Software Requirements Specification

for

UniEvent

Version 1.0

Prepared by

Group Name:

MEER SHAH ZAAHID	SE22UCSE165	se22ucse165@mahindrauniversity.com
A.V.S. AKASH	SE22UCSE288	se22ucse288@mahindrauniversity.edu.in
MIHIR VOHRA	SE22UCSE170	se22ucse170@mahindrauniversity.edu.in
VARSHITHA REDDY	SE22UCSE284	se22ucse284@mahindrauniversity.edu.in
KHYATHI	SE22UCSE289	se22ucse289@mahindrauniversity.edu.in
SATHWIKA	SE22UCSE105	se22ucse105@mahindrauniversity.edu.in

Instructor:	Dr.Vijay Rao
Course:	Software Engineering
Lab Section:	*PW*
Teaching Assistant:	*Swapna Ma'am*
Date:	10-03-24

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft Type and Number	Full Name	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	00/00/00

1 Introduction

UniEvent is an **automated university event management system** designed to streamline event approvals and management at **Mahindra University**. Student clubs often face a cumbersome and time-consuming approval process involving multiple levels of authorization. UniEvent simplifies this process by automating approvals, improving communication between clubs and university authorities, and enhancing event organization efficiency.

1.1 Document Purpose

This document outlines the **functional and non-functional requirements** of UniEvent. It serves as a reference for the faculty and stakeholders, ensuring that the system meets the intended objectives. The SRS defines the scope of UniEvent, detailing its functionalities, user roles, and constraints to guide the development process.

1.2 Product Scope

UniEvent is a **comprehensive event management platform** catering to over 30 student-run clubs at Mahindra University. It provides an **integrated system** for:

- Event proposals and submissions and automates the workflow
- Venue booking and logistics coordination
- Tracking student engagement and participation rewards
- Volunteer management for event organization

The system benefits students, club heads, and university authorities by reducing manual workload, improving transparency, and ensuring seamless event organization.

1.3 Intended Audience and Document Overview

This document is intended for:

- **Professors** - To evaluate the entire system
- **University Administration & Club Heads** – To ensure the system meets approval workflows
- **Developers** – To understand system requirements and functionality
- **UI/UX Designers** – To design an intuitive interface
- **Testers** – To verify system performance and correctness

The document covers system functionalities, constraints, and implementation details. Readers should begin with the **overview sections** before proceeding to technical specifications.

1.4 Definitions, Acronyms and Abbreviations

API – Application Programming Interface
Badge System – Recognition system tracking student event participation
SQL – Structured Query Language
SRS – Software Requirements Specification
UI/UX – User Interface / User Experience

1.5 Document Conventions

This document follows **IEEE formatting standards**, using:

- **Arial 11/12 pt font** for text
- **Italics** for comments and additional notes
- **Single spacing** and **1-inch margins**
- **Bold headings** for sections

1.6 References and Acknowledgments

This document refers to:

- Mahindra University’s event approval policies
- Interface design guidelines for the web application
- System requirement specifications and scope documents

2 Overall Description

2.1 Product Overview

UniEvent is a **newly developed system** aimed at replacing the traditional, manual event approval and management process at Mahindra University. It provides an automated, centralized platform to handle event proposals, venue bookings, and student participation tracking.

System Overview Diagram: *(To be inserted, showing user interactions with UniEvent.)*

2.2 Product Functionality

The system will offer the following major functionalities:

- **Multi-role login system:** Different roles for Club Heads, Approval Authorities, Logistics, and Students.
- **Event Proposal & Approval Workflow:** Automating the event approval process.
- **Venue Booking & Logistics Integration:** Managing venue reservations and logistical arrangements.
- **Student Engagement Portal:** Tracking participation with a **badge-based recognition system**.
- **Volunteer Management System:** Streamlining volunteer recruitment and assignments.

2.3 Design and Implementation Constraints

Design and Implementation Constraints

1. Software Design Methodology and Modeling
The COMET (Concurrent Object Modeling and Architectural Design Method) will be used to ensure modularity and structured development. This method is well-suited for concurrent, distributed, and real-time applications like UniEvent, where multiple stakeholders interact simultaneously.
UML (Unified Modeling Language) will be used for system design, including class diagrams, sequence diagrams, and state transitions to model event approvals, role-based access control, and logistics integration.
References:
Gomaa, H. (2011). Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures. Cambridge University Press.
Gomaa, H. (2000). Designing Concurrent, Distributed, and Real-Time Applications with UML. Addison-Wesley.

2. Hardware and Performance Constraints
University IT Infrastructure: UniEvent must be deployed within Mahindra University's server environment, which imposes hardware limitations such as storage capacity, processing power, and concurrent user handling.
Real-Time Processing: The system must efficiently handle multiple event proposals, approvals, and student interactions in real time while preventing performance bottlenecks.

3. Technology Stack Constraints
Frontend: Developed using React.js for a user-friendly interface that allows event proposal, approval tracking, and student engagement.
Backend: Implemented using Node.js with Express to manage workflows for event approvals, venue booking, and logistics coordination.
Database: PostgreSQL will be used to store event data, user roles, approvals, and student participation records.
Hosting: Deployment will be on university-provided cloud servers, with an alternative option of AWS in case of resource limitations.

4. Security Considerations
Role-Based Access Control (RBAC): The system must support different roles, including Club Heads, Approval Authorities, Logistics Teams, and Students, ensuring proper access privileges.
Authentication Integration: UniEvent must integrate with Mahindra University's student authentication system to provide a secure login mechanism.
Data Privacy and Protection: All event approval records, student participation data, and logistics details must be encrypted and securely stored.

5. Integration and Communication Protocols
Student Portal Integration: The system must interface with the university's student portal to validate users and streamline approvals.
Calendar Synchronization: Approved events must be synchronized with student calendars via Google Calendar API or a university-maintained scheduling system.
Notification System: Email and push notifications must be implemented to keep users informed about approvals, upcoming events, and changes.
6. Institutional Constraints
Compliance with University Policies: The system must follow Mahindra University's IT policies regarding data security and event approval regulations.
Project Timeline Restrictions: The development and deployment must be completed within the academic semester, aligning with the milestones outlined in the project plan.
Budgetary Limitations: Limited funds may affect hosting choices, API integrations, and third-party software dependencies.

2.4 Assumptions and Dependencies

Assumptions

- Timely Feedback: University administration, club heads, and logistics teams will provide prompt feedback on system functionality.
- IT Support and Resources: The university will allocate server access and IT infrastructure for hosting the platform.
- User Participation: Club representatives and faculty members will actively engage in system testing and feedback sessions.

Dependencies

- Authentication System: The project depends on the university's existing student authentication system. Any changes to this system may require modifications to UniEvent's login mechanism.
- Third-Party APIs: The system relies on Google Calendar API (or equivalent university calendar integration) to sync approved events.
- Regulatory Compliance: The project is subject to Mahindra University's policies for event approvals and data privacy.
- Project Timeline: The completion of the project depends on timely approvals from university administration and faculty advisors.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The UNIEvent website will feature an intuitive and user-friendly interface to ensure smooth interaction for all users, including students, faculty members, and event coordinators.

- ***Design & Navigation**: The interface will have a modern and clean layout with well-structured menus and easily accessible buttons.*
- ***User Interaction**:*
  - *Users will engage with the platform through a **graphical user interface (GUI)**, comprising interactive elements such as drop-down menus, buttons, search bars, and calendar views.*
  -
  - *The website will be optimised for **desktop and mobile browsers** to ensure accessibility across different devices.*
  -
  - *Event organizers will have exclusive dashboards to create, modify, and manage events, while attendees can register, bookmark events, and receive notifications.*

3.1.2 Hardware Interfaces

The system will communicate with various hardware components to ensure a seamless experience. The primary hardware interfaces include:

- **Web Servers**: The platform will be hosted on a server capable of managing multiple user requests, event listings, and registrations efficiently.
- **User Devices**: The website will be accessible from computers, laptops, tablets, and smartphones with an internet connection.
- **Notification Systems**: Integration with SMS and email servers will facilitate automated event confirmations, reminders, and updates.
- **Printers (Optional)**: Event organizers may connect to printers to generate physical copies of event tickets, schedules, or attendee lists.

3.1.3 Software Interfaces

The event management system will interact with multiple software components to enable key functionalities, including:

- ***Database Management System**: A relational database (such as MySQL or PostgreSQL) will store user accounts, event details, and registration data securely.*
- ***Email & Notification APIs**: Services like SendGrid or Twilio will handle automated email and SMS notifications.*
- ***Authentication System**: The platform will integrate with the university's Single Sign-On (SSO) system to ensure secure user authentication.*

- ***Third-Party Calendar Synchronization**: Users can sync event schedules with Google Calendar or Outlook for better event management.*

3.2 Functional Requirements

< Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. This section is the direct continuation of section 2.2 where you have specified the general functional requirements. Here, you should list in detail the different product functions.

3.2.1 **F1**: The system shall enable Club Heads to submit event proposals via an online submission form.

F2: The system shall offer an automated workflow to facilitate event approval by designated university authorities.

F3: The system shall provide real-time notifications to club heads regarding the status of their event approvals.

F4: The system shall allow students to explore and sign up for events.

F5: The system shall record and monitor student attendance for each event.

F6: The system shall issue digital badges and certificates to students based on their participation in events.

F7: The system shall maintain and display an event calendar that is accessible to all users.

F8: The system shall generate detailed reports on events, including participation and attendance metrics.

F9: The system shall implement role-based access controls for students, club heads, and university authorities.

F10: The system shall provide tools for collecting feedback from event organizers and participants.

3.3 Use Case Model

3.3.1 Use Case

1. Author

Akash, Sathiwika, Varshitha, Mihir

2. Purpose

This use case defines how students, event organizers, and admins interact with the University Event Management System to create, manage, discover, and register for events. It also covers user authentication, notifications, and report generation.

3. Requirements Traceability

- REQ-001: Users must be able to register and log in.
- REQ-002: Event organizers can create and manage events.
- REQ-003: Admins must approve events before publishing.
- REQ-004: Students can browse and register for events.
- REQ-005: The system must send event notifications.
- REQ-006: The system should generate reports for analytics.

4. Priority

High – This use case is essential for system functionality.

5. Preconditions

- The user must have an account to log in.
- The event organizer must have the necessary permissions to create events.
- The payment system must be active (if ticketed events are involved).

6. Postconditions

- Users successfully log in or register.
- Events are created, approved, and published.
- Students register for events and receive confirmation.
- Notifications are sent to relevant users.
- Reports are generated based on event analytics.

7. Actors

- Students – Register, log in, browse events, and register.
- Event Organizers – Create and manage events.
- Admins – Approve events and generate reports.
- Notification System (External) – Sends event updates.

8. Extends

- "Getting Approval from Admin" extends "Event Creation & Management" because some events require admin approval before publishing.

9. Flow of Events

Basic Flow (Main Scenario)

1. A student registers and logs into the system.
2. An event organizer creates an event.
3. If required, the event is sent for admin approval.
4. Admin reviews and approves the event.
5. The event is published, and students can browse/register.
6. A student registers for an event and receives a confirmation email.
7. Notifications are sent to registered users about event updates.
8. Admin generates reports for event performance.

Alternative Flow

- If an event does not require approval, it is published immediately after creation.
- If the user forgets their password, they can reset it.

Exceptions

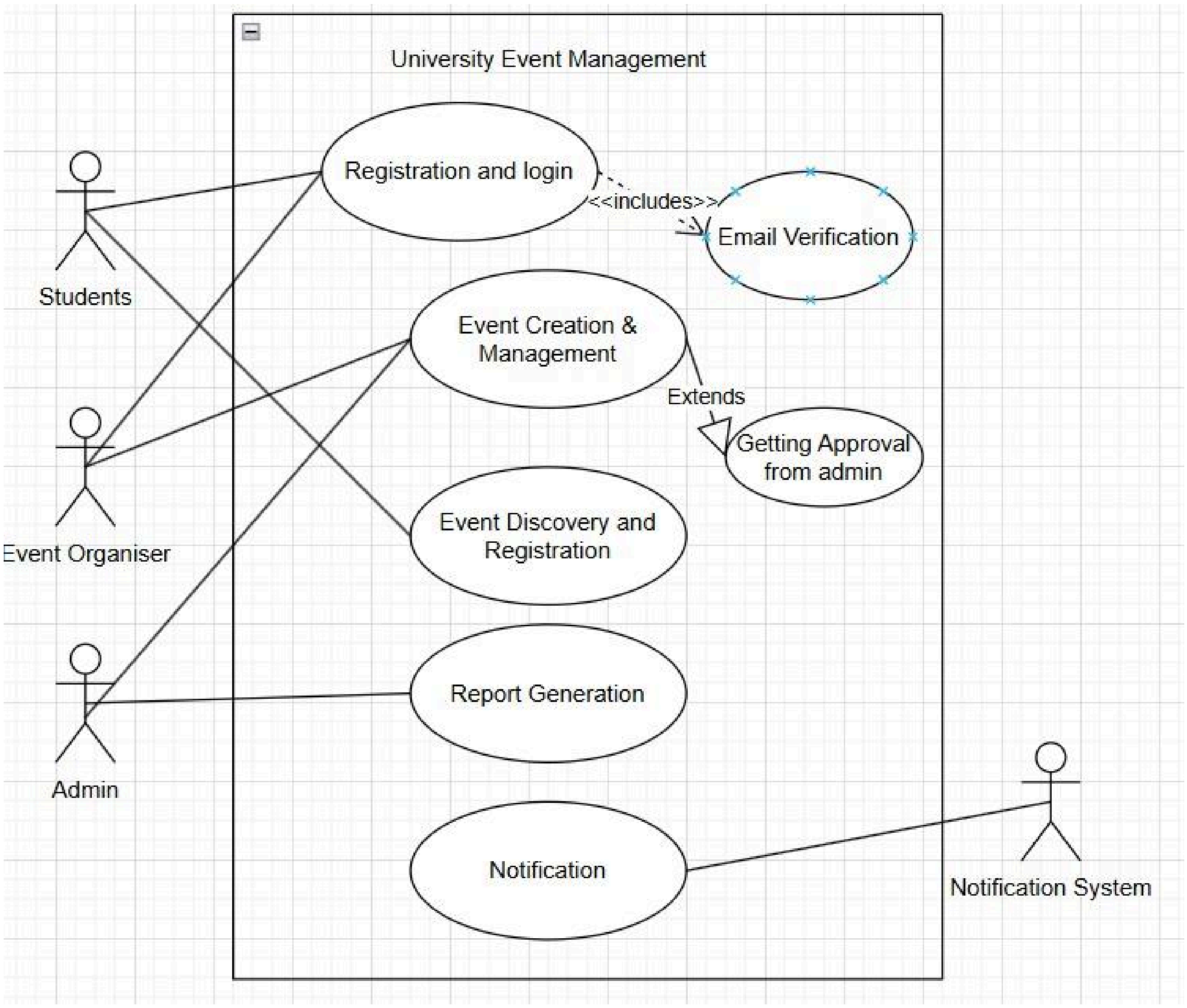
- If login fails due to incorrect credentials, an error message is displayed.
- If event approval is rejected, the organizer is notified.
- If payment fails, registration is canceled.

10. Includes (Other Use Cases)

- "Registration & Login" «includes» "Email Verification" because every user must verify their email.

11. Notes/Issues

- Ensure seamless integration with the Notification System.
- Implement a user-friendly event approval process.
- Consider adding a payment gateway for ticketed events.


University Event Management
Registration and login
<<includes>>
Email Verification
Event Creation & Management
Extends
Getting Approval from admin
Event Discovery and Registration
Report Generation
Notification
Students
Event Organiser
Admin
Notification System

4 Other Non-functional Requirements

4.1 Performance Requirements

P1: The system must support at least 100 concurrent users without performance degradation.

P2: Event approvals should be processed within 5 seconds of submission.

P3: The system should be capable of handling peak loads during university events without a noticeable decline in performance.

P4: Database queries should return results promptly to ensure efficient data retrieval.

P5: Automated reports should be generated swiftly upon request.

P6: The system should maintain consistent performance during simultaneous event registrations and approvals.

4.2 Safety and Security Requirements

4.2.1 User Authentication and Access Control

Role-Based Access Control (RBAC):

- Users will be assigned different roles: **Club Heads, Students, Approval Authorities, Logistics Team**.
- Permissions will be enforced based on roles (e.g., only Club Heads can submit event proposals).

Login System:

- Users will log in using their **university-provided credentials** (email and password).
- Basic **password hashing** (e.g., bcrypt) will be implemented for security.

Session Timeout:

- Users will be logged out **after 20 minutes** of inactivity.

4.2.2 Data Protection and Privacy

Basic Data Encryption:

- User passwords will be stored **hashed** to prevent direct access.
- Sensitive event details (approvals, rejections) will be protected using **basic database security mechanisms**.

Limited Data Access:

- Students can view events but cannot modify them.
- Club Heads can only manage events they created.

Data Storage and Retention:

- Event data will be **deleted at the end of the semester** to free up space.

4.2.3 System Security Against Basic Threats

Preventing Unauthorized Access:

- Input validation to prevent **SQL injection attacks**.
- Users will be restricted from accessing pages outside their role.

Basic Protection Against Bots & Spam:

- CAPTCHA will be added to the login page and event submission form.

Simple Audit Logging:

- Key actions (event approval, and role changes) will be **logged** in the database.

4.2.4 Compliance with University Guidelines

- **Adherence to University IT Policies:**
  - The system will follow **basic university IT guidelines** for data security.
- **No External Data Sharing:**
  - All student data will be used **only within the university environment**.

4.2.5 Handling System Failures & Errors

Automated Weekly Backups:

- The database will be backed up **once a week** to prevent data loss.

Error Handling Messages:

- Instead of generic system crashes, users will see a **friendly error page** if something goes wrong.

Basic Server Monitoring:

- If the system slows down or crashes, an email notification will be sent to the development team.

4.3 Software Quality Attributes

4.3.1 Reliability

Reliability is crucial since UniEvent will manage critical university event approvals and logistics. The system will achieve high reliability through:

- ***Automated Error Handling**: Built-in exception handling mechanisms to detect and recover from failures.*
- ***Database Redundancy**: Data backup mechanisms to prevent data loss in case of failures.*
- ***Load Testing**: Stress tests to ensure the system functions correctly under high user load.*
- ***Uptime Monitoring**: Ensuring at least **99% uptime** with proactive monitoring and alerts.*

4.3.2 Usability

As UniEvent will be used by students, club heads, and university administrators, its usability is a key focus. To achieve ease of use:

- ***Intuitive UI Design**: A user-friendly, role-based interface ensuring a smooth workflow.*
- ***Minimal Learning Curve**: A guided onboarding process and tooltips to help new users.*
- ***Mobile Responsiveness**: Ensuring full functionality across mobile and desktop devices.*
- ***Accessibility Compliance**: Adhering to WCAG standards for visually impaired users.*

4.3.3 Maintainability

Maintainability ensures that future changes and enhancements can be made efficiently. This will be addressed by:

- ***Modular Codebase**: Using a microservices architecture to enable independent feature updates.*
- ***Code Documentation**: Well-documented APIs and internal comments to ease future modifications.*
- ***Automated Testing**: Unit and integration tests for early bug detection and smoother updates.*

4.3.4 Interoperability

Since UniEvent may need to integrate with existing university systems, interoperability is key. It will be ensured through:

- ***API-First Approach**: Providing RESTful APIs for seamless integration with university databases.*
- ***Third-Party Compatibility**: Supporting common authentication methods (e.g., OAuth for university login).*
- ***Data Standardization**: Using JSON and XML formats for consistent data exchange.*

4.3.5 Security

Security is paramount since the system will store sensitive events and student data. The security measures include:

- ***Role-Based Access Control (RBAC)**: Restricting permissions based on user roles.*
- ***Data Encryption**: Using AES-256 encryption for stored data and TLS for transmission security.*
- ***Audit Logging**: Recording user activities for accountability and compliance.*
- ***Regular Security Audits**: Periodic vulnerability assessments and penetration testing.*

5 Other Requirements

NIL

Appendix A – Data Dictionary

<Data dictionary is used to track all the different variables, states and functional requirements that you described in your document. Make sure to include the complete list of all constants, state variables (and their possible states), inputs and outputs in a table. In the table, include the description of these items as well as all related operations and requirements.>

Appendix B - Group Log

<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist in determining the effort put forth to produce this document>