "Create Event" button to save the event.
    - The event will now appear under "Your Events" in the dashboard.

3. Sending Invitations

1. **Manage Your Guest List:**
    - On the Event Management page, select the event you created from the "Your Events" list.
    - Navigate to the "Guest List" section.
    - Add guests by entering their names and email addresses.
2. **Send Invitations:**
    - After populating the guest list, click the "Send Invitations" button.
    - EventConnect will send an email invitation to each guest, with details about the event and a link to RSVP.

4. Managing RSVPs

1. **View RSVPs:**
    - Return to the dashboard and select the event you wish to manage.
    - Click on the "RSVP Management" tab to view the list of attendees who have responded.
2. **Filter and Update RSVP Status:**

    - You can filter the RSVPs by status (e.g., confirmed, pending, declined).
    - Update the RSVP status manually if necessary, or send a reminder to those who have not yet responded.
3. **Communicate with Attendees:**
    - Use the "Send Updates" feature to communicate any changes or important information to the attendees.
    - Attendees will receive notifications via email and within the app.

Additional Features

- **Hotel Suggestions and Travel Options:** Attendees can use the "Find Your Way To The Venue" feature to search for nearby hotels and travel routes. This helps them make arrangements to attend the event.
- **Event Page Access:** Both organizers and attendees can access a detailed event page that includes all relevant information such as the venue, date, time, and guest list.

Conclusion

EventConnect simplifies the process of planning and attending birthday events. By following this guide, you can quickly set up your account, create events, manage guest lists, and ensure a smooth experience for all participants.

12.0 Non-functional Specifications

The overall design of EventConnect focuses on creating a user-centric application where functionality is driven by user input data. The technical foundation includes a database structure with various aggregate relationships, ensuring efficient data management. The API layer, built with a focus on performance and reliability, is responsible for handling all incoming requests, performing operations such as querying and writing data to the database. The API processes requests through an event-driven architecture, responding with success messages for valid requests and error messages for invalid ones. The server-side logic within the API routes determines the success of each request, ensuring smooth interaction with the application.

12.1 Node.js and Express

Node.js is a powerful JavaScript runtime built on Chrome's V8 JavaScript engine, which allows for efficient server-side scripting. Express, a minimal yet robust Node.js web application framework, offers a comprehensive set of features for building web and mobile applications.

- **Documentation:** Node.js Documentation, Express Documentation

12.1.1 Modules

Express facilitates the creation of modular components to handle routing, middleware, and server-side logic. Each system defined in the use cases is encapsulated within its own module, promoting maintainability and scalability.

12.1.2 Middleware

Express employs middleware to manage requests, with functions dedicated to tasks such as parsing request bodies, handling authentication, and managing user sessions. This layered approach allows for the seamless integration of additional functionalities and the efficient handling of client-server interactions.

12.1.3 REST API

Express provides a robust framework for building RESTful APIs, making it straightforward to develop the API layer of the application. The API is designed to handle CRUD operations, facilitating interaction between the front-end and the database.

12.2 React

React, a leading JavaScript library, enables efficient management of client-side state changes and enhances the overall structure of client-side projects. It simplifies event handling and promotes a well-organized project structure compared to traditional HTML.

- **Documentation:** React Documentation

12.2.1 Components

React operates using components, which are written in JSX, an HTML-in-JavaScript syntax. Each component is essentially a JavaScript function that returns HTML, allowing for high levels of reusability and modularity. This approach makes it easier to

manage complex user interfaces by breaking them down into smaller, reusable components.

12.3 MongoDB

Given the aggregate nature of the entities within EventConnect, MongoDB is utilized as the database solution. MongoDB, a NoSQL database, is well-suited for handling large volumes of data with flexible schema designs. It enables the efficient storage and retrieval of data, which is critical for the performance of the application.

- **Documentation:** MongoDB Documentation

12.4 Additional Requirements

- **Google Maps API for venue search:** Google Maps API integration is essential for providing users with accurate venue locations and options.
- **Postman for API Testing:** Postman is used to test and document the API endpoints, ensuring they function correctly and efficiently before deployment.

12.5 Non-Software Requirements

- **GitHub:** GitHub serves as the platform for version control, code sharing, and bug tracking, facilitating collaboration and maintaining project history.
- **Discord:** Discord is used for team communication, offering a platform for real-time collaboration and discussion.
- **Microsoft Teams:** Teams is utilized for meetings and the sharing of non-code files, providing a central hub for project-related communication and documentation.

These non-functional specifications ensure that EventConnect not only meets its functional requirements but also delivers a high-quality, secure, and user-friendly experience. The application is designed to be efficiently maintained and scalable to accommodate future growth.

13.0 Project Timeline

The project implementation for EventConnect will be carried out in five phases (sprints), each spanning approximately two weeks. Below is the detailed project timeline with milestones, sprint plans, and deliverables.

Sprint 1: Foundation and Setup (Weeks 1-2)

Goals:

- Set up the development environment
- Implement core backend functionality
- Create basic frontend framework

Deliverables:

- Project repository initialized on GitHub
- Node.js and Express server setup
- Initial database schema creation
- Basic React frontend with navigation

Tasks:

- Initialize GitHub repository
- Set up Node.js and Express server
- Define initial database schema (users, events, venues)
- Create basic React components (Header, Footer, Home Page)

Sprint 2: User Authentication and Profile Management (Weeks 3-4)

Goals:

- Develop user authentication and profile management

Deliverables:

- User registration and login functionality
- Profile management interface

Tasks:

- Implement user registration and authentication
- Create profile management interface

Sprint 3: Event Creation and Venue (Weeks 5-6)

Goals:

- Implement event creation and venuefeatures
- Set up API integrations

Deliverables:

- Event creation form and venue search functionality
- Integration with Google Maps API

Tasks:

- Develop event creation form
- Integrate Google Maps API for venue search

Sprint 4: Guest Management and Social Features (Weeks 7-8)

Goals:

- Develop guest list and RSVP management
- Implement social media interaction features

Deliverables:

- Guest list management and RSVP functionality
- Social media integration for event sharing

Tasks:

- Create guest list and RSVP management interface
- Implement social media interaction features

Sprint 5: Notification System and Finalization (Weeks 9-10)

Goals:

- Implement notification system
- Conduct testing and prepare documentation

Deliverables:

- Notification system for event updates
- Comprehensive testing and final project documentation

Tasks:

- Develop notification system
- Conduct unit and integration testing
- Prepare final project documentation and user manual

Sprint 6: Buffer and Final Touches (Week 11)

Goals:

- Buffer for any pending tasks
- Final touches and refinements

Deliverables:

- Final refinements based on feedback and testing
- Project ready for final submission

Tasks:

- Review and refine all project components
- Prepare for final submission

14.0 Documentation of the Implementation

This section provides an overview of the core algorithms and external APIs/libraries used in the development of the EventConnect application. It includes specific code excerpts from the project files, along with explanations of how these elements were implemented and integrated to achieve the desired functionality.

Core Algorithms

1. Event Creation

Algorithm Overview: The event creation process involves several key steps, including collecting user input, geocoding the event location using the Google Maps API, and storing the event details in the MongoDB database. The following code snippet from the `OrganizerInterface.js` file demonstrates the event creation logic:

javascript

Copy code

```
const handleCreateEvent = async (e) => {
    e.preventDefault();
    try {
        const location = await geocodeLocation();
        if (!location.lat || !location.lng) {
            setMessage('Latitude and Longitude are required.');
            return;
        }

        const response = await axiosInstance.post('/events', {
            name: eventName,
            date: eventDate,
            time: eventTime,
            description: eventDescription,
            venue: eventLocation,
            latitude: location.lat,
```

```
            longitude: location.lng,
            guestList: [],
            specialRequirements: '',
            userId: userId
        });
        setMessage('Event created successfully');
        setEventName('');
        setEventDate('');
        setEventTime('');
        setEventLocation('');
        setEventDescription('');
        setEvents([...events, response.data]);
    } catch (error) {
        setMessage('There was an error creating the event!');
        console.error('Error creating event:', error.message);
    }
};
```

Explanation:

- The `handleCreateEvent` function handles the form submission for event creation.
- It uses the `geocodeLocation` function to convert the event location into latitude and longitude coordinates using the Google Maps API.

- If the geocoding is successful, the event details, including the location, are sent to the backend via an Axios POST request, where they are stored in the MongoDB database.

---

2. RSVP Management

Algorithm Overview: RSVP management allows users to submit their attendance status for events, and organizers to view and manage these responses. The following code from the `TaskManagement.js` file demonstrates the core RSVP functionality:

javascript

Copy code

```
const handleRsvpSubmission = async (eventId, rsvpStatus) => {
    try {
        const response = await
axiosInstance.post(`/events/${eventId}/rsvp`, { status:
rsvpStatus });
        setRsvpList([...rsvpList, response.data]);
        setMessage('RSVP submitted successfully');
    } catch (error) {
        setMessage('There was an error submitting the RSVP!');
        console.error('Error submitting RSVP:', error);
    }
};
```

Explanation:

- The `handleRsvpSubmission` function allows attendees to submit their RSVP for an event.

- The function sends an Axios POST request to the `/events/${eventId}/rsvp` endpoint, updating the RSVP status in the database.
- Upon successful submission, the RSVP list is updated, and a confirmation message is displayed to the user.

---

3. Google Maps API Integration

Algorithm Overview: The Google Maps API is integrated into EventConnect to allow users to search for venues and geocode event locations. The following snippet from the `Map.js` file shows how the API is used to handle venue searches:

javascript

Copy code

```
const handlePlaceSelected = (place) => {

    setEventLocation(place.formatted_address);

    setMapCenter(place.geometry.location);

    setMapZoom(15);

};



useEffect(() => {

    const searchBox = new
window.google.maps.places.SearchBox(searchBoxRef.current);


mapInstance.current.controls[window.google.maps.ControlPosition.
TOP_LEFT].push(searchBoxRef.current);



    searchBox.addListener('places_changed', () => {
```

```
        const places = searchBox.getPlaces();

        if (places.length === 0) {

            return;

        }


        const place = places[0];

        handlePlaceSelected(place);

    });

}, []);
```

Explanation:

- The `handlePlaceSelected` function updates the event location based on the selected place from the Google Maps search box.
- The `useEffect` hook initializes the Google Maps SearchBox and listens for changes. When a user selects a place, it triggers the `handlePlaceSelected` function to update the map and event location.

External APIs/Libraries

1. Google Maps API

Purpose: The Google Maps API is used for geocoding event locations and providing venue search functionality.

Integration: In `Map.js`, the Google Maps API is used to allow users to search for and select venues. The API's SearchBox is employed to provide an interactive search experience, and the selected place is then geocoded and displayed on the map.

2. Axios

Purpose: Axios is used for making HTTP requests to the backend API.

Integration: Axios is configured in `axiosInstance.js` and used throughout the application to interact with the backend. For example, in `OrganizerInterface.js`, Axios handles event creation requests, while in `TaskManagement.js`, it manages RSVP submissions and task management requests.

javascript

Copy code

```
import axios from 'axios';


const axiosInstance = axios.create({
    baseURL: process.env.REACT_APP_BACKEND_URL,
    headers: {
        'Content-Type': 'application/json',
    },
});


axiosInstance.interceptors.request.use(
    (config) => {
        const token = localStorage.getItem('token');
        if (token) {
            config.headers.Authorization = `Bearer ${token}`;
        }
        return config;
```

```
    },

    (error) => Promise.reject(error)

);


export default axiosInstance;
```

Explanation:

- The Axios instance is configured with a base URL and default headers.
- An interceptor is added to attach the JWT token to every request header, ensuring authenticated access to the API.

3. JSON Web Token (JWT)

Purpose: JWT is used for securing the user authentication process.

Integration: In `auth.js`, JWT is used to generate and validate tokens for user authentication. The token is stored in the client's local storage and attached to each API request to ensure secure communication.

javascript

Copy code

```
const jwt = require('jsonwebtoken');


const generateToken = (userId) => {

    return jwt.sign({ userId }, process.env.JWT_SECRET, {
expiresIn: '1h' });

};
```

```
const verifyToken = (token) => {

    return jwt.verify(token, process.env.JWT_SECRET);

};
```

Explanation:

- The `generateToken` function creates a JWT token with the user's ID, which is used to authenticate users across the application.
- The `verifyToken` function is used to validate the token on protected routes, ensuring that only authenticated users can access specific endpoints.

Installation/Deployment Guide

This guide provides step-by-step instructions to set up the EventConnect application in a local or production environment.

Prerequisites

Before you begin, ensure that you have the following installed on your system:

1. **Node.js** (v14 or later)
2. **npm** (Node Package Manager, comes with Node.js)
3. **MongoDB** (Ensure MongoDB is installed and running)
4. **Git** (for cloning the repository)

Step 1: Clone the Repository

Start by cloning the EventConnect repository from GitHub to your local machine:

```
git clone <https://github.com/DiamonGoff/CSC400.git>
```

Navigate to the project directory:

```
cd EventConnect
```

Step 2: Set Up the Backend

Navigate to the Backend Directory:

```
cd event-management-backend
```

1.

Install Dependencies:
Install the necessary npm packages:

```
npm install
```

2.

Set Up the Environment Variables:
Create a `.env` file in the root of the `event-management-backend` directory and add the necessary environment variables:
makefile
Copy code

```
MONGODB_URI=mongodb://localhost:27017/eventconnect

JWT_SECRET=yourSuperSecretKey

GOOGLE_MAPS_API_KEY=yourGoogleMapsAPIKey

EMAIL=yourEmail@example.com

EMAIL_PASSWORD=yourEmailPassword
```

3. Make sure to replace the placeholders with your actual values.

Start the MongoDB Server:
Ensure that the MongoDB server is running. You can start it using the following command:

```
mongod
```

4.

Run the Backend Server:
Start the backend server using npm:

```
npm start
```

5. The backend server should now be running and accessible at `http://localhost:3001`.

Step 3: Set Up the Frontend

Navigate to the Frontend Directory:
Go back to the root of the project and navigate to the `my-app` directory:

```
cd ../my-app
```

1.

Install Dependencies:
Install the necessary npm packages:

```
npm install
```

2.

Set Up the Environment Variables:
Create a `.env` file in the root of the `my-app` directory and add the necessary environment variables:
makefile
Copy code

```
REACT_APP_GOOGLE_MAPS_API_KEY=yourGoogleMapsAPIKey

REACT_APP_BACKEND_URL=http://localhost:3001
```

3. Make sure to replace the placeholders with your actual values.

Run the Frontend Server:
Start the frontend server using npm:

```
npm start
```

4. The frontend should now be running and accessible at `http://localhost:3000`.

Step 4: Verify the Installation

1. **Access the Application:**
   Open a web browser and navigate to `http://localhost:3000`. You should see the EventConnect landing page.
2. **Test the Application:**
   - Create a new account by registering a user.
   - Log in to the application.
   - Test creating an event, managing guest lists, sending invitations, and other functionalities.

Deployment Considerations

For deploying EventConnect to a production environment, consider the following:

1. **Database Hosting:** Use a cloud-based MongoDB service like MongoDB Atlas for better scalability and reliability.
2. **Environment Variables:** Ensure that environment variables are securely managed, especially in a production environment.
3. **HTTPS:** Implement HTTPS for secure communication between the client and server.
4. **Scaling:** Use containerization tools like Docker to deploy the application in scalable environments.

Conclusion

Following these steps will set up the EventConnect application in a local environment. Ensure that all dependencies and environment variables are correctly configured to avoid any issues during setup. Once everything is set up, you can begin using EventConnect to manage and attend events.

State of the Implementation

The section provides a overview of the current state of EventConnect

Feature Overview

1. **User Registration and Login**
    - **Feature Description:** Allows users to create an account and log in to access the platform's functionalities.
    - **Completion Status:** Fully functional. Users can register with their email, validate their account through a verification process, and securely log in using JWT-based authentication.
    - **Known Issues:** No known issues.
2. **User Profile Management**
    - **Feature Description:** Enables users to create and manage their personal profiles, including updating contact information and event preferences.
    - **Completion Status:** Fully functional. Users can edit their profiles and update preferences, which are saved and retrieved from the database.
    - **Known Issues:** No known issues.
3. **Event Creation**
    - **Feature Description:** Organizers can create events, specifying details such as event name, date, time, location, and guest list.
    - **Completion Status:** Fully functional. Event creation is smooth, and all necessary details are saved correctly. Integration with Google Maps for venue location is implemented.
    - **Known Issues:** No known issues.
4. **Venue Search**
    - **Feature Description:** Organizers can search for venues based on location, capacity, amenities, and budget using the Google Maps API.
    - **Completion Status:** Fully functional. The venue search feature is integrated with Google Maps API, allowing organizers to search and view venues based on the specified criteria.
    - **Known Issues:** The ability to book venues directly through the app is not implemented. This is a planned enhancement for future versions.
5. **Guest List Management**
    - **Feature Description:** Allows organizers to add, edit, and manage guest lists, including assigning roles and managing RSVPs.
    - **Completion Status:** Fully functional. Organizers can manage guest lists effectively, assign roles, and send out invitations.
    - **Known Issues:** No known issues.
6. **Sending Invitations**
    - **Feature Description:** Organizers can send email invitations to guests, with validation for email addresses and error handling for invalid addresses.

    - **Completion Status:** Fully functional. Invitations are sent out via an email service, and the system handles invalid email addresses appropriately.
    - **Known Issues:** No known issues.
7. **RSVP Management**
    - **Feature Description:** Attendees can RSVP to events, and organizers can view, filter, and update guest responses.
    - **Completion Status:** Fully functional. The RSVP management system is user-friendly and allows for efficient tracking of guest responses.
    - **Known Issues:** No known issues.
8. **Hotel Suggestions**
    - **Feature Description:** Attendees can view hotel suggestions near the event venue, filtered by distance.
    - **Completion Status:** Fully functional. Hotels are displayed in a grid format, and filtering works as expected.
    - **Known Issues:** No known issues.
9. **Travel Search**
    - **Feature Description:** Attendees can search for transportation options near the event venue.
    - **Completion Status:** Fully functional. The feature is integrated with the Google Places API, and attendees can find and view travel options.
    - **Known Issues:** No known issues.
10. **Notifications**
    - **Feature Description:** The system sends real-time notifications to users about event updates, RSVP changes, and other relevant information.
    - **Completion Status:** Fully functional. Notifications are sent out promptly, keeping users informed about critical updates.
    - **Known Issues:** No known issues.
11. **Social Media Sharing**
    - **Feature Description:** Attendees can share event details through social media platforms via a generated link.
    - **Completion Status:** Fully functional. Attendees can share a link to the event on their social media accounts.
    - **Known Issues:** No known issues.
12. **Task Management**
    - **Feature Description:** Organizers can create and assign tasks related to event planning, track their progress, and manage deadlines.
    - **Completion Status:** Fully functional. Task management features are integrated with the event system, and users can assign tasks and monitor their completion.

    - **Known Issues:** No known issues.
13. **User Interface (UI) Design**
    - **Feature Description:** The user interface is designed to be intuitive and user-friendly, with a focus on ease of navigation and accessibility.
    - **Completion Status:** Fully functional. The UI is well-designed, with responsive layouts and a cohesive design language across all pages.
    - **Known Issues:** No known issues.

Known Issues

1. **Venue Booking:** The ability to book venues directly through the app was planned but not implemented in this version. This feature remains a priority for future development.
2. **Advanced Analytics:** While basic analytics are in place, more sophisticated analytics features (e.g., tracking guest engagement or event success metrics) are not yet implemented.
3. **Further Testing:** Although the core functionalities are fully functional, some features would benefit from additional testing to ensure they perform well under various scenarios, such as high user load or edge cases.

13.0 Lessons Learned and Reflection

The development of EventConnect was a journey filled with both successes and challenges. Reflecting on the process, I gained valuable insights that will undoubtedly inform future projects and endeavors. This section outlines the key lessons learned, the successes achieved, the challenges faced, and considerations for how I might approach the project differently if given the chance to start over.

Successes

1. **Effective Use of React and Node.js:**
    - One of the standout successes of this project was the effective implementation of the React and Node.js stack. By leveraging React's component-based architecture, I was able to build a responsive and dynamic front end that provided a smooth user experience. The use of Node.js for the backend allowed for efficient handling of API requests and integration with MongoDB, resulting in a robust and scalable application.
2. **Integration of Google Maps API:**

    - Successfully integrating the Google Maps API to facilitate venue searches was another significant achievement. This feature added real value to the application, enabling organizers to search for venues based on location, which is crucial for event planning. The seamless integration of this API demonstrated the project's ability to incorporate external services effectively.
3. **JWT-Based Authentication:**
    - Implementing JWT-based authentication was a critical success in ensuring the security of user data. This approach not only provided a secure method for user authentication but also simplified the session management process, making the app more user-friendly and secure.
4. **User-Centric Design:**
    - The focus on creating a user-centric design was a highlight of the project. By continually refining the user interface based on feedback and testing, I was able to create an intuitive and visually appealing application. The inclusion of features like RSVP management and event detail pages catered directly to the needs of both organizers and attendees, enhancing overall user satisfaction.

Challenges and Changes

1. **Technical Difficulties with Venue Booking Integration:**
    - One of the most significant challenges I faced was the technical difficulty of integrating a venue booking feature. The complexity of working with various booking APIs and ensuring real-time availability data presented unforeseen obstacles. As a result, I had to deprioritize this feature in the final implementation, though it remains a key enhancement for future versions.
2. **Time Management and Scope Creep:**
    - Managing time effectively while balancing the scope of the project was a consistent challenge. As new features and ideas emerged, there was a constant need to reassess priorities and make difficult decisions about what could realistically be achieved within the project's timeframe. This led to certain features, like advanced social media integration, being postponed for future versions.
3. **Adapting to Database Changes:**
    - Initially, the project was designed with a SQL database in mind. However, as the development progressed, I recognized that MongoDB would be a better fit for the application's data structure and scalability requirements. This mid-project shift required revisiting and revising the

data models and queries, which presented both a challenge and a learning opportunity.

If Done Differently

1. **Earlier Focus on Core Features:**
    - If I were to start the project again, I would place a greater emphasis on completing core features like user authentication, event creation, and RSVP management earlier in the development cycle. This would allow more time for testing and refining these critical components, ensuring they are as polished as possible before moving on to additional features.
2. **Better Time Allocation for Complex Features:**
    - I would allocate more time to complex features like the venue booking integration, acknowledging from the outset that these would require significant development and testing time. By doing so, I could have potentially overcome the technical hurdles that led to the postponement of this feature.
3. **Improved Project Management Strategies:**
    - Adopting more rigorous project management strategies, such as detailed sprint planning and regular progress reviews, could have helped to keep the project on track and within scope. This approach would also have allowed for earlier identification of potential delays or issues, enabling quicker adjustments and ensuring a more streamlined development process.
4. **Exploration of Alternative Technologies:**
    - In hindsight, I would also consider exploring alternative technologies or frameworks that might better align with the specific needs of the project. For example, researching more specialized APIs for venue booking or considering other database options from the beginning might have provided additional benefits or simplified certain aspects of the development process.

15.0 Version 2 (Further Work

As with any project, there are always opportunities to build upon the foundation that has been established. Given more time, there are several features and improvements I

would like to implement to further enhance EventConnect and provide a richer experience for both organizers and attendees.

Planned Enhancements

1. **Venue Booking Integration:**
    - One of the most significant enhancements I had initially planned but could not implement due to time constraints is the ability for organizers to book venues directly through the application. Integrating this feature would involve working with external booking platforms or venue management systems via APIs, allowing users to secure a venue without leaving the app.
2. **Expanded Social Media Integration:**
    - Currently, EventConnect allows attendees to share event details via a link on their social media accounts. In a future version, I would expand this integration to include deeper social media interactions. This could involve real-time social feeds on the event page, allowing guests to post comments or share photos, making the event planning process more interactive and engaging.
3. **Enhanced Analytics:**
    - Adding an analytics dashboard for organizers would be a valuable addition. This feature could provide insights into guest engagement, such as the number of RSVPs, the most popular event times and venues, and guest feedback. Enhanced analytics could also track social media interactions, helping organizers better understand their audience and plan future events more effectively.
4. **Advanced Event Management Features:**
    - In the future, I would like to implement more sophisticated event management tools, such as task automation, budgeting tools, and vendor management. These features would help organizers manage all aspects of their events more efficiently, from setting reminders for tasks to tracking expenses and managing vendor contracts directly within the app.

Technical Improvements

1. **Backend Optimizations:**
    - To improve the performance and scalability of EventConnect, I would focus on optimizing the backend architecture. This could involve refining the database schema for better query performance, implementing caching strategies for frequently accessed data, and

optimizing API responses to reduce latency and improve the user experience.

2. **Additional API Integrations:**
    - Future versions could benefit from additional API integrations to expand the functionality of EventConnect. For example, integrating payment gateways would allow for secure online transactions, enabling organizers to collect event fees or donations directly through the app. Integrating with travel and lodging services like Airbnb or Uber could also enhance the attendee experience by providing seamless booking options.
3. **Security Enhancements:**
    - While EventConnect already implements JWT-based authentication, future versions could incorporate more advanced security measures. This might include multi-factor authentication (MFA) for added account security, encryption of sensitive data both in transit and at rest, and regular security audits to identify and address potential vulnerabilities.
4. **Improved User Interface and Experience:**
    - As the application grows, continually refining the user interface (UI) and user experience (UX) will be essential. Future versions could offer a more customizable interface, allowing users to tailor the app to their preferences. Additionally, implementing accessibility features would ensure that EventConnect is usable by a broader audience, including individuals with disabilities.

Conclusion

Version 2 of EventConnect would focus on expanding the platform's capabilities, making it a more comprehensive and user-friendly tool for event planning and management. By implementing venue booking, deeper social media integration, enhanced analytics, and more robust event management features, EventConnect could offer a truly seamless experience for organizers and attendees alike. With technical improvements such as backend optimizations, additional API integrations, and enhanced security, the application would be well-equipped to handle a growing user base and meet the evolving needs of its users.