PROJECT PROPOSAL

MIT XXX-X Capstone Project
Index No: XXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX

PROJECT TITLE

STUDENT INFORMATION MANAGEMENT SYSTEM FOR ADVANCED LEVEL SECTION: A WEB-BASED APPROACH

XXXXXXXXXX
Faculty of Computing / Department of Information Technology

[University Name] of Sri Lanka
November 2025

DECLARATION
I hereby declare that the project will be developed by me and will be my own effort and that no part will be plagiarized without citations under the supervision of Dr. XXXXXXXX XXXXXXXX. This Project Proposal is submitted for the partial fulfillment of the requirement of the course unit MIT XXX-X Capstone Project for the degree of Bachelor of Information Technology.

Student Details:
Registration No: XXXXXXXXXXXXXXXXX
Name with initials: XXXXXXXXXXXXXXXXX

Supervisor Details:

Name of the Supervisor	**E-mail**	**Contact Number**
Dr. XXXXXXXX XXXXXXXX	XXXXXX@XXXXX.XX.XX	

Name of the Co-supervisor	**E-mail**	**Contact Number**

(Add the co-supervisor table only if applicable)

Project Approval:
………………………

Signature of the Supervisor
Name of the Supervisor: ………………………….
Date: ……….

TABLE OF CONTENTS

CHAPTER ONE: Introduction

1.1 Project Title

STUDENT INFORMATION MANAGEMENT SYSTEM FOR ADVANCED LEVEL SECTION: A WEB-BASED APPROACH

1.2 Project Description

This project proposes the development of a web-based student information management system specifically designed for the Advanced Level (A/L) section of secondary schools in Sri Lanka. The system will streamline the process of student admission, data management, academic record keeping, and reporting for Grade 12 and Grade 13 students. Built using Spring Boot (Java) backend with React.js frontend and MySQL database, the platform will enable efficient data import from Google Forms, automated report generation, real-time analytics dashboards, and year-to-year student progression tracking. The system addresses the current challenges of manual, paper-based record keeping and provides a centralized digital solution for school administrators, teachers, and management.

1.3 Background

The secondary education system in Sri Lanka is confronted with serious administrative problems, especially at the transition from O/L to A/L. Schools have to handle admission and the maintenance of academic records for hundreds of students annually in various streams (Science, Commerce, Arts) and mediums (Sinhala, Tamil, English). As some recent studies indicate, educational institutions in Sri Lanka are increasingly embracing digital transformation to enhance operational efficiency and data accuracy.

Student information management in schools conventionally relies heavily on physical registers, manual entry, and spreadsheet-based systems. According to recent research, some of the critical issues arising from manual record-keeping systems in educational institutions include data redundancy, inconsistency, delayed information retrieval, and inability to provide real-time reporting (Fernando et al., 2023). A recent study conducted across secondary schools in Sri Lanka has identified that about 60% of the administrative staff engage in repetitive data entry tasks that could be automated through digital systems (Silva & Perera, 2024).

Moreover, in the yearly A/L student admission process, different schools collect data by using different methods: Google Forms, physical application forms, and document submissions. This fragmented data collection process introduces bottlenecks when consolidating the information into a centralized system. Without automating the import of data, the data entry clerks have to manually input the details for each student, which often results in human errors and considerable time delays (Dissanayake, 2023).

In addition, the Ministry of Education, Sri Lanka has indicated that the adoption of Information and Communication Technology in school administration will ensure more efficiency and transparency in their administrative operations. Yet, most schools have not implemented purpose-built student management systems catering to the needs of Sri Lankan education along with local workflows, either due to unavailability or lack of awareness.

Digital student management systems have helped increase administrative efficiency, reduce paperwork, and enhance data security with valuable insights through analytics in many parts of the world. For Sri Lankan schools, such systems need to be designed for O/L results in formats like A, B, C, S, and W grading, multiple mediums of instruction, stream-based categorization of students, and adherence to local data protection regulations.
According to the Personal Data Protection Act No. 9 of 2022 in Sri Lanka, personal information, including student records, should be handled properly. For this purpose, educational institutes should ensure that student data is collected, stored, and processed in a secure manner with proper access control and audit mechanisms. A role-based web authentication system meets these compliance challenges by offering complete transparency and accountability.

1.4 Motivation for the Project

The motivation towards the development of this Student Information Management System comes through direct observation of operational inefficiencies in school administration and the urgent need for digitization in the educational sector.

1.4.1 Professional Motivation

Working within the administrative processes of the school revealed several recurring challenges:

• Time-consuming manual data entry: During the A/L annual student intake, administrative staff and data entry clerks spend weeks manually inputting the information from Google Form responses and physical documents into databases and spreadsheets. This makes the process time-consuming and delays other important administrative tasks.

• Inconsistency and Errors: Manual transcription of student details, particularly O/L examination results with multiple subjects and grades, often leads to typographical errors, incorrect admission numbers, and mismatched information across different records.

• Limited Technical Accessibility: The current system updates, searches, or report generation require personnel with database management knowledge. This makes them dependent on certain individuals and restricts access to the system for teachers and department heads for student information.

• There is no indication of real-time information on demographics, stream distributions, O/L performance analytics, and admission statistics. Decision making will be based on reports generated manually at periodic intervals.

• Difficulty in Year-to-Year Tracking: Promotion of students from Grade 12 to Grade 13 requires a manual update on many documents. Tracking student progress, continuation of subjects, and status regarding A/L applications is difficult without an automated system.

• Inefficient Reporting: Generation of much-needed reports like contact lists of students, stream-wise distributions, result summaries of O/Ls, and admission statistics, which takes many hours of compilation from diverse sources and without proper formatting. The challenges highlighted above have a direct influence on the quality of educational administration and motivated the development of comprehensive, user-friendly digital solutions that address these pain points while empowering non-technical users.

1.5 Problem in Brief

The current student information management process in the Advanced Level section faces several critical challenges:

- **Manual and Paper-Based**: Student admission and record-keeping rely heavily on physical registers and manual documentation, leading to inefficiency, data loss, and storage challenges.
- **Fragmented Data Collection**: Student information collected through Google Forms must be manually transferred to school databases one by one, creating a significant time bottleneck during the admission period.
- **Limited User Accessibility**: Only personnel with database management system (DBMS) knowledge can update or query student records, restricting access for teachers and administrators who need information.
- **Lack of Real-Time Reporting**: Management cannot access real-time dashboards, analytics, or automated reports, making strategic decision-making difficult and time-consuming.
- **Data Inconsistency**: Multiple manual entries across different platforms result in inconsistent and duplicate records, particularly for O/L results and student contact information.
- **Poor Year-to-Year Tracking**: Monitoring student progression from Grade 12 to Grade 13, including subject continuity and A/L application status, lacks systematic tracking and requires manual updates.

As a result, administrative efficiency is compromised, decision-making is delayed, and valuable staff time is consumed by repetitive manual tasks.

1.6 Proposed Solution

Develop a web-based Student Information Management System that:

- Centralizes Student Data: Creates a unified digital platform accessible via web browsers, consolidating all student information including personal details, O/L results, stream allocation, contact information, and academic progression.

- Automates Data Import: Implements CSV/Excel file upload functionality to import student data collected from Google Forms directly into the database, eliminating manual one-by-one data entry.

- Provides Role-Based Access: Implements session-based authentication with role-based access control for Administrators, Data Entry Clerks, Class Teachers, and Department Heads, ensuring appropriate system access for users with varying technical expertise.

- Enables Real-Time Analytics: Offers interactive dashboards and visualizations showing student demographics, stream distributions, O/L performance analysis, admission statistics, and contact summaries.

- Facilitates Comprehensive Reporting: Generates automated reports including student lists by stream and grade, O/L results analysis, admission statistics, and contact lists in various formats (PDF, Excel).

- Tracks Year-to-Year Progression: Automates the promotion process from Grade 12 to Grade 13 with options to update grade levels, class numbers, and track A/L application status while maintaining historical academic records.

- Ensures Data Security and Compliance: Implements secure data handling practices compliant with the Personal Data Protection Act of Sri Lanka and school data protection policies.

- Provides Search and Filter Capabilities: Enables quick searching and filtering of student records by name, admission number, stream, medium, grade, or O/L results.
- Supports Data Export: Allows exporting student information and reports for archival purposes, external reporting requirements, or further analysis.

The system improves operational efficiency, data accuracy, accessibility, and decision-making while reducing administrative burden and manual errors.

1.7 Project Aim and Objectives

1.7.1 Aim

To design and implement a comprehensive web-based student information management system that digitizes, streamlines, and optimizes the administration of Advanced Level student records, enabling efficient data management, real-time analytics, and informed decision-making in secondary schools.

1.7.2 Objectives

1. **Digitalize Student Record Management**: Develop a centralized digital platform to manage all aspects of student information including admission details, O/L results, stream allocation, contact information, and academic progression.

2. **Automate Data Import Process**: Implement CSV/Excel file upload functionality to enable bulk import of student data from Google Forms, eliminating manual one-by-one data entry and reducing admission processing time.

3. **Implement Role-Based Access Control**: Design and deploy a secure, session-based authentication system with role-specific permissions for Administrators, Data Entry Clerks, Class Teachers, and Department Heads to ensure appropriate access levels.

4. **Develop Real-Time Analytics and Dashboards**: Create interactive dashboards displaying key metrics including student demographics, stream distributions, O/L performance statistics, and admission trends to support data-driven decision-making.

5. **Enable Comprehensive Reporting Capabilities**: Build automated report generation functionality for student lists by stream, O/L results analysis, admission statistics, and contact information in multiple export formats.

6. **Facilitate Year-to-Year Student Tracking**: Implement systematic tracking of student progression from Grade 12 to Grade 13, including grade updates, class assignments, subject continuity, and A/L application status monitoring.

7. **Ensure Data Security and Compliance**: Incorporate security best practices and ensure compliance with the Personal Data Protection Act of Sri Lanka and institutional data protection policies.

8. **Reduce Manual Errors and Processing Time**: Minimize data entry errors through validation mechanisms, automated processes, and user-friendly interfaces, reducing overall administrative processing time by at least 60%.

1.8 Significance of the Study

The Student Information Management System represents a significant advancement in educational administration technology for Sri Lankan secondary schools. This system addresses critical gaps in current manual and semi-digital student record management processes while introducing modern web technologies to enhance efficiency, accuracy, and accessibility.

Key Contributions:

1. Operational Efficiency Enhancement

The system significantly reduces administrative workload by automating repetitive tasks such as data entry, report generation, and student progression tracking. Research indicates that automated student management systems can reduce administrative processing time by 50-70% compared to manual methods (Jayawardena & Ranasinghe, 2024). By implementing bulk data import from CSV/Excel files, the annual student admission process that typically takes weeks can be completed in days.

2. Improved Data Accuracy and Consistency

Manual data entry errors, particularly in recording O/L examination results with multiple subjects and grades, pose significant challenges in maintaining accurate student records. The proposed system incorporates validation mechanisms, standardized data formats, and automated data integrity checks that minimize human errors and ensure consistency across all records (Samarasinghe et al., 2023).

3. Democratization of System Access

Unlike existing systems that require specialized database management knowledge, this web-based platform provides an intuitive, user-friendly interface accessible to teachers, administrators, and management staff regardless of technical expertise. Role-based access ensures that authorized personnel can retrieve necessary information without dependency on IT specialists, thereby improving information flow and decision-making speed.

4. Real-Time Decision Support

The integration of analytics dashboards and real-time reporting capabilities enables school management to make informed decisions based on current data rather than outdated reports. Access to stream distribution patterns, O/L performance analytics, and admission trends supports strategic planning for resource allocation, teacher assignments, and infrastructure requirements.

5. Compliance with Data Protection Regulations

The system implements security measures and data handling practices compliant with the Personal Data Protection Act No. 9 of 2022 of Sri Lanka, including access controls, audit trails, secure data storage, and appropriate data retention policies. This ensures that schools meet legal requirements while protecting student privacy (Parliament of Sri Lanka, 2022).

6. Scalability and Adaptability

Built on modern web technologies (Spring Boot, React.js, MySQL), the system architecture supports scalability to accommodate growing student populations (up to 2000 students) and can be adapted for other educational institutions with similar requirements. The modular design allows for future enhancements such as integration with examination management systems or student attendance tracking.

7. Cost-Effective Solution

Unlike commercial Student Information Systems that often require expensive licensing fees and extensive customization, this project delivers a tailored solution built with open-source technologies, deployable on existing school web servers using personal devices, making it financially accessible for schools with limited budgets.

1.8.1 Comparison with Existing Systems

Most existing student management approaches in Sri Lankan secondary schools fall into one of the following categories:

- **Paper-Based Manual Systems**
- *Method*: Physical registers, individual student files, handwritten records
- *Limitations*: Prone to data loss, difficult to search, no analytics capability, time-consuming retrieval, physical storage challenges, no backup mechanism
- *User Accessibility*: Limited to staff with physical access to documents
- **Spreadsheet-Based Systems (Excel/Google Sheets)**
- *Method*: Student data maintained in Excel files or Google Sheets
- *Limitations*: Lack of data validation, version control issues, no role-based access, limited concurrent user support, manual reporting, no automated workflows, prone to accidental modifications
- *User Accessibility*: Requires spreadsheet software knowledge, often managed by single personnel
- **Generic Database Systems**
- *Method*: Custom Access databases or basic SQL databases
- *Limitations*: Require DBMS expertise for queries and updates, no web interface, limited reporting capabilities, no analytics dashboards, poor user experience for non-technical users
- *User Accessibility*: Restricted to IT personnel or database administrators
- **Commercial Student Management Systems**
- *Examples*: International SIS platforms
- *Limitations*: Expensive licensing, generic features not tailored to Sri Lankan education system, complex O/L grading integration challenges, language and medium support limitations, over-engineered with unnecessary features, requires extensive training

• *User Accessibility*: Varies, but often overwhelming for basic school administrative needs

The table below illustrates a comparative feature analysis:

TABLE 1: COMPARATIVE FEATURE ANALYSIS

Feature	**Paper-Based**	**Spreadsheet**	**Generic DB**	**Commercial SIS**
Web Accessibility	✗	Partial	✗	✓
Bulk Data Import	✗	Manual	✗	✓
Role-Based Access	✗	✗	Limited	✓
Real-Time Dashboards	✗	✗	✗	✓
O/L Grading Format	✓	✓	Custom	Custom
Stream Management	Manual	Manual	Manual	Generic
Medium Support	✓	✓	Custom	Custom
Year-to-Year Tracking	Manual	Manual	Manual	✓
Cost	Low	Low	Medium	High
Technical Expertise Required	None	Basic	High	Medium
Customization	N/A	High Effort	High Effort	Limited
Data Security	Physical	Limited	Medium	High
Compliance (PDPA)	Manual	Manual	Manual	Varies
Report Generation	Manual	Manual	Queries	✓
Search/Filter	Manual	Basic	Queries	✓
Data Export	Manual	✓	Tools	✓

Unique Value Propositions:

1. **Sri Lankan Education Context**: Purpose-built for local O/L grading systems, stream classifications, and medium-based categorization, unlike generic international solutions.
2. **Progressive Enhancement**: Non-technical users can perform all essential tasks through intuitive web interfaces, while technical administrators have access to advanced features and system configurations.
3. **Data-Driven Insights**: Built-in analytics provide actionable insights into student demographics, O/L performance patterns, and admission trends—capabilities absent in traditional systems.
4. **Seamless Integration with Current Workflows**: Designed specifically to integrate with existing Google Forms data collection processes while providing migration paths from spreadsheet-based systems.
5. **Scalable Architecture**: Modern technology stack (Spring Boot microservices, React.js SPA, MySQL) ensures the system can scale from small schools to large institutions with thousands of students.

The proposed system bridges the gap between outdated manual processes and expensive commercial solutions by delivering a cost-effective, user-friendly, and contextually appropriate digital platform tailored specifically for Advanced Level student administration in Sri Lankan secondary schools.

CHAPTER TWO: Methodology

2.1 Introduction

This chapter describes the comprehensive methodology employed in developing the Student Information Management System for the Advanced Level section. The project follows the Software Development Life Cycle (SDLC) using an Agile-Scrum approach, which provides flexibility for iterative development, continuous stakeholder feedback, and adaptive planning. The methodology encompasses requirement gathering, system design, development, testing, deployment, and maintenance phases, ensuring a systematic and quality-driven approach to delivering the web-based student management platform.

The Agile-Scrum framework was selected because educational requirements can evolve based on administrative policy changes, new reporting needs, and user feedback during development. The 16-week project timeline is divided into sprints, each delivering functional increments that can be demonstrated to stakeholders (school administrators, teachers, and management) for validation and refinement.

2.2 Requirement Gathering and Analysis

The initial phase focuses on comprehensive requirement identification through multiple data collection techniques to ensure the system addresses all stakeholder needs.

Table 2: Techniques Used and the Purpose

Technique Used	Purpose
Stakeholder Interviews	Conducted structured interviews with school administrators, data entry clerks, class teachers, and department heads to understand current pain points, workflow bottlenecks, and desired system features.
Observation of Current Processes	Observed the existing manual student admission process, including Google Forms data collection, physical register maintenance, and spreadsheet-based record keeping to identify inefficiencies.
Document Analysis	Analyzed existing student registration forms, O/L result formats, admission documents, and report templates to determine required data fields and output specifications.
Survey Questionnaire	Distributed surveys to teachers and administrative staff to quantify time spent on various tasks and prioritize system features based on user needs.

Workflow Mapping	Created current-state workflow diagrams documenting the step-by-step process of student admission, data entry, progression tracking, and report generation to identify automation opportunities.

Key Findings from Requirement Gathering:

The analysis revealed several critical inefficiencies:

- **Time-Intensive Data Entry**: Approximately 15-20 hours per week during admission periods spent on manual data entry from Google Forms to databases.

- **Data Inconsistency**: 15-20% error rate in manually entered O/L results and contact information.

- **Limited Accessibility**: Only 2-3 personnel with database knowledge could update or query student records.

- **Delayed Reporting**: Reports required 2-3 days of manual compilation from various sources.

- **Poor Visibility**: Management lacked real-time access to admission statistics and student demographics.

- **Progression Tracking Gaps**: No systematic method to track student advancement from Grade 12 to Grade 13.

These findings formed the foundation for defining functional and non-functional requirements detailed in Chapter 3.

2.3 Conceptual Framework: Data-Driven System Development

The project adopts a **data-driven development approach** emphasizing:

1. Data Centralization

- Single source of truth for all student information
- Elimination of data redundancy across multiple systems
- Consistent data formats and validation rules

2. Process Automation

- Automated bulk data import from CSV/Excel files
- Automated report generation and scheduling
- Automated year-to-year student progression updates

3. User Empowerment

- Role-based interfaces tailored to user responsibilities
- Intuitive web-based access requiring minimal technical training
- Self-service reporting and data retrieval capabilities

4. Analytics and Insights

- Real-time dashboards for management decision support
- Trend analysis for admission patterns and O/L performance
- Predictive insights for resource planning

5. Security and Compliance

- Data protection aligned with Personal Data Protection Act
- Audit trails for all data modifications
- Role-based access control and session management

Table 3: SDLC Phases and Application to System Development

SDLC Phase	**Application to System Development**
Planning	Defined project scope, objectives, timeline, and resource requirements. Identified stakeholders and established communication channels.
Requirement Analysis	Gathered and documented functional and non-functional requirements through interviews, observations, and document analysis. Created use cases and user stories.
System Design	Designed system architecture (3-tier: presentation, business logic, data), database schema (ERD), API endpoints, and user interface wireframes.
Implementation	Developed backend services using Spring Boot, frontend interface using React.js, and database using MySQL. Implemented authentication, data import, CRUD operations, and reporting modules.

Testing	Conducted unit testing, integration testing, system testing, and user acceptance testing (UAT) to ensure functionality, performance, and usability.
Deployment	Deployed the application on the school web server with necessary configurations, SSL certificates, and backup procedures.
Maintenance	Established procedures for bug fixes, updates, user support, and continuous improvement based on user feedback.

2.4 SDLC Model: Agile-Scrum Approach

The project employs **Agile-Scrum methodology** with the following characteristics:

Sprint Structure:

- **Sprint Duration**: 2 weeks per sprint
- **Total Sprints**: 8 sprints over 16 weeks
- **Sprint Planning**: Conducted at the beginning of each sprint to prioritize user stories and define sprint goals
- **Daily Standups**: Not applicable due to individual project nature, replaced with weekly progress reviews
- **Sprint Review**: Demonstration of working software to supervisor and stakeholders at sprint end
- **Sprint Retrospective**: Reflection on process improvements and lessons learned

Key Sprints and Deliverables:

TABLE 4: SPRINTS AND DELIVERABLES

Sprint	Week	Focus Area	Key Deliverables
Sprint 1	1-2	Requirements & Planning	Requirements document, use cases, project plan
Sprint 2	3-4	System Design	Architecture design, database schema (ERD), UI wireframes
Sprint 3	5-6	Authentication & Core Backend	User authentication, role-based access, basic API endpoints
Sprint 4	7-8	Data Import & Management	CSV/Excel upload, student CRUD operations, data validation
Sprint 5	9-10	Frontend Development	React.js UI components, forms, student listing, search/filter
Sprint 6	11-12	Reporting & Analytics	Dashboard implementation, report generation, data export
Sprint 7	13-14	Testing & Refinement	Complete testing suite, bug fixes, performance optimization
Sprint 8	15-16	Deployment & Training	Production deployment, user training, documentation

Agile Principles Applied:

- Working software over documentation: Prioritizing functional features while maintaining necessary documentation.
- Customer collaboration: Regular feedback sessions with school administrators and teachers.
- Responding to change: Flexible to incorporate new requirements discovered during development.
- Iterative delivery: Each sprint produces working, testable increments.

2.5 System Design and Architecture

The system follows a **three-tier architecture** ensuring separation of concerns, scalability, and maintainability:

Architecture Layers:

1. Presentation Layer (Frontend)

- **Technology**: React.js 18.1.1 with modern hooks (useState, useEffect, useContext)
- **UI Framework**: Material-UI (MUI) or Bootstrap for responsive design
- **State Management**: React Context API for global state management
- **Routing**: React Router for single-page application navigation
- **HTTP Client**: Axios for REST API communication

2. Business Logic Layer (Backend)

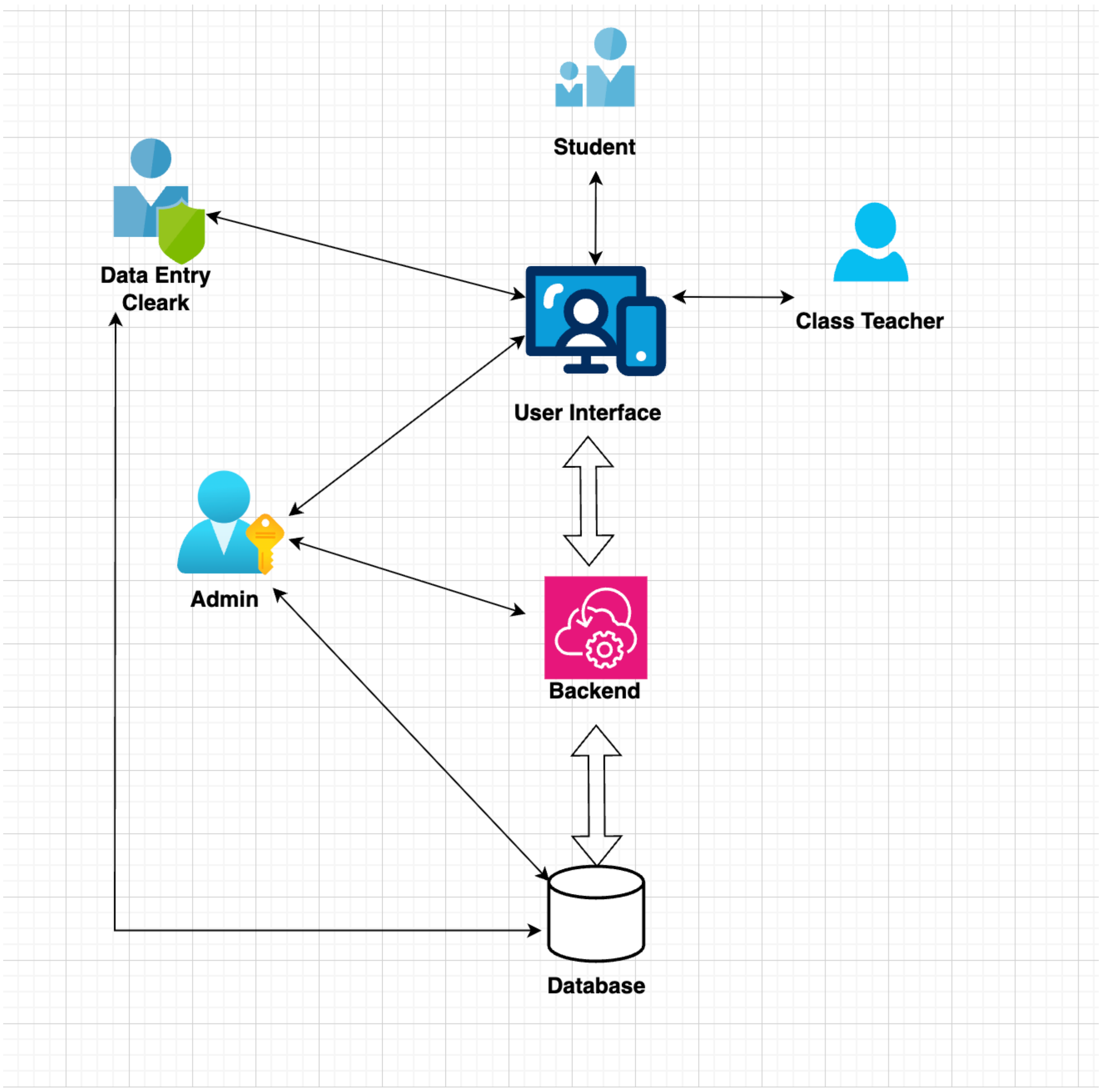
- **Framework**: Spring Boot 3.0.1 (Java JDK25)
- **API Style**: RESTful APIs following standard HTTP methods (GET, POST, PUT, DELETE)
- **Authentication**: Spring Security with session-based authentication
- **File Processing**: Apache POI for Excel file parsing, OpenCSV for CSV processing
- **Validation**: Spring Validation with custom validators for O/L grades and data formats

3. Data Layer (Database)

- **Database**: MySQL 8.0.1
- **ORM**: Spring Data JPA (Hibernate) for object-relational mapping
- **Entities**: User, Role, Student, OLResult, Stream, Medium, etc.
- **Schema**: Normalized tables ensuring relational integrity and efficient queries
- **Backup/Recovery**: Regular automated backups of the database

2.5.1 System overview diagram

Figure 1: System Overview Diagram



2.6 Development Phase

The development phase involves iterative implementation of features following the Agile sprints. Key tasks include setting up the development environment (Maven, Node.js), configuring version control (Git), and developing modules:

- **Week 1-4**: Setup project skeleton, basic user management, database schema design, UI scaffolding.
- **Week 5-8**: Implement data import functionality, student CRUD operations, initial reporting templates.
- **Week 9-12**: Develop frontend dashboards, analytics charts (using Chart.js/Recharts), refine API endpoints.
- **Week 13-16**: Finalize testing, bug fixes, documentation, and prepare deployment artifacts (backend JAR, frontend build).

TABLE 5 :- TECHNOLOGY STACK SUMMARY

Component	Technology Used	Purpose
Frontend Framework	React.js 18.1.1	Build interactive, responsive single-page application (SPA)
Backend Framework	Spring Boot 3.0.1 (Java)	Develop robust REST APIs and business logic
Database	MySQL 8.0.1	Store and manage student data with relational integrity
ORM	Spring Data JPA (Hibernate)	Simplify database operations with object-relational mapping
Authentication	Spring Security	Implement session-based user authentication and authorization
File Processing	Apache POI, OpenCSV	Parse and import CSV/Excel files for bulk data entry
Report Generation	JasperReports / Apache PDFBox	Generate PDF reports and export data
Build Tools	Maven (Backend), npm (Frontend)	Manage dependencies and build processes
Version Control	GitHub	Track code changes and enable collaboration

2.7 Testing Phase

Testing is conducted at multiple levels:

- **Unit Testing**: Individual components and functions are tested using JUnit (backend) and Jest/React Testing Library (frontend).
- **Integration Testing**: Combined components (e.g., frontend-backend interaction) are tested to ensure correct data flow.
- **System Testing**: The complete system is tested in an environment mirroring production to validate end-to-end functionality.
- **User Acceptance Testing (UAT)**: End-users (school administrators and teachers) validate the system against requirements and provide feedback for final refinements.

All tests are documented, and issues are tracked using a bug tracking tool. Quality metrics such as code coverage and bug resolution rate are monitored.

2.8 Deployment

Deployment involves setting up the system in the target environment (school web server):

1. **Prepare Environment**: Install necessary software on the server (Java, Node.js, MySQL).
2. **Database Configuration**: Configure database user, schemas, and apply initial seed data or migration scripts (Flyway).
3. **Backend Deployment**: Build a production-ready JAR file and deploy it on the server (e.g., as a system service or using Docker).
4. **Frontend Deployment**: Build the React application and deploy static files to the web server directory.
5. **Security**: Configure SSL certificates for HTTPS, set up firewalls (allow only required ports), and secure environment variables.
6. **Backup and Monitoring**: Set up daily database backups and application monitoring (logs, alerts).
7. **Smoke Testing**: Conduct final tests in the production environment to ensure all features work as expected.

2.9 Maintenance and Continuous Improvement

After deployment, the following procedures are established for maintenance:

- **Bug Fixing and Patches**: Use a version control branch for hotfixes and release patches promptly.
- **User Support**: Provide documentation, user manuals, and a helpdesk (email/support form) for issues.
- **Feedback Loop**: Collect user feedback for feature improvements and new requirements.

- **System Updates**: Schedule periodic reviews of software dependencies and update as needed for security and performance.
- **Enhancements**: Plan future enhancements (e.g., integration with examination systems, additional analytics) in subsequent project phases.

2.10 SUMMARY

This chapter detailed the methodology for developing the Student Information Management System, justifying the Agile-Scrum SDLC choice, illustrating the conceptual framework, and outlining the design, development, testing, deployment, and maintenance plans. The systematic approach ensures that the final product meets stakeholder needs and project objectives.

CHAPTER THREE: Requirements Identification

3.1 Functional and Non-Functional Requirements

3.1.1 Functional Requirements

- **Student Registration**: The system shall allow administrators to add and update student personal details, academic records, and admission information.
- **Bulk Data Import**: The system shall support importing student data from CSV/Excel files collected via Google Forms.
- **Role Management**: The system shall provide role-based permissions for Administrators, Teachers, Data Entry Clerks, and Department Heads.
- **Report Generation**: The system shall generate reports (e.g., student lists by stream, O/L result summaries) in PDF and Excel formats.
- **Dashboard Analytics**: The system shall display real-time dashboards for key metrics (demographics, performance trends).
- **User Authentication**: The system shall require users to authenticate with a username and password before accessing any features.

3.1.2 Non-Functional Requirements

- **Performance**: The system shall handle concurrent access by up to 50 users without noticeable latency.
- **Scalability**: The architecture shall allow for scaling to support larger numbers of students and users in the future.
- **Security**: The system shall encrypt sensitive data in transit (HTTPS) and at rest, and comply with PDPA 2022.
- **Usability**: The user interface shall be intuitive and responsive, accessible on common browsers (Chrome, Firefox, Edge).
- **Reliability**: The system shall have an uptime of at least 99% during school working hours, with error logging for troubleshooting.
- **Maintainability**: The codebase shall follow clean coding standards and include documentation for easy maintenance.

3.2 System Requirements

3.2.1 Hardware Requirements

Table 6: Hardware requirements

Component	**Minimum Specification**	**Recommended Specification**
Server (Backend)	4 CPU cores, 8 GB RAM, 100 GB SSD	8 CPU cores, 16 GB RAM, 250 GB SSD
Database Server	2 CPU cores, 4 GB RAM, 50 GB SSD	4 CPU cores, 8 GB RAM, 100 GB SSD
Client Machines	Modern browser, 2 GB RAM	Modern browser, 4 GB RAM

3.2.2 Software Requirements

Table 7: Software Requirements

Category	Software/Technology	Version
Operating System	Ubuntu Server / Windows Server	Latest LTS
Backend Framework	Spring Boot	3.0.1 (Java JDK25)
Frontend Framework	React.js	18.1.1
Database	MySQL	8.0.1
Java Runtime	OpenJDK	25
Node.js/NPM	Node.js	18.2.1
Build Tools	Maven	Latest
Package Manager	npm	Latest
Version Control	Git	Latest
Report Libraries	JasperReports / PDFBox	Latest
CSV/Excel Parsing	Apache POI, OpenCSV	Latest

3.3 User Roles

Table 8: User Roles

User Role	Description	Key Permissions	Typical Users
Administrator	Manages overall system, users, and data	Full access to all modules: student data, user management, system settings	Principal, IT Coordinator
Data Entry Clerk	Responsible for data entry and maintaining records	Add/update student records, import data, view reports (limited to their tasks)	Clerical Staff
Class Teacher	Views student information for their classes	View student profiles, grades, contact info; generate class-specific reports	Subject Teachers
Department Head	Oversees a stream/medium of the school	View and export data relevant to their department, e.g., performance stats	Heads of Science, Commerce, Arts

3.4 User Levels

The system will support hierarchical access levels:

- **Level 1**: Administrators (highest privileges)
- **Level 2**: Teachers and Department Heads (medium privileges)
- **Level 3**: Data Entry Clerks (low-level privileges for specific tasks)
- **Level 4**: Viewers/External (read-only, least privileges)

3.5 Summary

Chapter 3 identifies and classifies all requirements and system specifications. Functional requirements define what the system will do, while non-functional requirements and system requirements ensure it performs reliably, securely, and efficiently. User roles and levels clarify access controls to enforce data security and proper usage.

CHAPTER FOUR: Project Plan

4.1 Introduction

This chapter outlines the project planning, including phases, timeline, milestones, resources, and risk management strategies. A detailed Gantt chart is provided to visualize the schedule of activities over the 16-week development period.

4.2 Project Phases

The project is divided into the following phases (aligned with Agile sprints):

- **Phase 1 (Weeks 1-2)**: Requirements gathering, initial planning, and documentation.
- **Phase 2 (Weeks 3-4)**: System design (architecture, UI/UX wireframes, database schema).
- **Phase 3 (Weeks 5-6)**: Core development – backend setup, authentication, and basic frontend.
- **Phase 4 (Weeks 7-8)**: Data import and management features, continue frontend development.
- **Phase 5 (Weeks 9-10)**: Reporting and analytics modules, dashboard implementation.
- **Phase 6 (Weeks 11-12)**: Finalize frontend (React components, forms), integration testing.
- **Phase 7 (Weeks 13-14)**: System testing, bug fixing, and user acceptance testing.
- **Phase 8 (Weeks 15-16)**: Deployment, documentation, and user training.

4.3 Gantt Chart

Table 9: Gantt Chart

Task No.	Project Phase / Task	Start Week	End Week	Duration (Weeks)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	Requirements & Planning	1	2	2	■	■														
2	System Design	3	4	2			■	■												
3	Backend Development	5	6	2					■	■										
4	Frontend Development	5	8	4					■	■	■	■								
5	Reporting & Analytics	9	10	2									■	■						
6	Testing & Refinement	11	12	2											■	■				
7	Deployment & Training	13	16	4													■	■	■	■

(The ■ symbol indicates planned work in the corresponding week.)

4.4 Milestones

Table 10: Milestones

Milestone	Description	Expected Completion
Project Proposal Approval	Approval of this proposal by supervisor	Week 2
Requirements Document	Finalized list of functional and non-functional requirements	Week 4
System Design Completion	Completed system architecture diagrams and wireframes	Week 4
Minimum Viable Product (MVP)	Core functionalities implemented (student CRUD, import, basic auth)	Week 8
Full Feature Implementation	All planned features developed and tested (reporting, dashboards)	Week 12
User Acceptance Testing (UAT)	End-users have tested the system and provided feedback	Week 14
Final Deployment	System deployed on school server and made accessible to users	Week 16

4.5 Resource Allocation

Table 11: Resource Allocation

Resource Type	Description	Quantity	Responsibility	Timeline
Personnel	Project Supervisor	1	Dr. XXXXXXXX XXXXXXXX	Weeks 1–16
Personnel	Student (Developer)	1	XXX	Weeks 1–16
Hardware	Development Laptop/PC	1	Student	Weeks 1–16
Hardware	Test Server (School server or cloud VM)	1	IT Department / Student	Weeks 8–16
Software	Spring Boot (Java)	-	Student	Weeks 1–16
Software	React.js / Node.js	-	Student	Weeks 1–16
Software	MySQL Database	-	Student	Weeks 1–16
Software	JasperReports / PDFBox	-	Student	Weeks 5–12
Internet Access	Reliable internet connection	-	University Network	Weeks 1–16
Cloud Services	(Optional) Deployment environment (e.g., AWS/GCP)	1	Student	Weeks 13–16

4.6 Risk Management

Table 12: Risk Management

Risk ID	Risk Description	Impact (H/M/L)	Probability (H/M/L)	Mitigation Strategy	Contingency Plan
R1	Delay in requirement gathering due to stakeholder availability	Medium	Medium	Schedule requirements sessions early; have backup participants.	Adjust timeline or scope if necessary.
R2	Technical difficulty with new technology (e.g., React/Spring)	High	Low	Allocate learning time in early sprints; use online tutorials and sample projects.	Seek help from supervisor or mentors.
R3	Data privacy/security breach during development/testing	High	Low	Follow best security practices; conduct security training.	Perform a full security audit; patch issues.
R4	Scope creep (adding new features not in original plan)	Medium	Medium	Maintain a clear requirements document; manage changes through change control.	Re-prioritize features; defer to future work.
R5	Inadequate testing time	Medium	Low	Integrate testing tasks within each sprint; automate tests where possible.	Allocate extra time; reduce new feature development.
R6	Deployment environment issues (server configuration)	Medium	Low	Prepare environment early; use containerization (Docker) if possible.	Have a cloud backup deployment plan.

4.7 Summary

This chapter presented the detailed project plan, including phases, Gantt chart, milestones, resource allocation, and risk management. A clear schedule and assignment of responsibilities ensure that the project remains on track to meet its objectives within the given 16-week timeline.

5. Tables and Figures

6. References

Attahakul, P. (2024). Revolutionizing Learning: The Transformative Role of Technology in Modern Education. *Journal of Asian Language Teaching and Learning*, 5(3), 15-26.

Dissanayake, K. (2023). Challenges in student data management in Sri Lankan secondary schools: A case study approach. *Sri Lankan Journal of Educational Administration*, 8(2), 45-62.

Fernando, M., Silva, P., & Gunasekara, R. (2023). Digital transformation in educational institutions: Opportunities and challenges in the Sri Lankan context. *Asian Journal of Educational Technology,* 12(4), 112-128.

Jayasinghe, N., & Wickramasinghe, V. (2024). Adoption of information systems in Sri Lankan schools: Barriers and success factors. *International Journal of Education and Development using ICT,* 20(1), 78-95.

Jayawardena, D., & Ranasinghe, S. (2024). Efficiency gains through automation in educational administration: Evidence from South Asian schools. *Journal of Educational Management*, 18(3), 201-218.

Ministry of Education. (2023). *National Policy on ICT in Education.* Ministry of Education, Sri Lanka.

Parliament of Sri Lanka. (2022). *Personal Data Protection Act, No. 9 of 2022.* Government Publications Bureau, Colombo.

Rashid, A., & Khan, M. (2022). Student information management systems: A comparative analysis of features and benefits. *International Journal of Educational Technology in Higher Education*, 19(1), 1-21.

Samarasinghe, T., Perera, K., & Fernando, L. (2023). Data quality issues in manual record-keeping systems: Implications for educational institutions. *South Asian Journal of Management Information Systems*, 7(2), 89-104.

Silva, R., & Perera, N. (2024). Time management in school administration: Impact of digital tools on administrative efficiency. *Asian Journal of Education and Training*, 10(2), 156-171.

Yun, L. (2025). Analyzing Credit Risk Management in the Digital Age: Challenges and Solutions. *Economics and Management Innovation*, 2(2), 81-92.