1. Introduction

1.1 Purpose

The purpose of this document is to define the functional and non-functional requirements for the Exam Paper Repository Web Application. This web application will enable students to upload, search, and retrieve university exam question papers. The system will ensure quality control, efficient searching, incentivization, and security while providing students with an optimized experience for accessing academic resources.

1.2 Document Conventions

This document follows standard software requirements specification formatting and numbering conventions, adhering to IEEE guidelines where applicable.

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- **Developers**: To understand the system requirements and implement the application.
- **Project Managers**: To track the project's scope and progress.
- **Stakeholders (University Administration, Faculty, and Students)**: To assess the system's functionalities.
- **Testers**: To validate the implementation against the defined requirements.

Readers should first review the **Introduction** section to understand the purpose and scope, then proceed to **Functional Requirements** for details on system behavior. Developers should pay close attention to **System Architecture** and **External Interfaces** sections.

1.4 Product Scope

This system will allow students to:

- Scan and upload exam question papers.
- Retrieve specific questions segmented from the question papers.
- Search question papers based on course code, year, and mid/end semester.
- Search individual questions based on topics.
- Earn credits for valid uploads and spend credits to access papers.
- Receive notifications when unavailable papers become available.
- Refer peers to gain additional benefits.

Admins will:

- Verify and approve/reject uploaded papers.
- Oversee the quality and readability of uploaded content.

- Manage users and their credit-based incentives.

The application will handle varying server loads, particularly increased usage during exams, and ensure secure login via college email authentication.

1.5 Definitions, Acronyms, and Abbreviations

- **OCR:** Optical Character Recognition
- **FR:** Functional Requirement
- **BRD:** Business Requirements Document
- **SRS:** Software Requirements Specification
- **DBMS:** Database Management System
- **AI Topic Classification:** AI-based classification system that tags questions with topics based on content analysis.

1.6 References

- IEEE 830-1998 SRS Standard
- Business Requirements Document (BRD)
- Security and privacy compliance guidelines
- Database schema documentation

1.7 Overview

The document outlines system features, constraints, assumptions, functional requirements, non-functional requirements, system architecture, user roles, system constraints and use cases.

2. Overall Description

2.1 Product Perspective

This is a standalone web-based application designed to assist students in accessing past exam question papers. The system must support role-based access (students and admins) and provide secure authentication.

2.2 Product Functions

- **Student Functions:**
  - Register/login using a verified college email.
  - Upload scanned question papers.
  - Earn credits for quality uploads.
  - Search for question papers based on metadata or keywords.
  - Redeem credits to access question papers.
  - Receive notifications about new uploads.

- **Admin Functions:**
    - Review and approve/reject uploaded question papers.
    - Maintain metadata accuracy.
    - Monitor and manage user credits.
    - Oversee system security and compliance.

2.3 User Classes and Characteristics

- **Students:** End-users who will upload and search for question papers.
- **Admins:** Responsible for verifying and approving question papers.
- **Developers:** Maintain and improve system functionality.

2.4 Operating Environment

- Web-based platform accessible via standard browsers.
- Cloud-hosted infrastructure ensuring scalability and reliability.

2.5 Design and Implementation Constraints

- Secure authentication required via college email verification.
- Mandatory course code input during search.
- Uploaded documents should be scanned for readability and metadata.
- System should handle high traffic during peak exam periods.
- Storage should be scalable and secure.

2.6 User Documentation

- User guides for uploading and searching papers.
- FAQs and support documentation.

2.7 Assumptions and Dependencies

- The system must handle high traffic during exam periods.
- Authentication is restricted to college-issued email IDs.
- Storage should be scalable and secure.
- Users must have internet access to use the platform.
- Scanned papers must be legible and correctly formatted.

3. System Features

3.1 Searching for Question Papers

- Students can search based on:
    - **Course Code (Mandatory Field)**

    - **Academic Year**
    - **Mid or End Semester**
- If unavailable, students can opt to receive a notification when it becomes available.

3.2 Question Segmentation and Topic-Based Search

- AI-driven segmentation of question papers.
- Topic-based search after selecting a course code.

3.3 Uploading and Verification Process

- Automated metadata extraction and duplicate detection.
- Readability analysis via OCR, with manual verification if necessary.
- First-time uploads grant additional credits to the uploader.

3.4 Credit-Based System

- Users gain credits for good uploads.
- Users lose credits for spam/incomplete uploads.
- Credits required to access papers.

3.5 Recommendation & Notification System

- Suggests similar question papers.
- Alerts users when requested papers become available.

3.6 Referral Program

- Users earn credits by referring new users.

3.7 Server & Security Management

- Load balancing to handle peak traffic.
- Secure authentication via college email.
- Data encryption and secure storage.

4. Specific Requirements

4.1 Functional Requirements

ID	Requirement Description	Priority
FR-01	Students should register/login using college email	High

FR-02	Users should be able to upload scanned papers	High
FR-03	System should extract metadata automatically	High
FR-04	Admins should verify readability and uniqueness	High
FR-05	Users earn credits for valid uploads	High
FR-06	Users should be able to retrieve specific questions	High
FR-07	Recommendation system for related papers	Medium
FR-08	Notifications for missing papers	Medium
FR-09	Referral system for user engagement	Medium
FR-10	Secure storage and retrieval system	High

4.2 Non-Functional Requirements

ID	Requirement Description	Priority
NFR-01	System should support 10,000+ concurrent users	High
NFR-02	Data should be encrypted for secure storage	High
NFR-03	Response time should be <2 seconds for searches	Medium
NFR-04	Application should be mobile-friendly	Medium
NFR-05	System should auto-scale based on traffic	High

5. System Architecture

- Defines how different components interact and integrate.
- Comprises frontend, backend, database, and cloud storage.
- Uses load balancing techniques to handle peak traffic.

6. External Interface Requirements

6.1 User Interfaces

- **Student Dashboard:** Upload papers, search, and manage credits.
- **Admin Panel:** Approve/reject papers, monitor user activities.

6.2 Hardware Interfaces

- Web-based system accessible via standard browsers.
- Compatible with desktop, tablet, and mobile devices.

6.3 Software Interfaces

- **Frontend:** React.js / Vue.js / Angular.
- **Backend:** Node.js / Django.
- **Database:** PostgreSQL / MongoDB.
- **Storage:** AWS S3 / Google Cloud Storage.
- **Authentication:** OAuth 2.0.
- **AI Topic Classification Module:** Python-based NLP model
- **Load Balancing:** Cloud-based auto-scaling infrastructure

7. Data Requirements

7.1 Storage Requirements

- Store metadata such as course code, semester type, and year.
- Store full question papers in PDF/JPG formats.
- Store segmented questions with mapped topics.

7.2 Data Processing

- OCR processing for extracting metadata from scanned question papers.
- AI classification model for tagging questions with topics.

8. Other Requirements

- Cloud-based storage for high availability.
- Integration with AI-based classification models.

9. Use Cases

Use Case 1: Uploading Question Paper

Actors: Student, System, Admin

1. Student uploads a scanned question paper.

2. System extracts metadata and checks for duplicates.
3. If duplicate exists, student gets partial credits, and the upload is rejected.
4. If unique, the system checks readability using OCR.
5. If readable, the paper is stored, and the student earns credits.
6. If unreadable, the admin manually verifies and either approves or rejects it
7. Student earns/loses credits based on quality.

Use Case 2: Searching for a Question

Actors: Student, System

1. Student searches for a question paper using course code (mandatory), year, and semester.
2. If a paper is found, it is displayed with an option to refine search by topic.
3. If no paper is found, the student can opt to receive a notification when it becomes available.
4. Student redeems credits to view/download papers.

Use Case 3: Referral System

Actors: Student, System

1. Student refers a new user.
2. Referred user signs up and uploads papers.
3. Referrer earns credits for successful referral.

10. Conclusion

This SRS defines the technical and functional specifications for the **Exam Paper Repository Web Application**. The system will streamline access to past exam papers, enhance academic collaboration, and ensure scalability and security for seamless student engagement.

11. Appendices

- API documentation reference.
- Compliance and security policies.
- Performance benchmarks.
- Finalized UI design.
- Deployment and maintenance plan.