Software Requirements Specification

for

Exam Management System

Version 1.0 approved

Prepared by
AP23110011211
AP23110011213
AP23110011216
AP23110011265

SRMUAP

27-01-2026

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This document serves as the primary roadmap for the Exam Management System platform. It defines how we use artificial intelligence to create, manage, and grade student assessments. Our goal is to provide a clear technical guide for developers while ensuring educators understand the system's core value.

1.2 Document Conventions

We follow the standard IEEE 830 formatting rules to ensure consistency and clarity. Every requirement is prioritized based on its impact on the student's exam experience and the teacher's workflow. We use bold text to highlight critical system behaviors and mandatory security protocols.

1.3 Intended Audience and Reading Suggestions

This SRS is designed for a diverse group of readers, including software engineers, school administrators, and academic stakeholders. Developers should focus on the technical interfaces in Section 3, while educators may find the "System Features" in Section 4 most relevant to their needs.

1.4 Product Scope

ExamZone is an end-to-end ecosystem that bridges the gap between traditional paper exams and modern AI. It handles everything from indexing textbooks to generating questions and grading handwritten scans. It is built to reduce the administrative burden on teachers while maintaining high academic integrity.

1.5 References

Our development process is guided by the IEEE Standard for Software Requirements. We also rely heavily on the technical documentation provided by the Google Gemini API, LangChain framework, and ChromaDB. These resources ensure that our RAG architecture remains cutting-edge and reliable.

2. Overall Description

2.1 Product Perspective

ExamZone operates as a specialized educational tool that sits between a school's document storage and the final result sheet. Unlike standard quiz apps, it uses a Retrieval-Augmented Generation (RAG) backend to "read" specific course materials. This ensures that the AI's logic is always grounded in the teacher’s own curriculum.

2.2 Product Functions

The system performs four major roles: indexing documents into a searchable vector database, generating balanced exam papers, monitoring students for cheating, and grading scans. It also features a 24/7 AI Tutor that helps students learn from their mistakes after the exam is over.

2.3 User Classes and Characteristics

Teachers are the "Admins" who oversee the knowledge base and finalize grades. They need a simple, intuitive dashboard that requires no coding knowledge. Students are the "Learners" who need a secure, distraction-free environment to showcase their knowledge and receive instant feedback.

2.4 Operating Environment

The platform is built to run on any modern web browser like Chrome or Firefox. The heavy lifting is done on a Python-based FastAPI server, while student data is managed by Node.js. This setup ensures that the system is fast, responsive, and compatible with most school hardware.

2.5 Design and Implementation Constraints

The most significant constraint is the need for high accuracy in AI grading. We use Google Gemini to minimize "hallucinations" and ensure the AI only uses the provided documents. Additionally, the system must remain lightweight enough to run smoothly on standard student laptops and tablets.

2.6 User Documentation

ExamZone provides tailored digital manuals: a "Command Center" guide for teachers to manage AI indexing and an "Exam Readiness" guide for students to navigate secure testing. The dashboard includes interactive tooltips and video walkthroughs that guide users through their first submission or exam creation. This integrated support ensures technical hurdles never disrupt the learning process, effectively minimizing the need for external IT assistance during high-stakes testing windows.

2.7 Assumptions and Dependencies

The platform assumes students have functional cameras and stable internet for proctoring and OCR tasks. It depends on the uptime of the Google Gemini API to process RAG-based evaluations and AI question generation in real-time. Finally, the system requires high-quality document uploads from teachers to ensure the vector database creates an accurate and reliable knowledge base for the learning assistant.

3. External Interface Requirements

3.1 User Interfaces

The Teacher Dashboard focuses on data visualization and content management through a clean, professional layout. The Student Portal is designed with a "minimalist" approach to prevent test anxiety and technical confusion. Both interfaces are fully responsive and work seamlessly across desktop and mobile devices.

3.2 Hardware Interfaces

ExamZone interacts primarily with the user's camera and microphone for proctoring purposes. The system also requires a high-quality camera interface for the OCR module. This allows students to take clear photos of their handwritten answer sheets, which the AI then converts into digital text.

3.3 Software Interfaces

Our platform connects to the Google Gemini API for natural language processing and question generation. We use MongoDB for traditional data like usernames and passwords, while ChromaDB handles the complex AI embeddings. Google Drive API is integrated to provide teachers with easy storage for exam reports.

3.4 Communications Interfaces

ExamZone uses secure HTTPS for all data transfers between the browser and servers, ensuring student privacy. It integrates WebRTC for real-time proctoring alerts and RESTful APIs to connect with Google Gemini and MongoDB. This combination ensures a synchronized, fast, and reliable flow of information .

4. System Features

4.1 RAG-Powered Content Indexing

4.1.1 Description and Priority

This feature allows the system to convert uploaded educational documents into a searchable vector database for precise AI interaction. It is a **High Priority** feature (Benefit: 9, Risk: 3) as it forms the foundation for grounded, hallucination-free AI grading and tutoring.

4.1.2 Stimulus/Response Sequences

When a teacher uploads a PDF, the system initiates document parsing and embedding generation. Once processed, the system confirms successful indexing by updating the "Knowledge Base" status and enabling question generation for that specific material.

4.1.3 Functional Requirements

REQ-1: The system must support PDF, DOCX, and TXT file formats for vector indexing.
REQ-2: The system shall store document embeddings in ChromaDB to enable semantic search retrieval.

4.2 AI Evaluation and Feedback

4.2.1 Description and Priority

This feature uses LLMs to grade student answers by comparing them against the indexed course materials. It is a High Priority feature (Benefit: 9, Risk: 5) designed to provide instant, objective feedback and significantly reduce teacher workload.

4.2.2 Stimulus/Response Sequences

When a student submits an answer, the system retrieves relevant context from the Vector DB and passes it to the LLM. The system then generates a score and a detailed explanation, which is displayed on the teacher's review dashboard.

4.2.3 Functional Requirements

- **REQ-3:** The system must provide a citation or source reference for every grade generated by the AI.
- **REQ-4:** If the OCR text is illegible, the system shall flag the response for manual teacher intervention.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system must generate AI responses and grading feedback within 10 seconds to maintain a smooth user experience. Semantic search queries against the vector database must return relevant context in under 2 seconds, ensuring that the 24/7 AI Learning Assistant remains responsive and helpful.

5.2 Safety Requirements

ExamZone must ensure that AI-generated content is filtered for educational appropriateness and follows safety guidelines to prevent biased or harmful output. The system must include a "Human-in-the-Loop" safeguard, requiring teacher approval before any AI-calculated grades are officially published to students.

5.3 Security Requirements

All student data and exam submissions must be protected using AES-256 encryption both at rest and during transmission. User identity is managed through secure JWT authentication, and all API keys for Google Gemini must be stored in encrypted environment variables to prevent unauthorized access.

5.4 Software Quality Attributes

The platform prioritizes Reliability and Usability, ensuring that the interface is simple enough for non-technical users to navigate. We also emphasize Interoperability, allowing the system to easily export result data to common formats like Google Sheets and PDF for institutional record-keeping.

5.5 Business Rules

Only users with the "Teacher" role can upload source documents and modify the AI's grading parameters. Students are strictly restricted to the "Attempt" and "Review" modes, and their access to the AI Assistant is limited until after the official exam submission is locked.

6. Other Requirements

Appendix A: Glossary

- **RAG:** Retrieval-Augmented Generation, a method to improve AI accuracy using external facts.
- **Vector DB:** A database that stores text as mathematical vectors for semantic similarity searching.

Appendix B: Analysis Models

B.1 System Sequence Diagram (The Exam Lifecycle)

This model illustrates the step-by-step communication between the Student, the Node.js Backend, and the FastAPI RAG service. It tracks the process from the moment a student initiates a secure session to the final OCR text extraction and AI evaluation. This diagram ensures that the timing and data flow between the proctoring logic and the grading engine are perfectly synchronized.

B.2 Data Flow Diagram (RAG Ingestion & Retrieval)

This diagram visualizes how raw data—such as teacher-uploaded PDFs—is transformed into mathematical embeddings. It maps the movement of information from the document parser to the ChromaDB vector store and finally to the LLM during a query. This model is essential for developers to understand how the system "remembers" course content and retrieves it to answer student questions accurately.

Appendix C: To Be Determined List

- Final latency targets for concurrent OCR processing of 100+ users.
- Exact storage limits for free-tier MongoDB document metadata.