SOFTWARE REQUIREMENTS SPECIFICATION

for

Kaiju Academy

Version 1.1

Prepared by
Group A2

YU Ching Hei	**1155193237**	**chyu@link.cuhk.edu.hk**
Lei Hei Tung	**1155194969**	**1155194969@link.cuhk.edu.hk**
Ankhbayar Enkhtaivan	**1155185142**	**1155185142@link.cuhk.edu.hk**
Yum Ho Kan	**1155195234**	**1155195234@link.cuhk.edu.hk**
Leung Chung Wang	**1155194650**	**1155194650@link.cuhk.edu.hk**

The Chinese University of Hong Kong
Department of Computer Science and Engineering

CSCI3100: Software Engineering

May 7, 2025

Contents

Document Revision History

Version	Revised By	Revision Date	Comments
0.1	C. H. Yu	2025-02-08	Updated: –Initial document structure –Basic template setup
0.2	C. H. Yu	2025-02-08	Updated: –Formatting
0.3	C. H. Yu	2025-02-08	Updated: –Titlepage formatting –Page numbering –Chapters title formatting –Chapter and sections arrangement
0.4	H. T. Lei	2025-02-09	Added: –Specific requirements
0.4.1	C. H. Yu	2025-02-10	Updated: –Specific requirements: fixed compilation error –Titlepage formatting: fixed alignment issue
0.5	C. H. Yu	2025-02-10	Added: –Acronyms and Abbreviations table –Product Overview –Product Functionality –User Classes and Characteristics –Design and Implementation Constraints –Assumptions and Dependencies
0.6	A. Enkhtaivan	2025-02-10	Updated: –System Features –Specific requirements Use Case Model
0.7	H. K. Yum	2025-02-10	Updated: –Nonfunctional Requirements
0.8	C. W. Leung	2025-02-10	Added: –Document Purpose –Project Scope –Intended Audience and Document Overview –Document Conventions –References and Acknowledgments
0.9	H. T. Lei	2025-02-10	Updated: –Refine Nonfunctional Requirements
1.0	C. H. Yu H. T. Lei A. Enkhtaivan H. K. Yum C. W. Leung	2025-02-10	Updated: –Finalized draft with all requirements completed and reviewed
1.1	C. W. Leung	2025-04-28	Modified: –Updated requirements and features according to meeting1.docx instructions: roles, deleted/added features, error handling, credit system, page list, PICs.

1. Introduction

1.1 Document Purpose

This document encompasses all essential functionalities, including course delivery, user progress tracking, assessment tools, user and educator profiles, and credit/payment features. All requirements reflect the current system scope as revised per stakeholder and instructor feedback.

1.2 Project Scope

Kaiju Academy is an online self-learning platform that makes learning to code easier and more engaging. The platform delivers interactive coding courses for all skill levels through a self-paced learning environment with progress tracking and instant feedback. Features now focus on interactive code assessment, course credit/-payment, and profile management. Community features, code competitions, dashboards, forum, notifications, calendar, keyboard shortcuts, and accessibility customizations have been removed from scope.

1.3 Intended Audience and Document Overview

This document serves as a guide for developers, managers, testers and stakeholders of Kaiju Academy. It contains sections covering product description, requirements, features, and non-functional requirements. We recommend starting with the product description section before moving to specific requirements.

1.4 Definitions, Acronyms and Abbreviations

Abbreviation	Definition
2FA	Two-factor Authentication
API	Application Programming Interface
CDN	Content Delivery Networks
GDPR	General Data Protection Regulation
LMS	Learning Management System
MC	Multiple Choice
REST	Representational State Transfer
SRS	Software Requirements Specification
UI	User Interface
UX	User Experience
MFA	Multi-Factor Authentication

1.5 Document Conventions

This Project Requirements Specification adheres to the IEEE Std 830-1998 standard. The document utilizes Times New Roman font in size 11 for consistency and readability. Important terms are highlighted in bold, while supplementary notes are presented in italics. Each requirement is uniquely identified, with higher-level requirements inheriting their priority unless otherwise specified. Sections and requirements are sequentially numbered to facilitate easy navigation. Additionally, technical terms and acronyms are clearly defined in the Glossary to ensure clarity and understanding for all readers.

1.6 References and Acknowledgments

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

2. Overall Description

2.1 Product Overview

Kaiju Academy is a web-based e-learning platform that provides interactive programming education through modern LMS capabilities and specialized coding features, facilitating the learning of programming anytime and anywhere as long as you are connected to the internet.
The platform serves as a comprehensive learning environment where students can engage with programming concepts through a structured curriculum. Users access the platform through web browsers, eliminating the need for software installation and reducing the barrier to entry for users.
The platform employs a module-based learning approach, where content is organized into discrete units that build upon each other. Each module typically contains:

- Video lectures with synchronized transcripts
- Interactive coding exercises
- Supplementary reading materials
- Practice problems and assignments
- Progress assessments

Students can track their progress through a personalized user profile, which displays completed modules, registered courses, recommended courses, and achievement metrics. The integrated code assessment editor allows students to practice coding directly within the browser, with immediate feedback and automated grading capabilities.
Educators can utilize the platform to create courses, manage course materials, and monitor student progress. The system enables educators to update course modules, manage course content, and view taught courses with a sidebar of categories.
A new platform feature is the credit system, which allows users to purchase course access via payment and credits.

2.2 Product Functionality

Kaiju Academy provides the following major functionalities:

- User Management and Authentication (including MFA if supported)
- Course Creation and Management
- Interactive Learning Environment (code assessment editor)
- Assessment and Progress Tracking
- User Profile (view all/registered/recommended courses, progress)
- Educator Profile (taught/manage courses, sidebar)
- Course Credit and Payment System

2.3 User Classes and Characteristics

Kaiju Academy serves three primary user classes:

Admin: Platform managers with full access for maintenance and monitoring.

User: Students who access courses, register, track learning progress, and manage credits.

Teacher/Educator: Subject matter experts who create/manage course content and monitor students.

2.4 Design and Implementation Constraints

The implementation of Kaiju Academy is governed by the following key constraints:

Hardware Constraints

- The platform should run with low memory usage.

Performance Constraints

- Page load time must not exceed 3 seconds.
- System must support at least 1000 concurrent users.
- Video streaming must adapt to user bandwidth.
- Code execution response time must be under 5 seconds.
- Database queries must complete within 1 second.

Operational Constraints

- System must achieve 99.9% uptime.
- Must implement automated backup systems.
- Must support horizontal scaling.
- Must implement monitoring and logging.
- Must provide disaster recovery procedures.

2.5 Assumptions and Dependencies

2.5.1 Assumptions

- **Internet access:** Since the application relies on a web interface, users must have stable internet connectivity with minimum 5 Mbps bandwidth or access through institution's local network.
- **Minimum System Requirements:** All client devices must feature:
  - Computer or mobile device with minimum 4GB RAM for smooth performance
  - Web browser (Chrome, Firefox, Safari, Edge - latest 2 versions)
  - HTML5 and JavaScript enabled

2.5.2 Dependencies

- **Third-Party Libraries and Services:**
  - Authentication protocols (OAuth2)
  - Database management systems (PostgreSQL, MongoDB, Redis)
  - Cloud infrastructure services (AWS/Google Cloud/DigitalOcean)
  - Content delivery networks (CDN)
  - Video hosting platforms
  - Code editor components (CodeMirror/Monaco)
- **Development Methodologies:**
  - UML Modeling: System architecture documentation and design specifications
  - COMET Methodology: Concurrent object modeling for workflow integration
  - Container orchestration with Docker and Kubernetes
  - CI/CD pipeline implementation

2.5.3 Critical Risks

- **External System Failures:**
  - Third-party service disruptions (authentication, video hosting)
  - Cloud infrastructure outages
  - CDN performance issues
  - Database system failures

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Kaiju Academy provides a graphical user interface (GUI) tailored for users, educators, and admins. The UI components include:

· **Online Code Assessment Editor:** An interactive online editor for code assessment tasks, supporting in-browser code editing and running.

· **Course Pages:** Structured pages for each course, displaying modules, materials (videos, PDFs), and assessments.

· **User Profile:** View all available courses, registered courses (showing modules), recommended courses for registration, and progress tracking.

· **Educator Profile:** View taught courses (with modules), manage courses with a sidebar of categories.

· **Course Credit/Payment:** Users can purchase credits via payment to unlock course access.

· **Login/Register:** Login (with MFA if available) and registration forms.

· **Error Handling:** Consistent error messages for invalid inputs, failed logins, or system errors.

· **Responsive Design:** All interfaces are accessible on desktops, tablets, and mobile devices.

3.1.2 Hardware Interfaces

The platform is designed to support various hardware components. Key considerations include:

· **User Devices:** Compatible with desktops, laptops, tablets, and smartphones supporting modern web browsers.

· **Server Infrastructure:** Hosted on cloud-based servers (AWS, Google Cloud, or Azure) with auto-scaling.

· **Peripheral Support:** Keyboards, mice, and touchscreens.

· **Network Requirements:** Requires a stable internet connection for streaming and online coding.

3.1.3 Software Interfaces

The system will integrate with various software components to ensure smooth operation and functionality. These include:

· **Operating Systems:** Compatible with Windows, macOS, Linux, Android, and iOS.

· **Web Browsers:** Supports the latest versions of Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge.

· **Database Management System:** Uses PostgreSQL or MySQL.

· **Authentication Services:** Integration with OAuth2 and MFA where possible.

· **APIs:** RESTful APIs for frontend/backend communication (user management, course, assessment, payment).

· **Content Delivery Networks (CDN):** For efficient media delivery.

· **Payment Gateway:** Integrated to support course credit purchases.

· **Logging and Monitoring:** Centralized error and usage monitoring.

3.2 Functional Requirements

The following table lists all functional requirements kept in scope (removed: dashboard, forum, code competition, notification, calendar, keyboard shortcuts, data backup, screen reader, color blindness):

Function ID	Function Name	Description
FR01	User Registration	Users can create an account by providing necessary details.
FR02	User Login	Users can log in using credentials; MFA supported if available.
FR03	Profile Management	Users can update personal details and account settings.
FR04	Role Management	Assign roles (User, Teacher, Admin) with specific permissions.
FR05	Course Creation	Educators can create and manage courses.
FR06	Course Enrollment	Users can register/enroll in courses (via credit/payment if required).
FR07	Course Content Management	Educators can upload and organize course materials.
FR08	Quiz/Code Assessment Management	Educators can create and assign coding/quiz assessments.
FR09	Automated Grading	System auto-grades multiple-choice, short-answer, and code assessment questions.
FR10	Manual Grading	Educators grade long-answer questions.
FR11	Progress Tracking	Users can track their learning progress in profile.
FR12	Course Credit/Payment	Users can buy credits to unlock courses.
FR13	Learning Recommendations	System recommends courses for registration.
FR14	Security	Implements authentication, authorization, and encryption.
FR15	Error Handling	Consistent user-facing error messages for invalid input, failed login, and system errors.

Table 3.1: Functional Requirements Table

3.3 Use Case Model

3.3.1 Use Case 1: Code Assessment

Description

A user edits and runs code in an online editor as part of a course assessment.

Actors and Preconditions

- **Actors**: User (Student)
- **Preconditions**: The user is logged in and enrolled in the course/module.

Steps

1. The user opens the code assessment in a course/module.
2. The user writes or edits code in the online editor.
3. The user runs code to see results/output.
4. The system provides immediate feedback or grading.

3.3.2 Use Case 2: Course Credit Purchase

Description

A user buys credits to unlock/register for a course.

Actors and Preconditions

- **Actors**: User (Student)
- **Preconditions**: The user is logged in; has access to payment methods.

Steps

1. The user navigates to the credit purchase page.
2. The user selects the amount of credits to buy.
3. The user completes payment.
4. Credits are added to the user's account.
5. The user uses credits to register for a course.

3.3.3 Use Case 3: Educator Course Management

Description

An educator creates or manages courses and modules.

Actors and Preconditions

- **Actors**: Teacher/Educator
- **Preconditions**: The educator is logged in and authorized.

Steps

1. The educator opens the manage courses page (with sidebar categories).
2. The educator creates/updates/deletes a course or module.
3. The system validates data and updates course content.
4. Students are notified of new/updated course materials (if applicable).

4. System Features

This section describes the functional requirements of the product by its major features (all removed features are omitted):

4.1 Online Code Assessment Editor

4.1.1 Description and Priority

The code assessment editor is a high-priority feature, providing an online text editor where users can write, edit, and run code for assignments and practice within the course module.

4.1.2 Stimulus/Response Sequences

- **User Action**: Opens code assessment in a module.
  System Response: Loads the online editor with problem description.
- **User Action**: Edits code and runs it.
  System Response: Executes code and displays result/output.
- **User Action**: Submits code for grading.
  System Response: Grades submission and provides feedback.

4.1.3 Functional Requirements

- The code editor must support editing and running code in the browser.
- Students must receive immediate feedback on code execution.
- Submissions must be automatically graded.

4.2 Course Pages and Profile Features

4.2.1 Course Pages

- Display list of modules, materials, and assessments.
- Allow access only to registered users (or after credit purchase).

4.2.2 User Profile

- View all courses and registered courses (show modules).
- View recommended courses for registration.
- Track progress across all courses.

4.2.3 Educator Profile

- View taught courses and modules.
- Manage course content (sidebar with categories).

4.3 Course Credit and Payment

- Users can purchase credits via integrated payment gateway.
- Credits are used to register/unlock courses.
- System tracks user balance and transaction history.

4.4 Authentication

- Users can register and log in.
- MFA is supported if available.

4.5 Error Handling

- Consistent error messages for all invalid actions and system failures.
- Error handling for payment, registration, login, and code execution.
- Error handling is under the responsibility of Ankr.

4.6 Frontend

- All frontend UI/UX development is overseen by Hazel and Daniel.
- Ensure all features/pages are responsive and user-friendly.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The platform must load within **2 seconds** for 95% of users under normal load conditions.
- Code execution results must be returned within **5 seconds** for 99% of submissions.
- The platform must support up to **10,000 concurrent users**.
- The platform must have an uptime of **99.9%**.

5.2 Safety Requirements

- The platform must prevent users from executing malicious code (sandboxing).
- The platform must comply with **GDPR** and **COPPA**.

5.3 Security Requirements

- Users must authenticate via **OAuth 2.0** or email/password with **MFA** support.
- Role-based access control (User, Teacher, Admin).
- All sensitive data must be encrypted using **AES-256**.
- All data in transit must be secured using **TLS 1.2** or higher.
- The platform must prevent brute-force, SQL injection, and XSS attacks.

5.4 Software Quality Attributes

- **Usability:** The platform must have an intuitive UI with a high satisfaction rate.
- **Maintainability:** The codebase must follow modular design principles.
- **Portability:** The platform must be compatible with major browsers and OSs.
- **Performance:** The platform must support **10,000 concurrent users** with a response time of less than **2 seconds**.

5.5 Additional Requirements

· **Database Requirements:** The system should support relational databases with ACID compliance.

· **Internationalization:** The platform should support multiple languages, including English, Spanish, and Chinese.

· **Legal Compliance:** The platform must adhere to data protection regulations (e.g., GDPR, CCPA).

· **Reuse Objectives:** Modular architecture should be adopted to allow for code reuse across projects.

· **Scalability Requirements:** The system should be designed to handle high concurrent users efficiently.

· **Customization and Extensibility:** The system should allow customization for branding and feature addition.