

SRS Document:

Purpose

The purpose of this Software Requirements Specification (SRS) document is to define the functional and non-functional requirements for extending an existing React Mini Project into a full-stack application by integrating a Node.js backend. This document serves as a reference for developers, evaluators, and stakeholders to understand the system behavior, constraints, and implementation expectations.

Marks Card Upload Feature integrated into the existing Node.js + Express backend. The feature enables secure upload, validation, and storage of marks cards

Scope

This project transforms a frontend-only React application into a full-stack web application by integrating a Node.js backend with MongoDB Atlas for persistent data storage.

All client-side static data, mock logic, and LocalStorage usage are replaced with RESTful APIs. The backend manages all Create, Read, Update, and Delete (CRUD) operations, while the React frontend consumes these APIs asynchronously.

The feature extends the current backend API with a single endpoint for multi-file uploads, including comprehensive validation, secure storage, and detailed JSON responses.

Functional Requirements

Backend API Development

The system shall provide a Node.js backend within the existing project.

The backend shall expose RESTful APIs for all data operations.

The backend shall connect to MongoDB Atlas for persistent data storage.

The backend shall accept and return data in JSON format.

The backend shall handle validation and send proper error responses.

The system shall allow creating new records using a POST API.

The system shall allow retrieving all records using a GET API.

The system shall allow updating existing records using a PUT API.

The system shall allow deleting records using a DELETE API.

Accept marks cards (PDF/JPG/PNG) in a single HTTP request, validate files, store securely with random filenames, and return detailed file information.

Key Capabilities:

- Multi-file upload via `marksheets[]` array field
- File type validation (PDF, JPG, PNG only)
- Individual file size limit ($\leq 2\text{MB}$ per file)
- Maximum 8 files per request
- Secure random filename generation
- Comprehensive error handling with HTTP status codes

Endpoint Details:

Method: `POST`

URL: `/api/students/upload-marksheets`

Content-Type: `multipart/form-data`

Non-Functional Requirements

- The system should provide fast API responses and handle multiple requests efficiently.
- Input validation must be implemented, and sensitive data should be stored using environment variables.
- The backend should handle errors properly and return meaningful messages for failures.
- The application should follow a modular structure with clear separation between frontend and backend.
- The system should be scalable to support future features like authentication without major changes.

Conclusion

- This SRS explains the requirements for converting a React-only project into a full stack application.
- The project focuses on proper API usage, backend integration, and clean separation of frontend and backend.
- The original project design and domain are preserved.