EduManage - Student Management System

A Comprehensive Project Report

Submitted by:

1. **Md Shad Alam**
2. **Rajan Kumar**
3. **Md Kashif**

Department of Computer Science & Engineering

Kameshwer Narayan Singh Government Polytechnic College

Abstract

EduManage is a state-of-the-art Student Management System designed to streamline the administrative and academic processes of educational institutions. In an era where digital transformation is paramount, EduManage replaces traditional, paper-based record-keeping with a robust, cloud-ready web application.

This project leverages the most modern web technologies available in 2025, including **React.js** for a dynamic user interface, **Node.js** and **Express** for a high-performance backend, and **PostgreSQL** for reliable data storage. Unlike legacy systems built on PHP or JSP, EduManage offers a Single Page Application (SPA) experience, ensuring instant page transitions, real-time data updates, and a mobile-responsive design.

Key features include a comprehensive Student Dashboard, real-time Attendance tracking with visual analytics, a digital Library Management System, automated Result/Marks calculation, and an AI-powered Chatbot to assist students with common queries. This report details the system's architecture, implementation, and the significant advantages it offers over existing solutions.

Table of Contents

Chapter 1: Introduction

1.1 Background

Educational institutions handle massive amounts of data daily, from student registrations to exam results. Managing this manually is error-prone and inefficient. **EduManage** was conceived to digitize these operations, providing a centralized platform for students, teachers, and administrators.

1.2 Problem Statement

Existing systems in many polytechnic colleges rely on manual registers or outdated software that is slow, non-responsive on mobile devices, and difficult to maintain. Students often have to physically visit notice boards or administrative offices to get information about attendance or marks.

1.3 Objectives

- To develop a **centralized database** for student records.
- To provide **instant access** to attendance and exam results for students.
- To automate **grade calculations** and report generation.
- To simplify **library management** (book issues/returns).
- To implement an **AI Chatbot** for 24/7 student support.

1.4 Scope

The current version of EduManage covers the Academic Module (Attendance, Marks), Library Module, and General Administration (Notices, Student Profiles).

Chapter 2: System Analysis

2.1 Existing System Limitations

- **Manual Errors:** Calculation mistakes in compiling attendance or marks.
- **Data Redundancy:** Same student data repeated across multiple physical registers.
- **Lack of Privacy:** Physical registers are accessible to unauthorized personnel.
- **Time-Consuming:** Retrieving a student's past record takes hours.

2.2 Proposed System Advantages

- **Accuracy:** Automated calculations eliminate human error.
- **Speed:** Instant data retrieval using optimized SQL queries.
- **Security:** Role-Based Access Control (RBAC) ensures only authorized users access sensitive data.
- **Accessibility:** Accessible from any device (Phone, Tablet, Laptop) via a web browser.

Chapter 3: Technology Stack (Why EduManage is Advanced)

EduManage stands out because it rejects outdated technologies (like classic PHP/MySQL or Java Swing) in favor of the **Modern JavaScript Stack (MERN/PERN)**.

3.1 Frontend: React & Vite

- **React.js:** A library for building user interfaces. Unlike traditional HTML pages that reload on every click, React creates a **Single Page Application (SPA)**. This means the page never reloads; content updates instantly, providing an "app-like" feel.
- **Vite:** A next-generation build tool that makes the application start almost instantly. It is significantly faster than older tools like Webpack.
- **TanStack Query:** Manages data fetching. It automatically caches data, so if a student visits the "Marks" page twice, the second load is instant, saving bandwidth.

3.2 Backend: Node.js & Express

- **Node.js:** Allows us to run JavaScript on the server. This unifies the language (JS) across the entire stack, simplifying development.
- **Express:** A minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It handles our API routes efficiently.

3.3 Database: PostgreSQL & Drizzle ORM

- **PostgreSQL:** The world's most advanced open-source relational database. It offers superior data integrity and complex query capabilities compared to MySQL.
- **Drizzle ORM:** A modern TypeScript ORM (Object-Relational Mapper). It allows us to interact with the database using type-safe TypeScript code instead of writing raw

SQL strings. This prevents SQL Injection attacks and reduces bugs.

3.4 UI/UX: Radix UI & Tailwind CSS

- **Tailwind CSS:** A utility-first CSS framework that allows us to build custom designs without leaving your HTML. It ensures the site is fully responsive (looks good on mobile).
- **Radix UI:** A library of unstyled, accessible components. We use this to build complex interactive elements like Dialogs, Dropdowns, and Toasts that work perfectly for screen readers and keyboard navigation.

Chapter 4: System Design

4.1 System Architecture

The system follows a **Client-Server Architecture**.

- **Client:** The React application running in the user's browser.
- **Server:** The Express API running on a Node.js server.
- **Database:** PostgreSQL storing all persistent data.

4.2 Database Schema (ER Diagram Description)

The database is normalized to ensure data integrity. Key tables include:

1. **Students:** Stores personal details (Name, Roll No, Password).
2. **Admins:** Stores administrative credentials.
3. **Subjects:** Courses offered (Code, Name, Instructor).
4. **Attendance:** Links Students and Subjects. Tracks monthly attendance stats (Present Days, Total Days, Percentage).
5. **Marks:** Stores exam results. Includes `testName` , `marksObtained` , `grade` (A, B, C...), and `percentage` .
6. **LibraryBooks:** Inventory of books (Title, Author, Copies Available).
7. **BookIssues:** Tracks which student has which book, issue date, and due date.
8. **Notices:** Broadcast messages from admins to students.

4.3 Data Flow

1. **User Request:** User clicks "View Attendance" on the dashboard.

2. **API Call:** Frontend sends a secure GET request to `/api/attendance`.
3. **Processing:** Server verifies the user's session (Authentication).
4. **Query:** Drizzle ORM translates the request into SQL and queries PostgreSQL.
5. **Response:** Database returns JSON data.
6. **Rendering:** React updates the DOM to display the attendance chart.

Chapter 5: Modules Description

5.1 Authentication & Security

- **Secure Login:** Uses hashed passwords (never stored in plain text).
- **Session Management:** Uses secure HTTP-only cookies to maintain user sessions, preventing XSS attacks.
- **Middleware:** `requireAuth` and `requireAdmin` middleware ensure students cannot access admin features.

5.2 Student Dashboard

The landing page for students. It provides a "Bird's Eye View" of their academic standing.

- **Visual Charts:** Uses **Recharts** to display attendance trends and marks distribution graphically.
- **Quick Stats:** Cards showing "Total Subjects", "Books Issued", and "Pending Assignments".

5.3 Admin Dashboard

A control center for college staff.

- **User Management:** Add/Remove students and teachers.
- **Data Entry:** Interfaces to input daily attendance and exam marks.
- **Notice Board:** Post announcements that instantly appear on student dashboards.

5.4 Attendance Management

- **Monthly Tracking:** Records attendance month-wise.
- **Auto-Calculation:** Automatically calculates the percentage.
- **Status Indicators:** Color-coded status (Green for Good, Red for Short Attendance).

5.5 Marks & Result Analysis

- **Grade Logic:** Automatically assigns grades (A+, A, B...) based on percentage.
- **Performance Tracking:** Students can see their performance curve across different unit tests and semesters.

5.6 Library Management

- **Real-time Availability:** Students can check if a book is available without going to the library.
- **Issue Tracking:** Admins can issue/return books. The system prevents issuing books if a student already has overdue items.

5.7 AI Chatbot

A unique feature of EduManage.

- **Context:** Integrated with OpenAI (simulated/actual) to answer queries like "What is the syllabus for CSE?" or "When are the exams?".
- **24/7 Availability:** Reduces the burden on administrative staff by handling routine queries.

Chapter 6: UI/UX Features (User Interface & User Experience)

- **Responsive Design:** The layout automatically adjusts for Mobile, Tablet, and Desktop screens using Tailwind's responsive prefixes (`md:` , `lg:`).
- **Dark Mode Support:** The UI is designed with contrast in mind, suitable for long study sessions.
- **Interactive Feedback:** Toast notifications (popups) appear for every action (e.g., "Login Successful", "Book Issued").
- **Loading States:** Skeleton screens are shown while data loads, preventing the "jerky" feel of older websites.

Chapter 7: Conclusion & Future Scope

Conclusion

EduManage successfully demonstrates how modern web technologies can transform educational administration. By automating routine tasks and providing transparent access to data, it empowers both students and faculty. The system is secure, scalable, and user-friendly.

Future Scope

- **Payment Gateway:** Integration for online fee payment.
- **Parent Portal:** A separate login for parents to monitor their child's progress.
- **Mobile App:** Wrapping the React app into a native Android/iOS app using React Native.
- **Biometric Attendance:** Integration with fingerprint sensors for automated attendance marking.

This project report serves as a comprehensive guide to the design, development, and implementation of the EduManage system.