**Task 5 -**

**Software Requirements Specification**

**for**

**Student Club Management System**

**Version <1.0>**

**Group No.: 5**

|  |  |
| --- | --- |
| **MARCUS CHONG WEN XUAN** | **241UC2414J** |
| **KENG JING LI** | **1211106628** |
| **CHUAH YUN SHAN** | **1211107031** |
| **TAN YUN XUAN** | **1221309436** |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
| **Date created:** | **16 May 2025** |

# Table of Content

[Table of Content 2](#_Toc199082696)

[1. Introduction 4](#_Toc199082697)

[1.1. Purpose 4](#_Toc199082698)

[1.2. Scope 4](#_Toc199082699)

[1.3. Product Overview 5](#_Toc199082700)

[1.3.1. Product Perspective 5](#_Toc199082701)

[1.3.2. Product Functions 6](#_Toc199082702)

[1.3.3. User Characteristics 7](#_Toc199082703)

[1.3.4. Limitations 7](#_Toc199082704)

[1.4. Definitions 7](#_Toc199082705)

[2. References 8](#_Toc199082706)

[3. Requirements 8](#_Toc199082707)

[3.1. Functions 8](#_Toc199082708)

[3.1.1. Use case diagram 10](#_Toc199082709)

[3.1.2. Use Case Specification 11](#_Toc199082710)

[3.1.2.1. View club 11](#_Toc199082711)

[3.1.2.2. Join club 13](#_Toc199082712)

[3.1.2.3. Leave club 15](#_Toc199082713)

[3.1.2.4. View event 17](#_Toc199082714)

[3.1.2.5. RSVP to event 19](#_Toc199082715)

[3.1.2.6. View forum 21](#_Toc199082716)

[3.1.2.7. Create forum post 23](#_Toc199082717)

[3.1.2.8. Create event proposal 25](#_Toc199082718)

[3.1.2.9. Submit budget request 27](#_Toc199082719)

[3.1.2.10. Book venue 29](#_Toc199082720)

[3.1.2.11. Organize event committee 31](#_Toc199082721)

[3.1.2.12. View event history 33](#_Toc199082722)

[3.1.2.13. View financial report 35](#_Toc199082723)

[3.1.2.14. Manage forum 37](#_Toc199082724)

[3.1.2.15. Manage event 39](#_Toc199082725)

[3.1.2.16. Track attendance 41](#_Toc199082726)

[3.1.2.17. Track budget request status 43](#_Toc199082727)

[3.1.2.18. Track venue request status 45](#_Toc199082728)

[3.1.2.19. Handle budget request 47](#_Toc199082729)

[3.1.2.20. Handle venue request 49](#_Toc199082730)

[3.1.2.21. Login 51](#_Toc199082731)

[3.1.2.22. Update profile 53](#_Toc199082732)

[3.2. Performance Requirements 55](#_Toc199082733)

[3.3. Usability Requirements 55](#_Toc199082734)

[3.4. Interface Requirements 56](#_Toc199082735)

[3.4.1. System Interfaces 56](#_Toc199082736)

[3.4.2. User Interfaces 56](#_Toc199082737)

[3.4.3. Hardware Interfaces 56](#_Toc199082738)

[3.4.4. Software Interfaces 57](#_Toc199082739)

[3.4.5. Communications interfaces 57](#_Toc199082740)

[3.5. Logical Database Requirements 58](#_Toc199082741)

[3.6. Design Constraints 63](#_Toc199082742)

[3.7. Software System Attributes 63](#_Toc199082743)

[**4.** **Verification** 65](#_Toc199082744)

[4.1. Verification Approach 65](#_Toc199082745)

[4.2. Verification Criteria 66](#_Toc199082746)

[**5.** **Appendices** 68](#_Toc199082747)

[5.1. Assumptions and Dependencies 68](#_Toc199082748)

[5.1.1 Assumptions 69](#_Toc199082749)

[5.1.2 Dependencies 69](#_Toc199082750)

[5.2. Acronyms and Abbreviations 69](#_Toc199082751)

# Introduction

## Purpose

The main purpose of this document is defining the software requirements for the Student Club Management System with Budget and Venue Integration. This system will be built to streamline and centralize all the operations of student clubs and organizations within a university. This will enable efficient management of club memberships, discussion forum for club members, event planning, budget tracking, and campus venue reservations.

## Scope

Below is the student club management system’s scope:

1. **Manages administration**: The system will handle the university staff’s task for student clubs and organizations.
2. **Handles submission and approval**: It will manage the process of submitting and getting approval for various club-related items (likely event proposals, funding requests, etc.).
3. **Tracks event proposals**: The system will keep track of event proposals.
4. **Facilitates membership management**: It will help manage club memberships
5. **Manages club budgets and financial transactions**: The system will handle the financial aspects of clubs, including budgets and transactions.
6. **Allows reservation for campus venues**: Clubs will be able to reserve campus spaces through the system.
7. **Allow Able club members to discuss clubs related topics:** The system provides a platform for clubs’ members to have discussions among themselves.

## Product Overview

The Student Club Management System is a platform that will support the club operations by integrating with the university financial system and campus space reservation database. It also allows students to manage memberships as well as participate in events and have discussion in forum, while admins including finance staff and venue staff can handle budget and venue bookings respectively. Club leaders can organize events, submit budget and venue request.

### Product Perspective

The Student Club Management System is a web application that integrates with the university’s digital infrastructure. It is built to operate as a broader university ecosystem while connecting with multiple core systems to support the student club activities. The system interfaces with the university authentication system to secure login control for students, club leaders, and university staff such as finance and venue staff.

Therefore, it integrates with the financial system to enable budget tracking, funding requests, and financial approvals related to the club operations. Through these integrations, the system also enhances operational efficiency, promotes transparency, and supports coordination among students and administrative departments.

|  |
| --- |
|  |

*Figure 1.3.1 Context diagram*

### Product Functions

Below are the primary functions of the software:

1. View available clubs and events
2. Allow students to join or leave student clubs
3. Allow club leaders to check venue details
4. Enable staffs to approve budgets and venue bookings
5. Display available campus venues for event planning
6. Provide a dashboard to track club activities and notifications
7. Provide forum platform for students to discuss

### User Characteristics

The users of the Student Club Management System are students, club leaders, finance and venue staff. Students usually have basic computer and technological skills. Club leaders probably would have experience in digital tools, as they will be needed to perform tasks such as event creation. Finance and venue staff should have technical proficiency, particularly in reviewing and approving submissions.

### Limitations

The system is not limited to managing student organizations, it’s recognized by the university. This can only process budget requests through the university’s existing financial system, and venue reservations are usually restricted to spaces listed in the campus reservation system. This system doesn’t support external users or off-campus events. Therefore, performance may be dependent on the availability of the systems, such as authentication, finance, and venue databases.

## Definitions

**Application:** A web-based platform used to manage student club activities, events, budgets, and venue bookings.

**Venue Booking:** The page of reserving a university venue spaces for club events through the reservation system

**Financial Management System:** The university’s financial system to manage budgets and transactions, which the application connects to for financial operations.

**Authentication System:** The university’s login system used to verify user identity and provide access level-based roles such as student, staff, and others.

**SSO:** Single Sign On, a feature that allows users to log in to multiple applications or services with a single set of credentials to eliminate the need for remembering multiple usernames and passwords.

# References

IEEE. (2018). *ISO/IEC/IEEE 29148:2018 Systems and software engineering—Life cycle processes—Requirements engineering*. <https://www.iso.org/standard/72089.html>

Sheketa, V., Pasieka, M., Serman, T., Pasieka, N., Chupakhina, S., & Krul, L. (2021). System analysis and example of using SHA-512 Hash functions to protect students’ personal data on educational platforms. *2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)*, 132–136. <https://doi.org/10.1109/csit52700.2021.9648699>

Grassi, P. A., Fenton, J. L., Newton, E. M., Perlner, R. A., Regenscheid, A. R., Burr, W. E., Richer, J. P., Lefkovitz, N. B., Danker, J. M., Choong, Y., Greene, K. K., & Theofanos, M. F. (2017). *Digital identity guidelines: authentication and lifecycle management.* <https://doi.org/10.6028/nist.sp.800-63b>

Pohl, K. (2010). *Requirements engineering: Fundamentals, principles, and techniques*. Springer.

# Requirements

* 1. Functions

This section outlines the functional requirements for a student club management system that includes budget and venue integration. The functions are organized by system actors in the Table 3.1 below and represented using use case diagram in Section 3.1.1. The interactions and flows of each function are further explained in use case specifications and activity diagrams in Section 3.1.2.

*Table 3.1 Functions by Actors*

|  |  |  |
| --- | --- | --- |
| **Actor** | **Function** | **Description** |
| Student | * View club * Join club * Leave club * View event * RSVP to event * View forum * Create forum post | The system shall allow a student to view available clubs, join a club, and leave a club. The system shall also allow a student to browse upcoming events, RSVP to attend events, view the club discussion forum, and create new posts. |
| Club leader | * Create event proposal * Submit budget request * Book venue * Organize event committee * View event history * View financial report * Manage forum * Manage event * Track attendance * Track budget request status * Track venue request status | The system shall allow a club leader to create event proposals, submit budget requests, and book venues for upcoming events. The system shall also allow them to organize event committees, view past event history, view financial reports, manage forum discussions, manage event details, and track attendance of participants. In addition, the system shall also allow a club leader to track budget and venue request status. |
| Finance staff | * View budget request | The system shall allow a finance staff to review budget requests submitted by club leaders for approval or reject. |
| Venue staff | * View venue request | The system shall allow a venue staff to process venue booking requests submitted by club leaders for approval or reject. |
| All actors | * Login * Update profile | The system shall allow a user to log in and update profile excluding student ID and student email address. |

### Use case diagram

|  |
| --- |
| A diagram of a diagram  AI-generated content may be incorrect. |

*Figure 3.1.2.1 use case diagram*

### Use Case Specification

### View club

|  |  |
| --- | --- |
| **Use Case ID** | UC001 |
| **Use Case Name** | View club |
| **Description** | Students can view a list of clubs with details like club’s name, description, and members. |
| **Goal** | To allow students browse available clubs. |
| **Actor** | Student |
| **Precondition** | Student must be logged into the system with valid credentials. |
| **Postcondition** | Clubs are displayed on screen. |
| **Main Scenario** | 1. Student logs into the system. 2. Student navigates to the “Clubs” section. 3. System retrieves a list of all available clubs. 4. Student browses through the clubs. 5. Student selects a club to view details. 6. System displays the selected club’s name, description, and member list. |
| **Alternative Scenario** | * If the club database is temporarily unavailable, an error message is shown. * If no clubs are available, the system displays “No clubs found.” |

|  |
| --- |
| A diagram of a program  AI-generated content may be incorrect. |

*Figure 3.1.2.1 Activity diagram - view club*

### Join club

|  |  |
| --- | --- |
| **Use Case ID** | UC002 |
| **Use Case Name** | Join club |
| **Description** | Students can join a club directly in the system. |
| **Goal** | To allow students to become a member of the club they are interested in. |
| **Actor** | Student |
| **Precondition** | Students must be logged into the system and not a member of the selected club. |
| **Postcondition** | Students become a member of the selected club. |
| **Main Scenario** | 1. Student logs into the system. 2. Student navigates to the “Clubs” section. 3. System displays a list of all available clubs. 4. Student views the selected club’s details. 5. Student clicks on “Join” button. 6. System verifies that the student is not a member of the club. 7. System adds the student to the member list. 8. System confirms the membership and displays a success message. |
| **Alternative Scenario** | * If the student is already a member, the system displays “You are already a member of this club.” |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.2 Activity diagram - join club*

### Leave club

|  |  |
| --- | --- |
| **Use Case ID** | UC003 |
| **Use Case Name** | Leave club |
| **Description** | Student can withdraw membership from a club. |
| **Goal** | To allow students manage their club involvement by leaving clubs they no longer wish to stay. |
| **Actor** | Student |
| **Precondition** | Student must be logged into the system and currently a member of the club. |
| **Postcondition** | Student is removed from member list of the selected club. |
| **Main Scenario** | 1. Student logs into the system. 2. Student views the list of clubs they have joined. 3. Student selects a club they wish to leave. 4. Student clicks “Leave” button, the system prompts a confirmation dialog to confirm deletion. 5. Student confirms the deletion. 6. System removes the student from the club’s member list. 7. System displays confirmation of successful message. |
| **Alternative Scenario** | * If the removal fails due to system issues, an error message is shown. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.3 Activity diagram - leave club*

### View event

|  |  |
| --- | --- |
| **Use Case ID** | UC004 |
| **Use Case Name** | View event |
| **Description** | Student can browse the details of upcoming events. |
| **Goal** | To allow students explore the event information. |
| **Actor** | Student |
| **Precondition** | * Student must be logged into the system. * Event has been created and posted. |
| **Postcondition** | Student can browse the details of upcoming events. |
| **Main Scenario** | 1. Student logs into the system. 2. Student navigates to the “Events” section. 3. System displays a list of upcoming events. 4. Student browses through the events. 5. Student selects an event to view details. 6. System displays the selected event’s information. |
| **Alternative Scenario** | * If the event database is temporarily unavailable, an error message is shown. * If no events are available, the system displays “No events found.” |

|  |
| --- |
| A diagram of a program  AI-generated content may be incorrect. |

*Figure 3.1.2.4 Activity diagram – view event*

### RSVP to event

|  |  |
| --- | --- |
| **Use Case ID** | UC005 |
| **Use Case Name** | RSVP to event |
| **Description** | Student can respond to an event invitation. |
| **Goal** | To allow students inform the club leader about their participation in upcoming event. |
| **Actor** | Student |
| **Precondition** | * Student must be logged into the system. * Event must be available to RSVP. |
| **Postcondition** | Student’s RSVP is recorded and saved. |
| **Main Scenario** | 1. Student logs into the system. 2. Student navigates to the “Events” section. 3. System displays a list of upcoming events. 4. Student selects an upcoming event to view details. 5. System displays the selected event’s information. 6. Student clicks “Yes” to confirm participation. 7. System records the RSVP by student. |
| **Alternative Scenario** | * If student selects “No” instead of “Yes”, the system records that the student will not attend the event. * If RSVP submission fails due to system error, an error message is shown. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.5 Activity diagram – RSVP to event*

### View forum

|  |  |
| --- | --- |
| **Use Case ID** | UC006 |
| **Use Case Name** | View forum |
| **Description** | Students can view forum posts related to the club they joined, they can browse posts and discussions in the forum. |
| **Goal** | To allow students to read forum discussions and stay informed about club activities. |
| **Actor** | Student |
| **Precondition** | * Student must be logged into the system. * Student must be a club member to access the forum. |
| **Postcondition** | Forum posts from the clubs the student joined are displayed. |
| **Main Scenario** | 1. Student logs into the system. 2. Student navigates to the “Forum” section. 3. System displays all forum posts’ topic or title. 4. Student browses posts. 5. Student selects a post to view details (author, content, date/ time created), and comments. |
| **Alternative Scenario** | * If no posts exist, the system displays “No posts available” message. |

|  |
| --- |
| A diagram of a system  AI-generated content may be incorrect. |

*Figure 3.1.2.6 Activity diagram – view forum*

### Create forum post

|  |  |
| --- | --- |
| **Use Case ID** | UC007 |
| **Use Case Name** | Create forum post |
| **Description** | Student can create a new post in the forum and publish it. |
| **Goal** | To allow students to create a new discussion and share information. |
| **Actor** | Student |
| **Precondition** | * Student must be logged into the system. * Student must be the club member of that club to assess the forum. |
| **Postcondition** | New post created and visible in the forum. |
| **Main Scenario** | 1. Student logs into the system. 2. Student navigates to the “Forum” section. 3. Student selects to add a new post. 4. Student enters post title and content. 5. Student clicks on “Submit” button. 6. System saves and displays the post in the forum. |
| **Alternative Scenario** | * If any required fields are empty, system prompts users to fill. * If submission fails, system shows an error message. |

|  |
| --- |
| A diagram of a website  AI-generated content may be incorrect. |

*Figure 3.1.2.7 Activity diagram – create forum post*

### Create event proposal

|  |  |
| --- | --- |
| **Use Case ID** | UC008 |
| **Use Case Name** | Create event proposal |
| **Description** | Club leader can create proposal for new event, including organize committees and submit budget requests. |
| **Goal** | To allow club leader to propose new event with details like event type, schedules. |
| **Actor** | Club leader |
| **Precondition** | Club leader must be logged into the system. |
| **Postcondition** | Event proposal is created with committee organization and budget request submitted for approval. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Event Planning” section. 3. Club leader fills in event details (title, type, date, time, description). 4. Club leader uploads a full document of the proposal. 5. Club leader organizes the event committee by entering the names and roles. 6. Club leader fills in budget request form. 7. Club leader books a venue. 8. Club leader clicks on “Submit” to confirm submission of the event proposal. 9. System saves proposal, committee details, and send request for review. |
| **Alternative Scenario** | * If required fields are missing, the system prompts the club leader to complete all fields. * If submission fails due to system error, the system notifies the club leader. |

|  |
| --- |
| A diagram of a company  AI-generated content may be incorrect. |

*Figure 3.1.2.8 Activity diagram – create event proposal*

### Submit budget request

|  |  |
| --- | --- |
| **Use Case ID** | UC009 |
| **Use Case Name** | Submit budget request |
| **Description** | Club leader can request an estimated budget for event. |
| **Goal** | To allow club leader to submit proposed budget for finance staff review. |
| **Actor** | Club leader |
| **Precondition** | Club leader must be logged into the system and creating an event proposal. |
| **Postcondition** | Budget request submitted for approval by finance staff. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Event Planning” section. 3. Club leader accesses the “Request Budget” page. 4. Club leader fills in estimated costs and descriptions. 5. Club leader submits the budget request. 6. System sends the request to finance staff for approval. 7. System displays the confirmation of submission. |
| **Alternative Scenario** | * If fields are incomplete or invalid, system prompts club leader to correct the errors. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.9 Activity diagram – submit budget request*

### Book venue

|  |  |
| --- | --- |
| **Use Case ID** | UC0010 |
| **Use Case Name** | Book venue |
| **Description** | Club leader can request a venue for an upcoming event. |
| **Goal** | To reserve a campus venue by submitting a booking request. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system. * Club leader must be working on event proposal. * Venue must be available. |
| **Postcondition** | Venue request is submitted for approval by venue staff. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Event Planning” section. 3. Club leader accesses the “Book Venue” page. 4. Club leader selects preferred venue, date, time. 5. System checks availability of the selected venue. 6. Club leader submits the booking request. 7. System sends the request to venue staff for approval. 8. System displays the confirmation of submission. |
| **Alternative Scenario** | * If the venue is already booked, the system prompts the club leader to choose another time, date or venue. * If the details are incomplete, the system prompts the club leader to enter the details. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.10 Activity diagram – book venue*

### Organize event committee

|  |  |
| --- | --- |
| **Use Case ID** | UC0011 |
| **Use Case Name** | Organize event committee |
| **Description** | Club leader can assign students to event committee roles. |
| **Goal** | To allow club leader to form a committee organization for the event. |
| **Actor** | Club leader |
| **Precondition** | Club leader must be logged into the system and creating an event proposal. |
| **Postcondition** | Committee structure is saved in the system. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Event Planning” section. 3. Club leader accesses the “Organize Committee” page. 4. Club leader adds committee roles and assigns students. 5. Club leader clicks on “Save” button to confirm the committee structure. 6. System stores the committee roles and members. |
| **Alternative Scenario** | * If students already assigned to other roles, an error message is shown. |

|  |
| --- |
| **A diagram of a flowchart  AI-generated content may be incorrect.** |

*Figure 3.1.2.11 Activity diagram – organize event committee*

### View event history

|  |  |
| --- | --- |
| **Use Case ID** | UC0012 |
| **Use Case Name** | View event history |
| **Description** | Club leaders can view past events organized by the club, including event details such as dates, participants, location, and budget. |
| **Goal** | To allow club leaders to review past events for planning and reporting purposes. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system. * Past events must exist in the system database. |
| **Postcondition** | Event history details are displayed. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Event History” section. 3. System retrieves and displays a list of past events. 4. Club leader selects an event to view details. 5. Event details including date, attendees, outcome, and financial summary are shown. |
| **Alternative Scenario** | * If no past events are found, system shows “No event history available.” |

|  |
| --- |
| **A diagram of a company  AI-generated content may be incorrect.** |

*Figure 3.1.2.12 Activity diagram – view event history*

### View financial report

|  |  |
| --- | --- |
| **Use Case ID** | UC0013 |
| **Use Case Name** | View financial report |
| **Description** | Club leader can view financial reports including budget allocations, expenses, income and financial summaries for events and overall club activities. |
| **Goal** | To allows club leaders to monitor financial status, track budget usage, and make informed financial decisions for club management. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system * Club leader must have financial viewing permissions * Financial data must exist in the system database |
| **Postcondition** | Financial report is displayed with current financial status and transaction details. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Financial Reports” section. 3. Club leader selects report type (monthly, quarterly, annual, or event-specific). 4. Club leader specifies date range or event filter. 5. System retrieves financial data from database. 6. System generates and displays financial report. 7. Report shows budget allocations, expenses, income, and balance summary. 8. Club leader reviews the financial information. |
| **Alternative Scenario** | * No financial data:   + If no financial records exist for selected period, system displays “No financial data available for selected period”. * Insufficient permissions:   + If club leader lacks financial viewing privileges, system displays “Access denied - insufficient financial permissions”. |

|  |
| --- |
|  |

*Figure 3.1.2.13 Activity diagram – view financial report*

### Manage forum

|  |  |
| --- | --- |
| **Use Case ID** | UC0014 |
| **Use Case Name** | Manage forum |
| **Description** | Club leaders can moderate and manage the club forum by moderating posts, manage user discussions, and maintaining forum organization. |
| **Goal** | To allow club leaders to effectively oversee forum activities, ensure appropriate content, and facilitate productive discussions among club members. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system. * Club leaders must have moderator privileges for the forum. |
| **Postcondition** | Forum content is moderated, organized, or updated according to the management action performed. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Forum Management” section. 3. System displays forum overview with management options. 4. Club leader selects management action (create topic, moderate posts, manage categories, review reported content). 5. Club leader performs the selected action. 6. Club leader confirms or saves the changes. 7. System processes the management action. 8. System updates forum content accordingly. 9. System displays confirmation of successful action. |
| **Alternative Scenario** | * No forum content:   + If no posts or topics exist, the system displays “No content to manage”. * System error:   + If management action fails due to technical issues, system displays error message and suggests retrying. |

|  |
| --- |
|  |

*Figure 3.1.2.14 Activity diagram – manage forum*

### Manage event

|  |  |
| --- | --- |
| **Use Case ID** | UC0015 |
| **Use Case Name** | Manage event |
| **Description** | Club leaders can view, edit, update or cancel existing event. |
| **Goal** | To allow club leaders modify event information throughout the event lifecycle. |
| **Actor** | Club leader |
| **Precondition** | * User must be logged into the system. * At least one event exists in the system |
| **Postcondition** | Event information is successfully updated. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Event Management” section. 3. System displays list of existing events. 4. Club leader selects an event to manage. 5. System displays event details and management options. 6. Club leader chooses management action (edit details, update committee). 7. Club leader makes necessary changes. 8. Club leader saves the changes. 9. System validates and updates the event information. 10. System displays confirmation of successful update. |
| **Alternative Scenario** | * No events available:   + If no events exist, system displays “No events to manage”. * Validation errors:   + If required fields are missing or invalid during update, system prompts club leader to correct errors. |

|  |
| --- |
|  |

*Figure 3.1.2.15 Activity diagram – manage event*

### Track attendance

|  |  |
| --- | --- |
| **Use Case ID** | UC0016 |
| **Use Case Name** | Track attendance |
| **Description** | Club leader can record and manage attendance of students who have responded will be attending event. |
| **Goal** | To allows club leader track attendance record. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system. * The event must exist. * Students must have RSVPed. |
| **Postcondition** | Attendance data is recorded and saved. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Events” section. 3. Club leader selects the event for which attendance is to be tracked. 4. The system displays a list of students who RSVPed for the event. 5. Club leader marks students as “Present” by ticking checkboxes beside their names. 6. Club leader clicks “Submit”. 7. The system saves the attendance records. |
| **Alternative Scenario** | * Attendance not saved due to system error:   + System error occurs while saving.   + An error message is displayed: “Failed to save attendance. Please try again.”   + Club leader can retry submission. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.16 Activity diagram – track attendance*

### Track budget request status

|  |  |
| --- | --- |
| **Use Case ID** | UC0017 |
| **Use Case Name** | Track budget request status |
| **Description** | Club leader can monitor the status of budget requests submitted for events. |
| **Goal** | To allow club leader to monitor the progress and status of budget requests. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system. * Budget requests must have been submitted previously. |
| **Postcondition** | * Budget request status is displayed and can be reviewed by the club leader. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Budget” section. 3. Club leader views a list of previously submitted budget requests. 4. Club leader can click on each request to view more details or comments from reviewers. 5. The system displays relevant details such as request date, requested amount, purpose and status. |
| **Alternative Scenario** | * System error while loading data   + System error occurs while retrieving data.   + An error message is displayed: “Failed to load budget requests. Please try again.”   + Club leader can retry submission. |

|  |
| --- |
|  |

*Figure 3.1.2.17 Activity diagram – track budget request status*

### Track venue request status

|  |  |
| --- | --- |
| **Use Case ID** | UC0018 |
| **Use Case Name** | Track venue request status |
| **Description** | Club leader can monitor approval status for a venue booking request. |
| **Goal** | To allow the club leader to track the status of the venue request until it is approved or rejected. |
| **Actor** | Club leader |
| **Precondition** | * Club leader must be logged into the system. * The event for which the venue is requested must exist. * Venue options must be available in the system. |
| **Postcondition** | * Venue request status is displayed and can be reviewed by the club leader. * Club leader can view the status of the request. |
| **Main Scenario** | 1. Club leader logs into the system. 2. Club leader navigates to the “Venue Requests” section. 3. Club leader views a list of previously submitted venue requests. 4. Club leader can click on a request to view additional information or comments from the staff. 5. The system displays each request’s details such as event name, requested date and time, venue name, and status. |
| **Alternative Scenario** | * **No Venue Requests Found**   + The system finds no submitted venue requests.   + A message is displayed: “No venue requests found.” |

|  |
| --- |
|  |

*Figure 3.1.2.18 Activity diagram – track venue request status*

### Handle budget request

|  |  |
| --- | --- |
| **Use Case ID** | UC0019 |
| **Use Case Name** | Handle budget request |
| **Description** | Finance staff can view a list of submitted budget requests with details and approve or reject the request. |
| **Goal** | To allow finance staff to view and process budget requests. |
| **Actor** | Finance staff |
| **Precondition** | Finance staff must be logged in. |
| **Postcondition** | Budget request is either approved or rejected with status updated. |
| **Main Scenario** | 1. Finance staff logs into the system. 2. Finance staff navigates to the “Budget Requests” section. 3. System displays all pending requests. 4. Staff clicks “Approve” to confirm request. 5. System updates the request status and notifies club leader. |
| **Alternative Scenario** | * If there are no budget requests, system displays “No request found.” * Reject request:  1. Finance staff clicks “Reject”. 2. System prompts for a mandatory comment. 3. Staff provides a valid reason. 4. System update status to “Rejected” and notifies club leader.  * If reject comment field is empty, system displays an error message. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.19 Activity diagram – handle budget request*

### Handle venue request

|  |  |
| --- | --- |
| **Use Case ID** | UC0020 |
| **Use Case Name** | Handle venue request |
| **Description** | Venue staff can view a list of submitted requests and approve or reject the request. |
| **Goal** | To allow venue staff to view and process requests for venue booking. |
| **Actor** | Venue staff |
| **Precondition** | Venue staff must be logged into the system. |
| **Postcondition** | Venue request is either approved or rejected with status updated. |
| **Main Scenario** | 1. Venue staff logs into the system. 2. Finance staff navigates to the “Venue Requests” section. 3. System displays all pending requests. 4. Staff clicks “Approve” to confirm request. 5. System updates the request status and notifies club leader. |
| **Alternative Scenario** | * If there are no requests, system displays “No request found.” * Reject request:  1. Venue staff clicks “Reject”. 2. System prompts for a mandatory comment. 3. Staff provides a valid reason. 4. System update status to “Rejected” and notifies club leader.  * If reject comment field is empty, system displays an error message. |

|  |
| --- |
| A diagram of a process  AI-generated content may be incorrect. |

*Figure 3.1.2.20 Activity diagram – handle venue request*

### Login

|  |  |
| --- | --- |
| **Use Case ID** | UC0021 |
| **Use Case Name** | Login |
| **Description** | Allows user to access the system by providing valid credentials. |
| **Goal** | To authenticate the user and grant access to system features. |
| **Actor** | All user |
| **Precondition** | * The user is registered in the system. * The user has a valid username/email and password. |
| **Postcondition** | The user is successfully logged in and redirected to the appropriate homepage. |
| **Main Scenario** | 1. The user navigates to the login page. 2. The user enters their username/email and password. 3. The user clicks the "Login" button. 4. The system validates the entered credentials. 5. If credentials are valid, the system starts a user session. 6. The user is redirected to the homepage. |
| **Alternative Scenario** | * Invalid Credentials   + The entered username/email or password is incorrect.   + The system displays an error message: “Invalid username or password.”   + The user retries login with correct credentials. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.21 Activity diagram – login*

### Update profile

|  |  |
| --- | --- |
| **Use Case ID** | UC0022 |
| **Use Case Name** | Update profile |
| **Description** | User updates their personal profile information in the system. |
| **Goal** | To allow the user to modify and save their personal information such as name, contact details, password. |
| **Actor** | All user |
| **Precondition** | * The profile exists in database. * The user must be logged into the system. |
| **Postcondition** | The user’s updated profile information is saved and reflected in the system. |
| **Main Scenario** | 1. The user logs into the system. 2. The user navigates to “Profile” section. 3. The system displays current profile details. 4. The user edits desired fields. 5. The user click “Save” button. 6. The system validates the input and updates the profile in database. |
| **Alternative Scenario** | * Validation Failure   + The input fails validation (e.g., invalid email format)   + The system highlights the errors and prompts the user to correct them.   + The user corrects the information and retries saving. |

|  |
| --- |
| A diagram of a flowchart  AI-generated content may be incorrect. |

*Figure 3.1.2.22 Activity diagram – update profile*

* 1. Performance Requirements

The Student Club Management System shall meet the following performance criteria to ensure efficient and reliable operation:

* The system will respond to user interactions (e.g., form submissions, data retrieval, navigation) within **2 seconds** under normal load conditions.
* The system shall support at least **500 concurrent users** without significant degradation in performance.
* The system shall process and display data for dashboard views and reports within **3 seconds** for 95% of requests.
* The authentication and login process shall complete within **2 seconds** for at least 90% of login attempts.
* The system shall be capable of handling a throughput of **100 budget or event requests per hour** during peak usage.
* The application shall maintain performance levels with up to **10,000 registered users** and scalable database storage for long-term growth.
* Background processes (e.g., report generation, data backup) shall not exceed **5 minutes** per operation and shall not impact real-time user functions.

## Usability Requirements

* Users should be able to complete key tasks, such as submitting event proposals or budget requests, in no more than **three clicks** from the main dashboard.
* First-time users should be able to learn how to use the system and complete basic functions within **15 minutes**, without formal training.
* The interface will follow a consistent design in terms of layout, icons, and labels, so that users can easily understand and predict how to navigate the system.
* The system will meet common accessibility standards (such as **WCAG 2.1 AA**) so that users with visual, motor, or cognitive impairments can also interact with it effectively.
* A search feature will help users quickly find events, clubs, or documents, with at least **90% accuracy** in search results.
* During user testing, at least **80% of users** should rate the system **4 out of 5 or higher** in overall usability.
* Help tips or brief instructions will be built into key parts of the system to guide users as they go, minimizing confusion or the need for external support.

## Interface Requirements

### System Interfaces

* The system will integrate the university system to secure logins for every user, such as students, club leaders, and staff. When a student logs in, their credentials are verified using the LDAP service.
* The system will connect to the campus reservation system to allow venue booking and availability checks. A club leader will have permission to view available rooms and book a venue for the event within the system.

### User Interfaces

* The system will have a responsive interface that can be able to adjust to desktops, tablets, mobile devices, etc. Students using a mobile phone can check the system using an optimized layout for small screens.
* Forms will include clear labels, tooltips, and calendar pickers to assist with data entry. Users can select a date for an event that will interact with a calendar widget instead of manually entering the date.

### Hardware Interfaces

* No special hardware is required for the operation of the system. Users are able to access the platform using standard devices such as laptops or smartphones.
* The system will support standard peripherals such as barcode scanners or USB card readers if integrated for attendance or identity verification at events. During an event, a student will be able to scan their ID card using a USB reader to record attendance.

### Software Interfaces

* The system will chat directly with the university’s finance and venue management systems. Think of it like a polite but efficient messenger, using a common language (called RESTful web services) to ask for and share important information.
* It might also reach out to its digital lockers in the cloud, like Google Drive or Amazon S3. This is where it will tuck away all the files and paperwork related to events, keeping them secure.

### Communications interfaces

* Each time a user’s device communicates with the system, the interaction is secured through encrypted channels. Specifically, the system employs HTTPS with TLS version 1.2 or higher to ensure that all data exchanged remains confidential and protected from unauthorized access.
* When the system interacts with other university systems, it communicates using a structured and widely accepted data format known as JSON. These interactions are governed by standardized RESTful protocols to ensure consistent and reliable data exchange.
* The system is readily accessible from any computer connected to the university’s internal network through a modern web browser, such as Google Chrome, Mozilla Firefox, or Microsoft Edge to provide users with convenient access experience.

## Logical Database Requirements

This section outlines the logical structure of the database required for the system. It describes the types of information processed by various functions, frequency of use, role-based access capabilities, and the relationships and constraints among data entities. The relationships between data entities illustrated using a class diagram in Section 3.5.4.

* + 1. **Types of Information Used by Various Functions**

|  |  |
| --- | --- |
| **Function** | **Information Used** |
| User management | Authentication credentials, profile information (name, email, role) |
| Club management | Club information, membership records, forum posts |
| Event management | Event details, proposals, RSVPs |
| Financial management | Budget requests, financial reports |
| Venue management | Venue bookings, requests, availability |

* + 1. **Frequency of Use**
* **User authentication**
  + Frequency: High
  + Users (students, club leaders, staff) log in daily to access system features such as viewing events, managing forums, or processing requests.
* **Discussion forum**
  + Frequency: High
  + Users (students and club leaders) post frequently to discuss club activities and share announcements. Forum posts are created, edited, or deleted multiple times a day by club leader.
* **Event RSVP**
  + Frequency: Medium
  + RSVP collected before upcoming events, usually peak during event seasons.
* **Notifications**
  + Frequency: Medium
  + Notifications are pushed during activities (event updates, forum posts, request approvals) and triggered by user actions.
* **Budget and venue requests**
  + Frequency: Low
  + Budget requests are submitted only during the early stages of event planning phases, often once per event planned.
* **Financial reports** 
  + Frequency: Low
  + Reports are generated at the end of each semester to summarize club spending.
    1. **Accessing Capabilities**

|  |  |
| --- | --- |
| **Role** | **Accessibility** |
| Student | View clubs, events, forums, join or leave club, RSVP to event, create posts |
| Club leader | Create proposals, submit budget and venue requests, manage events and forums, view financial reports |
| Finance staff | Approve or reject budget requests |
| Venue staff | Approve or reject venue requests |
| All users | Login, update profile information |

* + 1. **Data Entities and Relationships**

|  |  |  |
| --- | --- | --- |
| **Entity** | **Attributes** | **Relationships** |
| User | userID, name, email, password, role | Forum (one-to-many)  RSVP (one-to-many)  Membership (one-to-many) |
| Club | clubID, name, description | Membership (one-to-many)  Event (one-to-many)  Forum (one-to-many)  Proposal (one-to-many)  Report (one-to-one) |
| Membership | memberID, clubID, userID, joinDate | Club (many-to-one)  User (many-to-one) |
| Event | eventID, clubID, title, description, eventType, startDate, endDate | Club (one-to-one)  RSVP (one-to-many) |
| RSVP | rsvpID, eventID, userID, response, timestamp | Event (one-to-one)  User (many-to-one) |
| Forum | forumID, clubID | ForumPost (one-to-many)  User (many-to-one) |
| ForumPost | postID, userID, title, content, timestamp | User (many-to-one) |
| BudgetRequest | budgetRequestID, proposalID, amount, comment, status | Proposal (one-to-one) |
| VenueRequest | venueRequestID, proposalID, venueID, startDate, endDate, comment, status | Proposal (one-to-one)  Venue (many-to-one) |
| Venue | venueID, name, location, capacity, status | VenueRequest (one-to-many) |
| Proposal | proposalID, clubID, title, description, documentPath, submittedAt | Club (many-to-one)  VenueRequest (one-to-one)  BudgetRequest (one-to-one) |
| Report | reportID, clubID, totalBudget, totalSpent, balance | Club (one-to-one) |

|  |
| --- |
| **A diagram of a computer  AI-generated content may be incorrect.** |

*Figure 3.5.4 Class diagram*

* + 1. **Integrity Constraints**
* Primary keys:
  + All entities have unique identifiers (userID, clubID, eventID, venueID, and etc.)
* Foreign keys:
  + clubID in Event, Membership, Report, Proposal, and Forum
  + userID in Membership, ForumPost, RSVP
  + venueID in VenueRequest
  + eventID in RSVP
  + proposalID in BudgetRequest, VenueRequest
* Constraints:
  + Attribute “status” in BudgetRequest and VenueRequest classes must be pending, approved, rejected.
  + Attribute “role” in User class must be student, club leader, finance staff, venue staff.
  + Attribute “eventType” in Event must be workshop, meeting, competition, and etc.
    1. **Security**
* **Passwords using hashing**
  + All user passwords are hashed using secure hashing algorithm such as SHA-512 before stored to prevent unauthorized access if the database is hacked.
* **Role-based access control**
  + Each user is assigned a role, and permissions are defined based on these roles to ensure that users can only perform relevant actions.
* **Restricted access to sensitive data**
  + Data such as budget requests, approval statuses, and financial reports are protected and only visible to finance staff and club leaders. Unauthorized users such as students do not have permissions.
* **Session-based authentication**
  + Secure sessions used after login, with automatic timeouts for inactive users to enhance protection against unauthorized use.
    1. **Data Retention Requirements**
* Student data: Retained for 5 years following graduation before permanent deletion.
* Event data: Retained for 2 years after event completion.
* Forum post: Retained for 2 years after the date created, then archived for read-only.
* Financial report: Retained for 7 years for audit and compliance purposes before deletion.

## Design Constraints

* The system is required to follow the university’s branding and accessibility guidelines, such as including logo placement, color schemes, and text contrast for visibility
* Every single communication between client and server is required to use HTTPS to ensure secure data transmission
* The system is required to support responsiveness for any device, such as mobile, tablet, desktop, etc.
* Financial data are required to comply with university financial audit requirements and follow standard accounting practices

* 1. Software System Attributes
* **Reliability**: The system should ensure smooth and ongoing operation and should be able to recover from any unexpected failures within a minute.
* **Availability**: The system should maintain 99.9% uptime during university working hours, such as Monday to Friday 9am to 7pm, to ensure every student and staff be able to use the system.
* **Security**: The system must implement access control, enforce strong password policies, and use encryption to protect any sensitive information.
* **Maintainability**: The system should follow best coding practices and be modular to facilitate updates.
* **Portability**: The software should be able to run on both Linux and Windows servers without additional configuration.
* **Usability:** The interface should be user-friendly and able to assist users to complete tasks in a fewer click. An easy or a clean layout and responsive design should enhance users’ navigation.
* **Scalability:** This system is built to handle a large number of users, clubs, and events without any drop in performance. This should support future proof of the system.

* 1. **Supporting Information**

This section provides additional information to support understanding and implementation of the software requirements.

1. **Results of Questionnaire**

The questionnaire results show support for the club management system requirements. A total of 92% of respondents found the ability to join or leave clubs useful, which means it is a must-have feature. Similarly, 96% supported a notification system to improve engagement. Discussion forums received 96% positive responses, which classify them as satisfiers for enhancing club communication. Attendee management and committee role assignment features also gained 96% and 92% approval respectively. Financial report was considered useful by 86%, this shows its relevance for transparency. Overall, 96% of respondents expressed likelihood to use the system, which demonstrates high potential adoption.

1. **Background Information**

The management of student clubs currently primary done by manual procedures. Club leaders frequently use paper forms or email to submit requests, which causes delays, document loss, and inefficient approval processes. Communication between club members across various platforms such as group chats, social media, it might be challenging to monitor discussions or decisions. The lack of a centralized framework for students interested in joining clubs to view available clubs or activities reduces participation and engagement. The Student Club Management System address these operational challenges by providing a centralized platform for club management.

1. **Description of Problems to be Solved**

Clubs frequently experience delays due to the time-consuming manual request processing. Students find it difficult to explore and join clubs due to the absence of a centralized platform. Communication within the club was disconnected without a specific discussion forum to share updates and ideas. Event planning is unstructured, which leads to low participation in event. Furthermore, financial records lack of visible to stakeholders, and there is no method to review club spending. The proposed system aims to solve these issues by centralizing requirements to streamline workflows and enhance user experience for all stakeholders involved.

1. **Special Packaging Instructions**

The system will be deployed by the university IT team on secure campus servers. There is no export restrictions apply as the system only for internal university use only. To guarantee integrity and security compliance, only authorized IT staff will be able to access the deployment environment during initial setup.

***Note:***The supporting context in this section is to assist in understanding the requirements and is part of the software requirements.

1. Verification

## Verification Approach

The verification approach for the student club management system will ensure that all functional, performance, usability, interface and database requirements are validated. Below is the structured approach:

* **Verification Methods:**
  + **Functional testing**: Validate vital workflows such as club membership management, budget approvals, venue booking etc against Section 3.2 – 3.4 requirements.
  + **Unit testing**: Test individual component to ensure code-level accuracy.
  + **System integration testing**: Verify seamless interaction between subsystems (e.g., integration with university financial system)
  + **Database testing**: Ensure database structure supports scalability and maintains data integrity.
  + **Security testing**: Validate role-based access controls (Section 3.5.3) and align with university policies.
* **Responsible Parties:**
  + Development Team
  + QA Team
  + Stakeholders
    - Club leaders/Students
    - University Staff
* **Verification Schedule:**
  + **Sprint-level:** Unit and functional testing after each 2 weeks agile sprint.
  + **Phase Gates:** Integration testing after major milestones.
  + **Pre-Deployment:** Smoke testing in production post-deployment.
  + **Ongoing:** Regression testing for updates.
* **Verification Environments:**
  + **QA Testing Environment:** Unit, functional, and integration testing. Dummy data will be used to simulate production scenarios.
  + **Staging Environment:** Supports user acceptance testing and performance testing.
  + **Production Environment:** Final checks for whole production.

## Verification Criteria

Verification criteria for the Student Club Management System are defined to ensure alignment with the functional, performance, usability, interface, and database requirements. Below are the measurable benchmarks for validating system compliance:

* + 1. **Performance Criteria**
* Response Time:
  + User interactions (e.g., form submissions, data retrieval) shall complete within 2 seconds under normal load (Section 3.2).
* Concurrency:
  + The system shall support 500 concurrent users without performance degradation (e.g., lag, timeouts).
* Authentication Speed:
  + Login processes shall complete within 2 seconds for 90% of attempts.
* Scalability:
  + Performance shall remain stable with up to 10,000 registered users and scalable database storage.
* Portability:
  + The software should be able to run on both Linux and Windows servers without additional configuration.
    1. **Usability Criteria**
* Task Efficiency:
  + 80% of users shall complete key tasks (e.g., event proposals, budget requests) in ≤3 clicks from the dashboard (Section 3.3).
* Learnability:
  + First-time users shall perform basic functions within 15 minutes without training.
* Accessibility:
  + The interface shall meet WCAG 2.1 AA standards (e.g., screen reader compatibility, keyboard navigation).
* Search Accuracy:
  + Search results for events, clubs, or documents shall achieve ≥90% relevance.
* User Satisfaction:
  + At least 80% of users shall rate the system 4/5 or higher in post-testing usability surveys.
    1. **Interface Criteria**
* System Integration:
  + Secure login via university authentication systems and real-time venue availability checks shall function without errors (Section 3.4.1).
* Responsive Design:
  + The interface shall adapt seamlessly to desktops, tablets, and mobile devices.
* Hardware Compatibility:
  + Peripheral devices (e.g., barcode scanners) shall integrate without performance issues.
* Security Compliance:
  + All communications shall use HTTPS/TLS 1.2+ encryption (Section 3.4.5).
    1. **Database Criteria**
* Data Integrity:
  + Logical database structures (Section 3.5.1) shall maintain consistency and accuracy during high-frequency operations (e.g., user logins, forum updates).
* Role-Based Access:
  + Access controls (Section 3.5.3) shall restrict users to permitted actions (e.g., students cannot approve budget requests).
* Scalability:
  + The database shall support 10,000+ users and long-term data growth without latency.

1. Appendices

## Assumptions and Dependencies

This section outlines the assumptions and dependencies relevant to the software system. Assumptions are conditions expected to hold true during development and operation. Dependencies are external systems or services that the system relies on. Changes to any of these may require a revision of the system requirements.

### 5.1.1 Assumptions

* Internet Access:
  + Users are expected to have stable internet when using the system. If internet connection is unreliable, the system would require capability or caching.
* Accurate User Roles:
  + Roles such as students, club leader, finance and venue staff are expected to assign correctly. Incorrect roles may cause access and permissions issues.
* Browser Compatibility:
  + Users are expected to use browsers like Google Chrome, Firefox, Microsoft Edge. Older or unsupported browsers may cause system to function incorrectly.

### 5.1.2 Dependencies

* University Student Database:
  + The system will use the university’s student database for login and user data. Database downtime or changes may break login and profile features.
* Campus Server:
  + The system will be hosted on university servers. If server environment changes, it may affect performance and security.
* External System Integration:
  + The system depends on two external APIs: the Financial Management System API for submitting and tracking budget requests, and the Space Reservation Database API for handling venue bookings. Any API changes or downtimes may disrupt request processing.

## Acronyms and Abbreviations

This section provides definitions of the acronyms and abbreviations used in this document.

**RSVP:** Répond S’il Vous Plaît (Please respond)

**SHA-512:** Secure Hash Algorithm 512-bit

**ID:** Identifier

**API:** Application Programming Interface

**IT:** Information Technology

**HTTPS:** Hypertext Transfer Protocol Secure

**USB:** Universal Serial Bus

**TLS:** Transport Layer Security

**JSON:** JavaScript Object Notation

**SSO:** Single Sign On

**WCAG:** Web Content Accessibility Guidelines

**LDAP:** Lightweight Directory Access Protocol