SOFTWARE REQUIREMENTS SPECIFICATION

for

Kaiju Academy

Version 1.0

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

February 11, 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. Introduction

1.1 Document Purpose

This document encompasses all essential functionalities, including course delivery, user progress tracking, community interaction, and assessment tools, providing both high-level and specific requirements. This document shall form the basis for all stakeholders and developers to understand intended features and constraints of the platform.

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. Game-like elements drive student engagement, while community features enable collaboration between learners and educators. Through this technology-driven learning experience, Kaiju Academy aims to expand digital education access and develop a skilled tech workforce.

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

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 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 dashboard, which displays completed modules, upcoming assignments, and achievement metrics. The integrated code 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 provide targeted assistance where needed. They could also answer questions from students in the discussion forum. The system's analytics tools help identify areas where students may be struggling, enabling educators to provide additional teaching resources. The platform features automated grading and feedback for assignments, allowing educators to have more flexible arrangement.

2.2 Product Functionality

Kaiju Academy provides the following major functionalities:

- User Management and Authentication
- Course Creation and Management
- Interactive Learning Environment
- Assessment and Progress Tracking
- Community and Discussion Features
- Administrative Tools

2.3 User Classes and Characteristics

Kaiju Academy serves four primary user classes:

Administrators: System managers with full platform access for maintenance and monitoring

Students: Main users who access courses and track their learning progress

Educators: Subject matter experts who create and manage course content

Content Moderators: Forum managers who maintain community standards

2.4 Design and Implementation Constraints

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

Hardware Constraints

- We should able to run the platform 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 will have a graphical user interface (GUI) designed for ease of use and accessibility across different roles, including students, educators, and administrators. The UI components include:

· **Dashboard:** Upon login, users will be presented with a dashboard displaying their enrolled courses, upcoming deadlines, progress tracking, and notifications.

· **Navigation:** A global navigation bar providing quick access to Courses, Assessments, Calendar, Discussion Forum, and Profile settings.

· **Course Pages:** Each course contains a structured layout displaying modules, videos, PDFs, quizzes, and assessments with a progress tracker.

· **Assessment Interface:** Interactive assessment screens allowing multiple-choice (MC) questions with self-checking, short-answer autograded questions, and long-answer questions submitted for manual grading.

· **Progress Tracking:** A visual roadmap for students to track completed and pending modules and overall course progress.

· **Discussion Forum:** A interactive discussion forum with search, filter, and sort options.

· **Calendar:** An integrated calendar highlighting course schedules, assignment deadlines, and upcoming events.

· **Notifications:** A notification center alerting users about new content, deadlines, and updates.

· **Coding Environment** An embedded coding environment for practice exercises and coding competitions, similar to LeetCode.

· **Error Handling** Consistent error messages displayed in case of invalid input, failed login attempts, or system errors.

· **Keyboard Shortcuts** Common keyboard shortcuts for navigation and execution of commands, enhancing efficiency.

· **Mobile Compatibility:** A responsive design ensuring accessibility 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:** The platform will be 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 to accommodate user load.

· **Peripheral Support:** Users can interact using keyboards, mice, touchscreens, and audio devices for accessibility.

· **Network Requirements:** Requires a stable internet connection with minimum bandwidth to support video streaming and coding environment interaction.

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 to store user data, course materials, progress tracking, and assessment records.

· **Authentication Services:** Integration with OAuth2 for third-party authentication (Google, GitHub, etc.) and two-factor authentication (2FA).

· **APIs:** RESTful APIs to connect frontend and backend services, including: User authentication and management, Course content retrieval and management, Assessment grading and tracking, Notification and messaging services, Code execution API for online coding exercises

· **Content Delivery Networks (CDN):** Utilized for efficient media delivery and reduced latency.

· **Notification Services:** Integration with email and push notification services for alerts and reminders.

· **Logging and Monitoring:** Implementation of centralized logging and monitoring tools for system health tracking.

3.2 Functional Requirements

Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks, or functions the system is required to perform. Below is a detailed list of product functions:

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.
FR03	Profile Management	Users can update personal details and account settings.
FR04	Role Management	Assign roles (Student, Educator, Admin) with specific permissions.
FR05	Course Creation	Educators can create and manage courses.
FR06	Course Enrollment	Students can enroll or drop courses.
FR07	Course Content Management	Educators can upload and organize course materials.
FR08	Quiz Management	Educators can create and assign quizzes.
FR09	Automated Grading	System auto-grades multiple-choice and short-answer questions.
FR10	Manual Grading	Educators grade long-answer questions.
FR11	Progress Tracking	Users can track their learning progress.
FR12	Notifications	System alerts users of deadlines, updates, and activities.
FR13	Calendar Integration	Displays schedules, deadlines, and reminders.
FR14	Discussion Forum	Users can post, reply, and discuss course-related topics.
FR15	Online Coding Environment	Users can solve coding problems with real-time execution.
FR16	Code Competitions	Users can participate in coding challenges with leaderboards.
FR17	Search	Users can search and filter courses, quizzes, and discussions.
FR18	Learning Recommendations	AI-based suggestions for personalized learning paths.
FR19	Data Backup	System automatically backs up data periodically.
FR20	Security	Implements authentication, authorization, and encryption.

Table 3.1: Functional Requirements Table

3.3 Use Case Model

3.3.1 Use Case 1: View Course Progress

Description

This use case describes how a student can view their progress in any course

Actors and Preconditions

- **Actors**: Student
- **Preconditions**: The student must be logged in and enrolled in the course.

Steps

1. The student navigates to the "Courses" section.
2. The student selects a course.
3. The system displays the course progress.

3.3.2 Use Case 2: Course Modification

Description

This use case describes the process of an educator modifies a course on the Kaiju Academy platform.

Actors and Preconditions

- **Actors**: Educator
- **Preconditions**: The educator must be logged in and have the necessary permissions to modify courses.

Steps

1. The educator navigates to the "Course Management" section.
2. The educator enters course details (e.g., title, description, modules, and materials).
3. The system validates the course details.
4. The system adds the course to the platform and notifies enrolled students (if applicable).

Alternative Flow

- **Step 4a**: If the educator provides invalid details (e.g., missing title), the system displays an error and prompts corrections.
- **Step 5a**: If the educator cancels, the system discards the changes and returns to the "Course Management" section.

4. System Features

This sections serves to describe the functional requirements of the product by its major features. The features are described in detail, including their development priority, user interactions, and specific functions.

4.1 User Dashboard

4.1.1 Description and Priority

The User Dashboard is the High Priority feature as the hub is where users can access their courses, track their progress, view upcoming deadlines, and receive notifications.

4.1.2 Stimulus/Response Sequences

- **User Action**: Logs into the system.
  System Response: Displays the dashboard with course progress, deadlines, and notifications.
- **User Action**: Clicks on a course.
  System Response: Redirects to the course page, displaying modules, videos, and assessments.
- **User Action**: Marks a module as completed.
  System Response: Updates the progress tracker and highlights the next module.

4.1.3 Functional Requirements

Functional Requirements

- **REQ-1**: The dashboard must display enrolled courses, progress, and deadlines upon login.
- **REQ-2**: Users should be able to navigate to course pages directly from the dashboard.
- **REQ-3**: Progress tracking must update in real-time when a module is marked as completed.

4.1.4 Course Management

Description and Priority

The **Course Management** feature is the High Priority feature as the feature enables educators to create, update, and delete courses, as well as manage course materials such as videos, PDFs, and quizzes. It ensures that educators can efficiently organize and maintain course content, providing a structured learning experience.

Stimulus/Response Sequences

- **Educator Action**: Creates a new course.
  System Response: Displays section to the educator to upload course materials and set deadlines.
- **Educator Action**: Updates a course module.
  System Response: Saves changes and notifies enrolled students of updates.
- **Educator Action**: Deletes a course.
  System Response: Removes the course and all associated materials from the system.

Functional Requirements

- **REQ-1**: Educators must be able to create, update, and delete courses.
- **REQ-2**: Course materials (videos, PDFs, quizzes) must be uploaded and organized by modules.
- **REQ-3**: Students should receive notifications when course materials are updated.

4.1.5 Assessment and Grading

Description and Priority

The **Assessment and Grading** feature allows educators to create quizzes and assessments, while students can complete them and receive grades. This feature is of **Medium Priority** as it supports the learning process but is secondary to content delivery. It provides a mechanism for evaluating student understanding and progress.

Stimulus/Response Sequences

- **Educator Action**: Creates a quiz with multiple-choice (MC) and short-answer questions.
  System Response: Saves the quiz and makes it available to students.
- **Student Action**: Completes a quiz.
  System Response: Automatically grades MC questions and displays results.
- **Educator Action**: Reviews and grades short-answer questions.
  System Response: Updates the student's grade and provides feedback.

Functional Requirements

- **REQ-1**: Educators must be able to create quizzes with MC and short-answer questions along with coding assessments with the test cases.
- **REQ-2**: MC, and coding assessment questions must be automatically graded, and results displayed immediately.
- **REQ-3**: Educators should be able to manually and use AI prompt grade short-answer questions and provide feedback.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- **Response Time:**
  - 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, even during peak usage.
  - Real-time collaboration features (if any) must have a latency of less than **200ms** for seamless interaction.
- **Scalability:**
  - The platform must support up to **10,000 concurrent users** without degradation in performance.
- **Availability:**
  - The platform must have an uptime of **99.9%** (excluding scheduled maintenance).
  - Downtime for scheduled maintenance must not exceed **10 hours per month** and must be communicated to users at least **24 hours** in advance.
- **Resource Usage:**
  - Code execution environments must not exceed **10MB of memory** per user session.
  - The platform must handle **10,000 code submissions per minute** without performance degradation.

5.2 Safety Requirements

- **User Safety:**
  - The platform must prevent users from executing malicious code that could harm the system.
  - A sandboxed environment must be used for code execution.
- **Data Safety:**
  - Regular backups of user data must be performed daily and stored securely for at least **30 days**.
  - In case of data loss, user data must be restorable to a state no older than **24 hours**.
- **Compliance:**
  - The platform must comply with **GDPR** and **COPPA**.

5.3 Security Requirements

- **Authentication and Authorization:**
  - Users must authenticate via **OAuth 2.0** or email/password with **MFA** support.
  - Role-based access control (RBAC) must be implemented.
- **Data Security:**
  - All sensitive data must be encrypted using **AES-256**.
  - All data in transit must be secured using **TLS 1.2** or higher.
- **Prevention of Abuse:**
  - The platform must prevent brute-force attacks, SQL injection, and XSS attacks.
  - Rate limiting must be implemented (e.g., max **10 submissions per minute** per user).
- **Compliance:**
  - The platform must comply with **ISO/IEC 27001** and **SOC 2 Type II**.

5.4 Software Quality Attributes

- **Usability:**
  - The platform must have an intuitive UI with a **90% satisfaction rate**.
  - The learning curve for new users should not exceed **10 minutes**.
- **Reliability:**

  - The platform must have **99.9% uptime** and recover from failures within **5 minutes**.
- **Maintainability:**
  - The codebase must follow modular design principles, with at least **80% test coverage**.
  - The platform must support seamless updates with zero downtime.
- **Portability:**
  - The platform must be compatible with major browsers and OSs.
  - Mobile responsiveness must be ensured for screen sizes as small as **320px**.
- **Performance:**
  - The platform must support **10,000 concurrent users** with a response time of less than **2 seconds**.

5.5 Additional Requirements

This section defines additional requirements that are not covered elsewhere in the document.

· **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) and accessibility laws.

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

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

· **Backup and Recovery:** Automated daily backups must be implemented, with a recovery plan in place for data loss scenarios.

· **Customization and Extensibility:** The system should allow customization for branding and the addition of new features without major rework.