Event Management System – Fullstack Mini Project

🎯 Purpose:

This system is designed to allow users to **discover and register for events**, and allow administrators to **create and manage events** and **monitor registrations**. The goal is to assess the interns' skills in fullstack development, including backend API design, frontend UI implementation, database modeling, and basic reporting.

👥 Actors

Actor	Description
Public User (Attendee)	Any visitor who wants to view and register for events.
Admin	Internal user who creates, updates, and monitors events and registrations.

🧩 Use Cases

1. User Registration & Login

Actors: Public User
Description: Allows new users to create accounts and log in.
Precondition: User is not logged in.
Flow:

- User provides name, email, password.
- System validates and stores the user.
- On success, user is logged in and token/session is created.

---

2. Admin Login

Actors: Admin
Description: Allows admin users to log in and access the dashboard.
Precondition: Admin account exists.
Flow:

- Admin enters email and password.
- System verifies credentials and grants access with JWT/session token.

---

3. Create / Update / Delete Event (Admin)

Actors: Admin
Description: Admin can manage event lifecycle.
Precondition: Admin is logged in.
Flow:

- Admin fills event form (title, description, date, location, type, etc.)
- System stores/updates/deletes the event.
- On success, confirmation is shown.

---

4. View Events (Public)

Actors: Public User
Description: Any user can browse all upcoming events.
Precondition: None
Flow:

- User visits event listing page.
- User can search/filter by type, location, or date.
- System displays paginated event list.

---

5. View Event Details

Actors: Public User
Description: Shows full information of a selected event.
Precondition: Event exists.
Flow:

- User clicks on an event.
- System displays description, date/time, location, capacity, etc.

---

6. Register for an Event

Actors: Public User
Description: Logged-in users can register for an event.
Precondition: Event is open for registration.
Flow:

- User clicks "Register".
- Fills in name, phone, email (prefilled if logged in).
- System checks if event is not full and registration is within cutoff.
- Stores registration and shows success message.

---

7. Cancel Registration

Actors: Public User
Description: Allows users to cancel their registration before the cutoff.
Precondition: Registration exists and is not past cutoff date.
Flow:

- User goes to "My Registrations"
- Clicks "Cancel"
- System marks registration as canceled.

---

8. View My Registrations

Actors: Public User
Description: Displays all events a user has registered for.
Precondition: User is logged in.
Flow:

- User navigates to their profile
- System displays event list with options to cancel.

---

9. View Registration List for Event

Actors: Admin
Description: Admin views all users who registered for a specific event.
Precondition: Admin is logged in.
Flow:

- Admin selects an event
- System displays list of users with name, phone, email, time of registration.

---

10. Upload Event Image

Actors: Admin
Description: Admin can upload an image for each event.
Precondition: Valid image file provided.
Flow:

- Admin chooses an image
- System stores and links it with event

Option A - Identity & SSO Integration (Keycloak)

Feature: Integrate **Keycloak** for authentication and role-based access.

🎯 **Objectives:**

- Set up **Keycloak** (can be local or containerized).
- Configure login & logout flows using OAuth2/OpenID Connect.
- Define roles: User, Admin.
- Protect frontend routes and backend APIs using access tokens.
- Map Keycloak user claims to system roles.

Option B - Notification & Email

Feature: Send **email notifications** for key actions.

🎯 **Objectives:**

- Send confirmation email on:
  - Successful registration

    - Cancellation (if applicable)
    - Waitlist promotion (if integrated with Intern 2's part)
- Emails must include:
    - Event title, date, time
    - User's registration info
    - (Optional) QR code ticket attachment

Option C - Real-time Updates (WebSocket / SignalR)

Feature: Implement **real-time event update notifications**.

🎯 **Objectives:**

- Use WebSocket (or SignalR if using .NET) to:
    - Notify users when a new event is added or updated
    - Auto-refresh registration count when someone registers
- Bonus: notify admins when capacity is reached.

Option D - Calendar Integration & Event Sync (Fullcalendar)

Feature: Add a **calendar view** for browsing events and allow users to **add events to their personal calendars** (Google, Outlook, etc.).

🎯 **Objectives:**

1. **Implement a full calendar view** (month/week/day) to display upcoming events.
2. Allow users to **click a date** and see events scheduled on that day.
3. Provide an "Add to Calendar" button on the event details page:
    - Export event details as .ics file
    - OR offer "Add to Google Calendar" and "Add to Outlook" links