

# Smart College Management System (SCMS): A Comprehensive Web-Based ERP Solution for Educational Institutions

Rubigha M, Gokulraj R, Arun Prasanna TJ, Dhanushraja R, Abishake C

<sup>1</sup> Rubigha M: Assistant Professor, Dept. of Information Technology, Knowledge Institute of Technology, Tamil Nadu, India

<sup>2</sup> Gokulraj R: Student, Dept. of Information Technology, Knowledge Institute of Technology, Tamil Nadu, India

<sup>3</sup> Arun Prasanna TJ: Student, Dept. of Information Technology, Knowledge Institute of Technology, Tamil Nadu, India

<sup>4</sup> Dhanushraja R: Student, Dept. of Information Technology, Knowledge Institute of Technology, Tamil Nadu, India

<sup>5</sup> Abishake C: Student, Dept. of Information Technology, Knowledge Institute of Technology, Tamil Nadu, India

\*\*\*

**Abstract** - Managing day-to-day activities in colleges can become complicated when different tasks such as student records, fee payments, exams, and communication are handled separately. In many institutions, these processes are still done manually or using disconnected systems, which leads to delays and confusion.

To solve this problem, this project introduces a Smart College Management System (SCMS), which is a web-based platform designed to bring all institutional activities into one place. The system is developed using PHP with the CodeIgniter framework and uses MySQL for storing data.

SCMS combines multiple features such as student information management, fee handling with online payment options, attendance monitoring, examination processing, and communication tools. It also supports different user roles, allowing administrators, staff, students, and parents to access only the features they need.

By using this system, institutions can reduce manual work, improve accuracy, and make information easily accessible at any time. The solution is designed to support growing institutions and can handle large numbers of users efficiently.

**Key Words:** School Management System, College ERP, CodeIgniter, MVC Architecture, Role-Based Access Control, PHP, MySQL, Online Fee Payment, Student Information System, Web Application.

## 1. INTRODUCTION

In today's digital world, it is becoming increasingly important for educational institutions to move away from manual methods of management. Many colleges still depend on paperwork and separate systems to handle their operations, which often results in wasted time and errors.

Handling student data, fee collection, attendance, and exam records manually can become difficult as the number of students increases. It also becomes challenging to maintain proper communication between the institution and stakeholders such as students and parents.

Although some management systems are already available, they do not always meet all requirements. Some lack

important features, while others are expensive or difficult to use.

This project focuses on developing a Smart College Management System (SCMS) that brings all essential functions together in a single platform. The system is built using CodeIgniter, which follows the MVC approach, making the application organized and easier to maintain.

The main aim is to create a system that is simple, efficient, and capable of handling different institutional activities without complexity.

## 2. RELATED WORK

Different types of software have been created to support school and college management. Some of the commonly used systems include Fedena, OpenSIS, SchoolTool, and Gibbon. While these platforms provide useful features, they also have certain drawbacks.

For example, some systems are difficult to install or require technical expertise to manage. Others do not support modern requirements such as online payments or advanced communication methods

. The SCMS is designed to meet this requirement by offering multiple functionalities within a single, easy-to-use platform

Table -1: Comparison of Existing School Management Systems

System	Technology	Modules	Payment Gateways	SMS Providers	Multi-Branch
Fedena	Ruby on Rails	15+	3-5	Limited	No
OpenSIS	PHP	10+	0	No	No
SchoolTool	Python/Zope	8+	0	No	No
Gibbon	PHP	12+	2-3	No	No

SCMS (Proposed)	PHP/CI3	20+	28	15+	Yes
-----------------	---------	-----	----	-----	-----

As shown in Table 1, the proposed SCMS significantly surpasses existing solutions in payment gateway integration (28 vs. 0-5), SMS provider support (15+ vs. 0), and multi-branch capability.

### 3. SYSTEM ARCHITECTURE AND DESIGN

#### 3.1 Layered Architecture

The Smart College Management System is designed using a five-layer architectural model, as shown in Fig. 1, to ensure better organization and system efficiency. The first layer, known as the client layer, allows users to interact with the system through a web browser that displays a responsive interface built using Bootstrap. The presentation layer is responsible for managing the user interface, including layout design and interactive elements, using frontend technologies such as Bootstrap and JavaScript.

The web server layer processes incoming requests and manages navigation within the application, using server configurations to maintain clean and structured URLs. The application layer contains the main functional logic of the system and is developed using the CodeIgniter framework based on the MVC pattern. It includes multiple controllers, models, and supporting libraries that work together to handle system operations.

Finally, the data layer is used for storing and managing all application data. It relies on a MySQL database and supports multiple branches by allowing dynamic switching between different databases. In addition, it handles file storage for user-uploaded content.

Fig. 1: System Architecture — Five-Layer MVC Architecture of SCMS

#### 3.2 Controller Class Hierarchy

An important design choice in the system is the use of seven base controller classes, which are derived from a common parent controller (MY\_Controller) that extends the core CI\_Controller. Each of these base controllers is responsible for handling user-specific access control, including verifying user identity, managing permissions, and maintaining session data based on the assigned role.

Table -2: Controller Class Hierarchy and Authentication

Controller	Purpose	Authentication
MY_Controller	Root base — loads models, libraries	None

Admin_Controller	Admin panel access	Staff login + license + RBAC
Student_Controller	Student portal	Student login + lock panel
Parent_Controller	Parent portal	Parent login
Front_Controller	Public CMS website	None (maintenance check)
OnlineAdmission_Controller	Admission payments	None
Studentgateway_Controller	Fee payment gateways	Student login

Controller Class Hierarchy Diagram

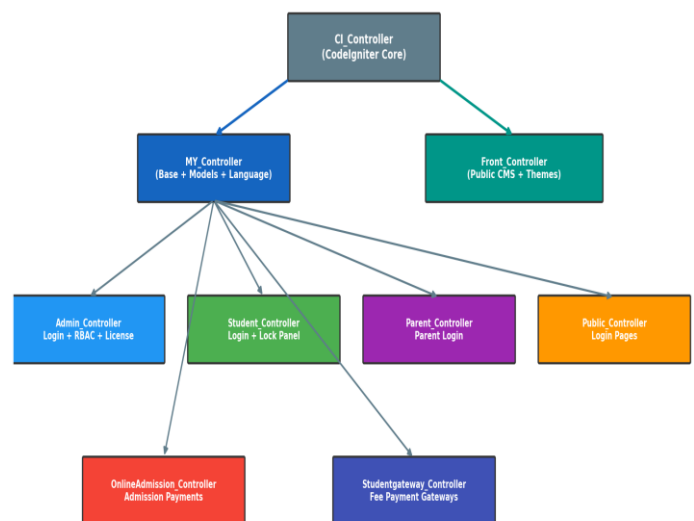


Fig. 2: Controller Class Hierarchy — Inheritance Structure

#### 3.3 Data Flow Diagrams

The system data flow is modelled using structured analysis.

Level 0 (Context Diagram): Fig. 3 shows six external entities (Admin, Teacher, Student, Parent, Payment Gateway, SMS Provider) interacting with SCMS as a central process.

Data Flow Diagram — Level 0 (Context Diagram)

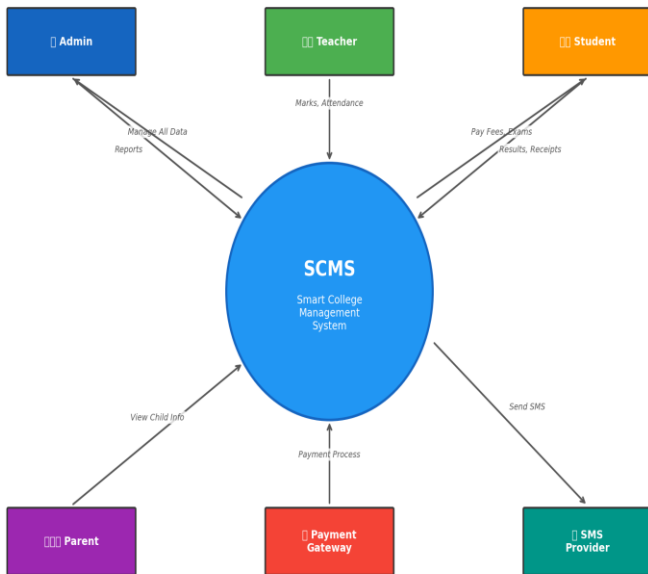


Fig. 3: DFD Level 0 — Context Diagram

Level 1 (Sub-processes): Fig. 4 decomposes the central process into eight major sub-processes: Authentication, Student Management, Fee Management, Academic Management, HR Management, Communication, Report Generation, and Front CMS & Admission.

Data Flow Diagram — Level 1

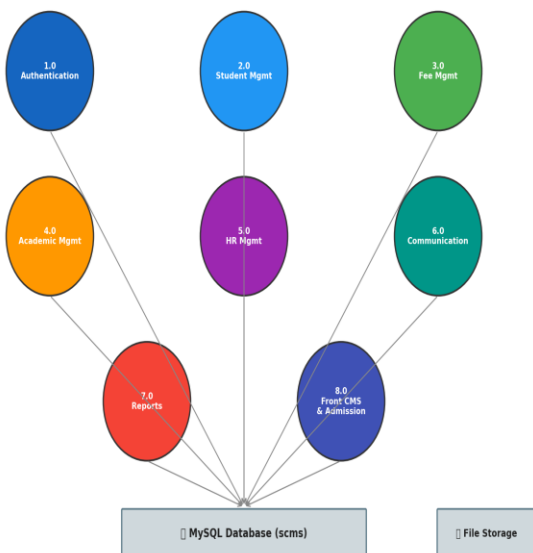


Fig. 4: DFD Level 1 — Major Sub-processes

Level 2 (Fee Management): Fig. 5 provides a granular decomposition of the Fee Management process into seven

sub-processes, including fee setup, assignment, viewing, gateway handling, payment recording, receipt generation, and reporting.

Data Flow Diagram — Level 2 (Fee Management Module)

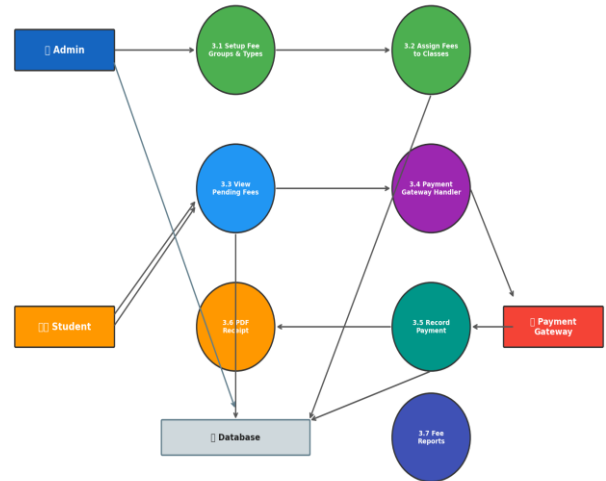


Fig. 5: DFD Level 2 — Fee Management Module

### 3.4 Use Case Diagram

Fig. 6 presents the use case diagram identifying 18 primary use cases across 5 actor types: Admin, Teacher, Student, Parent, and Public User.

Use Case Diagram — SCMS

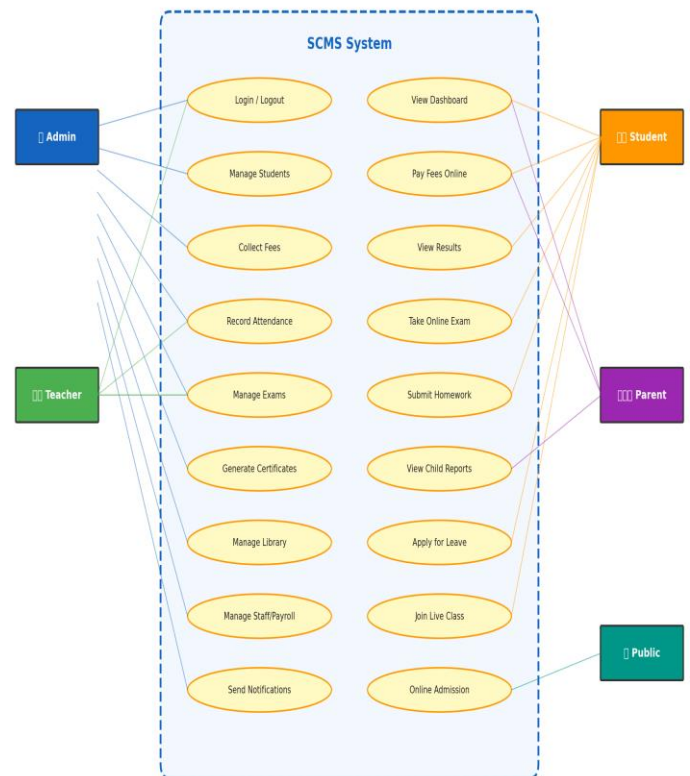


Fig. 6: Use Case Diagram — 5 Actors and 18 Use Cases



### 4.1 Student Management

The Student Management module is responsible for handling all activities related to student records throughout their academic journey. It stores detailed information such as personal details, parent or guardian data, contact address, banking information, and other custom fields required by the institution.

This module provides several useful features to simplify administration. It supports bulk data operations through CSV import and export, allowing large numbers of student records to be managed efficiently. It also includes options for organizing students into different categories, generating ID cards with customizable formats, and managing transfers between classes or branches.

### 4.2 Fee Management and Payment Integration

The Fee Management module is designed using a structured approach that includes multiple levels such as fee groups, fee categories, master fee definitions, and assignment of fees to individual students. This layered structure helps in organizing and managing different types of fees efficiently.

The system supports a wide range of online payment options by integrating multiple payment gateways from different regions. These include internationally recognized platforms as well as region-specific services from countries such as India, Africa, and Southeast Asia. This wide coverage ensures that users can make payments conveniently using their preferred methods.

Each payment gateway is implemented as an independent controller within the system, which makes it easier to manage, update, or add new gateways in the future. This modular design improves flexibility and scalability.

In addition to payment processing, the module also provides features such as fee concession management, automated notifications through SMS and email, and generation of digital receipts in PDF format. These features help streamline financial operations and improve communication with users.

### 4.3 Examination and Result Management

The Examination module is designed to handle both traditional and digital exam processes within the institution. For offline examinations, it allows administrators to organize exams based on terms or semesters, create schedules, record student marks, and define grading criteria. It also supports classification of results into divisions based on performance.

For online examinations, the system provides a flexible setup that includes different types of questions such as multiple-choice, multiple-select, and true/false. A question bank is maintained for easy exam creation, and exams can be conducted within a specified time limit. The system

automatically evaluates responses and generates results, reducing manual effort.

### 4.4 Communication Infrastructure

The Communication module allows the institution to send messages through multiple channels such as SMS, email, and notifications. It supports various service providers for reliable message delivery. The system also includes features like in-app chat and online meeting integration for real-time communication.

### 4.5 Additional Modules

The system also includes additional modules to support overall management. These cover attendance tracking, staff and payroll management, library operations, transport and hostel management, and inventory control. It also provides features for generating certificates and documents, managing website content, and creating various reports related to academics, finance, and attendance.

## 5. DATABASE DESIGN

The system uses a database to store all information in an organized manner. Data related to students, staff, fees, and exams is stored in separate tables, making it easy to manage and retrieve..

Table -3: Key Database Tables and Attributes

Table	Purpose	Key Attributes
students	Student master data	id, admission_no, name, dob, gender, image
student_session	Class enrollment	student_id, session_id, class_id, section_id
staff	Staff/Admin records	id, name, email, password, role_id
student_fees	Fee transactions	id, student_id, amount, payment_mode, date
student_attendances	Attendance records	id, student_session_id, date, type_id
exam_results	Exam mark entries	id, exam_group_id, student_id, marks
online_exams	Online exam config	id, name, duration, passing_pct
books	Library catalog	id, title, author, isbn, qty
roles	Role definitions	id, name, is_system
sidebar_menus	RBAC/menu config	id, short_code, url, permissions

multi_branch	Branch DB config	id, hostname, database_name
--------------	------------------	-----------------------------

## 6. SECURITY IMPLEMENTATION

SCMS implements a defence-in-depth security approach:

**Table -4: Security Implementation Summary**

Security Layer	Mechanism	Implementation
Authentication	Password hashing + sessions	SHA-1 hash, 2hr expiry, file driver
Authorization	RBAC	Per-page, per-action permission checks
Input Security	Validation + XSS clean	CI Form Validation library
SQL Security	Parameterized queries	CI Active Record / Query Builder
CSRF	Token validation	Configurable token-based protection
File Security	MIME + extension check	Upload library whitelist
Audit	Login tracking	user_logs table with IP, timestamp

Each controller checks user permissions using the hasPrivilege() function before executing any action. The sidebar menu is also generated dynamically, showing only the options that the logged-in user is allowed to access based on their role.

## 7. TESTING AND EVALUATION

### 7.1 Testing Methodology

The system was tested using four levels of testing. Unit testing was used to verify individual components, integration testing checked the interaction between modules, system testing ensured overall functionality, and user acceptance testing was carried out with real users to validate performance.

### 7.2 Test Results

Twenty test cases were executed across 10 functional modules. All test cases achieved PASS status, demonstrating functional correctness.

**Table -5: Test Case Execution Results (20 Test Cases)**

TC#	Module	Test Case	Expected Result	Status
TC-01	Auth	Valid admin login	Dashboard redirect	PASS

TC-02	Auth	Invalid credentials	Error message	PASS
TC-03	Auth	Empty field submission	Validation errors	PASS
TC-04	Student	Add a new student	Student ID generated	PASS
TC-05	Student	Duplicate admission no.	Duplicate error	PASS
TC-06	Student	CSV bulk import	Records created	PASS
TC-07	Fee	Fee structure setup	Fee master created	PASS
TC-08	Fee	Online payment	Payment + receipt	PASS
TC-09	Fee	Payment failure	Error, no record	PASS
TC-10	Attendance	Mark attendance	Records saved	PASS
TC-11	Exam	Marks entry	Grades calculated	PASS
TC-12	Exam	Marksheet generation	PDF downloaded	PASS
TC-13	Online Exam	MCQ attempt	Score calculated	PASS
TC-14	Admission	Online submission	Reference no. issued	PASS
TC-15	Library	Book issue	Availability updated	PASS
TC-16	Library	Book return	Stock restored	PASS
TC-17	SMS	Bulk SMS dispatch	Delivered	PASS
TC-18	RBAC	Unauthorized access	403 page shown	PASS
TC-19	Report	Fee collection report	Accurate totals	PASS
TC-20	Certificate	Certificate generation	PDF generated	PASS

### 7.3 Performance Metrics

**Table -6: Performance Evaluation Metrics**

Metric	Target	Achieved
Page load time	< 3 seconds	1.5 - 2.5 seconds
Concurrent users	100+	Tested with 150

		sessions
Database queries/page	< 50	15 - 40 queries/page
PDF generation time	< 5 seconds	2 - 4 seconds
Student capacity	10,000+/branch	Validated

### 7.4 Browser Compatibility

Cross-browser testing confirmed compatibility with Google Chrome 90+, Mozilla Firefox 85+, Microsoft Edge 90+, Safari 14+, and mobile browsers (Chrome and Safari) with a responsive layout.

### 8. CONCLUSIONS

This paper presented SCMS, a comprehensive web-based ERP system for educational institutions built on the PHP CodeIgniter 3 MVC framework. The system successfully integrates 20+ functional modules, 28 payment gateway integrations, multi-channel communication (15+ SMS providers), and RBAC for 7+ user roles.

Key achievements include: layered MVC architecture with 7 base controller classes, dynamic multi-branch database switching, comprehensive RBAC with page and action-level granularity, and modular payment gateway integration across 6 continents.

Testing across 20 test cases confirmed 100% pass rate. Performance metrics demonstrated sub-3-second page loads and support for 150+ concurrent sessions.

### 9. FUTURE SCOPE

In the future, the system can be improved by adding mobile applications, cloud-based services, and advanced analytics. Features such as AI-based predictions and biometric attendance can also be included to enhance functionality.

### ACKNOWLEDGEMENT

The authors would like to thank Rubigha M for their guidance and the Department of Information Technology at Knowledge Institute of Technology for providing the necessary infrastructure for this research.

### REFERENCES

[1] M. A. Khan, S. K. Sharma, "Role of ICT in Education: A Review," International Journal of Computer Science and Information Technologies, vol. 7, no. 2, pp. 800-803, 2016.

[2] Fedena, "Open Source School Management Software," [Online]. Available: <https://fedena.com/>.

[3] OpenSIS, "Open Source Student Information System," [Online]. Available: <https://www.opensis.com/>.

[4] Gibbon, "Flexible, Open Source School Management Platform," [Online]. Available: <https://gibbonedu.org/>.

[5] British Columbia Institute of Technology, "CodeIgniter 3 UserGuide," [Online]. Available: <https://codeigniter.com/userguide3/>.

[6] Oracle Corporation, "MySQL 8.0 Reference Manual," [Online]. Available: <https://dev.mysql.com/doc/refman/8.0/en/>.

[7] Shuttleworth Foundation, "SchoolTool — School Administration Software," [Online]. Available: <https://schooltool.org/>.

[8] I. Back, "mPDF — PHP Library for Generating PDF Files," [Online]. Available: <https://mpdf.github.io/>.

[9] Google, "Firebase Cloud Messaging Documentation," [Online]. Available: <https://firebase.google.com/docs/cloud-messaging>.

[10] Zoom Video Communications, "Zoom API Documentation," [Online]. Available: <https://marketplace.zoom.us/docs/api-reference/>.

[11] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.

[12] OWASP Foundation, "OWASP Web Security Testing Guide," [Online]. Available: <https://owasp.org/www-project-web-security-testing-guide/>.

[13] Stripe, "Stripe API Reference," [Online]. Available: <https://stripe.com/docs/api>.

[14] Twilio, "Twilio SMS API Documentation," [Online]. Available: <https://www.twilio.com/docs/sms>.

[15] Bootstrap, "Bootstrap Documentation," [Online]. Available: <https://getbootstrap.com/docs/>.