Software Requirements Specification

for

Undergrad Student Registration and Monitoring Platform

Prepared by Kachen Ruamek, Phubet Siriyasathain,

Suppapich Umchansa, Kamonchanok Phanthusin

Assumption University of Thailand

04 September 2025

Table of contents

Table of contents

1. **Introduction**
   1.1 Purpose
   1.2 Scope of the Project
   1.3 Intended Audience and Reading Suggestions
   1.4 Definitions, Acronyms, and Abbreviations
   1.5 Overview
2. **Overall Description**
   2.1 Product Perspective
   2.2 Product Features
   2.3 User Classes and Characteristics
   2.4 Operating Environment
   2.5 Design and Implementation Constraints
   2.6 User Documentation
   2.7 Assumptions and Dependencies
3. **External Interface Requirements**
   3.1 User Interfaces
   3.2 Hardware Interfaces
   3.3 Software Interfaces
   3.4 Communications Interfaces
4. **System Features**
   4.1 System Feature 1
   4.2 System Feature 2…
5. **Other Nonfunctional Requirements**
   5.1 Performance Requirements
   5.2 Safety Requirements
   5.3 Security Requirements
   5.4 Software Quality Attributes
6. **System Architecture and Design**
   6.1 High-Level Architecture
   6.2 Component Interaction
   6.3 Database Design
   6.4 Network Design
7. **Other Requirements**
   Appendix A: Glossary
   Appendix B: References
   Appendix C: Supporting Diagrams or Document

1. Introduction

1.1 Purpose

The product specified in this document is the Undergraduate Student Registration and Monitoring Platform (Release x.x, month 2025).
This platform integrates software and hardware components to support course registration, monitoring, and verification in real time.
It combines:

- Web-based system (course registration, schedule visualization, seat monitoring, QR receipt generation)
- Hardware-based module (ESP32 microcontroller with QR scanner and OLED for verification)

This is the first release of the system, developed as part of the Bachelor of Engineering program at Assumption University.

1.2 Scope of the Project

The scope of this SRS covers the entire platform, including software and hardware subsystems. The platform is designed to:

- Allow students to register for courses via a web application.
- Provide real-time seat availability and course schedules to both students and faculty.
- Generate a receipt with an embedded QR code upon registration.
- Enable staff or students to verify registration using a hardware-based QR scanning unit (ESP32 + OLED).
- Store and synchronize course and registration data in a Supabase cloud database.

This SRS describes the complete system, not just a single subsystem. It includes requirements for:

1. Web-based module (front-end interface for students and faculty, real-time dashboards, registration, database integration).
2. Hardware-based module (QR scanner + ESP32 microcontroller).

1.3 Intended Audience and Reading Suggestions

The intended audience for this document includes:

- Developers – to implement the software and hardware modules according to functional and non-functional requirements.
- Project Managers – to monitor progress, ensure deliverables align with specifications, and manage scope.

- Testers/QA Engineers – to design and execute test cases for validation and verification.
- Faculty Administrators/Users – to understand the system's capabilities and constraints for operational use.

The document is organized into sections that progress from a high-level overview to detailed specifications:

1. Section 1 (Introduction) – Defines the purpose, scope, intended audience, key definitions, and overview of the system.
2. Section 2 (Overall Description) – Describes the product perspective, major features, user classes, operating environment, constraints, and assumptions.
3. Section 3 (External Interface Requirements) – Specifies the user interfaces, hardware interfaces, software interfaces, and communication interfaces.
4. Section 4 (System Features) – Details the functional features of the system, broken down into individual feature specifications.
5. Section 5 (Other Nonfunctional Requirements) – Outlines performance, safety, security, and software quality attributes.
6. Section 6 (System Architecture and Design) – Provides the high-level architecture, component interactions, database design, and network design.
7. Section 7 (Other Requirements) – Captures any additional system requirements not covered in previous sections.
8. Appendices (A–C) – Include glossary, references, and supporting diagrams or documents.

Reading Suggestions:

- Project Managers and Administrators should first read Sections 1 and 2 for a high-level understanding.
- Developers should focus on Sections 3 for implementation details.
- Testers should emphasize Sections 5-7 to design validation processes.
- End users (faculty and students) may focus on Sections 2 and 4 to understand the user-facing functions.

1.4 Definitions, Acronyms, and Abbreviations

- AU Spark – Assumption University's existing registration system.
- ESP32 – A microcontroller with Wi-Fi and Bluetooth capability, used for hardware integration.
- QR Code – Quick Response code, used to encode student registration details.
- UART – Universal Asynchronous Receiver-Transmitter, a serial communication protocol.
- Supabase – A cloud-based database platform, used for data storage.

- OLED – Organic Light-Emitting Diode, a display module for showing verification results.
- SRS – Software Requirements Specification.

1.5 References

1. R. Santos, "ESP32-CAM QR code reader user management system (web server)," Random Nerd Tutorials, May 2025. [Online]. Available: https://randomnerdtutorials.com/esp32-cam-qr-code-reader-web-server/
2. M. Penica, M. Bhattacharya, W. O'Brien, S. McGrath, M. Hayes, and E. O'Connell, "Advancing interoperable IoT-based access control systems: A unified security approach in diverse environments," IEEE Access, vol. 11, pp. 79450–79464, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10870056
3. R. Onggo, M. S. Nugroho, D. T. Nugroho, and R. R. Hakim, "Revolutionizing classroom attendance: A wireless smart system using ESP-NOW protocol," International Journal of Interactive Mobile Technologies, vol. 18, no. 10, pp. 23–38, 2024. [Online]. Available: https://www.researchgate.net/publication/387107654
4. H. Elbehiery, "Enhancement of QR-code student's attendance management system using GPS," International Journal of Computer Applications, vol. 178, no. 25, pp. 1–7, May 2019. [Online]. Available: https://www.academia.edu/64553334
5. N. Kamaruddin and A. Alduais, "Development of a web-based IoT class attendance monitoring system for a primary school," Journal of Educational Technology and Engineering, vol. 6, no. 2, pp. 45–55, 2024. [Online]. Available: https://www.researchgate.net/publication/387031188
6. H. Mustafa and N. Mustafa, "Student records management system (SRMS) using IoT," International Journal of Computational and Experimental Science and Engineering, vol. 11, no. 2, pp. 123–130, 2025. [Online]. Available: https://www.ijcesen.com/index.php/ijcesen/ article/view/1309
7. "Online QR code attendance system," International Research Journal of Modernization in Engineering Technology and Science (IRJMETS), vol. 5, no. 5, pp. 2324–2331, May 2023. [Online].
8. J. H. Chang, "An introduction to using QR codes in scholarly journals," Science Editing, vol. 1, no. 2, pp. 113–117, 2014.
9. A. Bhargava, R. Kumar, and A. P. Bhondekar, "Generation and recognition of covert quick response codes," M.Tech Dissertation, Giani Zail Singh PTU, Bathinda, 2015.
10. A. Mishra and M. Mathuria, "A review on QR code," International Journal of Computer Applications, vol. 164, no. 9, pp. 17–19, 2017.
11. Denso Wave Incorporated, QR Code Essentials. [Online]. Available: https://www.qrcode.com/en/ (accessed: Sept. 10, 2025).

12. DSPMU Ranchi, “IEEE SRS template,” [Online]. Available: https://dspmuranchi.ac.in/pdf/Blog/srs template-ieee.pdf. [Accessed: Sept. 17, 2025].
13. Krazytech, “Sample software requirements specification (SRS) report - Airline database,” [Online]. Available: https://krazytech.com/projects/ sample-software-requirements-specificationsrs-report-airline-database.
14. TestRigor, “Software requirement specification (SRS) document - Best practices and guide,” [Online]. Available: https://testrigor.com/blog/ software-requirement-specification-document/.
15. I. Sommerville, Software Engineering, 10th Global Edition. Pearson, 2016.
16. ConceptDraw, “Flowchart examples – Six circles diagram,” [Online]. Available: https://www.conceptdraw.com/examples/6-circles-flow-chartpng. [Accessed: Sept. 17, 2025].

2. Overall Description

2.2 Product Perspective

The Undergraduate Student Registration and Monitoring Platform is a hybrid system that integrates software (web-based application and supporting scripts) with a hardware-based verification unit. It supplements the existing AU Spark system by automating data extraction through Python Selenium scripts, centralizing information in a Supabase cloud database, and presenting real-time updates via a web application. The platform is composed of two main subsystems:

- Web-Based Module – Provides the user interface for students and administrators to register for courses and receive a QR-encoded receipt, displays course schedules, seat availability in real time.
- Hardware-Based Module – Consists of an ESP32 microcontroller with a UART-based QR scanner and OLED display for validating registration receipts.

This integrated design bridges digital course management with physical verification, enhancing transparency and efficiency.

2.3 Product Features

Key features of the system include:

- Automated web scraping of course data from AU Spark.
- Real-time registration and seat monitoring through a centralized database.
- Generation of receipts with QR codes for each student registration.
- Hardware-based QR code scanning and verification using an ESP32 and OLED display.

- Administrative dashboards for monitoring enrollment and making informed decisions (e.g., opening new course sections).
- Scalable cloud-based backend (Supabase) to ensure reliable storage and synchronization.

2.4 User Classes and Characteristics

- Students – Register for courses, view schedules, and receive QR receipts. Expected to have basic familiarity with web and mobile applications.
- Faculty Administrators – Monitor enrollment data, verify student registrations, and manage seat availability. Require access to dashboards and validation tools.
- System Developers/Maintainers – Responsible for maintaining software, hardware firmware, and database integration. Require technical expertise in web development, cloud services, and embedded systems.

2.5 Operating Environment

- Software Environment:
  - Backend: Python (Selenium)
  - Database: Supabase
  - Frontend: HTML, CSS, JavaScript (designed with Figma, implemented in Visual Studio Code)
- Hardware Environment:
  - ESP32 microcontroller
  - UART QR code scanner
  - OLED display module
  - USB-TTL interface for communication and debugging
- Platforms:
  - Web applications are accessible via modern browsers (e.g., Chrome, Edge, Firefox)
  - Hardware verification unit operating as a standalone embedded system

2.6 Design and Implementation Constraints

- Dependency on AU Spark as the source of registration data; changes to AU Spark's structure may require updates to web scraping scripts.
- Internet connectivity required for real-time synchronization between the Supabase backend and the web application.
- Hardware QR scanner accuracy may be affected by environmental conditions (e.g., lighting, scanning angles).
- System performance is limited by Supabase's free-tier service restrictions.

2.7 User Documentation

The system will provide the following user documentation:

- USR _UserManual.pdf (PDF format)
    - Provides an overview of the platform's components and explains the functionality available to users in the web application (e.g., course registration, schedule visualization, seat monitoring, QR code receipt).
    - Includes step-by-step instructions with screenshots.

2.8 Assumptions and Dependencies

- Students and administrators have stable access to the internet and devices (PCs, laptops, smartphones) to use the web application.
- The university permits web scraping from AU Spark for educational/research purposes.
- Supabase cloud database service remains operational and continues to support required APIs for data storage and retrieval.
- ESP32 microcontroller and associated hardware modules (QR scanner, OLED display) are available and compatible with the system design.
- Supabase service remains operational and continues to support required APIs.
- Users are assumed to have at least basic digital literacy to interact with both the web application and the hardware verification module.

3. External Interface Requirements

3.1 User Interfaces

HTML, CSS, and JavaScript for web page display including:

- Time axis (07:30 – 22:30) with tick marks every 90 minutes.
- Rows for days (Sunday–Saturday).
- Course blocks showing course code, section, seat badge, and colors by prefix.
- Controls: search box, session filter dropdown, refresh button, day tabs.
- Details popup when clicking a course block (course title, section, instructor, exam dates, seats info).

3.2 Hardware Interfaces

- QR code scanner: Using for scanning QR code, that generate after registered courses and send ID to ESP32
- ESP32: Receive ID from QR code scanner, then communicate with server and send data to OLED
- OLED: Display data of registered courses from EPS32

3.3 Software Interfaces

- Supabase JavaScript Client Library (`window.supabase.createClient`) is used for database access.
- Supabase Database Service hosts the `data_vme` table.
- DOM APIs (`document.createElement`, `querySelector`) used for rendering.

3.4 Communications Interfaces

- The client communicates with Supabase REST API over HTTPS.
- Data exchange format is JSON.
- Authentication uses the Supabase public anon key.
- Requests: .select("*") queries to retrieve course data.
- Responses: JSON objects with course attributes (code, title, section, times, instructor, seats, exam info).

4. System Features

4.1 User Interfaces

5. Other Nonfunctional Requirements [5.1,2,4]

5.1 Performance Requirements

- The system should support at least *****(time needed e.g., 1 second, 1minute) concurrent users performing registration without performance degradation.
- The web application shall respond to user actions (e.g., submitting a registration form) within ***** seconds under normal load.
- The Supabase database shall process registration queries and updates with an average response time of less than ***** second.
- The hardware-based QR scanner shall decode and display registration data within ***** seconds of scanning.

5.2 Safety Requirements

- The system shall ensure data integrity so that registration records cannot be lost or corrupted during processing.
- The hardware subsystem shall be designed with safe voltage and current levels to prevent hazards to users.
- Backup and recovery procedures shall be available to restore registration data in case of database failure.
- The system shall avoid race conditions or duplicate entries during simultaneous registrations to prevent schedule conflicts.

5.3 Security Requirements

- User authentication shall be required for all students and administrators accessing the web application.
- Student credentials and registration data shall be protected using encryption during transmission (HTTPS/TLS). ******Not sure
- QR codes shall be generated with unique identifiers to prevent tampering or unauthorized duplication.
- The system shall log all access attempts, including failed logins, for auditing purposes. ***** Not sure
- The hardware verification module shall only display data for valid QR codes generated by the system.

5.4 Software Quality Attributes

*******Not Sure**

- Reliability: The system shall maintain 99% uptime during registration periods.
- Scalability: The system shall be able to scale by increasing database capacity and server resources to handle growing numbers of users.
- Usability: The web application interface shall follow usability principles (clear navigation, consistent layout) to support students with minimal training.
- Maintainability: The system architecture shall be modular, allowing updates to web scraping scripts, database schema, or hardware firmware independently.
- Portability: The web application shall be compatible with major browsers (Chrome, Firefox, Edge, Safari).

6. System Architecture and Design [not]

6.1 High-Level Architecture

6.2 Component Interaction

6.3 Database Design

6.4 Network Design

7. Other Requirements [not]

Appendix A: Glossary

Term	Acronym	Definition
AU Spark	-	Assumption University's existing registration system platform designed for educational

ESP32	-	A microcontroller with Wi-Fi and Bluetooth capability, widely used for hardware integration and IoT application
Quick Response code	QR Code	A two-dimensional barcode that can store information like URLs, text, or other data, readable by smartphones or scanners
Universal Asynchronous Receiver-Transmitter	UART	A hardware communication protocol used for serial communication between devices, transmitting data one bit at a time asynchronously
Supabase	-	An open-source backend platform that provides database, authentication, and API services
Organic Light-Emitting Diode	OLED	A type of display technology where each pixel emits its own light, providing high contrast and low power consumption
Software Requirements Specification	SRS	A formal document that describes the expected behavior, features, and constraints of a software system, serving as a guide for development and testing