Event Management Platform Project Report

1. Project Overview

The Event Management Platform is a comprehensive web application designed to facilitate event management for organizers and attendees. The platform allows users to create events, manage bookings, and process payments. The system supports authentication using Spring Security with JWT, and data management using JPA with a PostgreSQL database.

User Roles

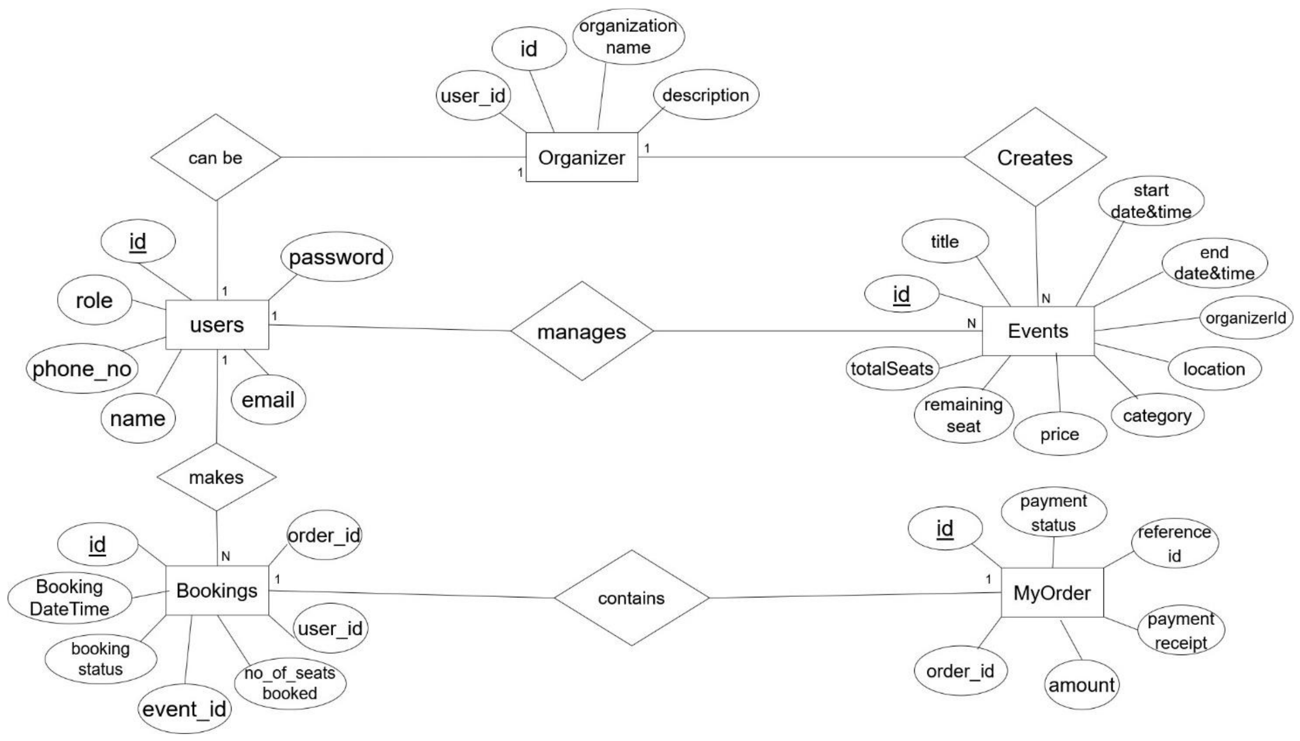
1. **Organizer**
    - Can create, update, and delete events.
    - Each organizer is associated with an organization, which includes additional metadata like organization name and description.
    - Can view the list of attendees registered for their events.
2. **Attendee**
    - Can browse and view available events.
    - Can book a seat for an event via Razorpay payment gateway.
    - Can cancel their bookings, which restores the available seats for the event.
    - Can view their booking history.

2. System Design

2.1 Entity-Relationship Diagram (ERD)

The system consists of key entities like User, OrganizerDetails, Event, Booking, MyOrder, connected using appropriate relationships.

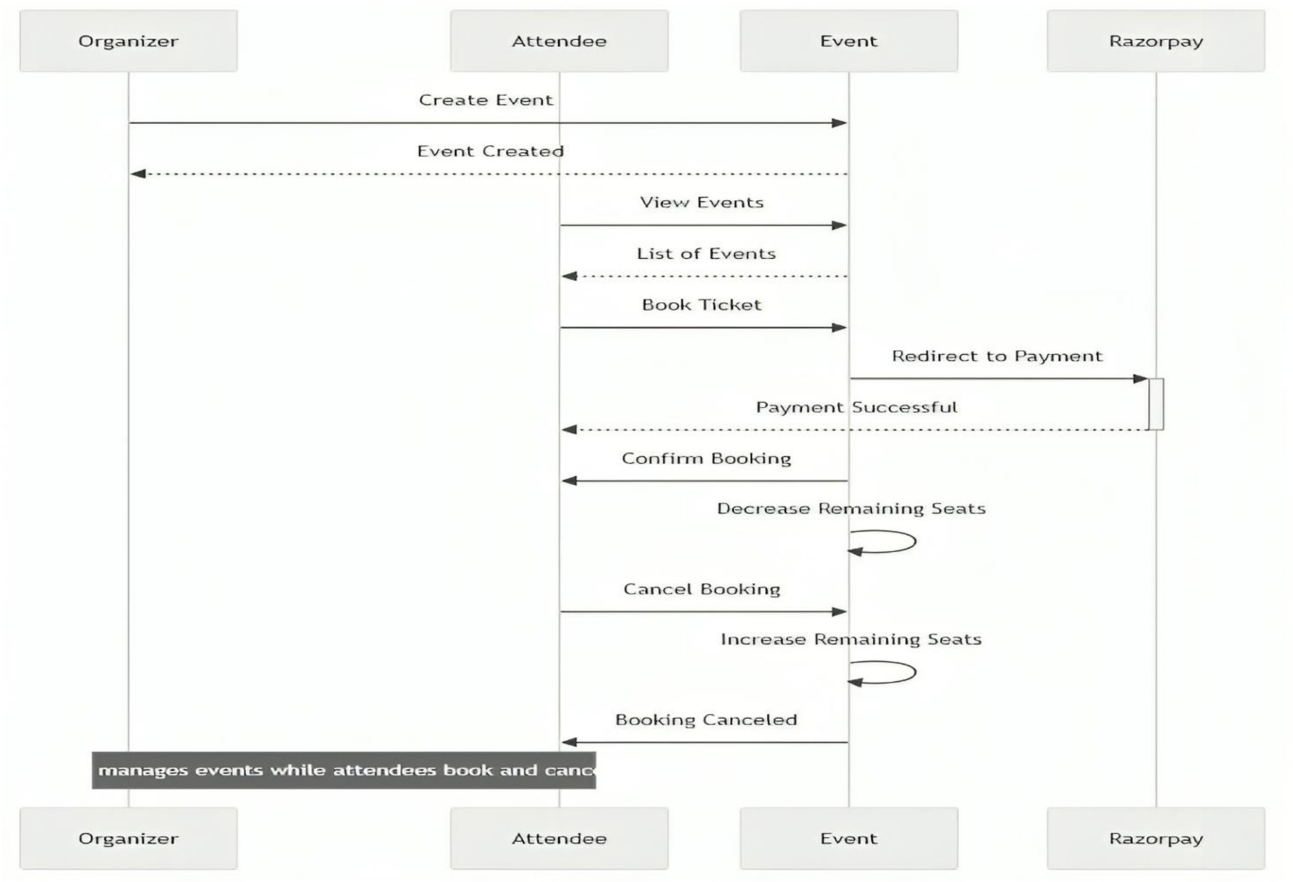
- **User**: Represents both attendees and organizers.
- **OrganizerDetails**: Having some extra information related to organization.
- **Event**: Contains event details with start and end times, categories, and seat availability.
- **Booking**: Manages user event registrations and payments.
- **MyOrder**: Tracks payment details for bookings.



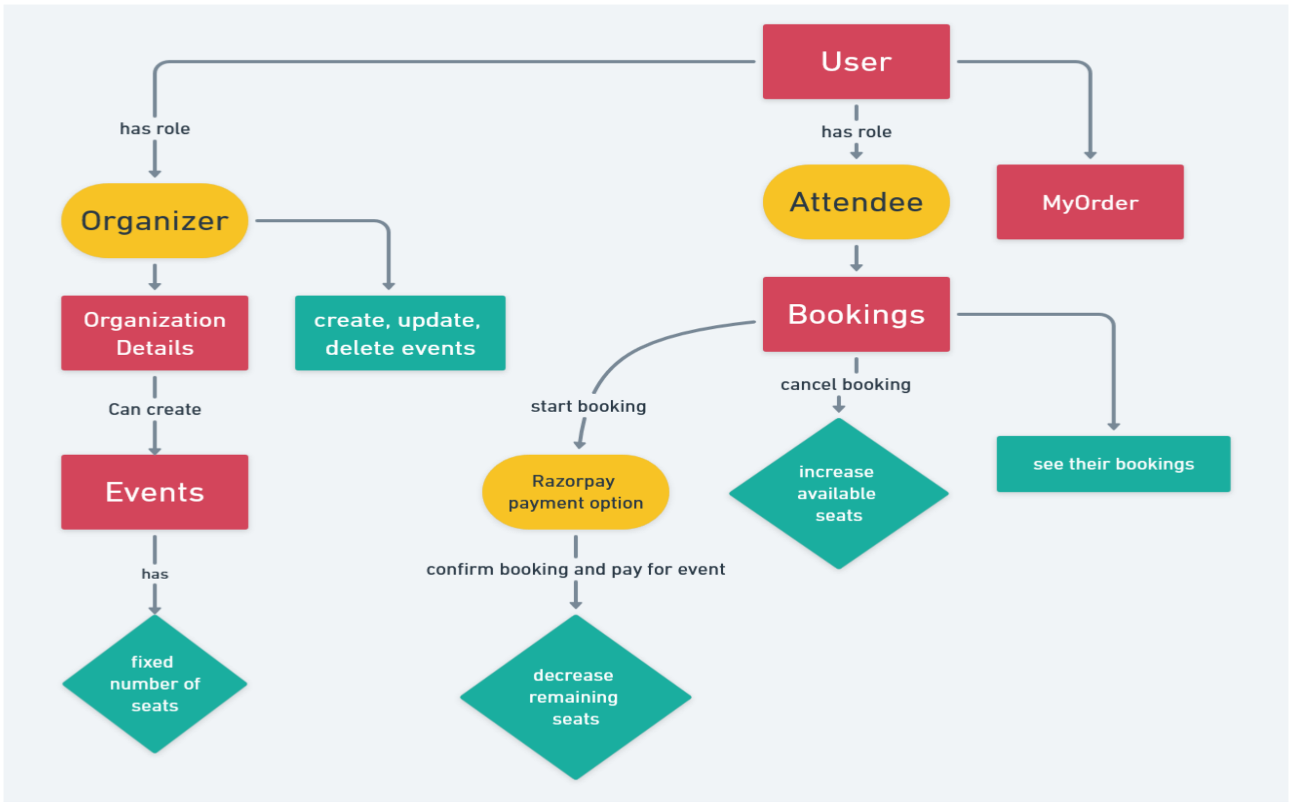
2.2 Sequence Diagram

A typical sequence includes the following steps:

1. Event Creation by Organizers
2. Booking by Attendees
3. Payment Processing
4. Booking Confirmation

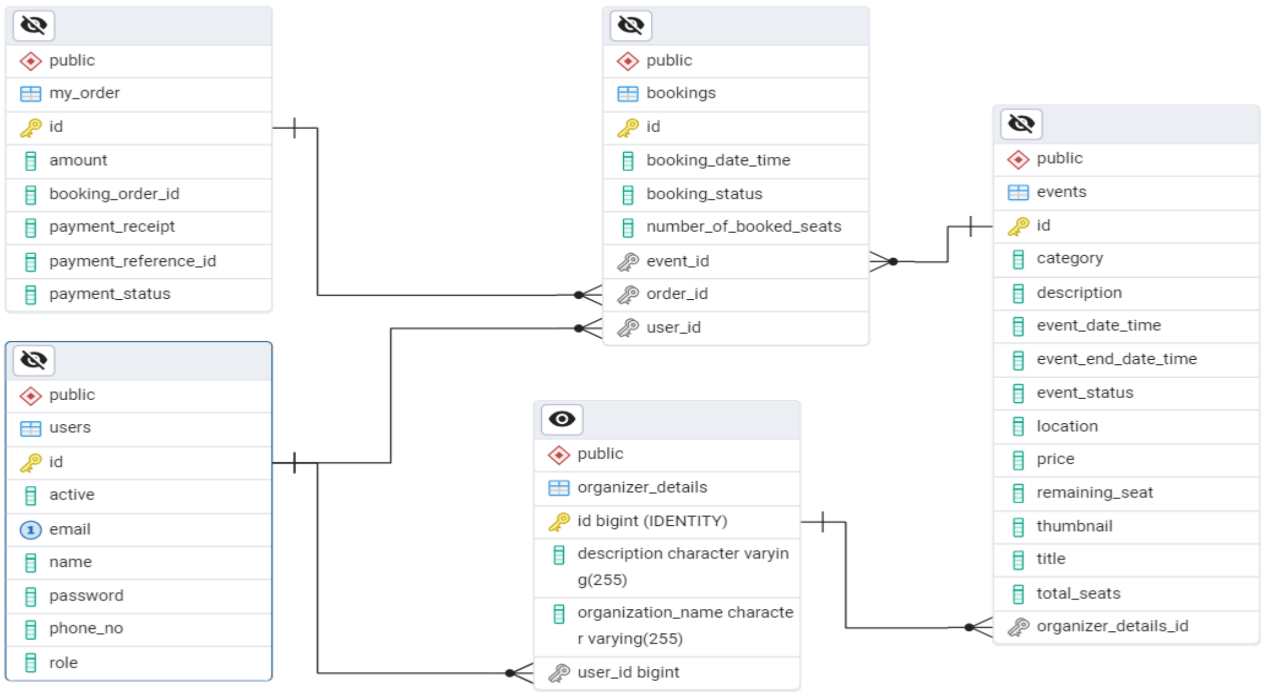


2.3 Data Flow Diagram



3. Database Schema

- **users**: Stores user details with fields like email, password, phoneNo, and role etc.
- **events**: Contains event-specific information with fields like title, description, eventDateTime, category, price, and remainingSeat, totalSeats etc.
- **bookings**: Tracks event registrations and payment information.
- **my_order**: Manages order details such as paymentStatus, amount, and paymentReferenceId.
- **organizer_details**: Stores organization-related data linked to users.



4. API Endpoints

HTTP Method	Endpoint	Description
POST	/user/create-user	Register a new user.
POST	/events/create-event	Create a new Event.
POST	/generate-token	User login and JWT token generation
GET	/events/organizer/{userId}	View events created by organizer.
PUT	/events/update/{id}	View all evenUpdate event details.ts
DELETE	/events/{id}	Delete an event.
GET	/events/category/{category}	Fetch events on the basis of category.
GET	/events/getEnrolledPeople/{eventId}	Get Enrolled user info for a particular event.
GET	/events	View all events.
GET	/bookings/all/{id}	Get all bookings made by the logged-in user.
POST	/bookings/create-booking/{refId}	Book a particular event.
PUT	/bookings/cancel/{bookingId}	Cancel a booking.
POST	/user/create-order	Initiate payment for an event (create an order).
PUT	/user/update-order	After payment done, status from "created" to "paid"
PUT	/user/update	User can update their profile

5. Features and Functionalities

- **User Authentication**: Secure authentication using Spring Security and JWT.
- **Role-Based Access**: Different roles with different privileges (Organizer, Attendee).
- **Event Management**: Organizers can create, edit, and delete events.
- **Booking Management**: Users can book events, view booking status, and make payments.

- **Payment Tracking**: Integrated payment tracking using the MyOrder entity.

6. Implementation Details

6.1 User Management

- **Registration**: Users can register as attendees or organizers. Organizers provide additional details like organization name and description.
- **Login**: Authentication is handled using Spring Security, generating JWT tokens for session management.

6.2 Event Management

- **Create Event**: Organizers can create new events using the /events endpoint. Event details like title, description, date, and category are required.
- **Update Event**: Organizers can edit event details using the /events/{id} endpoint.
- **Delete Event**: Events can be deleted by organizers if there are no bookings.

6.3 Booking Management

- **Book Event**: Attendees can book events using the /bookings endpoint. Seats are reserved based on availability.
- **Cancel Booking**: Users can cancel bookings, releasing the seats.

6.4 Payment Management

- **Payment Integration**: The platform uses Razorpay for payment processing. Payment details are stored in the MyOrder table.
- **Order Tracking**: Users can check their order status using the payment reference ID.

7. Authentication and Authorization

- **Spring Security**: Protects endpoints with role-based access.
- **JWT Tokens**: Provides secure authentication using signed tokens.
- **BCrypt**: Ensures password encryption.

8. Conclusion

This Event Management Platform is an efficient solution for event organizers to manage events and track bookings. It ensures secure transactions and user management through role-based access control. The application is scalable and maintainable, making it ideal for deployment in real-world scenarios.