Software Design Document (SDD)

# 1. Introduction

This Software Design Document (SDD) outlines the technical design and architecture for the Elective Course/Project Registration System, which streamlines the process of course/project registration for students and course management for faculty and administrators. This document is intended to guide the development and implementation of the system.

# 2. System Overview

The Elective Course/Project Registration System is a web-based application designed to allow students to view available electives and projects, check prerequisites, and enroll in the courses/projects based on their eligibility and availability. Faculty and administrators will be able to manage course offerings, track enrollments, and generate reports.

## Key Features

- User Authentication: Secure login for students, faculty, and administrators.  
- Course/Project Catalog: View a list of available electives or projects with descriptions, prerequisites, and seat availability.  
- Automated Registration: Students can enroll based on eligibility and seat limits.  
- Waitlist Management: Automatic waitlisting when a course/project is full.  
- Faculty Dashboard: Faculty can approve or disapprove student enrollments for projects.  
- Admin Controls: Admin users can add, modify, or remove electives/projects and manage registrations.  
- Notifications & Alerts: Email/SMS alerts for registration confirmations, deadlines, and updates.  
- Reporting & Analytics: Generate reports on enrollment trends and student preferences.

# 3. System Architecture

The system follows a typical client-server architecture with three main components:  
  
- Client Side (Frontend): The user interface for students, faculty, and administrators.  
- Server Side (Backend): Handles logic, requests, and data processing.  
- Database: Stores user data, courses, enrollments, and other related information.

## Technology Stack

- Frontend: HTML, CSS, JavaScript, React or similar framework for the UI.  
- Backend: Node.js, Express, or Python (Flask/Django).  
- Database: MySQL, PostgreSQL, or similar relational database.

# 4. Detailed Design

## 4.1. User Interface

The UI will be designed with simplicity and usability in mind, with wireframes created for the following sections:  
  
- Login Page: User authentication form for all roles (students, faculty, administrators).  
- Course/Project Catalog Page: A list of available electives or projects with filtering options.  
- Registration Page: Students can register for available electives/projects.  
- Faculty Dashboard: Faculty members can approve or disapprove enrollments.  
- Admin Dashboard: Admins can manage the courses/projects and track enrollment.

## 4.2. Database Design

The database will include the following main tables:  
  
- Users: Stores information about students, faculty, and admins.  
- Courses/Projects: Stores course/project details (title, description, prerequisites, seat availability).  
- Enrollments: Links students to their enrolled courses/projects.  
- Waitlist: Manages students who are waiting for a full course/project.

## 4.3. System Components

The system components include:  
  
- Authentication System: Secure login and role-based access.  
- Course Registration System: Handles registration logic based on eligibility and availability.  
- Admin & Faculty Dashboards: For course management and enrollment approvals.  
- Notification System: Sends alerts for registration confirmations and updates.

# 5. Security and Compliance

The system must comply with relevant data security and privacy regulations.  
  
- Authentication: The system will implement secure login using OAuth2 or JWT.  
- Data Protection: All sensitive data, such as passwords and personal information, will be encrypted using industry-standard encryption techniques.  
- Compliance: The system will follow institutional policies on data protection and privacy.

# 6. System Interaction

### User Workflow  
- Students: Log in, browse courses/projects, check eligibility, register, waitlist if full, receive notifications.  
- Faculty: Log in, approve/disapprove enrollments for projects.  
- Administrators: Log in, manage courses/projects, generate reports.

# 7. Non-Functional Requirements

- Scalability: The system must be able to handle peak traffic during registration periods.  
- Performance: The system should maintain performance with multiple concurrent users.  
- Usability: The user interface must be simple, intuitive, and easy to navigate.

# 8. Testing and Debugging

The system will undergo the following types of testing:  
  
- Functional Testing: Ensure that all features work as expected.  
- Security Testing: Ensure that the system is secure and data privacy is maintained.  
- Performance Testing: Test system performance under high load.

# 9. Conclusion

The Elective Course/Project Registration System aims to streamline the registration process, enhance accessibility, and reduce manual efforts. With a robust design and automated features, it will simplify course management for students, faculty, and administrators alike.