**Software Requirements Specification (SRS)**

CERIA Check-in & Payment System

Version 1.0

Prepared by:

MUHAMAD AIMAN BIN MOHD EHWAL (1211103064 )

MUHAMMAD AQIL BIN RAHMAT (1211107976 )

AHMAD FAIZ BIN ANUAR (1211109154 )

WAN MUHAMMAD ILHAN BIN WAN ZIL AZHAR (1211102908 )

Date created: 24 May 2025

Table of Content

[**1. INTRODUCTION 2**](#_79udo36v5y2e)

[1.1 Purpose 2](#_36um889sjpu)

[1.2 Scope 2](#_2eehptluzjdq)

[1.3 Product Overview 2](#_g5nzkz5bqysm)

[1.3.1 Product Perspective 3](#_1f49m6fhoqi9)

[1.3.2 Product Functions 4](#_pcbnfo652tqp)

[1.3.3 User Characteristics 5](#_lyea73nbgy2c)

[1.3.4 Limitations 5](#_mijopvexsmf)

[1.4 Definitions 6](#_an5ju9jg41fv)

[**2. REFERENCES 7**](#_numpd0s98hdw)

[**3. REQUIREMENTS 8**](#_8ju1595p6k5b)

[3.1 Functions 8](#_6ozg02yb6fiz)

[3.1.1 Functional Requirements 8](#_75ar00bcpty0)

[3.1.2 User Functional Requirements 9](#_iksfvpm0qsf)

[3.1.2.1 Login and Authentication 9](#_ljecqleg42xz)

[3.1.2.2 Student Event Registration 12](#_pz12nrmflbq)

[3.1.2.3 Student Event Check-In 15](#_lwxy4opfry9h)

[3.1.2.4 Student Payment System 18](#_iwufiia78x51)

[3.1.2.5 Admin Event Management 21](#_x82lns1iwohn)

[3.1.2.6 Admin Attendance Management 24](#_qbatnnybrve1)

[3.1.2.7 Vendor Payment Management 27](#_feotk9mq0z71)

[3.1.2.8 Vendor Sales Report 30](#_pl4gj820imh8)

[3.2 Performance Requirements 33](#_pcv78q8yta3c)

[3.3 Usability Requirements 33](#_cxpxv4cagbo4)

[3.4 Interface Requirements 34](#_nayzsplftkc1)

[3.4.1 System Interfaces 34](#_ne782shyqtr6)

[3.4.2 User Interfaces 35](#_jrlg37qnvats)

[3.4.3 Hardware Interfaces 45](#_x9k41jqkujf)

[3.4.4 Software Interfaces 45](#_dy8tl1voupin)

[3.4.5 Communication Interfaces 46](#_uiqgk1y7dt3b)

[3.5 Logical Database Requirements 47](#_7694wli938g0)

[3.6 Design Constraints 47](#_htht7vjpwwd5)

[3.7 Software System Attributes 48](#_drmdrwq7tg)

[3.8 Supporting Information 48](#_k46qf71ynuo3)

[**4. VERIFICATION 49**](#_fnpv4il5wh1w)

[4.1 Verification Approach 49](#_isjbe2167zz)

[4.2 Verification Criteria 50](#_rlhzgw46zu4t)

[**5. APPENDICES 51**](#_wk6m1pnj68yh)

[5.1 Assumptions and Dependencies 51](#_pfpof9kqj37s)

[5.2 Acronyms and Abbreviation 51](#_ocmv86f7w8ga)

# **INTRODUCTION**

## 1.1 Purpose

The purpose of this project is to develop a digital check-in system for campus events that integrates with the university’s student identification database and payment processing system. This system aims to streamline event attendance tracking by automating check-ins through student ID verification and ticket validation, reducing manual effort and errors. Additionally, it facilitates secure on-site purchases such as food, merchandise, or services, and provides organizers with real-time data and analytics for better event management and reporting. The platform ultimately enhances the efficiency, security, and convenience of campus event operations for both students and staff.

## 1.2 Scope

The system would allow participants and vendors to check in for an event via QR for a seamless, user-friendly experience. Features include real-time notifications, on-the-spot payment integration, and a rating system for the event. It will not include mapping integration, third-party map systems, or outdoor navigation.

## 1.3 Product Overview

The digital check-in system designed to manage campus event attendance and transactions efficiently. It allows students to check in to events using their university IDs and supports both digital and physical ticket verification. The system integrates with the university’s student database to ensure accurate identification and with payment gateways to enable secure on-site purchases. Admin can use the system to monitor attendance in real-time, manage ticketing, and access reports. The platform is intended for use by students, vendors and admin.

### 

### 1.3.1 Product Perspective

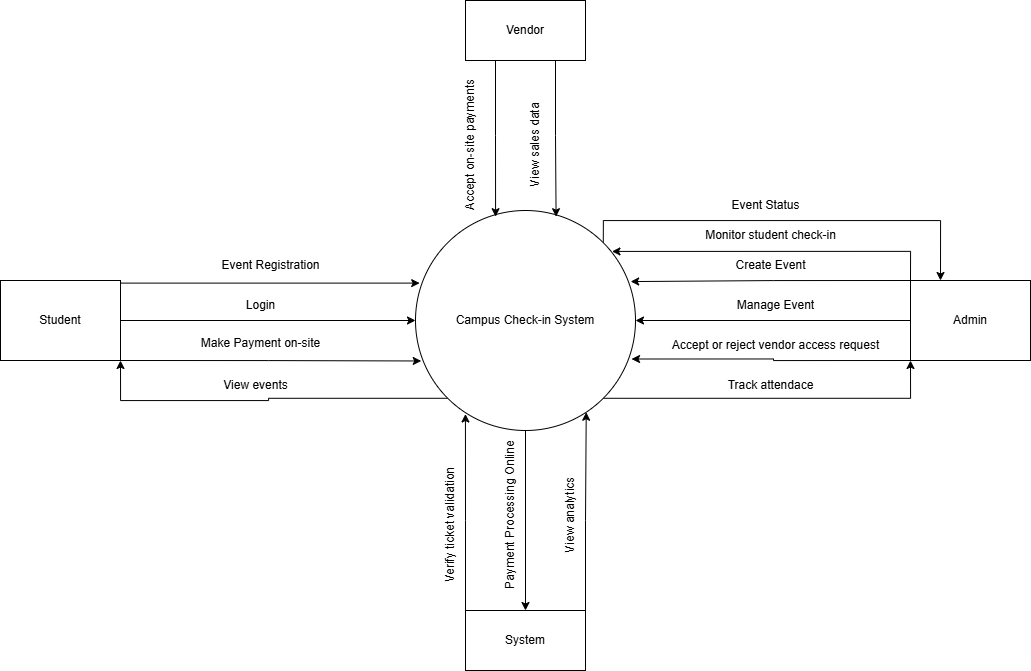


Figure 1. shows the context of Campus Check-In System

### 1.3.2 Product Functions

| Functions | Description |
| --- | --- |
| Student Check-in | Allow students to check in to events using their university ID cards or student numbers. |
| Ticket Verification | Validate tickets during entry to ensure only authorized attendees are allowed. |
| Payment Processing | Enable secure on-site purchases such as event merchandise, food, or add-on services through integrated payment gateways. |
| Real-time Attendance Tracking | Monitor and log attendance as students check in, providing up-to-date data to the admin. |
| Student Database Integration | Connect with the university's student identification database for real-time verification of student status and information. |
| Analytic and Reports | Generate post-event reports including attendance summaries, purchase history, and ticket scan logs for students and vendors |

Table 1 Product Functions

### 

### 

### 

### 

### 1.3.3 User Characteristics

| Role | Description | Required Knowledge |
| --- | --- | --- |
| Student | A university students who use the system to check-in and make purchases | Able to navigate a web or mobile interface, |
| Admin | Admin who manage track attendance, generate reports, and oversee the system. | Familiar with event management workflows, admin dashboards, and data/report handling. |
| Vendor | On-site service providers (e.g., food stalls, merchandise sellers) who use the system for processing transactions and viewing sales data. | Basic familiarity with digital payment systems and able to operate an interface. |

Table 2 User Characteristics

### 1.3.4 Limitations

* **Dependence on Internet Connectivity:** The system requires stable internet connectivity access for real-time check-in, payment processing and database integration.
* **Scalability Limits:** The system may have performance issues if it’s a large-scale event without a proper server.
* **No Offline Support:** The initial version may not support offline check-in or any payment result, which can cause disruption in areas with poor connectivity.

* **User Training Requirement:** Users, especially admin and vendor, may require brief training to use the system correctly and to avoid any issues or errors.
* **Device Constraints:** Check-in admin and vendor must use compatible devices such as tablet or laptop as it can lack proper hardware and can hinder the functionality.

## 

## 1.4 Definitions

| Term | Definition |
| --- | --- |
| Integration | Connecting the system with other platform (e.g database, payment) |
| Analytics | The analysis of data to gain insight, such as reports, |
| On-site purchases | Buying foods, merchandise, accessories etc. at the event of the location. |
| Payment Gateway | A service that processes credit card,debit card or digital payment securely. |
| Check-in System | A digital platform used to verify students or admins at events. |
| Ticket Verification | The process of checking whether an event ticket is valid or invalid. |
| End users | The person who will be using the software or the system |
| Backend | The server-side components and operations of an application, including the logic, data storage, and infrastructure that make the application work behind the scenes |

Table 3 Definitions

# **REFERENCES**

**References**

*The 12 Event Check-In Apps Every Planner Should Know About in 2025*. (2025, January 24). InEvent. Retrieved May 25, 2025, from https://inevent.com/blog/events/event-check-in-app.html

*Elastic Stack: (ELK) Elasticsearch, Kibana & Logstash*. (n.d.). Elastic. Retrieved May 25, 2025, from https://www.elastic.co/elastic-stack

*How to Write an SRS Document (Software Requirements Specification Document)*. (2023, January 17). Perforce. Retrieved May 25, 2025, from https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document

*ISO/IEC/IEEE 29148 Requirements Specification Templates*. (n.d.). ReqView. Retrieved May 25, 2025, from https://www.reqview.com/doc/iso-iec-ieee-29148-templates/

Jain, S. (2024, June 19). *Classification of Software Requirements – Software Engineering*. GeeksforGeeks. Retrieved May 25, 2025, from https://www.geeksforgeeks.org/software-engineering-classification-of-software-requirements/

Jain, S. (2025, March 17). *Flutter Tutorial*. GeeksforGeeks. Retrieved May 25, 2025, from https://www.geeksforgeeks.org/flutter-tutorial/

Lemburg, M. (n.d.). *sqlite3 — DB-API 2.0 interface for SQLite databases*. Python documentation. Retrieved May 25, 2025, from https://docs.python.org/3/library/sqlite3.html

# **REQUIREMENTS**

## 3.1 Functions

### 3.1.1 Functional Requirements

This section details the functional requirement of our Campus Event Check-In System starting with the overall requirements followed by requirements of each feature of the system. Figure 1.0 shows the overall use case of Campus Event Check-In System.

## 

Figure 2 Ceria Check In and Payment System Overall Use Case

### 3.1.2 User Functional Requirements

#### 3.1.2.1 Login and Authentication

The functional requirement for login and authentication are as followed:

| Requirement ID | FR001 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow users to log in using their unique credentials | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 4.

## 

| Requirement ID | FR002 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall validate login credentials to ensure only validated users are allowed in | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 5.

## 

| Requirement ID | FR003 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall display an error message if invalid credentials are entered | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 6.

## 

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC001 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F001 Login | | |
| Purpose | To enable user to enter | | |
| Actor | Student / Admin / Vendor | | |
| Trigger | User opening the app and entering login credentials | | |
| Precondition | User must have valid account credentials | | |
| Scenario Name | Action | | |
| Main Flow | 1. User opens app 2. System displays login form 3. User enter credentials 4. System validates input field (non-empty, correct format) 5. System verifies credentials 6. If valid, user redirects to appropriate interface | | |
| Alternate Flow - Invalid Credentials | 5.1 System shows: “Invalid Username or Password” | | |
| Alternate Flow -  Empty Fields | 4.1 System shows: “Please fill in all required fields” | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 7 Use Case Specification - Login & Authentication

## 

Figure 3 Sequence Diagram - User Login

#### 

#### 3.1.2.2 Student Event Registration

The functional requirement for student event registration are as followed:

| Requirement ID | FR004 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow students to book tickets for events if tickets are available. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 8.

| Requirement ID | FR005 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall process payments for event registrations through the payment system. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 9.

| Requirement ID | FR006 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall update the student database with ticket information after successful payment. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 10.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC002 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F002 Event Registration | | |
| Purpose | To allow students to register for events by booking tickets, making payments, and updating ticket records. | | |
| Actor | Student | | |
| Trigger | A student initiates the event registration process by attempting to book a ticket. | | |
| Precondition | A student must be logged in to the system, and the event ticket must be available. | | |
| Scenario Name | Action | | |
| Main Flow | 1. Student selects an event and books a ticket. 2. System checks ticket availability. 3. If tickets are available, the system requests payment. 4. Student completes payment. 5. Payment system verified payment. 6. If successful, tickets are updated in the student database. 7. Student receive confirmation. | | |
| Alternate Flow  (Tickets Unavailable) | 2.1 System shows an error message: “Ticket not available.”. | | |
| Alternate Flow  (Payment Failed) | 4.1 System shows an error message: "Payment not completed.”. | | |
| Rules | * Tickets are booked on a first-come-first-serve basis. * Payment must be verified before ticket confirmation. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 11 Use Case Specification - Student Ticket Registration

#### 

Figure 4 Sequence Diagram - Student Event Registration

#### 

#### 3.1.2.3 Student Event Check-In

The functional requirement for student event check in are as followed:

| Requirement ID | FR007 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow students to check in for registered events | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 12.

| Requirement ID | FR008 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall verify the student’s identity and event registration status before granting access | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 13.

| Requirement ID | FR009 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall mark the student as checked in and update the attendance record | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 14.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC003 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F003 Check In For Event | | |
| Purpose | Allow student to check in to an event using a QR Code | | |
| Actor | Student | | |
| Trigger | Students scans QR Code to enter the event venue | | |
| Precondition | Student must be registered and have a valid ticket | | |
| Scenario Name | Action | | |
| Main Flow | 1. Student arrive at event location 2. Student scans QR code to check in 3. System verifies registration and payment status 4. If verified, mark attendance and log time | | |
| Alternate Flow  (No Registration) | 3.1 System shows: “You are not registered for this event” | | |
| Rules | Only registered students can check in | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 15 Use Case Specification - Student Event Check-In

#### 

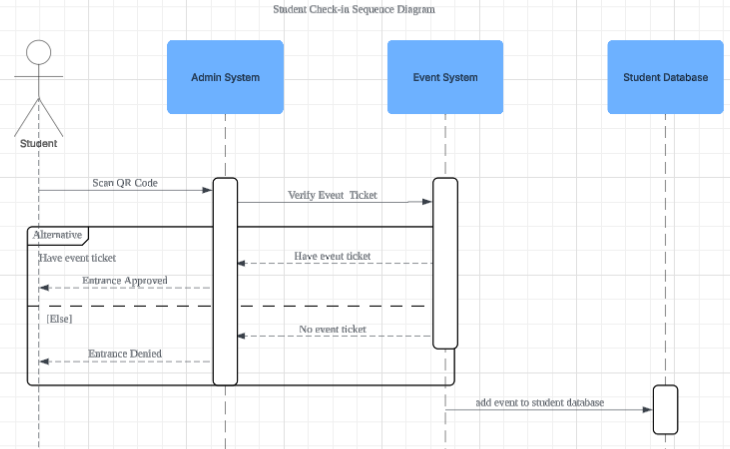


Figure 5 Sequence Diagram - Student Event Check-In

#### 

#### 3.1.2.4 Student Payment System

The functional requirements for student payment system are as follows:

| Requirement ID | FR010 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall verify the payment made by the student using the payment gateway. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 16.

| Requirement ID | FR011 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall update the payment database when the payment is successfully verified. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 17.

| Requirement ID | FR012 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall notify the student whether the payment was successful or unsuccessful. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 18.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC004 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F004 Process Student Payment | | |
| Purpose | Allow students to make payment and verify their payment status. | | |
| Actor | Student | | |
| Trigger | Students initiate payment using the payment system. | | |
| Precondition | Students must have valid payment details and access to the system. | | |
| Scenario Name | Action | | |
| Main Flow | 1. Students enter payment details. 2. Payment Gateway processes the payment. 3. System verifies payment status. 4. If successful, the payment database is updated. | | |
| Alternate Flow  (Failed Payment) | 3.1 System shows: “Payment failed. Please try again.” | | |
| Rules | Only registered students can make payments. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 19 Use Case Specification - Student Payment System

#### 

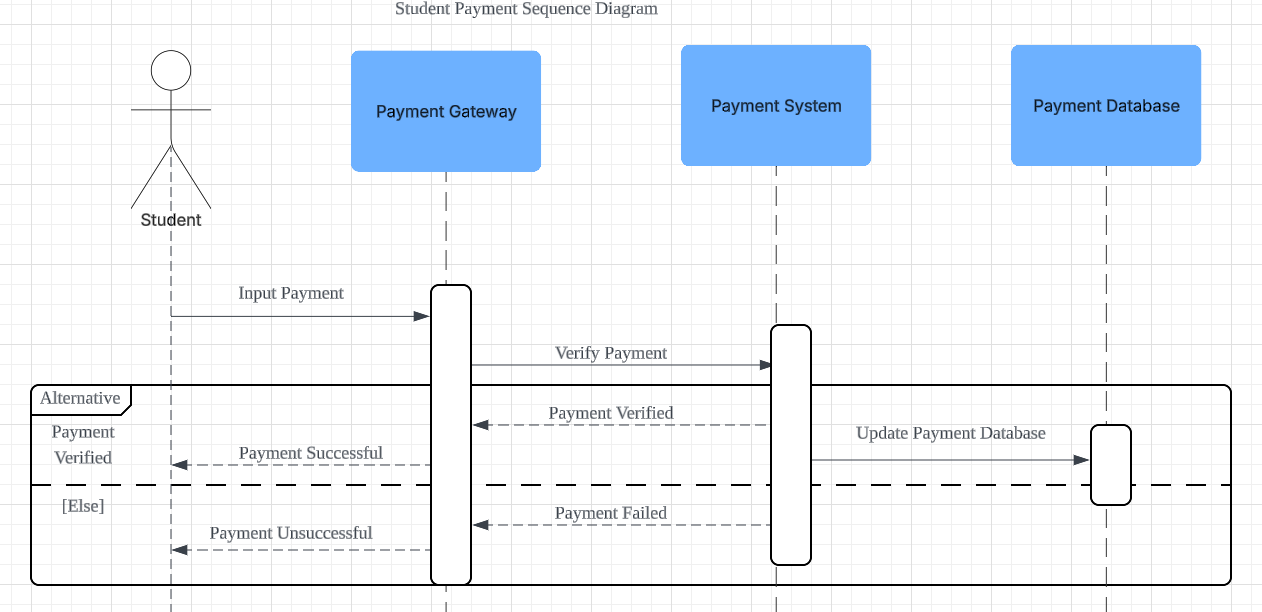


Figure 6 Sequence Diagram - Student Payment System

#### 

#### 3.1.2.5 Admin Event Management

The functional requirement for admin event management are as followed:

| Requirement ID | FR013 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow admins to create new events with details such as name, date, time, location and capacity | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 20.

| Requirement ID | FR014 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall enable admins to edit existing events | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 21.

| Requirement ID | FR015 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow admins to delete events that are no longer needed | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 22.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC005 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F005 Event Management System for Admin | | |
| Purpose | Allow admins to manage events | | |
| Actor | Admin | | |
| Trigger | Admin go to the event tab | | |
| Precondition | Admin must be logged in to manage the events | | |
| Scenario Name | Action | | |
| Main Flow | 1. Admin navigates to event management space 2. System display list of events with options to create new, delete or edit 3. System validates any changes made by the admins 4. System save the event to the database | | |
| Alternate Flow  (Invalid Data) | I. admin click the cancel button  Ii. back to main flow step 1 | | |
| Rules | All field must not be left blank before saving | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 23 Use Case Specification - Admin Event Management

#### 

#### 

Figure 7 Sequence Diagram - Admin Event Management

#### 

#### 

#### 3.1.2.6 Admin Attendance Management

The functional requirement for admin attendance system are as followed:

| Requirement ID | FR016 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow admin to generate a qr code for students to scan | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 24.

| Requirement ID | FR017 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow admin to associate the qr code with the event the admin is handling | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 25.

| Requirement ID | FR018 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow admins to view the list of students who attended the event | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 26.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC006 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F006 Attendance System | | |
| Purpose | Allow admins to take student attendance during events | | |
| Actor | Admin | | |
| Trigger | Admin go to the attendance page | | |
| Precondition | Admin must be logged in | | |
| Scenario Name | Action | | |
| Main Flow | 1. Admin navigate to the event attendance page 2. Admin click ‘generate qr code’ 3. Admin show it to event attendee | | |
| Alternate Flow  (Show List of Attendee) | I. go back to main flow 1  Ii. admin click ‘view attendance list’ | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 27 Use Case Specification - Admin Attendance System

#### 

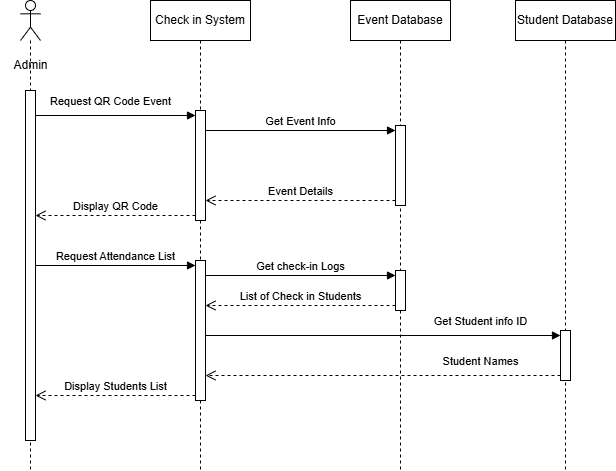


Figure 8 Sequence Diagram - Admin Attendance System

#### 

#### 3.1.2.7 Vendor Payment Management

The functional requirement for vendor payment management are as followed:

| Requirement ID | FR019 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow vendors to view all payments made for events they are associated with | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 28.

| Requirement ID | FR020 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall enable vendors to mark payments as received or pending | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 29.

| Requirement ID | FR021 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall update the seller dashboard accordingly after each purchase | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 30.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC007 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F007 Vendor Payment System | | |
| Purpose | Allow vendors to keep track of sales made during the event | | |
| Actor | Vendor | | |
| Trigger | Vendor enter In-app POS System | | |
| Precondition | Vendor must be registered to the event | | |
| Scenario Name | Action | | |
| Main Flow | 1. Vendor click ‘’ on the main dashboard 2. Vendor input the total bill 3. Customer pay the bill 4. Vendor click ‘New Sale’ button 5. Payment will be reflected on the main dashboard | | |
| Alternate Flow  (Empty Field) | 4.1 System will display: “Please enter amount” | | |
| Author | Muhammad Aqil Bin Rahmat | | |

Table 31 Use Case Specification - Vendor Payment Management

#### 

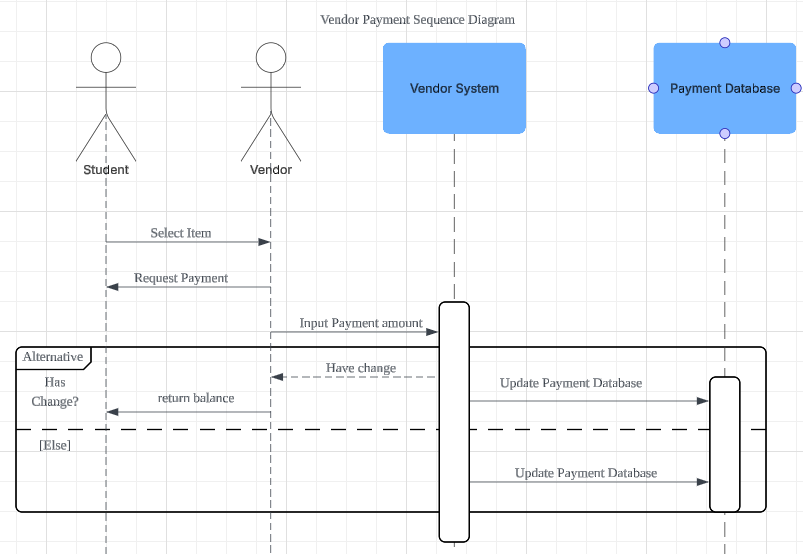


Figure 9 Sequence Diagram - Vendor Payment Management

#### 

#### 

#### 

#### 3.1.2.8 Vendor Sales Report

| Requirement ID | FR022 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall allow vendors to generate a sales report. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 32.

| Requirement ID | FR023 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall retrieve sales data from the payment database when requested by the vendor system. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 33.

| Requirement ID | FR024 | Version | 1.0 |
| --- | --- | --- | --- |
| Description | The system shall display the generated sales report to the vendor. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 34.

The following table shows the detail of this feature followed by a sequence diagram below it to illustrate.

| Use Case ID | UC008 | Version | 1.0 |
| --- | --- | --- | --- |
| Feature | F00 View Sales Report | | |
| Purpose | Allow vendors to view and analyze their sales performance. | | |
| Actor | Vendor | | |
| Trigger | Vendor requests to generate a sales report. | | |
| Precondition | Vendor must have access to the system and valid credentials. | | |
| Scenario Name | Action | | |
| Main Flow | 1. Vendor initiates the request to generate a sales report.  2. Vendor system requests sales data from the payment database.  3. Payment database returns the sales data.  4. Vendor system generates and displays the report | | |
| Alternate Flow  (No Data) | 3.1 System shows: “No sales data available for the specified period.” | | |
| Rules | Only vendors can generate sales reports. | | |
| Author | Wan Muhammad Ilhan Bin Wan Zil Azhar | | |

Table 35 Use Case Specification - Vendor Sales Report

#### 

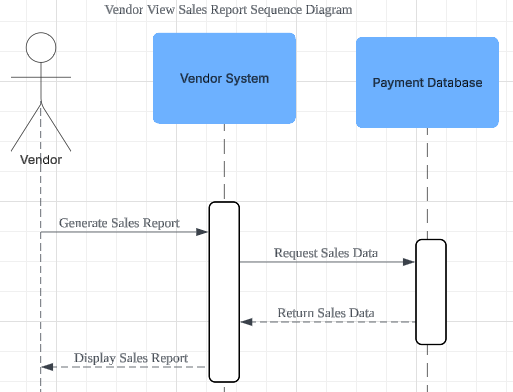


Figure 10 Sequence Diagram - Vendor Sales Report

## 

## 3.2 Performance Requirements

* The system must support up to **300 concurrent users** without performance degradation.
* Check-in validation and ticket verification must be completed in **under 5 seconds**.
* On-site purchase transactions must be processed correctly and smoothly.
* Event registration confirmation, including ticket issuance, should occur within **10 seconds** of submission.
* The system must scale to **3,000 total users** without functional errors or data loss.
* All admin-made changes (e.g., event status updates) must synchronize across all user devices.

## 3.3 Usability Requirements

* The system must offer a clean, intuitive interface accessible via web and mobile platforms.
* All users (students, vendors, admins) must be able to navigate the system smoothly without any errors to complete their primary tasks or needs (e.g., check-in, sales, or reporting).
* The platform must include readable fonts, logical page layouts, and keyboard navigability to meet accessibility standards.
* Error messages must be clear and informative, allowing users to recover from issues without technical support.

## 

## 3.4 Interface Requirements

### 3.4.1 System Interfaces

| **System** | **Description** |
| --- | --- |
| Student Database integration | The system must be interlinked with the university’s secure student identification database via an authenticated API or secure database connection to perform real-time identity validation. |
| Payment Gateway Integration | The system must integrate with a PCI-compliant digital payment service to allow vendors to process transactions (credit, debit, and digital wallets). |
| Event Management Backend | Admin interfaces should connect to the centralized backend to fetch, update, and sync event data, tickets, and attendance records. |
| Logging & Analytics Module | All check-ins, transactions, and administrative actions must be logged and made available |

Table 36 System Interface

### 

### 3.4.2 User Interfaces



Figure 11 UI - Student Home Page

**Home Page (Students)**

The home page will show a “Scan QR” button that lets students check in to an event by scanning a QR code. At the bottom, there are three buttons:

* The **Payment** button on the left takes students to the payment page where they can buy items from event vendors.
* The **Tickets** button on the right allows students to buy event tickets.
* In the middle is the main **QR** button that redirects to the home page.

At the top right, there’s also a **Contact** button that lets students quickly call event authorities in case of an emergency.

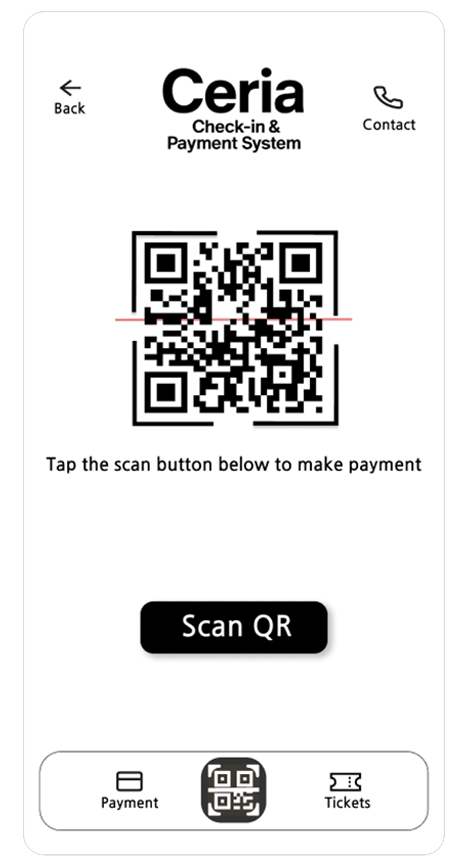


Figure 12 UI - Student Payment Page

**Payment Page (Students)**

This page lets students make payments to event vendors. There is a “Scan QR” button in the center, which students can use to scan a vendor’s QR code to complete a payment.

At the top, there is a **Back** button to return to the previous page, and a contact button to quickly call event staff for help.

At the bottom, the navigation bar includes:

* **Payment** (current page)
* **Home page** in the middle.
* **Tickets** on the right for buying event tickets.

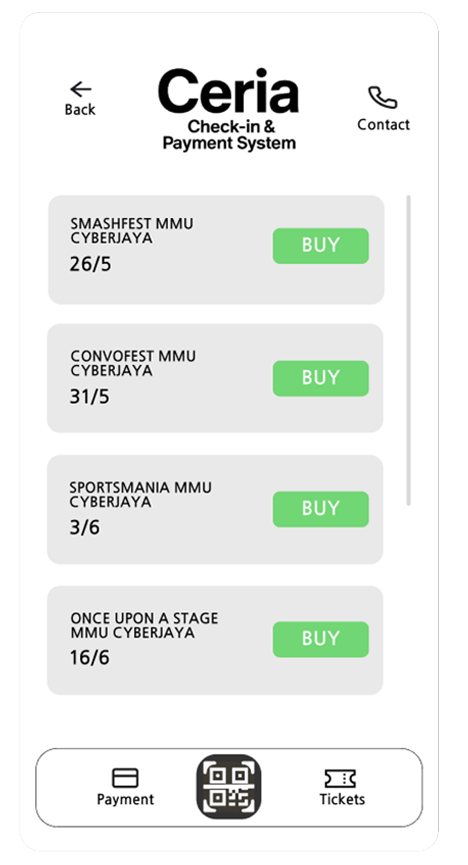


Figure 13 UI - Student Event Ticket Page

**Tickets Page (Students)**

This page shows a list of upcoming events that students can attend. Each event is displayed with its name, location (MMU Cyberjaya), and date. Students can tap the green “Buy” button next to any event to purchase a ticket.

At the top, there is a **Back** button to return to the previous page, and a **Contact** button for emergency calls to event staff.

At the bottom, the navigation bar includes,

* **Payment** to go to the payment page.
* **Home page** in the middle.
* **Tickets** (current page).

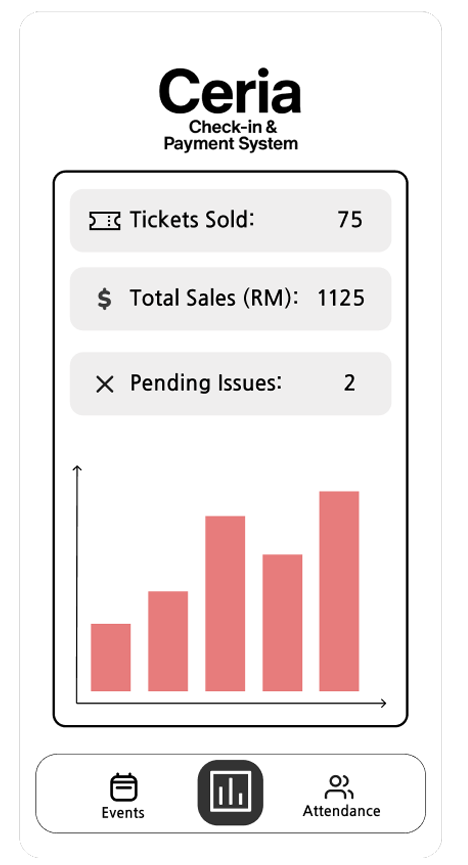


Figure 14 UI - Admin Home Page

**Home Page (Admin)**

This page shows a dashboard with multiple data for the admin. The dashboard shows the amount of tickets sold, total sales (RM), and pending issues. There’s also a bar chart with the X-axis as the time, and the Y-axis as tickets sold. Clicking on each bar would show the amount of tickets sold and the hour. (e.g. 11am).

At the bottom, the navigation bar includes:

* **Events** button to go to the events page.
* **Home page** which redirects here.
* **Attendance** button which shows a list of attendees.

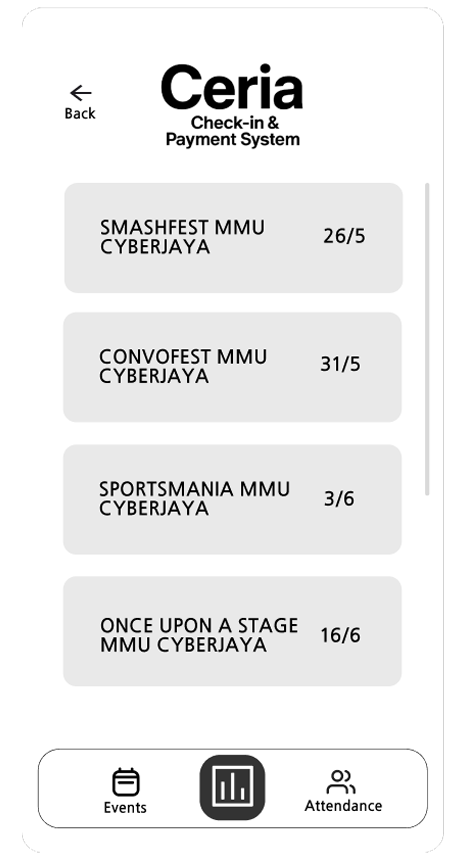


Figure 15 Admin Events Page

**Events Page (Admin)**

On the events page, admin can view all the upcoming events that use the Ceria Check-in & Payment System. It shows the name of the event, its location, and the date it will be held.

At the top, there is a **Back** button to return to the previous page.

At the bottom, the navigation bar includes:

* **Events** page button (current page).
* **Home** page button.
* **Attendance** page button.

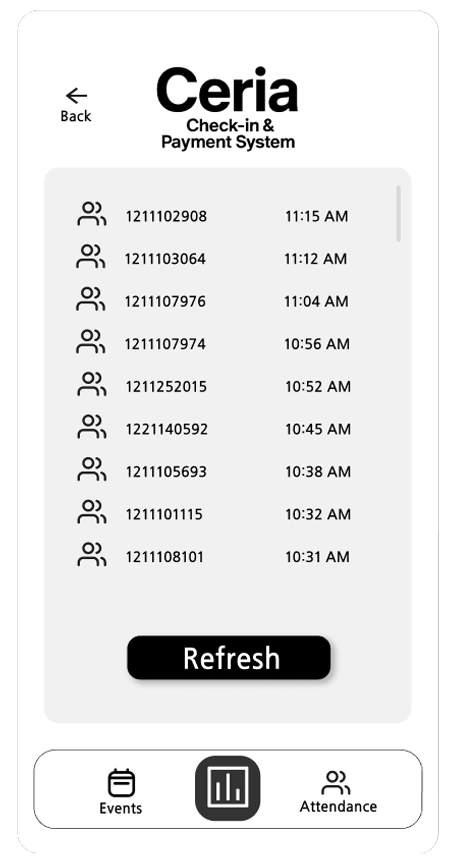


Figure 16 Admin Attendance Page

**Attendance Page (Admins)**

The attendance page shows the list of students that checked-in starting from the most recent time. It shows a list of student ID’s, and the time they checked-in. There’s also a “refresh” button to refresh the list and view the most recent attendees.

At the top, there is a **Back** button to return to the previous page.

At the bottom, the navigation bar includes:

* **Events** page button.
* **Home** page button.
* **Attendance** page button (current page).

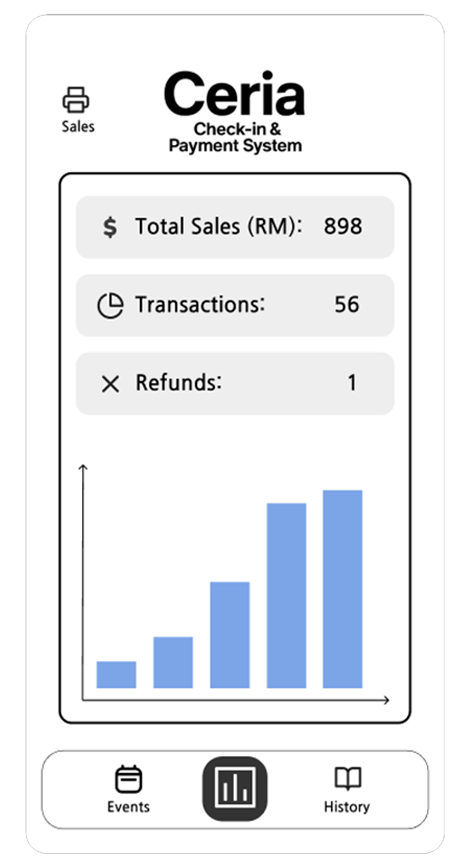


Figure 17 Vendor Home Page

**Home Page (Vendor)**

The home page for the vendor shows a dashboard with a few data. The data shown includes the total sales (RM), the transactions made between vendor and customers, and the number of refunds. There’s also a bar chart with the X-axis being the time, and the Y-axis being the amount of sales. Clicking on each bar chart will show the amount of sales and the hour (e.g. 10am).

At the top, there’s a **Sales** button which leads you to the Sales page.

At the bottom, the navigation bar includes:

* **Events** page button.
* **Home** page button (current page).
* **History** page button.

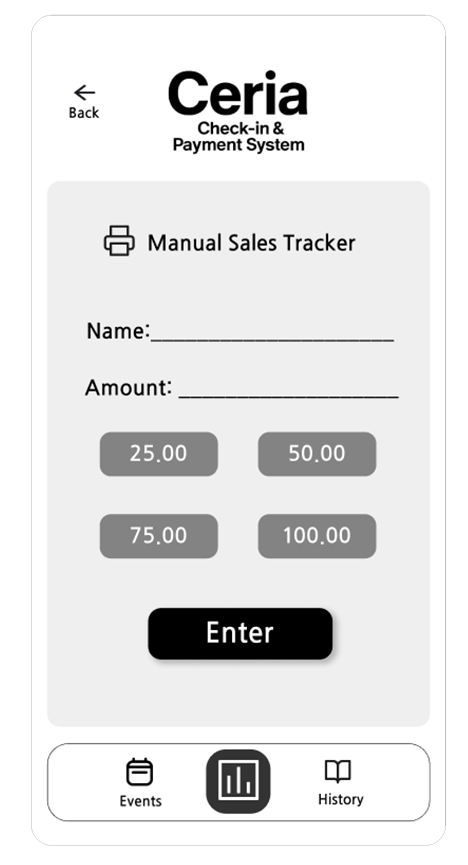


Figure 18 Vendor POS System

**Sales Page (Vendor)**

On the sales page, vendors can put the name of the customer and the amount they paid. There’s also a few buttons that, when pressed, automatically fills the amount (e.g. RM25.00). Then, you can press the “Enter” button to key-in your sales.

At the top, there is a **Back** button to return to the previous page.

At the bottom, the navigation bar includes:

* **Events** pagebutton.
* **Home** page button.
* **History** page button.



Figure 19 Vendor Events Page

**Events Page (Vendor)**

On the events page, vendors can view all the upcoming events that use the Ceria Check-in & Payment System. It shows the name of the event, its location, and the date it will be held.

At the top, there is a **Back** button to return to the previous page.

At the bottom, the navigation bar includes:

* **Events** page (current page).
* **Home** page button.
* **History** page button.



Figure 20 Vendor Event History Page

**History Page (Vendors)**

On the history page, vendors can see all the transactions from most recent to oldest. The transaction includes the name of the customer and the amount. The transaction is updated from the “Sales” page where the vendor manually inputs their sales and it gets reflected here.

At the top, there’s a **Back** button to return to the previous page.

At the bottom, the navigation bar includes:

* **Events** page button.
* **Home** page button.
* **History** page button (current page).

### 3.4.3 Hardware Interfaces

1. Student Devices

* Students use smartphones (Android/IOS) equipped with built-in cameras to scan event QR code displayed in the venue.
* Devices must be connected to Wi-Fi or mobile data.
* The camera will be used via the app.

1. Vendor Devices

* The system shall support vendors using tablets or smartphones to manage sales and process payments.
* Devices must have a stable internet connectivity via Wi-Fi

1. Admin Devices

* Admin shall operate the system from laptops or desktop computers for managing events, tracking attendance and generating reports.
* Devices must have secure access Wi-Fi or Ethernet and be compatible with standard web browsers.

1. Display Devices for QR Code

* Admin may use a tablet or laptop to display the generated QR code for the students to scan.
* These devices must support a high resolution display to make sure students can easily scan the QR code.

1. Server Infrastructure

* The backend system shall operate on a university hosted server.
* All communication between servers and user devices occurs over HTTPS using TLS for secure data transfer.

### 3.4.4 Software Interfaces

1. University Student Database API

* The system shall interact with the university’s student information system hosted on **SQLite**.
* This interface will be used for verifying student credentials during the login and check-in process.

1. Payment Gateway API

* The system will integrate with **Stripe** to handle all student payment transactions for vendor purchases.
* **Stripe** will be used for creating payment intents, confirming payments and handling transactions status.
* Communication will be over HTTPS and API keys will be securely stored in the server.

1. QR Code Generation Library

* The system will utilize the **qr\_flutter** package (for flutter apps) to dynamically generate QR codes for event entry.
* These QR codes will contain encrypted event ID and be displayed on screen.

1. Logging and Analytics

* The system will use **Elastic Stack** for logging events such as check-in and log in.
* Logs will be pushed via **Logstash**, indexed in **Elasticsearch** and visualized through **Kibana.**

### 3.4.5 Communication Interfaces

1. Student Database Communication

* Protocol: HTTPS over REST API
* Purpose: To securely validate students record

1. Payment Gateway Communication

* Protocol: HTTPS with TLS encryption
* Purpose: To securely process payment from students to vendor

1. Internal System Communication

* Protocol: HTTPS
* Purpose: For interactions such as QR generation, event sync, and check-in logging

1. Students, Admin and Vendor Interfaces

* Protocol: HTTPS over web interface
* Purpose: To secure interaction with the backend such as create event, view reports and transactions

## 

## 3.5 Logical Database Requirements

| **System Database** | **Description** |
| --- | --- |
| Users | Stores student, admin, and vendor information |
| Events | Stores event metadata |
| Tickets | Associates users with registered events |
| CheckIns | Logs each check-in occurrence |
| Transactions | Tracks on-site purchases and payments |
| Roles | Defines permissions and categorizes user types (student, admin, vendor). |

Table 37 Logical Database

## 3.6 Design Constraints

| **Design Constraints** | **Description** |
| --- | --- |
| Device Compatibility | Outdated or unsupported devices may cause degraded performance. |
| Third-party Integrations | Relies on university-provided student databases and a secure third-party payment gateway. |
| Network Dependence | The system requires stable internet access for real-time check-ins and transactions; offline mode is not supported. |
| Limited Scope | Outdoor navigation, venue maps, and social sharing are outside the scope of the initial system. |
| Security Compliance | Must comply with relevant data protection standards when handling user credentials and financial data. |

Table 38 Design Contraints

## 

## 

## 3.7 Software System Attributes

| **Attributes** | **Description** |
| --- | --- |
| Reliability | The system shall recover from crashes within 1 minute without data loss. Check-in, transactions and reports must be automatically saved and restored on restart. |
| Availability | The system shall maintain its uptime during event hours using auto-recovery to resume from the last stable state. |
| Security | The system shall use TLS Encryption for data in transit and AES Encryption for data at rest. Data integrity checks will ensure protection against unauthorized access and support data privacy. |
| Maintainability | The system will be modular and well-documented, allowing for easy updates, bug fixes. It is easier to separate into manageable modules. |
| Portability | The software shall run on mobile OS (Android/IOS). |

Table 39 Software System Attributes

## 3.8 Supporting Information

| **Category** | **Descriptions** |
| --- | --- |
| Documentation | * User manuals for students, vendors, and admins * Technical API documentation for backend and third-party integrations |
| Source Code Management | * Hosted on GitHub |
| Legal and Ethical Compliance | * Data collection complies with university policies and local privacy laws * - Payment data processed through PCI DSS-compliant gateway |
| Tools & Frameworks | * We will use Flutter for the app, and either SQLIte or MongoDB for the database. |

Table 40 Supporting Information

# **VERIFICATION**

## 4.1 Verification Approach

* **HOW:**

Verification will be carried out through testing methods such as user testing, system testing and integration testing. Test cases will be derived from the requirements from the document to ensure traceability.

* **WHO:**
* Developers will conduct system and integration testing during developments.
* End users (students, admins, vendors) will participate in user testing.
* Project manager will oversee the verification process and ensure all criteria are met before deployment.
* **WHEN:**
* Integration testing will do once component of the system are ready
* System testing will be performed after all modules are integrated.
* User testing will take place during the before deployment phase.
* Final verification will be done before the system goes public.
* **WHERE:**
* Development and testing environments hosted on internal university servers will be used for integration and system testing.
* Verification documentation and test results will be maintained in a shared project repository accessible to all stakeholders.

## 4.2 Verification Criteria

| **Requirement Area** | **Verification criteria** |
| --- | --- |
| **Login** | Students, admin and vendors must be able to log in with valid credentials. Login response time should be less than **3** seconds. |
| **Event Registration** | Students must be able to register for an event and receive a ticket within **6** seconds after submission. |
| **Ticket verification** | The system must be able to detect the validation within **2** seconds. |
| **On-site Purchases** | Vendors must be able to complete a transaction within **3** seconds after confirming payment. |
| **Data consistency** | All actions such as transaction or check-in must be recorded reliably in the database. |
| **Error handling** | If a failure occurs, the system must return an error message within **2** seconds. |
| **Accessibility** | The system must have accessibility standard such as readable font or proper navigation |
| **Load handling** | The system must maintain functionality with up to 300 concurrent users without significant delay ( no page load exceeds **3** seconds). |
| **Scalability** | The system must support scaling up to 3000 users with no errors. |
| **Data Synchronization** | All changes made by the admin such as event status must reflect across the system within **10** seconds. |
| **Timezone & Clock Sync** | All timestamps such as check-in time must be synced time format (MYT/GMT+8) to avoid mismatch. |
| **User Interface Response Time** | All users actions such as click on the navigation or form submission must trigger a response in under **3** seconds. |

Table 41 Verification Criteria

# **APPENDICES**

## 5.1 Assumptions and Dependencies

**Assumptions**

* The university should already provide access to its student identification database via a secure API or database connection.
* Events will have reliable internet connectivity to allow real-time check-in and payment processing.
* Students will have a valid student identification and ticket at the time of check-in.
* Students, admins and vendors will have access to the university internet connection to interact with the system.
* Admin operating the check-in system will be trained to use the platform effectively to ensure smooth operation.

**Dependencies**

* Device compatibility for check-in (e.g, device not compatible, device troubleshoot).
* Integration with the university’s student identification database for real-time identity verification
* Integration with a secure payment processing gateway to handle transactions.

## 5.2 Acronyms and Abbreviation

| API | Application Programming Interface |
| --- | --- |
| e.g | For example |
| GMT+8 | Greenwich Mean Time 8 hours ahead |
| MYT | Malaysian Time |
| TLS | Transport Layer Security |
| AES | Advanced Encryption Standard |
| PCI | Percutaneous coronary intervention |
| OPT | Optional Step |
| HTTPS | Hypertext Transfer Protocol Secure |

Table 42 Acronyms and Abbreviations