## Requirement Specification for

# Online Event Management System

Prepared by: Jibon Kumar Roy, Farhan Bin Masud,

Nirupama Yeasmin Nisa

# 1. Introduction

## 1.1. Purpose

The **Online Event Management System (OEMS)** is a digital platform that helps in planning, organizing, and managing events in an easy and efficient way. It reduces the problems of traditional methods such as manual registration, ticket booking, and event coordination.

This system allows event organizers to create and schedule events, manage venues, handle registrations, collect payments, and send real-time notifications to participants. At the same time, participants get a simple and user-friendly experience to register, book tickets, and stay updated. By automating tasks, OEMS saves time, reduces costs, and improves communication between organizers and participants. It provides a modern solution for successful event management.

## 1.2. Project Scope

The Online Event Management System (OEMS) will provide a digital platform for creating, organizing, and managing events. The system will allow organizers to schedule events, manage venues, handle registrations, process payments, and send notifications. Participants will be able to register, book tickets, and view event details online. The scope includes event creation, user management, and real-time updates, but excludes large-scale marketing or third-party integrations.

# 2. Overall Description

## 2.1. Product Perspective

The Online Event Management System (OEMS) is a web-based platform that replaces manual event planning with digital automation. It connects organizers and participants, offering features like event creation, registration, ticket booking, payments, and notifications, ensuring efficiency and ease of use.

## 2.2. Product Features

- **User Authentication**: Secure login and registration for both participants and organizers.
- **Profile Management:** Users can edit and update their personal details.
- **Event Creation & Management:** Organizers can create, update, and delete events.
- **Event Scheduling:** Organizers can set time, date, and venue for events.
- **Event Browsing:** Participants can search and view available events easily.
- **Event Registration:** Users can register, with confirmations stored in the system.
- **Ticket Booking:** Online ticket booking with seat/slot selection.
- **Payment Integration:** Secure online payment for event tickets.

· **Participant Tracking:** Organizers can track and manage registered participants.

· **Notifications:** Real-time alerts and reminders for events.

· **Feedback & Surveys:** Collect feedback from participants for improvement.

· **Admin Dashboard & Reports:** Centralized control with event insights and reports.

## 2.3. User Classes and Characteristics:

### 2.3.1. Administrators

· Manage the overall system.

· Control user accounts, events, payments, and reports.

· Require technical knowledge and full system access.

### 2.3.2. Event Organizers

· Schedule dates, venues, and ticketing.

· Create, update, and manage events.

· Track participants and analyze event reports.

· Need moderate technical skills.

### 2.3.3. Participants/Attendees

· Browse events, register, and book tickets.

· Make online payments and receive notifications.

· Provide feedback and surveys.

· Require simple and user-friendly access.

## 2.4. Operating Environment

This software, designed for managing and organizing events online, will be developed and will operate primarily using the Java programming language.

### 2.4.1. Hardware Platform:

· Web server with at least Intel i3/i5 processor, 8 GB RAM, 500 GB storage for hosting the application and database.

· Client devices including desktops, laptops, tablets, and smartphones for user access.

· Standard input/output peripherals like keyboard, mouse, and display.

### 2.4.2. Operating System:

· **Server**: Linux-based (Ubuntu 22.04 / CentOS 8) or Windows Server 2019+

· **Client devices:** Windows 10/11, macOS 12+, Android 12+, iOS 15+

· **Web browsers:** Google Chrome, Mozilla Firefox, Microsoft Edge, Safari

### 2.4.3. Software Dependencies:

· Java Runtime Environment (JRE 17+) for application execution

· Database: MySQL/PostgreSQL for storing event and user data

· Web frameworks: Spring Boot, JSP/Servlets, or similar

· Payment gateway APIs for online ticket payments

· Email/SMS services for notifications

### 2.4.4. Other Requirements:

· Must operate in environments with stable or intermittent internet access (offline caching optional for some features).

· Compatible with common consumer hardware; no specialized devices required

.

## 2.5. Design and System Constraints

### 2.5.1. Server Performance:

The system requires a minimum of Intel i3/i5 processor, 8 GB RAM, 500 GB storage. Lower-spec servers may lead to slow response times or crashes during high traffic.

### 2.5.2. Client Devices:

Older computers, tablets, or smartphones may not support all features or render the interface properly.

### 2.5.3. Concurrent Users:

Low-spec hardware may limit the number of simultaneous users that can access the system without performance issues.

### 2.5.4. Internet Dependence:

As a web-based system, it requires internet connectivity; offline access is limited.

### 2.5.5. Peripherals:

Certain features like ticket printing require compatible printers or other hardware devices.

# 3. System Features

**Feature 1: User Authentication**

**Description and Priority**

· **Description:** Provides secure access to the system by verifying participants' or organizers' identity before allowing event-related actions (registration, booking, or event management).

· **Priority:** High

· **Priority Components (1–9 scale):**

- Benefit: 10 (critical for system security and access)
- Penalty: 9 (without it, users cannot access core features)
- Cost: 4 (moderate implementation effort)
- Risk: 8 (high risk if authentication fails)

**Stimulus Sequences**

- User selects Register or Login.
- System prompts for credentials (email/phone + password or OTP).
- User enters input.
- System validates input against the database.
- If valid → access granted. If invalid → error message with retry option.

**Functional Requirements**

· FR1: The system shall allow users to log in using unique credentials (email/phone + password/OTP).

· FR2: The system shall lock the account after three consecutive failed login attempts.

· FR3: The system shall display clear error messages for invalid credentials.

· FR4: The system shall log all authentication attempts.

· FR5: The system shall enforce session timeouts after a set period of inactivity (e.g., 15 minutes)

**Feature 2: Profile Management**

**Description and Priority**

· **Description:** Allows users (participants and organizers) to view, edit, and update their personal information securely.

· **Priority:** High

· **Priority Components (1–9 scale):**

- Benefit: 8 (essential for personalized experience and accurate data)
- Penalty: 6 (without it, user data may become outdated or incorrect)
- Cost: 3 (moderate effort to implement)
- Risk: 5 (risk of incorrect data if not validated properly)

**Stimulus Sequences**

- User selects Profile from the dashboard.
- System displays current profile information.
- User edits fields (name, email, phone, password, etc.
- System validates the input for correctness and uniqueness.
- If valid → updates saved and confirmation shown.
  If invalid → error message displayed.

**Functional Requirements**

· FR1: The system shall allow users to view their profile information.

· FR2: The system shall allow users to edit and update profile fields.

· FR3: The system shall validate email and phone number formats and uniqueness.

· FR4: The system shall provide confirmation messages after successful updates.

· FR5: The system shall restrict access to profile management to logged-in users only.

**Feature 3: Event Creation & Management**

**Description and Priority**

· **Description:** Enables event organizers to create, edit, and delete events, including details like name, date, time, venue, and ticket availability.

· **Priority:** High

· **Priority Components (1–9 scale):**

- Benefit: 10 (core feature for the system)
- Penalty: 9 (without it, organizers cannot manage events)
- Cost: 5 (moderate implementation cost)
- Risk: 6 (risk of data inconsistency if not handled correctly)

**Stimulus Sequences**

- Organizer selects Create Event or Manage Event from the dashboard.
- System displays form for event details (title, date, time, venue, tickets).
- Organizer enters or edits information.
- System validates data for completeness and correctness.
- If valid → event is saved/updated/deleted successfully.
  If invalid → error messages shown.

**Functional Requirements**

· FR1: The system shall allow organizers to create new events with all required details.

· FR2: The system shall allow organizers to update existing event information.

· FR3: The system shall allow organizers to delete events.

· FR4: The system shall validate event dates and times to prevent scheduling conflicts.

· FR5: The system shall display confirmation messages for all successful event operations.

· FR6: The system shall allow organizers to manage ticket availability for events.

**Feature 4: Event Scheduling**

**Description and Priority**

· **Description:** Allows organizers to set event date, start/end time, duration, recurrence, venue, and visibility; enforces constraints (no overlapped booking for the same venue/organizer).

· **Priority:** High

· **Priority Components (1–9 scale):**

  - Benefit: 9 (enables the whole app)
  - Penalty: 8 (bad UX and conflicts if missing)
  - Cost: 5 (moderate; needs validation & calendar integration)
  - Risk: 6 (errors cause double-booking)

**Stimulus Sequences**

- Organizer opens “Edit Schedule” for an event.
- System shows scheduling form (date, start, end, repeat, venue, capacity).
- Organizer enters values and selects “Save”.
- System checks for conflicts, validates inputs, saves schedule and triggers notifications if published.
- If conflict/validation error → system shows clear error.
  If success → confirmation shown.

**Functional Requirements**

· FR1. The system shall allow organizers to set event start date/time and end date/time when creating or editing an event.

· FR2. The system shall validate scheduling input and reject events with end time earlier than start time or with start date in the past.

· FR3. The system shall detect and prevent scheduling conflicts for the same venue and time slot and notify the organizer with conflict details.

· FR4. The system shall support recurring events with at least daily, weekly, and monthly recurrence patterns and allow specifying an end date or number of

occurrences.

**Feature 5: Event Browsing**

**● Description and Priority**

· **Description:** Allows participants to discover events via search, filters, sorting, categories, and calendar/list views; includes event preview and bookmarking.

· **Priority:** High

**Priority Components (1–9 scale):**

· **Benefit:** 9  (required for participants to find and attend events)

· **Penalty:** 7 (poor browsing participation)

· **Cost:** 4 (straightforward implementation)

· **Risk:** 4 (low risk)

**● Stimulus/Response Sequences**

- User opens the “Browse Events” page or app home.
- System displays event list and calendar view with upcoming events.
- User enters a search term or chooses filters (date range, category, venue, organizer).
- System returns filtered/sorted results.
- User clicks an event → system displays full event details and registration & bookmark button.

**● Functional Requirements**

· **FR1:** The system shall allow participants to search events by keywords present in title, description, or organizer name.

· **FR2:** The system shall provide filters for date range, category, venue, price (free/paid), and event format (online/in-person).

· **FR3:** The system shall allow sorting of results by date (soonest), popularity (number of registrations), and newest (recently created).

· **FR4:** The system shall provide two viewing modes: list view (paginated) and calendar view (monthly/weekly) and allow switching between them.

· **FR5:** The system shall provide a detailed event page showing title, full description, schedule, venue map/link, ticket types and availability, organizer contact, and attendee count.

· **FR6:** The system shall allow logged-in users to begin registration for events.

· **FR7:** The system shall allow logged-in users to bookmark events and view their bookmarked list from their dashboard.

**Feature 6: Event Registration**

**● Description and Priority**

· **Description:** Enables participants to register for events (free or paid), handles ticket allocation, capacity checks, confirmation/ticket generation, cancellations, and waitlisting.

· **Priority:** High

**Priority Components (1–9 scale):**

· **Benefit:** 10 (directly enables attendance)

· **Penalty:** 9 (Registration failure affects attendance; payment failure is severe)

· **Cost:** 6 (requires payment integration, atomic seat allocation, and notifications)

· **Risk:** 4 (miscounts lead to overbooking and disputes)

**● Stimulus/Response Sequences**

- Participant clicks “Register” on an event page → system shows ticket types, prices, and remaining seats (if any).
- Participant selects ticket type and quantity and proceeds → system verifies availability in real-time.

- Participant provides required information and payment (for paid tickets) → system processes payment (or reserves ticket for free events).
- If payment successful and seats available → system issues confirmation, updates attendee count, and emails ticket/QR code to participant.
- If sold out → system offers to add the participant to the waitlist and notifies if a spot opens.

**● Functional Requirements**

· **FR1:** The system shall allow a logged-in user to register for an event by submitting required attendee information and selecting ticket type.

· **FR2:** The system shall enforce ticket availability and atomically decrement available seats when a registration completes to prevent overselling.

· **FR3:** For paid events, the system shall integrate with a payment gateway (or simulated payment for demo) and confirm payment before finalizing registration.

· **FR4:** The system shall generate and deliver a confirmation to the participant (on-screen + email) containing registration details and a unique ticket/QR code.

· **FR5:** The system shall allow registrants to cancel their registration; upon cancellation the system shall update availability and optionally issue refunds according to event policy.

· **FR6:** The system shall provide a waitlist mechanism for sold-out events; when seats free up, waitlisted users shall be notified in FIFO order and offered the seat for a limited hold period.

# 4. External Interface Requirements

## 4.1. User Interfaces

- **Logical Design**: Clean, intuitive layout with easy navigation; menu bar, dashboard, event list, and user profile sections.
- **GUI Standards**: Responsive web design compatible with desktops, tablets, and smartphones; consistent color scheme and fonts.
- **Error Messages**: Clear and user-friendly (e.g., “Invalid login credentials,” “Payment failed. Please try again”).
- **UI Components:** Login/registration forms, user dashboard, event creation forms, ticket booking page, payment page, reporting panel for admins.
- **Accessibility**: Supports multiple languages, readable fonts, high-contrast mode, and mobile-friendly navigation**.**

### Example:

- Dashboard shows upcoming events, registered events, and notifications.
- **Standard buttons**: Home, Events, Profile, Help, Logout on every screen.

## 4.2. Hardware Interfaces

- **Web Server:** Handles all application processing and database management; communicates with client devices over the internet.
- **Client Devices:** Desktops, laptops, tablets, and smartphones access the system via web browsers.
- **Payment Devices:** Optional card readers or POS terminals for offline ticket payments.
- **Network Equipment:** Routers, modems, and internet connections for real-time communication between server and clients.

- **Peripherals:** Standard input/output devices such as keyboard, mouse, and display; printers may be used for tickets or reports.

**Data/Control Interactions:**

- Client devices send HTTP/HTTPS requests to the server.
- Server processes requests, accesses the database, and sends responses back to clients.
- Payment devices communicate via secure APIs with payment gateways for transaction processing.

## 4.3. Software Interfaces

- **Database:**
    - MySQL or PostgreSQL for storing user data, event details, registrations, and payment records.
- **Backend Frameworks & Application Server:**
    - Java Spring Boot, JSP/Servlets to handle business logic and client requests.
- **Payment Gateway APIs:**
    - Stripe, PayPal, or similar services for secure online transactions.
- **Email/SMS Services:**
    - Integration with services like Twilio or SendGrid to send notifications, confirmations, and alerts.
- **Frontend Technologies:**
    - HTML, CSS, JavaScript for building responsive and interactive web interfaces.
- **Operating Systems Compatibility:**
    - Server: Linux (Ubuntu/CentOS) or Windows Server.
    - Client: Windows, macOS, Android, iOS.

- **Data Exchange Protocols:**
    - JSON or XML formats for API requests and responses between client and server.

Example Data Flow:

- User registers for an event → request sent via browser → server processes and stores data in the database → payment processed via gateway → confirmation sent via email/SMS → UI updated on client side.

## 4.4. Communication Interface

- **Protocols:**
    - **HTTPS** for secure communication between client devices (web browsers) and the server.
    - **REST API** for interactions between the frontend, backend, and third-party services (e.g., payment gateways, email/SMS services).
- **Network Requirements:**
    - Standard broadband, Wi-Fi, or mobile internet connection to access the system.
    - Reliable network needed for real-time updates, notifications, and payment processing.
- **Security Considerations:**
    - SSL/TLS encryption for all data transmitted over the internet.
    - Token-based authentication for API calls to ensure authorized access.
- **Performance Expectations:**
    - Page load within 3–5 seconds for smooth user experience.
    - Payment confirmation processed in under 5 seconds.
    - Notifications delivered in near real-time (within 10–15 seconds).
- **Third-Party Communication:**
    - Payment gateways (Stripe, PayPal) via secure API.
    - Email/SMS services (Twilio, SendGrid) for confirmations, alerts, and notifications.

### Example Data Flow:

· User books a ticket → request sent via browser → server validates and stores data → payment gateway processes transaction → confirmation sent via email/SMS → UI updated in real-time.

## 4.5. Nonfunctional Requirements

· **Performance Requirements:**

- System should handle at least 100 concurrent users at the same time.
- Page load time for event listings shall not exceed 3 seconds under average network conditions.
- Search queries must return results within 1 second for datasets of up to 10,000 events.

· **Security Requirements:**

- All passwords shall be stored using secure hashing algorithms
- Only authorized users may access features based on their roles (participant, organizer, admin).
- Data stored in the database should be encrypted.

· **Software Quality Attributes:**

- The system shall be available at least 99% of the time during academic semester hours.
- Only authorized users may access features based on their roles (participant, organizer, admin).
- The system shall provide a simple and consistent interface for all user roles.
- First-time users should be able to complete registration and join an event within 5 minutes without external help.

- In case of failure (e.g., crash, disconnection), user data shall not be lost and transactions must be recoverable.
- The system shall provide a simple and consistent interface for all user roles.
- Code should be modular so updates can be done easily.
- It should be easy to add new features like online payment or feedback in the future.

**Team Contribution**

This project was completed as a group work with the contributions of three members. The distribution of work among the team members is as follows:

**• Jibon Kumar Roy (2303003)**

- Product Features
- User Classes and Characteristics
- Design and System Constraints
- System Features
- User Interfaces
- Hardware Interfaces

**• Farhan Bin Masud (2303007)**

- Project Scope
- Project Perspective
- System Feature
- Communication Interface
- Nonfunctional Requirements

- **Nirupama Yeasmin Nisa (2303015)**
    - Introduction
    - Operating Environment
    - System Feature
    - Software Interfaces