Software Requirements Specification (SRS)

Secure Academic Audit Management Tool

Version 1.0
Prepared by: *Sugashini M*
Organization: *Bannari Amman Institute of Technology*
Date: *03-02-2026*

Table of Contents

1. Introduction

1.1 Purpose

This document specifies the software requirements for the **Secure Academic Audit Management Tool**, a web-based application developed using the **MERN stack**.

The purpose of this SRS is to provide a detailed description of system functionality, interfaces, constraints, and requirements for developers, evaluators, and stakeholders.

1.2 Document Conventions

- "Shall" indicates a mandatory requirement
- "Should" indicates a recommended feature
- All requirements are uniquely identified
- Section numbering follows the IEEE SRS standard

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- Project guides and evaluators
- Developers and testers
- System administrators

Readers should begin with **Section 1** for an overview and proceed to **Sections 4 and 5** for detailed functional and non-functional requirements.

1.4 Product Scope

The Secure Academic Audit Management Tool is designed to digitize and secure academic audit processes within educational institutions.

The system ensures transparency, role-based access, and secure maintenance of academic audit records, replacing manual and paper-based audit practices.

1.5 References

- IEEE Software Requirements Specification Standard
- MERN Stack Documentation
- Academic Audit Guidelines
- SRS Template by Karl E. Wiegers

2. Overall Description

2.1 Product Perspective

The system is a standalone web application that centralizes academic audit data. It replaces existing manual or spreadsheet-based audit processes and provides secure, traceable access to audit information.

2.2 Product Functions

- User authentication and role-based authorization
- Academic record creation and management
- Secure audit trail generation
- Audit review and approval workflows
- Report generation and data retrieval

2.3 User Classes and CharacteristicS

User Class	Description
Admin	Manages users, system settings, and audit assignments
Auditor	Reviews academic records and verifies compliance
Faculty	Uploads and updates academic audit data
Student	Views personal academic audit records

2.4 Operating Environment

- Frontend: Web browser (Chrome, Firefox, Edge)
- Backend: Node.js with Express.js
- Database: MongoDB
- Operating System: Windows / Linux / macOS

2.5 Design and Implementation Constraints

- MERN stack must be used
- JWT-based authentication required
- Secure password storage using hashing
- HTTPS communication mandatory

2.6 User Documentation

- User manual

- Admin guide
- Online help documentation

2.7 Assumptions and Dependencies

Assumptions

- Users have basic knowledge of web applications
- Internet connectivity is available

Dependencies

- Browser compatibility
- MongoDB server availability

3. External Interface Requirements

3.1 User Interfaces

- Responsive UI using React.js
- Role-based dashboards
- Standard navigation and error messages

3.2 Hardware Interfaces

No specific hardware interfaces are required.

3.3 Software Interfaces

- REST APIs between frontend and backend
- MongoDB for data storage
- JWT for authentication

3.4 Communications Interfaces

- HTTP/HTTPS protocol
- Secure API communication

4. System Features

4.1 User Authentication and Authorization

4.1.1 Description and Priority

Provides secure login and role-based access control.
Priority: High

4.1.2 Stimulus/Response Sequences

- User enters credentials
- System validates credentials
- User is redirected to role-specific dashboard

4.1.3 Functional Requirements

- **REQ-1:** The system shall authenticate users using valid credentials
- **REQ-2:** The system shall restrict access based on assigned roles
- **REQ-3:** The system shall securely store passwords

4.2 Academic Audit Record Management

4.2.1 Description and Priority

Allows creation and management of academic audit records.
Priority: High

4.2.2 Stimulus/Response Sequences

- Faculty submits academic data
- System stores and timestamps records

4.2.3 Functional Requirements

- **REQ-4:** Faculty shall be able to add and update audit records
- **REQ-5:** Students shall be able to view their audit records

4.3 Audit Review and Tracking

4.3.1 Description and Priority

Enables auditors to review and track academic audits.

Priority: Medium

4.3.2 Functional Requirements

- **REQ-6:** Auditors shall review submitted audit data
- **REQ-7:** The system shall maintain an immutable audit log

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- System response time shall be under 2 seconds
- Support multiple concurrent users

5.2 Safety Requirements

- Prevent accidental data deletion
- Backup mechanisms for recovery

5.3 Security Requirements

- JWT-based authentication
- Encrypted password storage
- Role-based access control

5.4 Software Quality Attributes

- Usability: Easy to learn and use
- Reliability: Consistent data storage
- Scalability: Supports future expansion

5.5 Business Rules

- Only authorized users may modify audit records
- Students have read-only access

6. Other Requirements

- Database must support audit logging
- System must comply with institutional data policies

7. Project Timeline

Phase	Activities	Duration	Dates
Phase 1	Problem definition, requirement analysis, finalizing SRS	3 days	Feb 3 – Feb 5
Phase 2	System design (architecture, flow diagrams, database schema)	3 days	Feb 6 – Feb 8
Phase 3	Frontend development (React UI, routing, forms, dashboards)	6 days	Feb 9 – Feb 14

Phase 4	Backend development (Node.js, Express APIs, MongoDB models)	6 days	Feb 15 – Feb 20
Phase 5	Authentication & security implementation (JWT, RBAC, hashing)	3 days	Feb 21 – Feb 23
Phase 6	Integration & testing (frontend–backend integration, bug fixing)	3 days	Feb 24 – Feb 26
Phase 7	Documentation, screenshots, final review	2 days	Feb 27 – Feb 28

Appendix A: Glossary

- **Audit Log:** Record of system actions
- **JWT:** JSON Web Token
- **RBAC:** Role-Based Access Control

Appendix B: Analysis Models

- Use case diagrams
- Data flow diagrams (DFD)

Appendix C: To Be Determined List

- Integration with external academic systems
- Mobile application support