**Project Report**

**Event Management System**

**Abstract**

The event management system aims to streamline event planning, organization, and execution. This report outlines the system’s design, functionality, and implementation details.

**1. Introduction**

Event management plays a crucial role in organizing successful gatherings, conferences, parties, and other events. This section introduces the purpose of the system and its significance in today’s fast-paced world.

**1.1 Purpose**

The purpose of our event management system is to:

Simplify event planning and coordination.

Provide a centralized platform for event organizers, attendees, and vendors.

Enhance communication and collaboration among stakeholders.

**1.2 Scope**

Our system covers the following aspects:

Event creation and scheduling.

Attendee registration and ticketing.

Venue management.

Payment processing.

Reporting and analytics.

**2. Functional Requirements**

Specify the functional requirements that the event management system must fulfill. These include user roles, use cases, and system behavior.

**2.1 User Roles**

Event organizers

Attendees

Administrators

**2.2 Use Cases**

Create an event

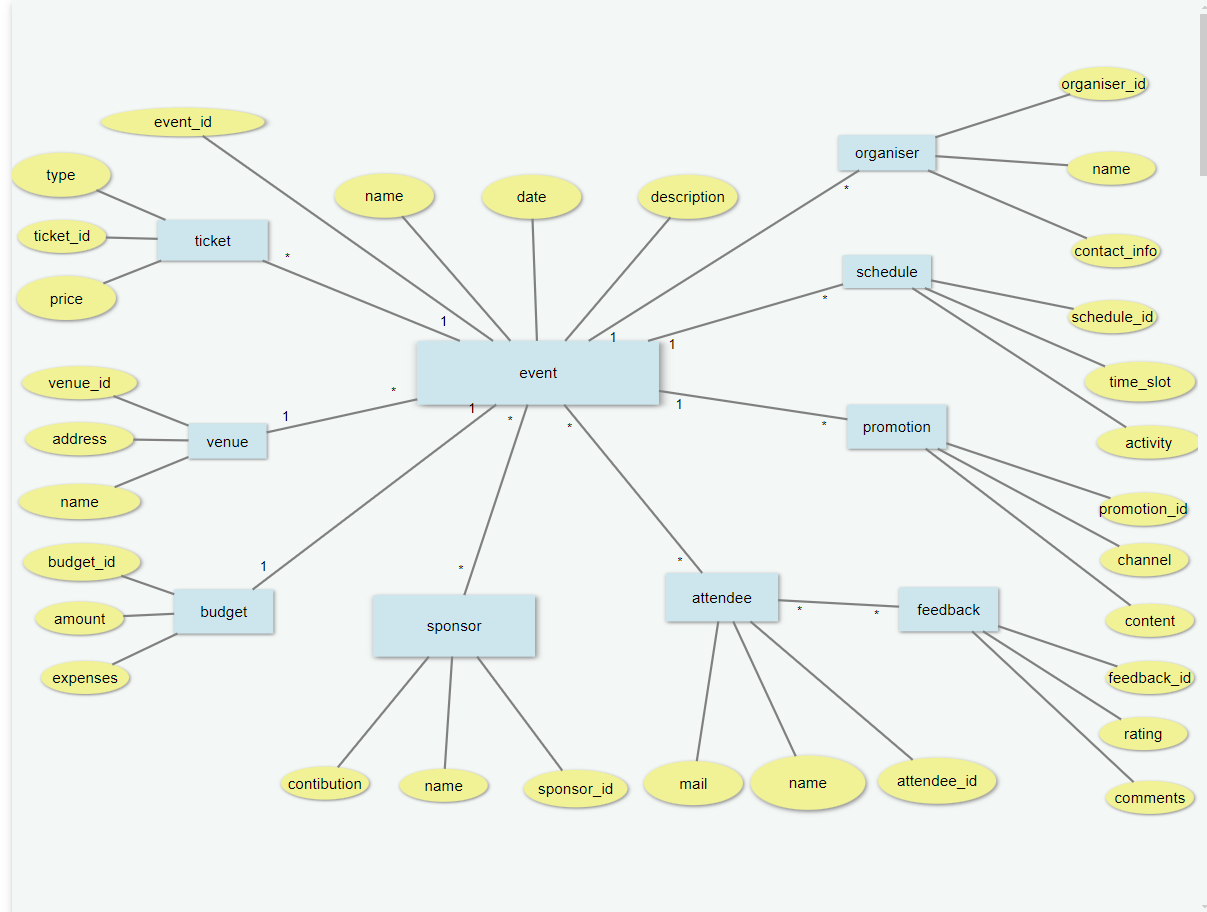
Register for an event

Manage event details

Generate event reports

**3. Entity-Relationship (ER) Diagram**

An ER diagram ,represent the relationships between different entities in the system. Identify entities, attributes, and their associations.



**4. Queries to Create the Database**

Provide MySQL queries to create the necessary database tables based on the ER diagram. Include data types, constraints, types of keys, and other details.

**SQL Query:**

**-- Example query for creating the 'Events' table**

--CREATE TABLE Events

(event\_id INT PRIMARY KEY,

event\_name VARCHAR(255) NOT NULL,

event\_date DATE);

**-- Example query for inserting values into the 'Events' table**

--INSERT into Event values(1,”TechicalEvent”,12-06-24);

**-- Example query for displaying the 'Events' table**

--SELECT \* from Event;

-- Similar queries for other tables (Attendees, Venues, Orders, etc.)

**5. Class/UML Attribute & Methods Diagram**

**Classes and Attributes**

**1.Event:**

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -Eventid  -Name  -Date  -Description | +CreateEvent()  +UpdateEvent()  +DeleteEvent() |

**2. Organizer:**

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -OrganizerId()  -Name  -ContactInfo | +AddOrganizer()  +UpdateOrganizer()  +RemoveOrganizer() |

**3.Attendee**:

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -AttendeeId  -Name  -Email | +RegisterAttendee()  +checkInAttendee()  +SendConfirmationEmail() |

**4.Ticket**:

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -TicketID  -EventID  -Price  -Type | +GenerateTicket()  +SellTicket()  +ValidateTicket() |

**5.Venue**:

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -VenueID  -Name  -Address | +ReserveVenue()  +UpdateVenueDetails()  +CancelVenueBooking() |

**6.Schedule**:

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -ScheduleID  -EventID  -TimeSlot  -Activity | +CreateSchedule()  +UpdateSchedule()  +DeleteSchedule() |

**7.Sponsor**:

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -SponsorID  -Name  -Contribution | +AddSponsor()  +UpdateSponsorContribution()  +RemoveSponsor() |

**8.Budget:**

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -BudgetID  -EventID  -ExpenseCategory  -Amount | +TrackExpenses()  +AllocateBudget()  +GenerateExpenseReport() |

**9.Promotion**:

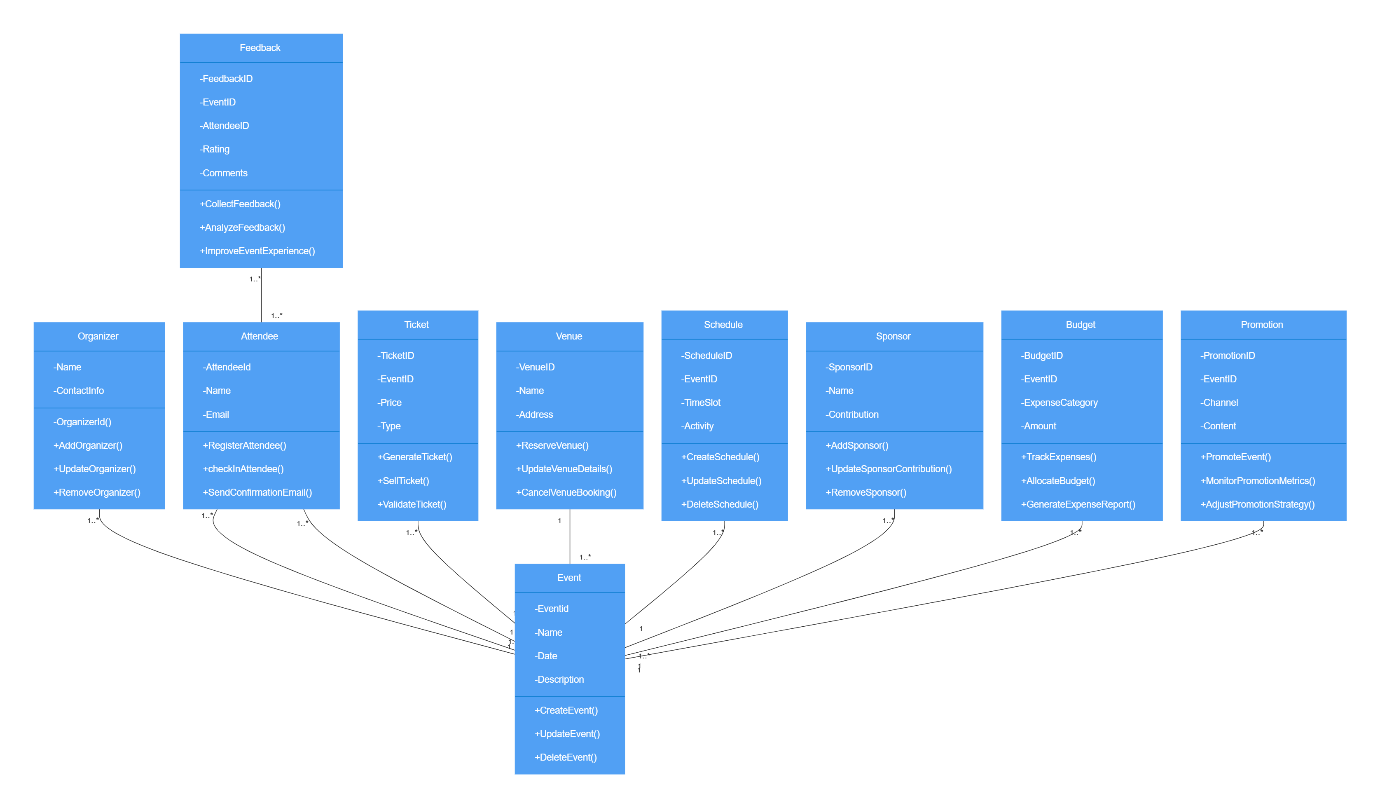
|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -PromotionID  -EventID  -Channel  -Content | +PromoteEvent()  +MonitorPromotionMetrics()  +AdjustPromotionStrategy() |

**10.Feedback**:

|  |  |
| --- | --- |
| **Attributes:** | **Methods**: |
| -FeedbackID  -EventID  -AttendeeID  -Rating  -Comments | +CollectFeedback()  +AnalyzeFeedback()  +ImproveEventExperience() |

A UML class diagram representing the classes and their relationships in the system. Specify class attributes and methods.

The UML Class diagram is a graphical notation used to construct and visualize

**Uml** **Diagram**:

**6. Java Code Files**

Write Java code for the classes defined in the UML diagram. Below is an example for the **Event class:**

**Sourcecode:**

public class Event {

private int eventId;

private String eventName;

private LocalDate eventDate;

private Venue venue;

}

**Conclusion**

The Event Management System project successfully developed a comprehensive application, covering requirement analysis, database design, UML diagram creation, and Java implementation. The system meets key functional requirements, providing efficient user and data management, secure operations. This project highlights the importance of careful planning and design in creating a scalable, maintainable, and secure application, demonstrating the practical application of software development principles.