****

**EVENTSPHERE**

**Theme: College Event Information System**

**Category: Full-Stack Application Development**

**Group name: HelloWorlders**

***Intructor: Anh Pham Nhat***

***Member:***

|  |  |
| --- | --- |
| ***Student1499598*** | ***Tuan Le Anh*** |
| ***Student1397910*** | ***Thanh Bui Duc*** |
| ***Student1566400*** | ***Hai Nguyen Thanh*** |
| ***Student1611328*** | ***Truong Nguyen Van*** |

**Ha Noi, 09/2025**

**ACKNOWLEDGMENTS**

*First and foremost, our team would like to express our deepest and most sincere gratitude to Mr. Phạm Anh Tuấn, our dedicated mentor, who has provided us with invaluable guidance, insightful advice, and unwavering support throughout the process of researching and developing the EventSphere – College Event Information System. His dedication and profound expertise have been a tremendous source of encouragement and an invaluable foundation for us to successfully complete this project.*

*We also extend our heartfelt thanks to the Organizing Committee of the competition for creating a meaningful platform that allows students to apply theoretical knowledge to practical scenarios, hone teamwork skills, develop systems thinking, and enhance project management capabilities.*

*Additionally, we would like to express our gratitude to our fellow students and friends who have provided feedback, encouragement, and suggestions, contributing to the continuous improvement of the system.*

*Despite our best efforts, due to time constraints and limited experience, the project inevitably has some shortcomings. Our team sincerely welcomes constructive feedback from the judging panel to further enhance and improve the system in the future.*

*Thank you very much for your support!*

*Sincerely, The Project Team!*

Contents

[I. Introduction 5](#_Toc208867624)

[1.1. Problem Definition 5](#_Toc208867625)

[1.2. Project Objectives 5](#_Toc208867626)

[1.3. Scope 6](#_Toc208867627)

[1.4. Assumptions and Constraints 7](#_Toc208867628)

[II. Technical Design Specifications 7](#_Toc208867629)

[2.1. System Architecture 7](#_Toc208867630)

[2.2. Functional Requirements 8](#_Toc208867631)

[2.3. Non – Functional Requirements 9](#_Toc208867632)

[III. System Diagrams 10](#_Toc208867633)

[3.1. Use Case Diagram 10](#_Toc208867634)

[3.2. Data Flow Diagram (DFD) 13](#_Toc208867635)

[3.3. Activity Diagram (Event Registration Process) 14](#_Toc208867636)

[3.4. Entity Relationship Diagram 15](#_Toc208867637)

[3.5. Architecture Diagram 15](#_Toc208867638)

[IV. Database Design 16](#_Toc208867639)

[4.1. ERD Model 16](#_Toc208867640)

[4.2. Relationships Between Tables 16](#_Toc208867641)

[V. Test Data 16](#_Toc208867642)

[5.1. Sample Dataset 16](#_Toc208867643)

[5.2. Test Scenarios 17](#_Toc208867644)

[VI. Project Installation Guide 17](#_Toc208867645)

[6.1. System Requirements 17](#_Toc208867646)

[6.2. Installation Steps 18](#_Toc208867647)

[6.3. Post-Installation Checks 18](#_Toc208867648)

[VII. Sample Login Information 18](#_Toc208867649)

[VIII. Appenddix 19](#_Toc208867650)

[8.1. Glossary 19](#_Toc208867651)

[8.2. References 19](#_Toc208867652)

# Introduction

## Problem Definition

In the vibrant academic environment of universities and colleges, professional, technical, and cultural events play a crucial role in enhancing knowledge, fostering social interaction, and developing soft skills for students. However, managing these events through traditional methods has revealed numerous limitations, posing significant challenges for both organizers and participants.

One of the core issues is the lack of an effective communication channel. Event information is often disseminated in a fragmented manner through notice boards, group messages, or word of mouth, leading to students easily missing important announcements or receiving inaccurate information. As a result, participation rates are low, as many interested students fail to receive timely notifications.

For organizers, event management also presents numerous difficulties. They must manually handle registration lists and attendance records on paper or scattered digital files, increasing the risk of errors, confusion, and lack of transparency. Accessibility to information is also limited, as students face obstacles when trying to review past event details or search for upcoming events due to the absence of a centralized information and media repository. Furthermore, the lack of analytical and reporting tools makes it challenging for institutions to accurately assess the effectiveness of organized events.

In summary, the core problem to be addressed is the absence of a centralized digital system capable of managing the entire lifecycle of an event—from announcement and registration to participation, feedback, and reporting. This not only reduces organizational efficiency but also limits students’ participation opportunities, creating unnecessary difficulties for all stakeholders involved.

## Project Objectives

The EventSphere project aims to develop a centralized and comprehensive event management platform that digitizes the processes of organizing and participating in events. This platform is designed to maximize benefits for students, organizers, and administrators, thereby enhancing overall efficiency and improving user experience.

For casual visitors (guests), the platform allows easy exploration without requiring a login. They can view public content such as upcoming events, event details, photo galleries, and general information pages. However, to perform interactive actions such as registering for events, submitting feedback, or downloading certificates, they will need to log in or create an account.

For students, EventSphere serves as a valuable tool, enabling them to easily search for and register for events online. Students can manage information about the events they have attended and receive timely notifications about schedule changes. Notably, event attendance and check-in processes are made faster than ever through QR code scanning technology. After each event, students can provide feedback and evaluations, contributing to the improvement of future events.

For organizers, EventSphere is a robust management system that allows them to create, edit, and manage events online. Organizers can track registration and attendance lists in real-time, making management more transparent and efficient. The system also supports managing participant feedback and issuing attendance certificates, streamlining the entire event organization process from start to finish.

Finally, for administrators, EventSphere provides a tool to manage the entire system and user data. Administrators have the authority to approve events before they are publicly announced, ensuring content quality. They can also monitor event quality through detailed reports and analytics while ensuring the security and integrity of user data. This ensures the platform operates smoothly, securely, and reliably.

The overarching goal of EventSphere is to build a digital event ecosystem where all stakeholders benefit. The platform not only enhances organizational efficiency but also improves access to high-quality events and elevates the participation experience for students, fostering a dynamic and effective interactive environment.

## Scope

The scope of the EventSphere system is clearly defined to ensure focus and efficiency during development. The system is designed to manage a diverse range of events, including academic workshops, cultural festivals, technical competitions, annual events, and extracurricular activities. This versatility allows EventSphere to serve as a comprehensive platform that meets the varied event organization needs within academic and community settings.

The system caters to four primary user groups, each with distinct roles and functionalities. Guests can view public event information without needing to log in. Registered students can sign up for events, provide feedback, and engage interactively. Organizers have full control over their events, from managing attendance to issuing certificates. Finally, administrators hold the highest authority, responsible for user management, event approval, and monitoring statistical reports.

Technologically, EventSphere is developed as a responsive web application, easily accessible through browsers on both computers and mobile devices. This optimizes the user experience across various devices without requiring a separate app installation. However, the project has certain limitations. The system does not include a native mobile application (iOS/Android), focusing solely on the web platform. Additionally, the certificate payment function is implemented at a simulation level only, without processing actual financial transactions, to maintain simplicity and avoid finance-related risks.

## Assumptions and Constraints

Assumptions:

* Users have a stable internet connection.
* Users have a valid email address for registration.
* Browsers support modern HTML5, CSS3, and JavaScript.

Technical Constraints:

* Frontend: ReactJS, Bootstrap
* Backend: Node.js
* Database: MySQL
* IDE: Visual Studio Code

# Technical Design Specifications

## System Architecture

The EventSphere system is designed based on a three-tier Client-Server architecture, ensuring scalability and clear separation between the user interface, business logic processing, and data management.

**Presentation Layer (Frontend – Client):** This layer is responsible for the user interface and direct interaction with users. Developed using ReactJS in combination with HTML5, CSS3, and Bootstrap, EventSphere’s interface is designed to be modern, responsive, and compatible with various mobile devices. The primary role of this layer is to display event data, manage interface states, and handle user actions such as event registration, submitting feedback, or scanning QR codes for attendance.

**Application Layer (Backend – Application Server):** This is the core of the system’s business logic processing. The backend layer, built using Node.js (Express.js), provides services through a RESTful API. All user requests are processed here. This layer handles critical tasks such as user authentication (using JWT), role-based access control for different user types (guests, students, organizers, administrators), event registration logic, feedback management, and generating statistical reports.

**Data Layer (Database – Data Layer):** The final layer is responsible for storing and managing all system data. MySQL is chosen as the database management system. All data related to events, users, registrations, attendance, feedback, and media libraries are stored in an organized manner. The database tables are designed with primary keys, foreign keys, and integrity constraints to ensure accuracy and consistency, thereby maintaining data integrity throughout the system’s operation.

## Functional Requirements

The EventSphere system supports multiple user groups with distinct roles and functionalities:

**a. Guests**

* Access the homepage and view the list of events (ongoing, upcoming, and past).
* View detailed event information: title, description, date and time, location, and organizing unit.
* Browse publicly available photo/video galleries.
* Not permitted to register for events or submit feedback.

**b. Students**

* Create a personal account with basic information (full name, email, faculty, student ID).
* Log in securely and access a personal dashboard.
* Register for events with available slots and cancel registrations if needed.
* Receive notifications about schedule or location changes.
* Check in at events using system-generated QR codes.
* Download electronic certificates after successfully attending events (if supported by the event).
* Submit feedback and evaluate the quality of events.
* View a history of attended events.

**c. Organizers**

* Log in using an account approved by administrators.
* Access an organizer dashboard with quick statistics (number of registrations, feedback).
* Create new events with details: name, description, category, date and time, location, maximum capacity, and images.
* Edit or cancel events before their scheduled date.
* Monitor and manage registration lists.
* Scan QR codes for attendance on the event day.
* Issue attendance certificates for eligible students.
* Manage photo/video albums for events after completion.

**d. Administrators**

* Log in securely (with optional two-factor authentication).
* Manage users: add, edit, suspend, or assign roles.
* Approve or reject events proposed by organizers before public announcement.
* Moderate feedback, event content, and uploaded media.
* Send system-wide or group-specific notifications.
* Export statistical reports (PDF/Excel) on event numbers, participation rates, and feedback.

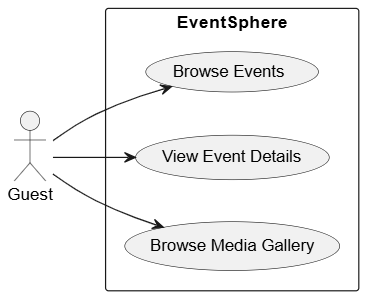
## Non – Functional Requirements

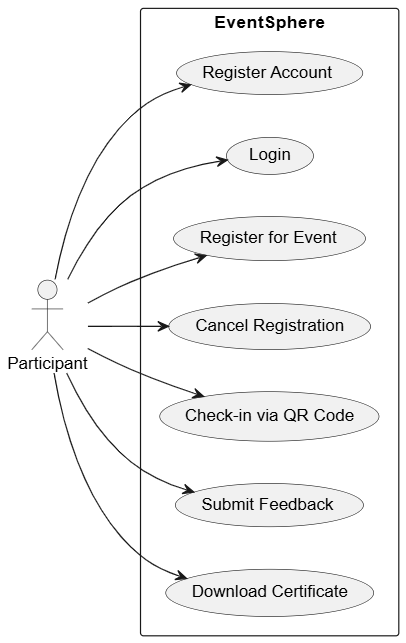
To ensure the efficient operation of the EventSphere system, it must meet the following criteria:

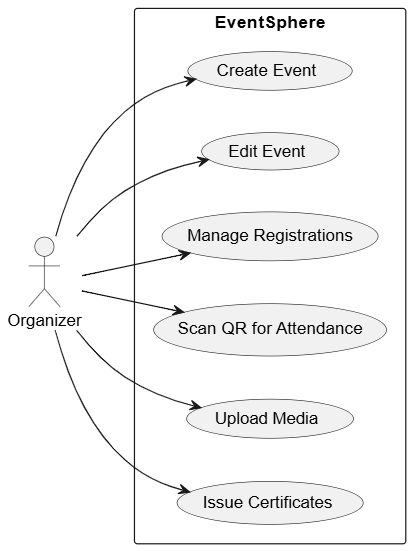
* **Performance**: Page load time under 3 seconds under stable network conditions. Capable of handling at least 100 concurrent requests without interruption.
* **Security**: Passwords stored using bcrypt hashing standard. Authentication via JWT, with role-based access control. Protection against common attacks (SQL Injection, XSS, CSRF).
* **Scalability**: Architecture supports horizontal scaling by deploying multiple backend instances. Supports database capacity expansion without service disruption.
* **Availability**: System operates 24/7 with downtime less than 1%. Includes periodic data backup mechanisms (daily/weekly).
* **Compatibility**: Supports major browsers: Chrome, Edge, Firefox, Safari. Responsive design, ensuring optimal display on desktops, tablets, and smartphones.
* **Usability**: Simple, user-friendly interface with clear menus. Includes user guides or tooltips for key operations.

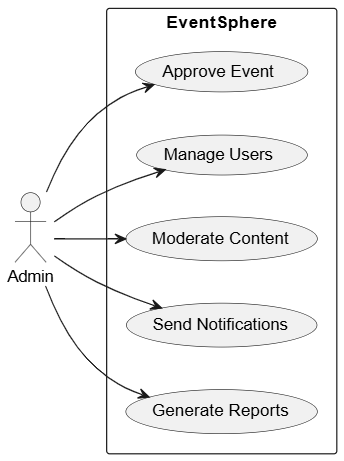
# System Diagrams

## Use Case Diagram

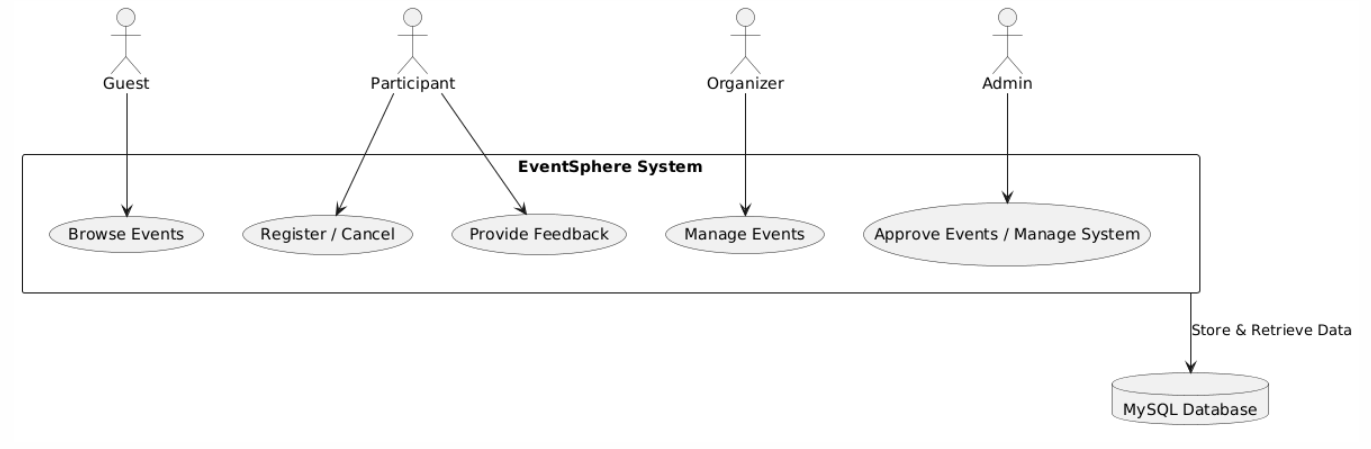
****

****

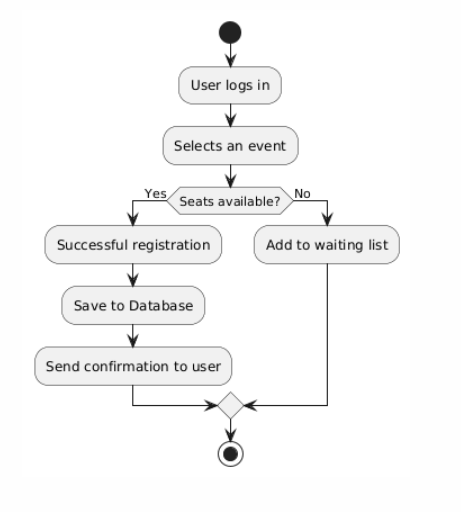
****

****

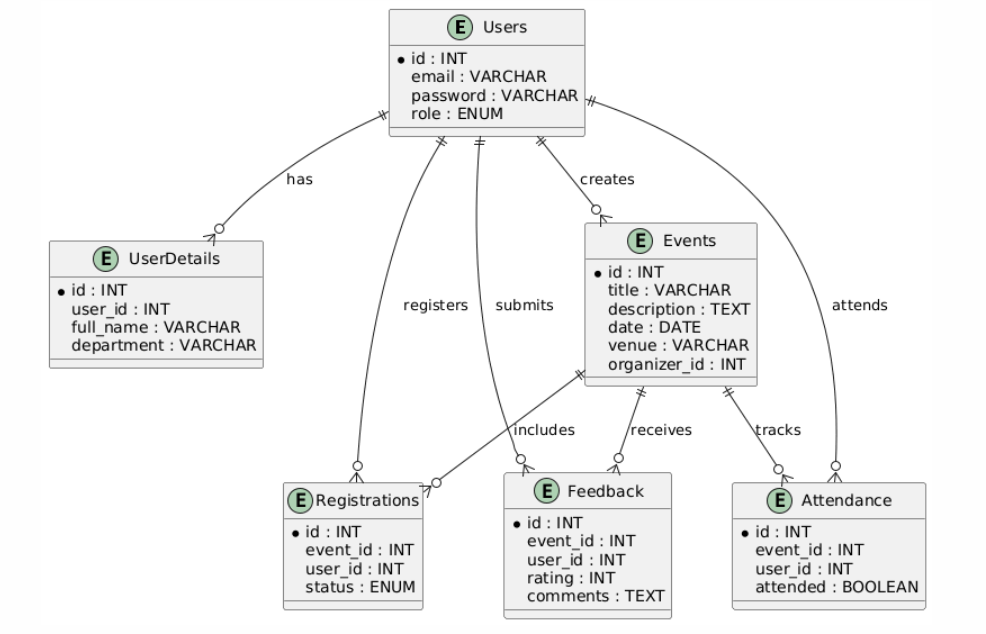
## Data Flow Diagram (DFD)

****

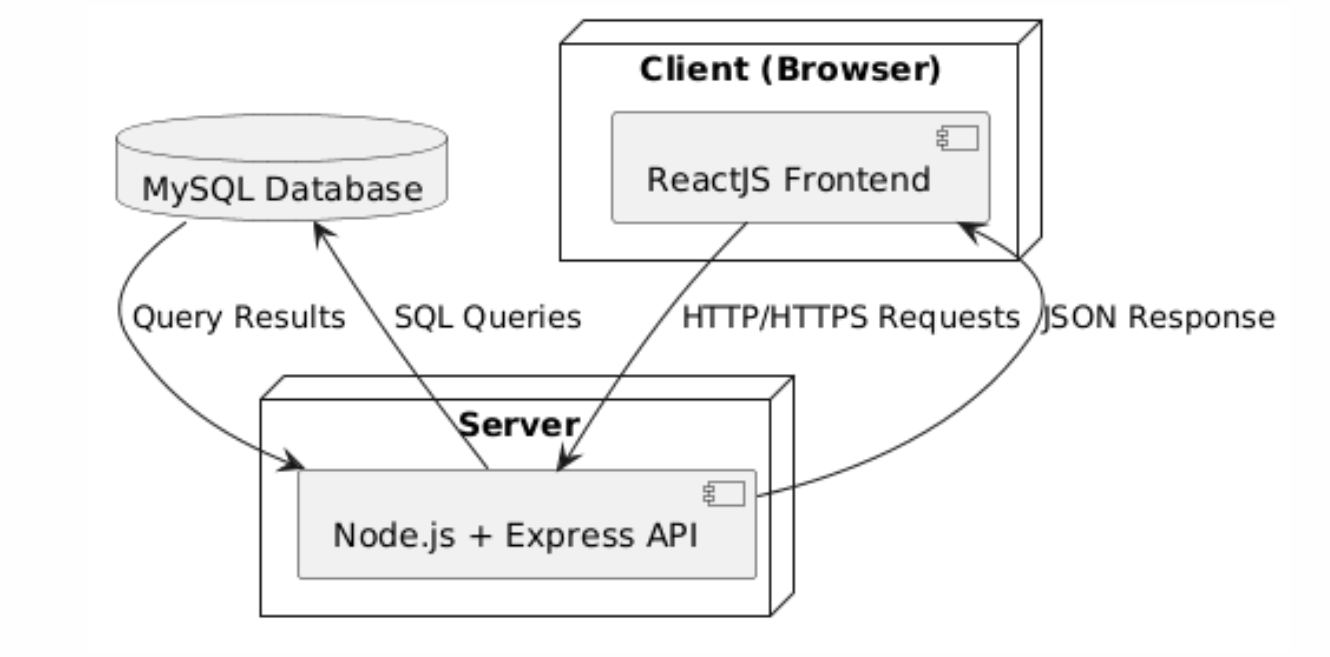
## Activity Diagram (Event Registration Process)

****

## Entity Relationship Diagram

****

## Architecture Diagram



# Database Design

## ERD Model

Ảnh có chứa văn bản, biểu đồ, Kế hoạch, sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

## Relationships Between Tables

* 1 - 1: Users – UserProfiles
* 1 – N: Roles — Users, Departments — UserProfiles, EventCategories — Events, Venues — Events, Users — Events (Organizer/Approver), Users — Notifications
* N – N: Users — Events (Registrations), Users — Events (WaitlistEntries), Users — Events (Attendances), Users — Events (QRCheckIns), Users — Events (Feedbacks), Users — Events (Certificates), Users — Media (UserFavorites)

# Test Data

## Sample Dataset

The sample dataset is constructed based on the defined tables in the ERD. Each table is populated with initial records to simulate real operations. The data is designed to cover the following purposes:

Verify primary key and foreign key constraints.

Check one-to-one, one-to-many, and many-to-many relationships.

Ensure referential integrity between tables.

Provide test cases for queries, reports, and functions of the system.

For example:

Users table: contains records of administrators, organizers, and students.

Events table: includes different types of events with corresponding organizers.

Registrations table: demonstrates the relationship between students and events.

## Test Scenarios

To ensure the system operates correctly, several main test scenarios have been developed:

1. Adding a new event

* The Organizer creates the Startup.
* The Admin approves the event → status changes from pending to approved.

1. Student registers for an event

* The student [student01@gmail.com](mailto:student01@gmail.com) registers to participate in the AI Workshop.
* The system checks the available seats and confirms successful registration.

1. Administrator approves the event

* The Admin logs in → checks the list of pending events.
* Selects the event and approves it.

1. Attendance via QR code

* The student arrives at the event, the Organizer scans the QR code.
* The system updates the attended column = TRUE.

1. Student submits feedback

* After attending the AI Workshop, the student leaves a 5-star rating along with comments.
* The feedback is stored and displayed in the report for the Organizer/Admin.

# Project Installation Guide

## System Requirements

Hardware

* + CPU: Minimum 2 cores, recommended 4 cores or more.
  + RAM: Minimum 4GB, recommended 8GB.
  + Hard Drive: 2GB free space (including code, database, and media files).

Software

* + Operating System: Windows 10/11, Linux Ubuntu 20.04+, or macOS Monterey+.
  + Recommended IDE: Visual Studio Code (latest version).
  + Node.js: v18 or higher.
  + MySQL: v8.0 or higher.
  + Browser: Latest Chrome, Edge, or Firefox.

## Installation Steps

* 1. Download source
  2. Set up environment :

Open the terminal and run

* + cd EventSphereBackend
  + node server.js
  1. Import file db.sql
  2. Run application

## Post-Installation Checks

* Access the homepage and verify that the event list displays correctly.
* Log in using admin, organizer, and participant accounts.
* Try creating a new event → check in the database.
* Try registering for an event using a student account.
* Verify that the QR code attendance function works.
* Submit feedback after an event, ensuring the feedback is saved and displayed.

# Sample Login Information

For testing purposes, several accounts have been pre-created in the database:

**Guest:** No login required → can only view public event information.

**Participant (Student):**

Email: student@test.com  
Password: password

**Organizer:**

Email: organizer@test.com  
Password: password

**Administrator (Admin)**

Email: admin@test.com  
Password: password

# Appenddix

## Glossary

* EventSphere: The name of the event management system being built.
* Frontend: The user interface layer, developed using ReactJS.
* Backend: The business logic and API processing layer, developed using Node.js + Express.
* Database (DB): The data storage repository for system information, using MySQL.
* Participant: Students or users who register to attend events.
* Organizer: The organizing committee responsible for creating and managing events.
* Admin: The highest-level manager with approval and system control privileges.
* QR Check-in: The attendance mechanism via QR code, which confirms student participation in the event.
* Feedback: Responses and evaluations from participants about event quality.
* JWT (JSON Web Token): The method for user authentication and authorization.

## References

* IEEE 1016-2009 – Systems and Software Engineering — Software Design Descriptions.
* Original SRS document of the College Event Management System project.
* Official documentation: ReactJS Documentation, Node.js Documentation, MySQL Documentation.
* UML/ERD design support tools: PlantUML Web Server.