PROJECT SPECIFICATION DOCUMENT:

Community Event Management Platform

1. Project Overview

1.1 Introduction

The Community Event Management Platform is a web application designed to streamline the organization, management, and promotion of community events. It aims to simplify the logistics for event organizers and enhance the experience for participants. The platform supports event creation, registration, communication, and analytics, providing a comprehensive tool for both organizers and attendees.

1.2 Objectives

- Simplify the process of creating and managing events for community organizers.
- Enhance user experience by making event discovery and registration seamless.
- Provide robust communication tools to foster engagement among participants.
- Offer insightful analytics to organizers for data-driven decision-making.

1.3 Target Audience

- **Event Organizers**: Individuals or organizations responsible for planning and managing community events.
- **Participants**: Community members interested in attending and engaging in local events.

2. Functional Requirements

2.1 User Registration & Authentication

2.1.1 Sign Up and Log In

- Users can create an account using an email address and password.
- *Password recovery via email.

2.1.2 Profile Management

- Users can view and edit their profile information, including name, contact details, and preferences.
- Profile pictures can be uploaded and updated.

2.1.3 User Roles

- **Creator**: Full access to create, manage, and delete the events related to them. Can view all analytics concerning their events.
- **Participants**: Can browse, register, and participate in events. Limited access to communication tools and personal analytics.

2.2 Event Creation & Management

2.2.1 Create Events

- **Event Details**: Name, description, date, time, location, and event capacity.
- **Ticketing**: Option to create free or paid events. For paid events, integrate with Stripe or PayPal for payment processing.
- **Customizable Forms**: Allow organizers to create custom registration forms with specific questions for attendees.

2.2.2 Manage Events

- **Edit Events**: Organizers can update event details, manage registration lists, and adjust capacity.
- **Event Cancellation**: Ability to cancel events, automatically notifying all registered participants.

2.3 Event Registration

2.3.1 Browse and Register

- **Event Listing**: Display all upcoming events with filtering options based on date, location, type, and user preferences.
- **Event Detail Page**: Detailed information about the event, including organizer contact info, schedule, and location map.
- **Registration**: Simple registration process with automated confirmation emails.

2.3.2 Seating Management

- **Capacity Management**: Automatically close registration when the event reaches capacity.

* 2.4 Calendar Integration

2.4.1 Calendar View

- **Monthly/Weekly View**: Visual calendar on the platform displaying all events the user is interested in or registered for.

2.5 Notifications & Reminders

2.5.1 Upcoming Events Notifications

- **Push Notifications**: Send reminders to users about upcoming events via email, SMS, or in-app notifications.
- **Event Updates**: Notify users of any changes (e.g., time, location) via their preferred communication channel.

2.6 Community & Communication

2.6.1 Discussion Boards

- **Event-Specific Boards**: Each event has a dedicated discussion board where participants can post and engage in discussions.
- **Moderation Tools**: Organizers can moderate discussions, pin important messages, and manage user participation.

2.6.2 Direct Messaging

- **Private Messaging**: Allow direct communication between organizers and participants or between participants.

- **Group Chats**: Enable group chats for specific events or interest groups within the community.

2.7 Analytics Dashboard

2.7.1 Event Insights

- **Registration Data**: Track the number of registrations, demographics, and engagement levels.
- **Revenue Tracking**: For paid events, track total revenue, ticket sales, and payment status.

2.7.2 Feedback Tools

- **Post-Event Surveys**: Automatically send surveys to participants to collect feedback.
- **Sentiment Analysis**: Analyze survey responses to gauge overall satisfaction and identify areas for improvement.

2.8 Search & Filters

2.8.1 Event Search

- **Keyword Search**: Users can search for events by keywords related to the event title, description, or location.
- **Advanced Filters**: Filters for date, category, location, ticket type (free/paid).

2.9 Responsive Design

2.9.1 Mobile-Friendly Interface

- **Responsive Design**: Ensure the platform is fully responsive, providing an optimal experience on all device types (desktop, tablet, mobile).
- **Touch-Friendly Navigation**: Optimize the user interface for touch interactions on mobile devices.

2.9.2 Progressive Web App (PWA)

- **App-Like Experience**: PWA will provide a native app-like experience with smooth transitions and fast load times.

3. Non-Functional Requirements

3.1 Performance

- The platform must handle concurrent access by multiple users without degradation in performance.
- Loading times for event lists and detailed pages should be minimized (target < 2 seconds).

3.2 Scalability

- The application should be designed to scale horizontally to handle increasing numbers of users and events.
- Cloud-based infrastructure (e.g., AWS, Heroku) will be used to allow dynamic scaling based on traffic.

3.3 Security

- *__Data Encryption__: All user data, especially sensitive information like passwords and payment details, must be encrypted both in transit and at rest.
- **Authentication**: Use JWT for secure user authentication and session management.
- **Role-Based Access Control (RBAC)**: Ensure that users only have access to features and data appropriate to their roles.

3.4 Usability

- **Intuitive Interface**: The platform must be user-friendly, with clear navigation and accessible language.
- *__Accessibility__: The platform should meet WCAG 2.1 standards, ensuring it is accessible to users with disabilities.

3.5 Reliability

- **Uptime**: The platform must be available 99.9% of the time, with scheduled maintenance periods clearly communicated to users.
- **Backup and Recovery**: Implement regular backups and a disaster recovery plan to prevent data loss.

4. Technical Stack

4.1 Frontend

- **React.js**: For building the dynamic and responsive user interface.
- **HTML/CSS**: For structuring and styling the web pages.
- **HookState**: For managing the global state of the application.

4.2 Backend

- **Node.js with Express.js**: For handling server-side logic and creating RESTful APIs.
- **JWT**: For user authentication and authorization.

4.3 Database

- **PostgreSQL**: For storing relational data like user profiles, event details, and transaction records.
- **MongoDB**: Optional use for storing non-relational data, such as logs or large datasets.

*4.4 Payment Integration

- **Stripe/PayPal**: For handling secure payment processing for paid events.

4.5 Hosting

- **Heroku/AWS**: For hosting the application, with options for scaling and deployment.

5. Development Plan

5.1 Milestone

1. **Day 1**: Requirements gathering, initial setup of the development environment, and database schema design.
2. **Day 2-3**: User registration & authentication, event creation & management features (API)
3. **Day 4-6**: Event registration, calendar integration, and notifications (API)
4. **Day 7-8**: Community features, analytics dashboard (API).
5. **Day 9-11**: Designing Frontend and Design implementation, and search functionality
6. **Day 11-13**: Adding AI based recommended Events, bug fixing and deployment.
7. **Day 14:** Final Presentation

5.2 Deployment

- **Staging Environment**: Deploy the application to a staging environment for final testing.
- **Production Deployment**: Launch the application in the production environment, with continuous monitoring for any issues.

6. Maintenance and Support

6.1 Post-Launch Support

- **Bug Fixing**: Address any issues that arise post-launch.
- **Feature Enhancements**: Gather user feedback for future feature enhancements.
- **System Monitoring**: Continuous monitoring for performance, security, and uptime.

Glossary

- **AI (Artificial Intelligence):** Technology that enables a system to perform tasks that typically require human intelligence, such as recommending events based on user behavior.
- **Analytics Dashboard:** A tool providing visual representations of data related to events, including registration numbers, revenue, and participant feedback.
- **Calendar Integration:** Feature that syncs event dates with a calendar view, allowing users to track and manage their events.
- **Community Management:** The process of overseeing and facilitating interactions within a community platform, including managing user discussions and activities.
- **Creator:** A user role with full access to create, manage, and delete events, as well as view analytics related to their events.
- **Direct Messaging:** A communication feature allowing private conversations between users, such as between organizers and participants.
- **Discussion Boards:** Online forums where users can post and engage in conversations related to specific events.
- **Event Creation & Management:** The functionality allowing organizers to create new events, manage existing ones, and handle event-related tasks such as registration and cancellations.
- **Event Registration:** The process by which participants sign up to attend events, including browsing event listings and confirming their attendance.
- **JWT (JSON Web Token):** A secure method for user authentication and session management in web applications.
- **Notifications & Reminders:** Alerts sent to users about upcoming events or changes to event details, delivered via email, SMS, or in-app notifications.
- **PWA (Progressive Web App):** A web application designed to deliver a native app-like experience, including fast loading times and smooth transitions.
- **Responsive Design:** Design approach ensuring that the platform functions well on various device sizes, including desktops, tablets, and mobile phones.
- **Role-Based Access Control (RBAC):** Security model that restricts system access based on the user's role, ensuring appropriate access to features and data.
- **Scalability:** The ability of the application to handle increasing numbers of users and events without performance degradation.
- **Stripe/PayPal:** Payment processing services integrated into the platform to handle transactions for paid events.
- **User Roles:** Defined access levels and permissions assigned to users, such as Creators and Participants, determining their capabilities within the platform.
- **WCAG (Web Content Accessibility Guidelines) 2.1:** Standards to ensure web content is accessible to people with disabilities.

- **Webhook:** A way for applications to provide real-time information to other applications via HTTP callbacks.
- **HookState:** A library used for state management in React applications, providing a global state management solution.
- **PostgreSQL:** An open-source relational database used to store structured data for the platform.
- **MongoDB:** An optional NoSQL database used for storing non-relational data, such as logs or large datasets.
- **Heroku/AWS:** Cloud-based platforms used for hosting and scaling web applications.

Table of Contents