Title of Project

"College Curriculum System"

Version 1.0

Date: 15 April 2024

Software Requirements Specification

Submitted By,

Mr. Aniket Rangari(BE21F05F056)

Mr. Prajwal Rangari(BE21F05F057)

Mr. Linesh Meshram(BE21F05F045)

Mr. Gajanan Ambekar(BE22S05F002)

Under the Guidance

of

Dr. Vikul Pawar

Lecturer in Computer Science and Engineering Department

Government College of Engineering Chh. Sambhajinagar

Prepared for Software Engineering

Revision History

Date	Description	Author	Comments

Document Approval

The following Software Requirements Specifications has been accepted and approved by the following.

Signature	Printed Name	Title	Date

Table of Contents

1. Introduction

College Curriculum System is a fully functional System that enables Student to create, consume, and rate college Curriculum . The platform is built using the MERN stack, which includes ReactJS, NodeJS, MongoDB, and ExpressJS.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to provide a detailed roadmap for the development of College Curriculum System. It outlines the functional and non-functional requirements essential for creating a robust, user-friendly learning environment.

1.2 Scope

The College Curriculum System is designed to streamline the management of college curriculums, including course offerings, student enrolment , instructor , rating & review and academic resources. It will facilitate efficient communication between students, instructors, and administrators, improving the overall educational experience.

1.3 Overview

The College Curriculum System Software Requirements Specification (SRS) serves as a comprehensive guide for the development of an innovative educational platform. This document outlines the functional and non-functional requirements, constraints, and assumptions necessary to create a seamless learning experience for students and instructors alike.

Within this SRS, you will find a detailed breakdown of the system's scope, including its purpose, user characteristics, and general constraints. Specific requirements such as course creation, user authentication, and course consumption are defined to ensure clarity and precision in development.

Additionally, the document highlights the interfaces—user, hardware, software, and communications—necessary for College Curriculum System's operation. Use cases, functional requirements, and non-functional requirements are detailed to guide the software design and implementation process effectively.

Throughout the document, the emphasis is on creating a user-friendly, secure, and reliable platform using the MERN stack. The SRS also includes provisions for future scalability and maintainability, ensuring College Curriculum System's adaptability to evolving educational needs.

This SRS lays the foundation for the College Curriculum System project, providing a roadmap for developers to build an engaging and efficient educational platform that empowers both students and instructors in their learning journey.

2. General Description

College Curriculum System is a comprehensive software solution designed to streamline and enhance the management of academic curriculums within colleges and educational institutions. It aims to provide a centralized platform for administrators, faculty members, and students to effectively manage, track, and access course information, academic resources, and student progress.

2.1 Product Perspective:

The College Curriculum System is conceived as a centralized platform to streamline academic processes within educational institutions. It operates as a comprehensive system that integrates various modules to manage curriculum planning, student enrollment, course offerings, and academic records.

In the realm of academic management systems, the College Curriculum System stands out as a versatile tool that offers administrators, faculty, and students a cohesive platform for efficient academic operations.

2.2 Product Functions:

The core functions of the College Curriculum System are designed to enhance academic administration and student experience:

Curriculum Planning and Management: Faculty and administrators can create, modify, and archive course curricula, ensuring alignment with institutional goals and accreditation requirements.

Student Enrollment and Registration: A user-friendly interface for students to enroll in courses, view available offerings, and register for classes online.

Academic Records Management: Efficient management of student academic records, including transcripts, grades, and course histories.

Faculty Dashboard: An interface for faculty members to view course rosters, input grades, and communicate with students.

Administrator Tools: Administrative features for managing user roles, permissions, and system configurations.

2.3 User Characteristics:

The College Curriculum System caters to three primary user groups:

Administrators: These users oversee the overall functioning of the system, manage user accounts, and configure system settings.

Faculty Members: Academic staff who utilize the system for curriculum planning, class scheduling, grading, and student communication.

Students: The primary beneficiaries, who interact with the system to enroll in courses, track academic progress, and access course materials.

2.4 General Constraints:

- The system must adhere to institutional policies and academic regulations set by accrediting bodies.
- Compatibility with modern web browsers and mobile devices to ensure accessibility for all users.
- Integration with existing university systems such as student information systems (SIS) and learning management systems (LMS).

2.5 Assumptions and Dependencies:

- The development of the College Curriculum System is based on several key assumptions and dependencies:
- Availability of necessary infrastructure and hardware within the institution to host and run the system.
- Compliance with data protection laws and regulations to safeguard student and faculty information.
- Reliance on faculty input and collaboration for accurate course offerings, descriptions, and scheduling.
- These factors form the foundation of the College Curriculum System, aiming to streamline academic processes, improve student success, and enhance overall institutional efficiency.

3. SPECIFIC REQUIREMENTS

3.1 External Interface Requirements

3.1.1 User Interface

The College Curriculum System offers intuitive and user-friendly interfaces tailored to the needs of its diverse user base:

- **Administrator Dashboard:** A centralized portal providing administrators with access to system-wide functionalities. Here, they can manage user accounts, configure system settings, and oversee curriculum planning.
- **Faculty Interface:** Faculty members interact with a dedicated interface for course management. This interface allows them to create new courses, input course descriptions, set prerequisites, and assign teaching schedules.
- **Student Portal:** The student portal serves as the gateway for students to access course catalogs, register for classes, view academic records, and track progress towards course completion. It provides a seamless and personalized experience for each student.

3.1.2 Software Interfaces:

The College Curriculum System interfaces with various software components to ensure seamless operation and data integration:

Student Information System (SIS) Integration: The system integrates with the institution's SIS to synchronize student enrollment data, academic records, and demographic information. This ensures accuracy and consistency across platforms.

Learning Management System (LMS) Integration: For course delivery and content management, the system interfaces with the institution's LMS. This allows faculty to seamlessly upload course materials, assessments, and facilitate online learning experiences.

Third-Party Tools for Analytics: Integration with analytics tools provides valuable insights into course performance, student engagement, and curriculum effectiveness. Data from these tools can inform decision-making for curriculum improvements.

3.1.3 Communications Interfaces

Efficient communication channels are essential within the College Curriculum System to facilitate collaboration and information exchange:

In-System Messaging: Users, including administrators, faculty, and students, can communicate within the system through messaging features. This includes announcements, course updates, and academic advisement.

Email Integration: The system is configured to send automated email notifications for important events such as course registration deadlines, grade postings, and academic advisement alerts. This ensures timely communication with all stakeholders.

Feedback Forms: A crucial aspect of communication is the provision of feedback mechanisms. The system includes user-friendly feedback forms for students to provide input on courses, instructors, and overall academic experiences. This feedback helps improve course offerings, teaching methods, and institutional policies.

These interfaces within the College Curriculum System are designed to enhance user experience, streamline operations, and foster effective communication and collaboration among administrators, faculty, and students.

3.2 Functional Requirements

The College Curriculum System is designed to provide a robust set of functionalities to meet the needs of administrators, faculty, and students. These functional requirements ensure efficient management of courses, seamless student enrollment, and effective communication within the academic community:

3.2.1 Administrator Functionalities:

User Management:

- Create, edit, and delete user accounts for administrators, faculty, and students.
- Assign roles and permissions to users based on their responsibilities within the system.

Curriculum Planning:

- Define and manage course catalogs, including course titles, descriptions, prerequisites, and credit hours.
- Create and modify academic programs, majors, minors, and specializations.

Enrollment Management:

- Set registration periods, enrollment caps, and waitlist options for courses.
- Monitor and manage student enrollment status, including approvals, overrides, and withdrawals.

Academic Calendar Management:

- Define and maintain the academic calendar, including semester dates, holidays, and examination periods.
- Schedule and publish course offerings for each academic term.

Reporting and Analytics:

- Generate reports on student enrollment, course popularity and other key metrics.
- Provide dashboards with visual analytics for administrators to make data-driven decisions.

3.2.2 Faculty Functionalities:

Course Management:

- Create and manage course syllabi, including learning objectives, assignments, and grading criteria.
- Upload course materials such as lectures, readings, and course content.

Communication Tools:

- Send announcements, notifications, and reminders to students regarding course updates and deadlines.
- Conduct virtual office hours and facilitate discussions through integrated communication channels.

Advising and Mentorship:

- View student profiles, academic histories, and progress towards degree requirements.
- Provide guidance and support to students on course selection, academic planning, and career pathways.

3.2.3 Student Functionalities:

Course Search and Registration:

- Search for courses based on criteria such as course title, instructor, department, or keyword.
- Register for courses, add to wishlist, and view enrollment status in real-time.

Course Progress Tracking:

- View personalized Course audit reports showing completed requirements, in-progress courses, and remaining credits.
- Plan course schedules for upcoming semesters and simulate different academic pathways.

Assignment Submission and Feedback:

- Submit assignments, projects, and assessments electronically through the system.
- Receive timely feedback from instructors on graded assignments, along with comments and suggestions for improvement.

Student Collaboration and Engagement:

- Participate in online discussions, group projects, and collaborative activities within course modules.
- Access learning resources, study materials, and multimedia content to supplement classroom learning.

These functional requirements of the College Curriculum System aim to streamline academic processes, enhance teaching and learning experiences, and promote effective communication and collaboration among all stakeholders within the educational institution.

3.3 Use Cases:

The College Curriculum System encompasses a range of use cases that illustrate the interactions and functionalities available to administrators, faculty, and students. These use cases demonstrate the system's ability to support essential academic processes and facilitate seamless operations within the institution:

3.3.1 Administrator Use Cases:

1. Manage User Accounts:

Description: The administrator creates, edits, or deletes user accounts for faculty, students, and other administrators.

Actors: Administrator

Flow:

1. Administrator accesses the user management interface.
2. Selects "Create User" to add a new user or "Edit User" to modify existing user details.
3. Enters user information such as name, email, role, and permissions.
4. Saves changes and confirms user account creation or modification.

2. Define Course Catalog:

Description: The administrator defines and updates the college's course catalog, including course titles, descriptions, prerequisites, and credit hours.

Actors: Administrator

Flow:

1. Accesses the course management section of the system.
2. Adds a new course or selects an existing course for modification.
3. Inputs course details such as title, description, prerequisites, and credit hours.
4. Saves changes and publishes the updated course catalog.

3. Manage Student Enrollment:

Description: The administrator oversees student enrollment processes, including setting registration periods, enrollment caps, and managing waitlists.

Actors: Administrator

Flow:

1. Accesses the enrollment management module.
2. Sets registration dates and deadlines for upcoming semesters.
3. Defines enrollment limits for courses and sections.
4. Manages student waitlists, approves overrides, and monitors enrollment status.

3.3.2 Faculty Use Cases:

1. Create Course Syllabus:

Description: The faculty member creates and updates a detailed syllabus for their course, outlining learning objectives, assignments, and grading criteria.

Actors: Faculty

Flow:

1. Navigates to the course management interface.
2. Selects the option to create a new syllabus or edit an existing one.
3. Adds sections for course overview, learning outcomes, weekly topics, assignments, and assessments.
4. Uploads course materials such as lecture slides, readings, and supplementary resources.
5. Publishes the completed syllabus for student access.

2. Grade Assignments and Provide Feedback:

Description: The faculty member grades student assignments, quizzes, and exams, providing timely feedback and comments.

Actors: Faculty

Flow:

1. Accesses the gradebook for the course.
2. Reviews submitted assignments, quizzes, or exams.
3. Enters grades and feedback for each student.
4. Saves grades and comments, which are automatically shared with students.

3. Conduct Virtual Office Hours:

Description: The faculty member offers virtual office hours for students to ask questions, seek clarification, and discuss course materials.

Actors: Faculty, Students

Flow:

1. Sets designated hours for virtual office availability.
2. Notifies students through the system or course announcements.
3. Students access the virtual office interface during scheduled hours.

4. Engages in one-on-one or group discussions, provides guidance, and addresses student queries.

3.3.3 Student Use Cases:

1. Search and Register for Courses:

Description: The student searches for available courses, views course details, and registers for desired classes.

Actors: Student

Flow:

1. Navigates to the course search and registration portal.
2. Filters courses by department, semester, or keyword search.
3. Views course descriptions, prerequisites, and available sections.
4. Selects desired courses and adds them to the registration cart.
5. Completes the registration process, confirming enrollment in selected courses.

2. Submit Assignments and View Feedback:

Description: The student submits assignments electronically through the system and accesses feedback from instructors.

Actors: Student, Faculty

Flow:

1. Navigates to the course page and selects the assignment submission link.
2. Uploads the completed assignment file and submits it.
3. Receives confirmation of successful submission and awaits grading.
4. Accesses the graded assignment with feedback and comments from the instructor.

4. Rate and Review Courses

Description: The student provides ratings and reviews for completed courses, sharing feedback with future students.

Actors: Student

Flow:

1. Visits the course overview page for completed courses.
2. Selects the option to rate and review the course.
3. Rates the course on a scale and provides written feedback.
4. Submits the review, which is displayed publicly for reference by other students.

These use cases showcase the comprehensive functionalities of the College Curriculum System, allowing administrators, faculty, and students to engage with the platform effectively. The addition of the "Rate and Review Courses" use case empowers students to contribute feedback, enhancing the overall learning experience for the college community.

3.5 Non-Functional Requirements:

The College Curriculum System is designed to meet various non-functional requirements to ensure optimal performance, reliability, security, maintainability, and portability. These requirements are crucial for the efficient functioning and long-term viability of the system within the academic environment:

3.5.1 Performance

- **Response Time:** The system shall respond to user interactions, such as course searches, registration, and grade submissions, within 2 seconds on average.

- **Scalability:** The system must support a concurrent user load of at least 500 users during peak hours without significant degradation in performance.

- **Data Processing:** Batch processes, such as generating reports and updating course catalogs, should be completed within 1 hour for moderate data volumes.

3.5.2 Reliability

- **Uptime:** The system shall maintain an uptime of 99.9% to ensure continuous availability during critical academic periods.

- **Error Handling:** Robust error handling mechanisms shall be implemented to minimize system downtime due to unexpected errors.

- **Data Integrity:** The system must ensure data integrity, preventing data loss or corruption through regular backups and transaction logging.

3.5.3 Availability

- **7 Accessibility:** The system should be accessible to users 24 hours a day, 7 days a week, except during scheduled maintenance windows.

- **Redundancy:** Critical system components, such as servers and databases, shall have redundant backups to minimize downtime in case of hardware failures.

- **Disaster Recovery:** A comprehensive disaster recovery plan shall be in place to restore system functionality within 4 hours of a major disruption.

3.5.4 Security

- **User Authentication:** Users must authenticate using secure methods such as password hashing and encryption.

- **Role-Based Access Control (RBAC):** The system shall implement RBAC to ensure that users only have access to functionalities and data relevant to their roles.

- **Data Encryption:** Sensitive data, including student records and grades, shall be encrypted both in transit and at rest using industry-standard encryption algorithms.

3.5.5 Maintainability

- **Modularity:** The system architecture shall be modular to allow for easy updates, enhancements, and bug fixes without affecting the entire system.

- **Code Documentation:** Comprehensive documentation of codebase, APIs, and database schemas shall be maintained to aid in system maintenance and future development.

- **Automated Testing:** Regular automated testing suites shall be implemented to ensure code integrity and detect regressions during updates.

3.5.6 Portability

- **Cross-Browser Compatibility:** The user interface shall be compatible with major web browsers such as Chrome, Firefox, Safari, and Edge.

- **Device Independence:** The system shall be responsive, providing optimal user experience across devices, including desktops, laptops, tablets, and smartphones.
- **Cloud Deployment:** The system shall be deployable on cloud platforms such as AWS, Azure, or Google Cloud, ensuring flexibility and scalability in hosting options.

These non-functional requirements are integral to the College Curriculum System, guaranteeing its performance, reliability, security, maintainability, and portability. They form the foundation for a robust and adaptable system that meets the diverse needs of administrators, faculty, and students within the academic institution.

3.6 Inverse Requirements

In considering the College Curriculum System, the following inverse requirements should be acknowledged. These represent scenarios where certain actions or outcomes should not occur, ensuring the system's integrity and user experience:

- **Inverse Requirement 1:** The system should not allow unauthorized users to access or modify course information, student records, or administrative settings.
- **Inverse Requirement 2:** Courses should not be removed from the system without proper authorization and confirmation, to prevent accidental loss of important data.
- **Inverse Requirement 3:** Students should not be able to register for courses that have reached maximum capacity or conflicting schedules to avoid overbooking and scheduling conflicts.

3.7 Design Constraints

Design constraints impose limitations on the development and implementation of the College Curriculum System. These constraints guide the design choices and influence the system's architecture:

- **Constraint 1:** The system must be developed using the MERN stack (MongoDB, Express.js, React.js, Node.js) to align with the institution's technology standards and available expertise.
- **Constraint 2:** Compliance with WCAG (Web Content Accessibility Guidelines) 2.0 standards for accessibility, ensuring that the system is usable by individuals with disabilities.
- **Constraint 3:** Utilization of a responsive design approach, ensuring the system's usability across a variety of devices, including desktops, tablets, and smartphones.

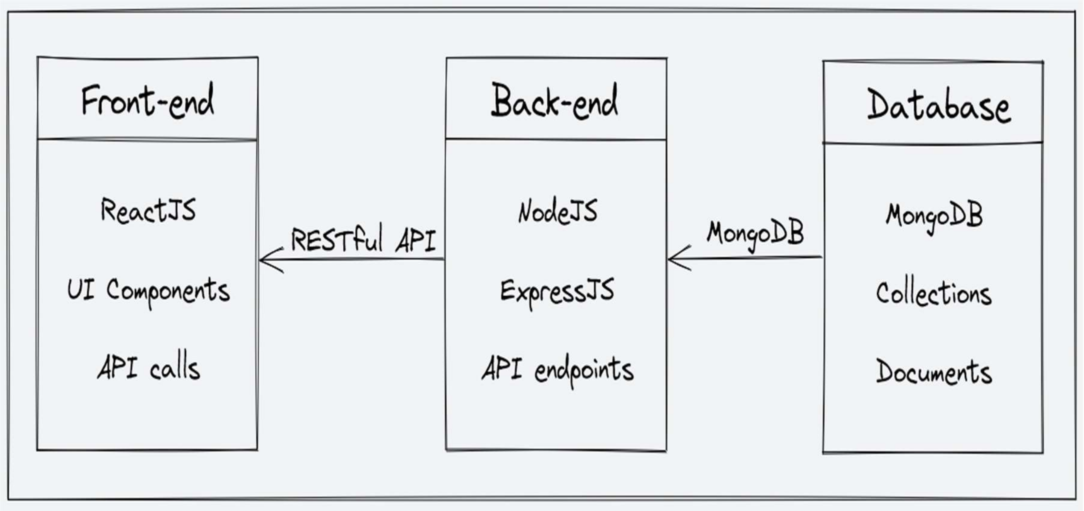
3.8 Logical Database Requirements

The College Curriculum System relies on a robust and efficient database structure to manage course information, student data, and administrative records. The logical database requirements include:

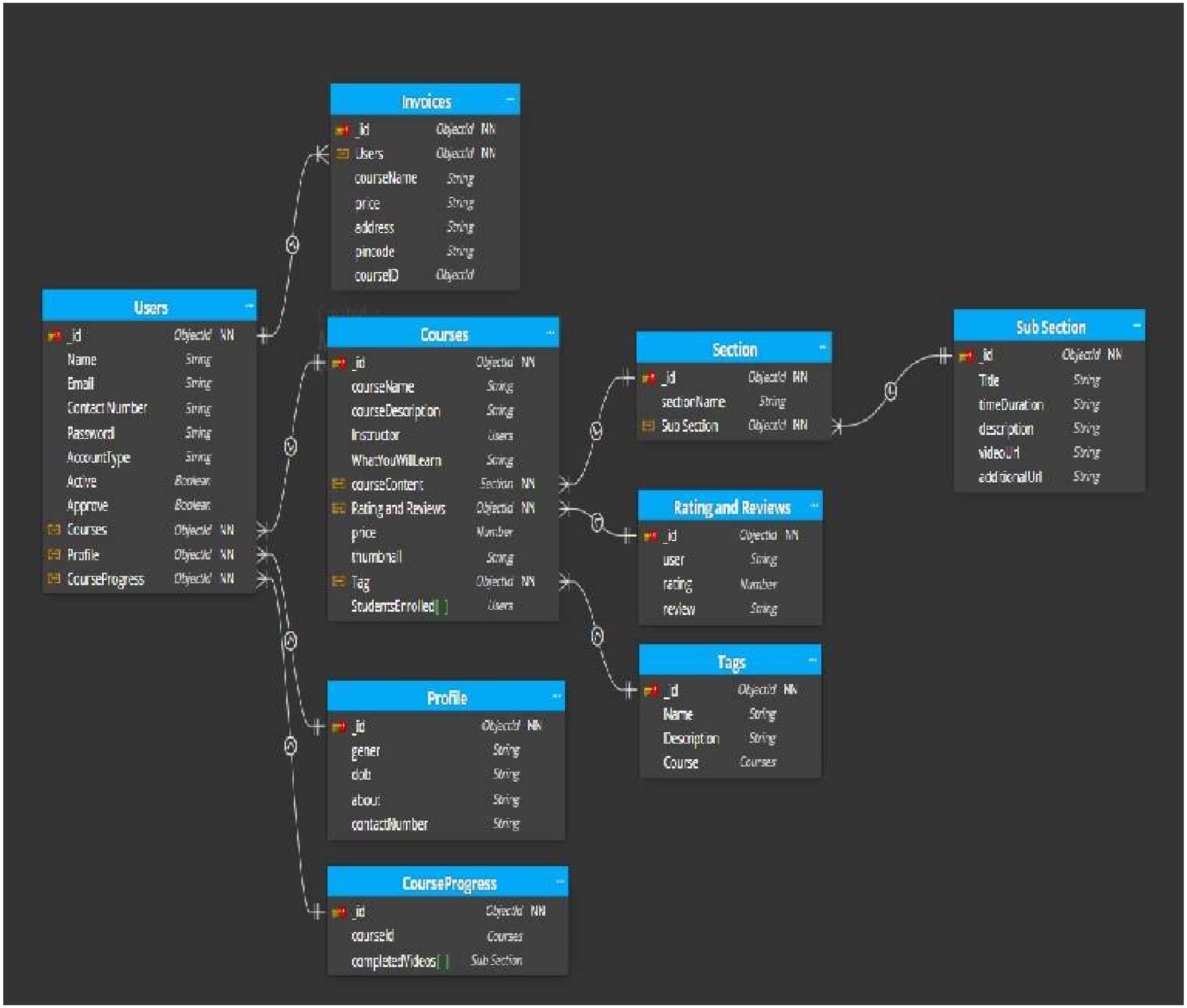
- **Requirement 1:** A relational database model using MongoDB to store course details, instructor information, student enrollment, and grading data.
- **Requirement 2:** Normalized database schema to minimize redundancy and ensure data integrity, with tables for courses, instructors, students, departments, and enrollment.
- **Requirement 3:** Implementation of database indexes to optimize query performance, especially for frequently accessed data such as course listings and student schedules.

4. ANALYSIS MODELS

4.1 DATA FLOW DIAGRAMS (DFD) / ARCHITECTURAL DIAGRAM:



4.2 Database Models and Database Schema Diagram:



5. Change Management Process:

The Change Management Process for the College Curriculum System outlines the procedures and protocols for handling modifications, updates, and enhancements to the system. This process ensures that changes are properly evaluated, approved, implemented, and documented to maintain the system's integrity and functionality.

5.1 Change Submission:

Submission Channels: Changes to the system can be submitted through the designated channels, including:

- User feedback forms available within the system interface.
- Emails to the system administrators or designated change management team.
- Direct communication with the IT department or system development team.

Change Request Form: A standardized change request form will be available for users to fill out, providing details such as:

- Description of the change or enhancement.
- Justification or rationale for the change.
- Impact assessment on system functionality or user experience.
- Priority level (e.g., low, medium, high).

5.2 Change Evaluation and Prioritization

Change Review Board: A Change Review Board (CRB) comprising system administrators, IT personnel, faculty representatives, and student stakeholders will convene to review change requests.

Evaluation Criteria: Each change request will be evaluated based on:

- Alignment with institutional goals and curriculum requirements.
- Technical feasibility and impact on existing system architecture.
- Potential benefits to users, such as improved efficiency, usability, or compliance.
- Resource requirements, including time, personnel, and budget.

Prioritization Levels: Changes will be categorized into priority levels to determine the order of implementation:

Low Priority: Minor enhancements or cosmetic changes that do not significantly impact system functionality.

Medium Priority: Moderate changes that improve system efficiency, user experience, or data management.

High Priority: Critical changes addressing system vulnerabilities, legal compliance, or essential feature additions.

5.3 Change Approval and Implementation

Approval Process: Once a change request is reviewed, the CRB will decide on its approval based on the evaluation criteria and priority levels.

Notification: Users submitting change requests will be notified of the CRB's decision, including the approval status and expected implementation timeline.

Testing and Deployment: Approved changes will undergo rigorous testing procedures to ensure compatibility, functionality, and security. This includes:

Unit testing for individual components.

Integration testing to verify interactions between modules.

User acceptance testing (UAT) involving stakeholders to validate changes in a real-world environment.

Rollout Plan: A phased rollout plan will be developed for implementing changes to minimize disruptions and ensure smooth transitions. This may include:

- Scheduled downtime for system updates.
- Communication of changes to users through announcements, emails, or in-app notifications.
- Training sessions or materials for users to adapt to new features or functionalities.

5.4 Documentation and Monitoring:

- Change Logs: Detailed change logs will be maintained, documenting:
- Description of the change request.
- Date of submission and approval.
- Implementation details, including testing outcomes and deployment dates.
- User feedback or issues encountered post-implementation.

Monitoring and Feedback: The system administrators will monitor the post-implementation performance and gather feedback from users regarding the effectiveness of the changes. This feedback loop informs future iterations and continuous improvement efforts.

5.5 Change Reversion and Rollback:

Contingency Plans: In the event of unforeseen issues or negative impacts post-implementation, a rollback plan will be enacted. This includes:

- Reverting the system to its previous state.
- Investigating the root cause of the issue and implementing corrective measures.
- Communicating with users regarding the situation, resolution, and any necessary actions.

The Change Management Process for the College Curriculum System ensures that changes are carefully evaluated, approved, implemented, and monitored to enhance system functionality, user experience, and overall effectiveness. By following structured procedures and involving key stakeholders, the system can adapt to evolving needs while maintaining stability and reliability.

A.1 Appendix 1