Project 6 “Smart Campus Management System”

Software Engineering course should involve the full software development life cycle (SDLC), foster collaboration, and demonstrate practical use of software engineering principles like requirements gathering, design, version control, testing, and deployment.

I. Project Summary:

** **A web-based or mobile application** that helps manage various aspects of a university or college campus—such as student information, attendance, exam schedules, room booking, and faculty management.

Core Features:

1. User Roles:

• Admin (manages the whole system)

• Faculty (marks attendance, posts assignments)

• Student (views grades, attendance, schedules)

2. Modules:

• Student Information System: Profile, academic records

• Attendance Management: Mark/view attendance

• Timetable & Room Booking: Dynamic scheduling

• Exam & Result Management: Upload/view results

• Notification System: For deadlines, announcements

3. Software Engineering Components to Include:

• Requirement Analysis Document (SRS)

• UML Diagrams: Use case, class, sequence

• Software Design Document

• Version Control: Git/GitHub for team collaboration

• Testing Plan: Unit tests, integration tests

• Deployment: On a local server or cloud (e.g., Firebase, Heroku)

Tech Stack (Suggested):

- Frontend: React.js / Angular / Flutter
- Backend: Node.js + Express / Django / Spring Boot
- Database: MySQL / PostgreSQL / MongoDB

II. Project Milestones

Note: “Smart Campus Management System” project into milestones along with a project report template that you can use for submission.

Milestone 1: Project Planning & Requirement Analysis

- Define project goals and scope
- Conduct stakeholder interviews or surveys (if simulated)
- Prepare a Software Requirements Specification (SRS)
- Identify user roles and system use cases

Deliverables:

- Project Proposal Document
- SRS Document
- Initial Use Case Diagrams

Milestone 2: System Design

- Create architectural design (MVC, 3-tier, etc.)
- Draw UML diagrams (class, sequence, activity)
- Design database schema (ER diagram)

Deliverables:

- Software Design Document (SDD)
- UML Diagrams
- ER Diagram

Milestone 3: Implementation (Frontend & Backend)

• Set up project repository and environment

• Develop core modules:

• User authentication

• Student info management

• Attendance system

• Integrate frontend with backend APIs

i. Deliverables:

• Source code (on GitHub)

• Screenshots of working features

• Progress report

Milestone 4: Testing

• Create and run unit tests for functions/modules

• Perform integration and system testing

• Gather bug reports and fix them

ii. Deliverables:

• Testing Plan

• Test Case Documentation

• Bug Tracking Report

Milestone 5: Deployment & Presentation

• Deploy the app on a server (Heroku, Firebase, or local host)

• Prepare user and admin documentation

• Present final demo to the class or instructors

iii. Deliverables:

• Working Web/Mobile App

• Final Presentation Slides

• User Manual

III. Project Report Template

1. Title Page

• Project Title

• Team Members

• Course Name

• Instructor Name

• Date of Submission

2. Abstract

• Brief overview of the project, objectives, and tools used.

3. Introduction

• Problem statement

• Motivation

• Objectives

• Scope

4. Requirement Analysis

• Functional requirements

• Non-functional requirements

• Tools and technologies used

5. System Design

• Architecture diagram

• UML diagrams (use case, class, sequence)

• Database schema (ERD)

6. Implementation

• Frontend and backend development

• Key code snippets

• Module breakdown

7. Testing

• Testing strategy

• Sample test cases

• Results

8. Deployment

• Deployment process

• Hosting details

9. Conclusion

• Achievements

• Limitations

• Future work

10. References & Appendix

• References (if any frameworks, libraries, or research papers are used)

• Screenshots, extra diagrams, etc.

Great! Let's start with the sample documents for the "Smart Campus Management System" project. I'll provide the following key ones:

1. Software Requirements Specification (SRS) – Summary Version
2. UML Diagrams – Use Case, Class, Sequence
3. ER Diagram – Database Schema Overview

1. Sample Software Requirements Specification (SRS) – Summary

1. Introduction

• Purpose: To create a centralized Smart Campus Management System for students, faculty, and admin to manage academic activities.

• Scope: Web-based system accessible to all users with role-based functionalities.

• Technologies: React, Node.js, Express, MongoDB

2. Functional Requirements

• User registration and login (Students, Faculty, Admin)

• View and edit student information (Admin)

• Mark and view attendance (Faculty, Students)

• Upload and access exam results (Faculty, Students)

• Timetable management (Admin, Faculty)

• Notifications/Announcements system

3. Non-functional Requirements

• Scalability for future modules (library, hostel, etc.)

• Secure login system using JWT

• Responsive design for desktop and mobile

2. UML Diagrams

Use Case Diagram (Text Description)

Actors: Admin, Faculty, Student

Use Cases:

• Login

• Manage Students (Admin)

• Mark Attendance (Faculty)

• View Attendance (Student)

• Manage Timetable (Admin/Faculty)

• Upload/View Results (Faculty/Student)

• Send Notifications (Admin)

III. Class Diagram (Text Version)

i. Classes:

- User
- id
- name
- email
- password
- role
- Student extends User
- enrollmentNo
- course
- year
- Faculty extends User
- department
- subjectList
- Attendance
- studentID
- date
- status
- ExamResult
- studentID
- subject
- grade
- Notification
- message
- recipientRole
- date

ii. Sequence Diagram (Text Summary: “Mark Attendance”)

1. Faculty logs in
2. Selects subject and date
3. Selects student list
4. Marks present/absent
5. Submits attendance
6. System saves to database
7. Confirms success

iii. ER Diagram (Text Summary)

Entities:

• Users (userID, name, email, role)

• Students (studentID, userID, course, year)

• Faculty (facultyID, userID, department)

• Attendance (attendanceID, studentID, date, status)

• Results (resultID, studentID, subject, grade)

• Notifications (noteID, message, targetRole, timestamp)