## Practical 4

# Software Requirement Specification (SRS) Development for the Selected System

**Aim:**Perform a requirement analysis and develop a Software Requirement Specification (SRS) sheet for the selected system. The SRS should include the following sections:

1. Functionality: Describe what the software is supposed to do.
2. External Interfaces: Explain how the software interacts with people, the system's hardware, other hardware, and other software.
3. Performance: Outline the expected speed, availability, response time, recovery time, and other performance-related characteristics of the software functions.
4. Attributes: Define considerations related to portability, correctness, maintainability, security, and other relevant attributes.
5. Design Constraints: Specify any design constraints imposed on the implementation, such as required standards, implementation languages, database integrity policies.

## 4. Software Requirement Specification (SRS)

A Software Requirement Specification (SRS) is a structured document that defines the functionalities, constraints, and requirements of a software system. It serves as a foundation for development, testing, and stakeholder communication.

### 4.1 Overview

A **College Management System (CMS)** is a digital solution designed to streamline and automate various administrative and academic operations within an educational institution. It replaces traditional manual processes with an efficient, organized system that enables seamless management of student records, faculty details, course enrollment, fee management, and examination processes. The CMS aims to enhance the overall efficiency of college administration by providing a centralized platform for data management, improving communication, and reducing manual workload.

This **Software Requirement Specification (SRS)** document follows the IEEE SRS standard and outlines the functional and non-functional requirements of the system, as well as external interfaces, constraints, and dependencies.

## 4.2 Purpose

The primary purpose of this **CMS** is to provide an automated platform for managing college administration, student records, course enrollment, and faculty details. The system enables students to register for courses, view academic progress, and manage fee payments, while administrators can oversee academic operations, track student performance, and generate reports. It also supports notifications for important academic deadlines, fee dues, and examination schedules.

## 4.3 Document Conventions

This document follows structured formatting using numbered sections, bullet points, and industry-standard terminology for clarity and ease of reference. This document is intended mainly for users such as **College Administrators**, who manage student records, faculty information, and course allocations; **Faculty Members**, who track attendance, upload grades, and communicate with students; **Students**, who enroll in courses, view academic records, and submit assignments; and **Developers**, who understand and implement system requirements. Other relevant stakeholders include **Testers**, who validate system functionality and performance, and **Management**, who review system capabilities and constraints.This document follows **IEEE 830-1998: Recommended Practice for Software Requirements Specifications.**

## 4.4 Overall Description

This section provides an overview of the system's purpose, functions, and constraints.

**Product Perspective**
The **CMS** is a standalone software system designed to digitize and streamline college administrative and academic operations. It integrates with existing institutional infrastructure, replacing manual record-keeping methods with a centralized database for improved efficiency and accessibility.

**Product Functions**

The system will provide:

- **User Account Management:** Role-based access control for students, faculty, and administrators.
- **Student Enrollment & Course Management:** Registration, course selection, and faculty allocation.
- **Attendance & Grade Management:** Tracking student attendance, recording grades, and generating progress reports.
- **Fee Management:** Handling tuition payments, generating invoices, and tracking due payments.
- **Notifications & Reminders:** Alerts via email/SMS for important deadlines, fee dues, and examination schedules.

- **Report Generation:** Statistical insights on student performance, course enrollments, and financial transactions.

Users of the CMS include:

- **College Administrators:** Manage student records, faculty details, and course allocations.
- **Faculty Members:** Track attendance, upload grades, and communicate with students.
- **Students:** Register for courses, access academic records, and receive notifications.
- **System Administrators:** IT personnel responsible for maintaining the system and ensuring data security.

**General Constraints**

The system must comply with **educational data privacy regulations**.It should support at least **50,000 student records** and scale as needed. The system must be **operational 24/7** with minimal downtime to support academic and administrative functions.

## 4.5 Specific Requirements

The Specific Requirements section of the SRS outlines detailed functional and non-functional needs of the **College Management System (CMS).** Functional requirements define what the system should do, describing specific actions and behaviors. These requirements directly contribute to the core functionalities of the software. Non-functional requirements define how the system performs its functions, focusing on system attributes like security, performance, usability, and scalability. The External Interfaces section of an SRS describes how the **College Management System (CMS)** interacts with users, hardware, and other software. These interfaces ensure seamless operation and integration within the existing environment.

### 4.5.1 Functional Requirements

Functional requirements describe what the system should do:
1. **User Authentication & Authorization:** Secure login and role-based access.
2. **Student Management:** CRUD (Create, Read, Update, Delete) operations for student records.
3. **Course Management:** Assign courses to students and faculty.
4. **Fee Management:** Track student fee payments and pending dues.
5. **Examination Management:** Manage exam schedules and result processing.
6. **Attendance Tracking:** Maintain student attendance records.
7. **Reports & Analytics:** Generate reports on student performance and financial transactions.

### 4.5.2 External Interfaces

External interfaces define how the system interacts with users, hardware, and other software components:

1. User Interface (UI): A web-based interface that provides an intuitive dashboard for users to interact with the system.
2. Hardware Interface**:** The system must support biometric scanners for attendance, printers for reports, and mobile devices for remote access.
**3.** Software Interfaces**:** The system integrates with SQL databases for storage.
**4.** Communication Interfaces: The system supports email, SMS, and push notifications to alert users about deadlines and updates**.**

### 4.5.3 Non-Functional Requirements

Non-functional requirements describe system performance, security, and usability expectations:

1. 1.Performance: The system must handle maximum concurrent users, with response times under 2 seconds.

2. 2.Security: Implements encryption, multi-level authentication, and role-based access control to prevent unauthorized access.

3. Usability: The UI should be user-friendly, accessible, and follow WCAG guidelines.

4. Scalability: The system must support expansion without performance degradation.

5. Availability: Ensures maximum uptime, with automated backups for data recovery.

### 4.5.4 Design Constraints

Design constraints are limitations that affect the implementation and architecture of the **College Management System (CMS).** These constraints ensure that the system is developed within predefined technical, security, and operational guidelines. Design constraints define system limitations:

1. Database Management: The system should support both SQL and NoSQL databases depending on scalability and performance needs.
2. Technology Stack: The system should be built using ASP.NET Core for the backend, Angular for the frontend, and MySQL as the primary database.
3. Security Compliance: The system must encrypt user data, follow GDPR compliance, and implement role-based access control (RBAC) with multi-factor authentication (MFA).
4. Performance Optimization: The system should implement load balancing and caching mechanisms for high availability and fast response times.

### 4.5.5 System Attributes

1. Portability: The system must be accessible on Windows, macOS, Linux, and mobile devices.
2. Maintainability: Modular design to allow easy updates and fixes.
3. Reliability: Ensures consistent operation with failover mechanisms.
4. Scalability: The system must handle increasing users and academic records efficiently.
5. Interoperability: Ability to integrate with third-party systems for extended functionality.
6. Extensibility: The system should allow future enhancements and new feature additions with minimal modifications.

### 4.4.6 Implementation details and Hardware Requirements

The College Management System (CMS) is developed as a web-based application using a three-tier architecture:

- Presentation Layer: Developed using Angular for a responsive and interactive user interface.
- Business Logic Layer: Built using ASP.NET Core, handling authentication, business rules, and data processing.
- Data Layer: Uses MySQL as the primary database to store student, faculty, and administrative records.

The system follows RESTful API architecture to ensure seamless communication between the frontend and backend. Security measures such as role-based access control (RBAC), multi-factor authentication (MFA), and data encryption are implemented to safeguard sensitive information.

**Hardware Requirements:**

**Processor:** Intel Core i5 / AMD Ryzen 5 Intel Core i7 / AMD Ryzen 7

**RAM:** 8 GB 16 GB

**Storage:** 250 GB SSD 512 GB SSD

**Operating System:** Windows 10 / macOS / Linux Latest Windows/macOS/Linux

**Database Server:** 4 CPU cores, 8 GB RAM 8 CPU cores, 16 GB RAM

**Network** :100 Mbps internet connection 1 Gbps internet connection