Capstone Project – Secure Online Event Management System (SOEMS)

Objective

Design and develop a secure web application for managing events, allowing event organizers to create/manage events and participants to register for them. The system will be implemented using Spring MVC (Spring 5), REST APIs, JWT authentication, and PostgreSQL with Hibernate following a Layered Architecture pattern and using Criteria Queries for dynamic data retrieval.

Technical Requirements

• Spring MVC (Spring 5) – No Spring Boot

• PostgreSQL database integration

• Hibernate ORM

• JWT-based authentication & authorization

• REST API endpoints returning JSON

• AJAX-based frontend interactions to consume REST APIs asynchronously

• Maven for dependency management

• Layered Architecture (Controller → Service → DAO → Entity)

• Criteria Query for dynamic searching/filtering

• Global Exception Handling for consistent JSON error responses

Core Requirements

User Authentication & Authorization

• JWT-based authentication for all protected endpoints.

• Role-based access control (e.g., ADMIN, USER).

CRUD Operations

• Create, Read, Update, Delete functionality for events.

Validation Requirements

• Event title: Required, min length 3, max length 100.

• Event date: Cannot be in the past.

• Location: Required, max length 255.

• Description: Optional, max length 500.

• User email: Must be unique, valid format.

• Password: Min 8 characters, must contain uppercase, lowercase, number, and special character.

Error Handling

• Consistent error response format with HTTP status codes.

• Use @ControllerAdvice and @ExceptionHandler for global exception handling.

Database Integration

• Use PostgreSQL for persistence.

API Requirements

Public Endpoints (No Auth):

• POST /register – Register new user (Admin/User)

• POST /login – Authenticate and return JWT

Protected Endpoints (JWT Required):

Admin:

• POST /events – Create new event

• PUT /events/{id} – Update event

• DELETE /events/{id} – Delete event

• GET /events/{id}/participants – View registered participants

User:

• POST /events/{id}/register – Register for an event

• GET /my-events – View registered events

Functional Requirements

User Management

• Two roles: Admin (event organizer) and User (participant).

• Registration endpoint for new users.

• Login endpoint with JWT token generation.

• Passwords securely hashed before storing.

Event Management (Admin Only)

• Create, update, delete events.

• Event details must include: Event ID, Event Name, Description, Location, Start Date & End Date, Registration Deadline, Maximum Participants Allowed.

Event Registration (User)

• View all available events (public).

• Register for an event (JWT required).

• Prevent overbooking beyond the maximum allowed participants.

• Prevent duplicate registrations for the same event.

Payment Simulation

• On event registration, simulate a payment transaction.

• Store payment details in DB: Payment ID, User ID, Event ID, Amount, Date, Status (Paid/Failed).

Reports

• Admin can view a list of all registered participants for an event.

• User can view their registered events.

Entities

• User

• Event

• Registration

• Payment

Deliverables Expected

- Fully functional web application fulfilling all core and technical requirements
- ER Diagram and Class Diagrams
- Individual documentation from each participant detailing errors faced during development and the steps taken to resolve them

Evaluation Criteria

Criteria	Weightage
Functionality Implementation	30%
Validation Implementation	15%
JWT Authentication & Authorization	15%
Database Design (ERD, Class Diagram)	10%
Naming Conventions, Module separation and Comments	10%
Documentation Completeness	10%
New Features Implemented Beyond Requirements	10%