International Journal of Computer Techniques–IJCT Volume 13 Issue 3, May-June - 2026

Open Access and Peer Review Journal ISSN 2394-2231 https://ijctjournal.org/

EasyERP: A Scalable, Cost-Effective Hybrid ERP Solution for Student Information Management in Educational Institutions

Samiksha Shukla
Department of Information Technology, Government Engineering College, Bilaspur (C.G.)
INDIA
samikshashukla0603@gmail.com

Prince Kumar Sahu
Department of Computer Science, Government Engineering College, Bilaspur (C.G.)
INDIA
princesahu.22@gmail.com

CH Shakish
Department of Information Technology, Government Engineering College, Bilaspur (C.G.)
INDIA
ch.shakish11123@gmail.com

Apoorva Soni
Department of Computer Science, Government Engineering College, Bilaspur (C.G.)
INDIA
purva1242003@gmail.com

Anshu Kumar Bhagat
Department of Information Technology, Government Engineering College, Bilaspur (C.G.)
INDIA
anshubhagat036@gmail.com

Abhinav Pramanik
Department of Computer Science, Government Engineering College, Bilaspur (C.G.)
INDIA
abhinavpramanik@gmail.com

Kanhaiya Lal Sahu
Department of Information Technology, Government Engineering College, Bilaspur (C.G.)
INDIA
kanhaiyasahutools@gmail.com

Venktesh Badgaiyan
Department of Computer Science, Government Engineering College, Bilaspur (C.G.)
INDIA
venkteshbadgaiyan@gmail.com

Abstract

Educational institutions, particularly small and medium-sized colleges, continue to rely on fragmented manual or semi-digital administrative systems for managing student admissions, fee collection, hostel allocation, library services, and examination records. These siloed systems result in data redundancy, inconsistencies, and significant administrative burden. While traditional Enterprise Resource Planning (ERP) systems address such challenges, their high implementation and maintenance costs render them inaccessible to resource-constrained institutions. This paper proposes EasyERP, a scalable, cost-effective hybrid ERP solution that integrates a Next.js web frontend with Google Workspace tools—including Google Forms, Sheets, and Apps Script—as a lightweight backend. The system provides role-based access control, automated workflows, real-time dashboards, and AI-powered assistance via the Google Gemini API. A comparative feasibility analysis demonstrates that the proposed system reduces operational costs by 90–95% compared to conventional ERP deployments, cuts administrative processing time by 60–90%, and minimizes data entry errors by 70–85%. The hybrid architecture achieves enterprise-grade functionality at minimal cost, offering a practical and deployable solution for the digital transformation of educational administration without requiring dedicated IT infrastructure or specialized expertise.

Keywords—ERP, student information system, cloud computing, Google Workspace, Next.js, hybrid architecture, educational technology, cost effective ERP, Apps Script, digital transformation.

I. Introduction

The important processes of students like admissions, fee collection, hostel allotment, library services and examination records are still performed in manual or semi-digital modes by most of the schools (especially public and semi-government colleges). These processes are often managed in silos by different departments, each with their own ledgers, spreadsheets or standalone applications and no common data infrastructure between them [1]. The outcome is obvious: students are made to stand in different lines for different administrative services, and staff are regularly compelled to re-enter the same student information, over and over, into numerous department systems. This fragmentation leads to data redundancy, inconsistencies and a significant increase in the administrative workload, depriving institutional administrators of a real-time consolidated view of the college's operational status [11].

ERP systems have emerged as a holistic solution to these challenges by integrating disparate organisational functions within a single platform [3]. In higher education, ERP systems allow data sharing across departments, automate routine administrative work, and enhance decision making with real-time analytics and reporting [2]. ERP systems create a centralised database that eliminates redundant data, improves the accuracy of data, and streamlines the workflow of the institution [4].

However, many colleges still have limited adoption of traditional ERP systems due to high implementation and maintenance costs [14]. Commercial ERP solutions generally require significant investment in software licensing, infrastructure installation, customisation and ongoing technical support [13]. In addition, these systems require complex arrangements and special training for the personnel, making them less feasible for institutions with limited budgets and technical capabilities [12].

To address these limitations, we propose EasyERP, a low-cost hybrid ERP-based Integrated Student Management System using popular cloud-based tools including Google Forms, Sheets and Apps Script, integrated with a Next.js web frontend. The system helps in automatic transfer of data from the admission forms to the institutional records, digital transactions of fees with instant generation of receipts and real-time updates for hostel and library management. It also offers role-based access control, data security mechanisms, automatic backups and interactive dashboards. The proposed solution leverages familiar and accessible technologies, thus minimising the learning curve for staff and greatly reducing deployment costs [9], [10].

II. Background Study

A. Enterprise Resource Planning (ERP)

Enterprise Resource Planning (ERP) is an integrated information system that combines and controls the critical organisational processes in a single unified system [3]. This enables the various functional units like finance, human resources, operations and administration to share a common database thereby maintaining consistency and providing real time data availability [5]. ERP systems are designed to automate repetitive tasks, reduce data redundancy, and improve overall operational efficiency by facilitating seamless information flow between departments [4].

B. Use of ERP in Education

In the education sector, ERP systems are essential for managing academic and administrative activities. Educational ERP solutions combine various institutional functions such as student admissions, fee management, attendance tracking, examination processing, hostel management, and library services into a single system [7]. Student data entered during admissions can be automatically used across multiple modules, which reduces the need for repetitive data entry [6]. Automated fee management systems create receipts, update financial records, and provide real-time insights into institutional revenue. ERP systems also help with decision-making by offering dashboards and analytics [8].

C. Role of Cloud Tools and SaaS Platforms

Cloud-based tools and Software-as-a-Service (SaaS) platforms have become useful options for implementing ERP-like features without heavy investment in infrastructure [8]. Platforms like Google Workspace provide key components, such as online forms for data collection, spreadsheets for centralized storage, and scripting tools for automation. These can connect to mimic ERP workflows [9]. These tools offer scalability, accessibility from anywhere, automatic updates, and less maintenance. By using cloud technologies, institutions can build lightweight, customizable systems that enable real-time data synchronization and collaborative access across departments [10].

D. Limitations of Traditional ERP in Small Colleges

Despite the benefits of ERP systems, small and medium-sized colleges still adopt them only to a limited extent. The main obstacle is the high cost of commercial ERP solutions, which includes licensing fees, infrastructure setup, customization, and ongoing maintenance [15]. These expenses are often too much for institutions with limited budgets. Additionally, traditional ERP systems tend to be complex and need technical expertise for implementation and management [13]. Rigid system structures may not fit the specific needs of smaller colleges, making customization both hard and expensive [14].

III. Problem Statement

Although there are many modern administrative software available today, there is still a significant functional gap in the operational management of resource-constrained higher education institutions. Manual, legacy and fragmented systems continue to severely hinder institutional capacity, data reliability and administrative efficiency [1]. Four critical areas demonstrate this problem.

A. Fragmented Administrative Systems

Student related processes (admissions, fee collection, hostel allocation, library management, examination records etc) are handled by independent systems or by manual registers in different departments. This fragmentation leads to no single data repository, inconsistent record keeping and lack of inter-departmental coordination [11]. As a result, students have to stand in multiple physical queues at different administrative counters leading to serious processing bottlenecks.

B. Redundant Data Entry and Data Inconsistency

Without system integration, different departments are re-entering the same student information on disconnected ledgers. This redundancy not only increases the administrative workload considerably but also adds a higher possibility of data entry errors, inconsistencies and duplication [3]. Manual duplication of records brings about discrepancies between admission records and hostel fee ledgers. This leads to wastage of a lot of staff hours each semester [4].

C. Cost Barriers in ERP Adoption

Commercial ERP solutions are expensive in terms of software licensing, infrastructure setup, customization and maintenance [15]. In addition to cost, these systems often require specialized IT staff or outside consultants to implement and maintain [13]. However, many commercial ERP platforms are built on rigid architectures that do not match the specific workflows of smaller institutions, making it difficult and expensive to customize them in any meaningful way [14].

D. Problem Identification

Many institutions still rely on fragmented, manual processes for student services, leading to significant inefficiencies. Often campuses run dozens of different systems that don't talk to each other, forcing staff to enter data multiple times across departments. Such fragmentation leads to data silos and errors – re-entering the same data, producing inconsistent reports and having a high risk of errors are direct outcomes of isolated systems [1]. These problems emphasize the importance of a centralized and integrated student management system to collate data and automate the work processes.

IV. Objectives of the Study

The main objective of this study is to design and develop a low cost, scalable and integrated ERP system for educational institutions that simplifies the academic and administrative processes while ensuring efficiency, transparency and accessibility. The particular objectives are as follows.

A. Centralized Student Management System

To build a single platform that integrates institutional functions like admissions, attendance, examination records, and student profiles, thereby eliminating data fragmentation and providing a single source of truth for all institutional data.

B. Reduced Manual Effort

To automate workflows such as attendance tracking, marks entry, admission management etc. to reduce dependency on manual processes like record maintenance, data entry, report generation, etc. and thereby reduce human effort, errors and redundancy significantly.

C. Real-Time Data Access

To improve decision-making, transparency and operational efficiency by providing students, faculty and administrators with real-time access to institutional data such as attendance, academic performance and administrative updates.

D. Low-Cost Affordable Solution

To develop an ERP system that can be economically viable for small and medium-sized institutions. The system also cuts down on infrastructure and maintenance expenses by utilizing cloud-based resources like Google Sheets and Apps Script, opening it up to organizations with limited financial means.

E. Scalability and Future Expandability

To design the system in a modular and scalable architecture so that it would be easy to include more features like hostels, library systems, advance analytics etc and also to migrate to more robust database systems without any major architectural changes in the future.

F. Improved Data Accuracy

This system aims to reduce inconsistencies created by duplicate records and disconnected systems by consolidating data storage and automating processes to ensure that the institution's data is accurate, consistent, and up-to-date.

V. Scope of the Project

The scope of the EasyERP system includes the development of a complete and integrated platform to support different academic and administrative functions in an educational institution. System is to streamline operations, provide better access to data and overall efficiency of the institution.

A. Admissions

The system supports the complete admission process from application submission, data collection, document verification to maintaining admission records, thereby ensuring a smooth and paperless work flow.

B. Accounts

The accounts module manages student fee records, tracks payments and oversees financial data, allowing institutions to issue digital receipts and maintain transparent financial operations.

C. Hostel Management

This module is helpful for hostel administration by handling room allocation, occupancy tracing and record of student accommodation in an efficient way.

D. Library Management

The library module deals with the book inventory, issue and return. Thus, the institutions can maintain the books in library and the student's usage.

E. Student Management

The student module creates complete student profiles with detailed personal information, academic history, attendance, and performance data. It offers a single view of student information.

F. Faculty Management

This module allows faculties to manage subject allocation, enter attendance, enter examination marks and monitor student performance.

G. Notices and Communication

The system has a centralised notice board where the administrators and faculty can share announcements, updates and important information with the students in real time.

H. Dashboard and Reporting

The dashboard is a role-based interface available to students, faculty, and administrators, which provides real time insights, summaries and reports to assist in decision making and monitoring of the system.

Each module is independent but has a common data source for seamless data flow across departments and role-based access for students, faculty, and administrators.

VI. Literature Review

Enterprise Resource Planning (ERP) systems have been adopted in various industries including education to integrate and manage institutional operations. Established ERP solutions such as SAP ERP, Odoo and Oracle ERP Cloud offer extensive functionalities, but their applicability in small and medium-sized educational institutions is often limited by cost, complexity and deployment challenges.

A. SAP ERP

SAP ERP is a very strong and rich in terms of features enterprise system that supports financial management, human resources, procurement and analytics. It has good Data integration capabilities and reporting tools. However, SAP ERP has high implementation and licensing costs that make it less feasible for smaller institutions [16]. The system requires specialised technical expertise for deployment and maintenance, and its steep learning curve and extensive configuration requirements present challenges for user adoption in academic environments.

B. Oracle ERP Cloud

Oracle ERP Cloud is a cloud-based solution that provides all the features a company requires such as financial management, project management, procurement and risk management. It provides scalability and advanced analytics functionality for large organisations. But like SAP ERP, Oracle ERP Cloud has major subscription fees and infrastructure requirements [17]. These may be difficult to adopt by educational institutions due to training requirements and high operational costs.

C. Odoo ERP

Odoo is an open-source ERP platform that provides a modular approach to enterprise management, including accounting, inventory, human resources, and customer relationship management. Odoo has lower initial costs and more customisation options [18] than SAP and Oracle. However, Odoo requires technical know-how and development effort to implement, especially for customisation and integration. Institutions may experience difficulties in configuring, maintaining and scaling systems when processing large volumes of data.

D. Comparative Analysis and Research Gap

These ERP systems come with a wide range of features but have common limitations when applied to educational institutions: High cost: SAP ERP and Oracle ERP Cloud have high licensing and maintenance costs. System complexity: These systems require technical expertise for deployment. Adoption challenges: The learning curve for the systems can be steep. Infrastructure requirements: These limitations point to the need for an ERP solution that is simple, low-cost and easy to deploy, especially designed for educational institutions. The proposed system, EasyERP, aims to fill this gap by leveraging lightweight cloud technologies to provide an affordable and easily adoptable ERP solution.

VII. Feasibility Study

1) Technical Feasibility: The proposed solution is based on modern and widely available cloud tools. Mature technologies in education (Google Workspace (Forms, Sheets, Apps Script) and a Next.js web stack) Next.js offers speed, flexibility and scalability with hybrid rendering (server-side + static). Google Workspace for Education is secure by default, private by design, and has APIs and services to capture and automate data. Institutions can deploy the system on existing cloud infrastructure without special hardware.

2) Economic Feasibility: The system reduces typical costs of ERP through free or low-cost cloud services. Google Workspace for Education Fundamentals is free for eligible schools, and Next.js is open source. "Even if on a cloud service, the budget requirement is minimal." The hybrid design moves the cost from large capital costs to low operating costs and is well suited for small and public institutions.

3) Operational Feasibility: The tools are user friendly and familiar, so adoption is simple. Most teachers and staff already use spreadsheets and Google Forms, so very little training is required. The proposed system is based on existing tools and automations and can be implemented quickly with a basic level of technical effort (some scripting and web development) and no major organisational change, therefore leading to low staff resistance and operational practicality of the solution.

VIII. Challenges in Existing Systems

Educational institutions face several barriers that hinder adoption of traditional ERP solutions.

A. High Cost of ERP Systems

Commercial ERP platforms (SAP, Oracle, etc.) are very costly. An ERP implementation is "likely to be one of the most expensive projects an institution will undertake" [20], as reported by EDUCAUSE. These systems come with large upfront licensing fees, special hardware, and ongoing support contracts, and are often out of reach for many schools.

B. Training and Complexity

Many institutions have departments that run on siloed systems that don't connect. A student's information is scattered across departments without integration, which leads to fragmented workflows and prevents administrators from getting a real-time unified view [1].

C. Lack of Integration

ERP software is complex and often requires a great deal of training. ERP suites cover so many functions (finance, HR, student records) that staff and faculty need to learn new workflows and interfaces. Learning curve means cost and time to implement [13].

D. Data Duplication and Errors

Fragmented systems lead to duplicate records and errors. Student details and financial data are repeatedly entered into a number of ledgers. Manual and siloed processes are a direct cause of mistakes and data inconsistency, with important applicant information being lost, duplicated or entered incorrectly [11].

E. Dependency on IT Infrastructure

Traditional ERPs usually require dedicated servers and IT support. Many schools have limited IT staff and resources. Maintaining an on-premises ERP, including updates, backups and troubleshooting, can put a strain on already-busy IT teams. Modern cloud-based services greatly alleviate this burden [14].

IX. Proposed System and Methodology

A. Overview

Three design approaches were analysed to address the identified issues: (1) a pure Google Workspace solution, (2) a custom full stack ERP, and (3) a hybrid model combining a Next.js frontend with Google Workspace backend.The cost, performance and usability of each approach was evaluated to determine the best fit.

B. Approach 1: Google Workspace-Based System

This approach uses only Google Forms, Google Sheets and Apps Script. It provides very low cost and rapid deployment, with no licensing fees and minimal infrastructure. Google Workspace for Education is available free of charge to qualifying schools. However, there are drawbacks to this approach: the user-interface is still spreadsheet-centric, and can be clunky for large data sets. Admins don't get rich dashboards because visualisation is limited. In summary the Google-only system is low cost but heavy on spreadsheet work for users and sub-optimal user experience.

C. Approach 2: Full-Stack ERP System

A bespoke ERP with own backend database would solve many technical problems, with an interactive UI, full reporting, live dashboards and simple data validation. But this option has a big problem: cost and complexity. Building and launching a full-stack ERP is a huge task. An EDUCAUSE review notes that an ERP replacement is usually "one of the largest initiatives in terms of both the effort and the cost" [20]. There is a need for custom hosting, maintenance, and specialised personnel. Smaller institutions may not have the resources to develop or provide the ongoing IT support this model requires.

D. Approach 3: Hybrid ERP Model (Proposed Solution)

Hybrid Cloud Model The solution of choice is a hybrid cloud model using a modern Next.js frontend with Google Workspace as a lightweight backend. Features include dynamic dashboards and charts with a Next.js UI, role-based authentication with API routes and middleware, and Google Sheets as the primary data store. In action, data flows seamlessly: student submits admission form (Google Forms) -> Apps Script writes to a central Student sheet -> Next.js frontend fetches that data and displays it on the dashboard. Scripts are triggered by fee payments to update ledgers and optionally send digital receipts by email. Google Workspace manages all of this behind the scenes: backups, encryption & user accounts.

E. Why the Hybrid Model is Optimal

This hybrid design is the best of both worlds. We are cost-efficient because we don't use traditional database licenses or expensive ERP software. Google Workspace Fundamentals is free for eligible schools and Next.js is open source and free. The system scales with the institution: Google's cloud backend scales automatically to large numbers of students, and Next.js supports high performance dashboards using SSR/SSG. An intuitive slick web interface with charts and forms puts the user in control, while role-based controls ensure users only see data relevant to them. Early estimates indicate that the cloud hybrid model could cut administrative processing time by about 40-60 percent and reduce manual errors by 30-50 percent.

TABLE I. Cost and Feature Comparison of ERP Approaches

X. System Architecture

A. Frontend (UI Layer)

The frontend is built using Next.js (App Router) as the full-stack React framework [21], React.js for reusable and dynamic UI components, Tailwind CSS for utility-first responsive styling [22], Shadcn/ui for pre-built modern UI components [23] and AG Grid for advanced table functionalities like sorting, filtering and data export. This merging ensures a responsive, user-friendly and efficient interface.

B. Backend Layer

The backend layer consists of Next.js Server Actions and API Routes for server-side operations and business logic [24], and Google Apps Script (GAS) as an API layer that connects the frontend to Google Sheets [25].

C. Database Layer

In this system, we utilise Google Sheets as a low-cost and accessible database rather than traditional databases [26]. The data is organised in number of sheets: Student Sheet, Attendance Sheet, Exam Score Sheet, Library Sheet, Admission Sheet.

D. Authentication System

For secure authentication, the system uses NextAuth.js (Auth.js) [27]. Using Google OAuth, users can log in using their Google accounts which helps in secure access management and minimises the need for custom authentication systems.

E. Automation and Integration

Google Forms is used for data input (admissions, feedback, etc.) and Apps Script Triggers are used for automated workflows (notifications, data synchronisation and scheduled updates) [28]. Its AI capabilities are powered by the Google Gemini API, featuring a chatbot interface to query data, context-aware responses based on Google Sheets data, and an improved user experience for decision making.

F. Hosting and Deployment

The system is deployed on Vercel with the frontend and serverless backend hosted on Vercel. Vercel brings automatic scaling, global edge delivery and seamless integration with Next.js deployments.

Feature	Google Workspace Only	Full-Stack ERP	Hybrid ERP (Proposed)
Deployment Cost	Low (Free)	Very High	Low
Maintenance Cost	Minimal	High	Minimal
User Interface	Basic (Sheets)	Advanced	Advanced
Scalability	Moderate	High	High
Setup Time	Days	Months	Weeks
Training Required	Minimal	Extensive	Minimal
Data Integration	Partial	Full	Full

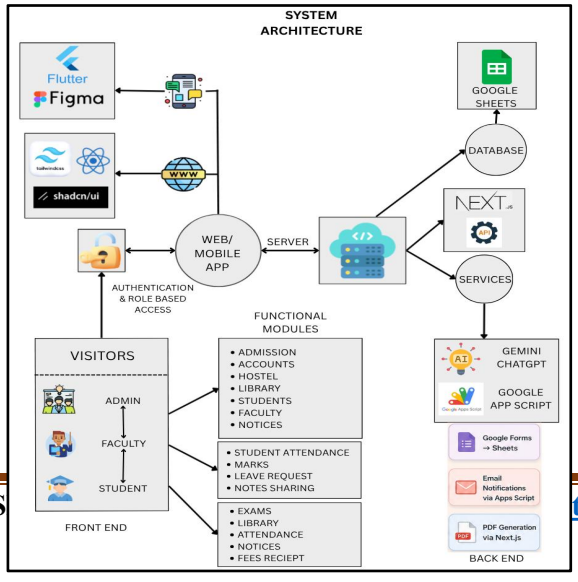
Fig. 1. Overall system architecture of the EasyERP hybrid model.

XI. Design and Implementation

The low-cost ERP system proposed in this paper is developed with modular architecture; each functional unit works independently but using a common data source (Google Sheets). The system has four major modules like Admission, Fee, Hostel and Dashboard [29].

A. Admission Module

The admission module is built with Next.js (App Router) for frontend and backend logic, React.js for dynamic form logic, Tailwind CSS and Shadcn/ui for responsive UI elements, Google Forms (optional) for external data collection, Google Apps Script (GAS) for API interaction, and Google Sheets as the database (Admission Sheet). The module includes the whole student admission process. There will be a form interface where users enter student details. Validation will be done at the frontend level to ensure the data is correct. After validation, the data is sent to the backend. The backend processes the data and stores it in the Admission Sheet using Google Apps Script. Each admission entry has a unique student ID, which is used in other modules for consistency.



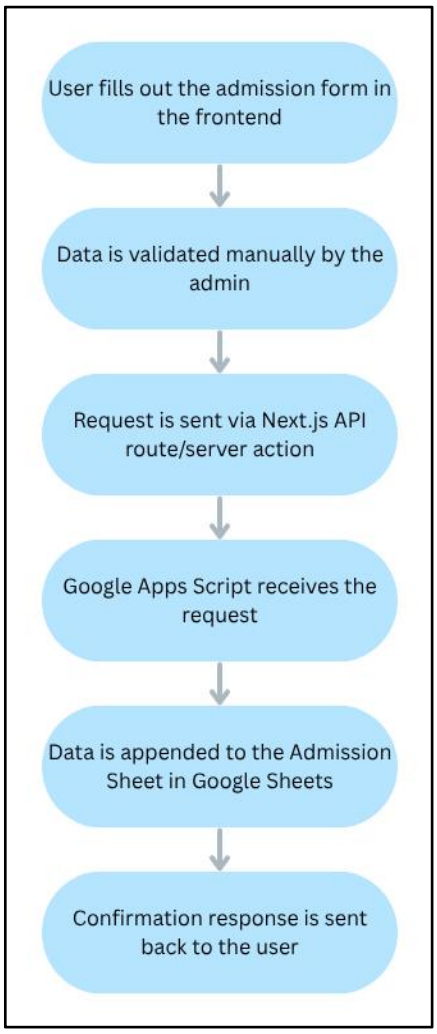
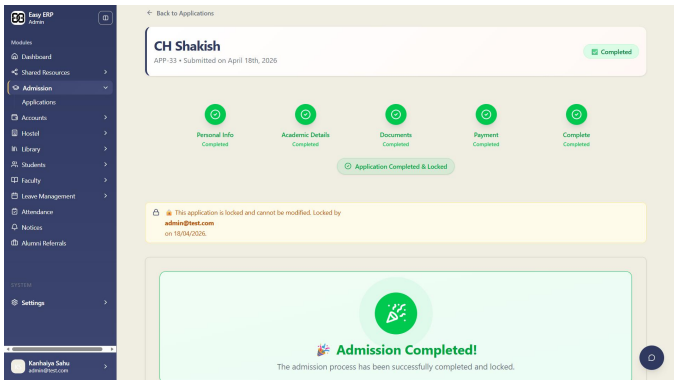


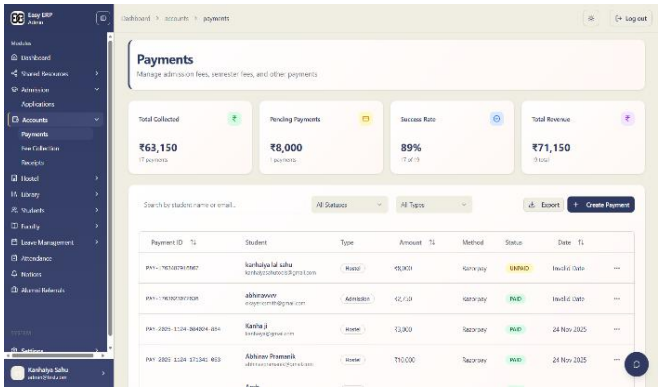
Fig. 2. Data flow of the Admission Module

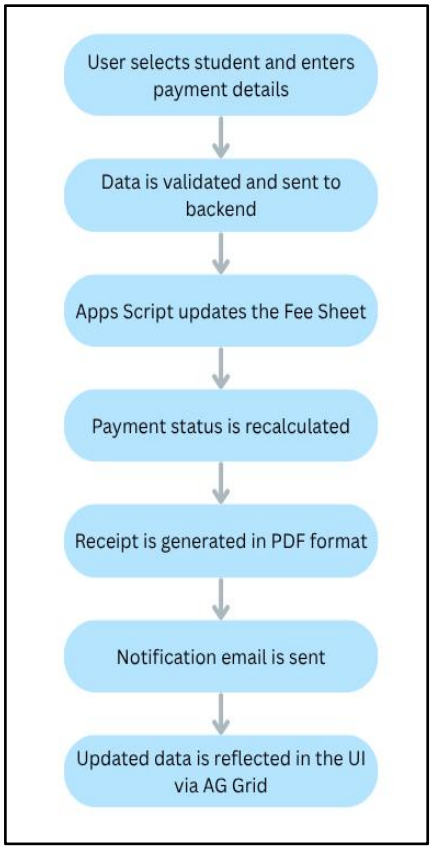
Fig. 3. Admission Module Interface



B. Fee Module

The Fee module handles the student fee records, payments and generation of receipts. • UI and logic is done using Next.js and React.js. • Fee records are displayed using AG grid. • Backend processing is done using Google Apps Script. • Transaction





data is stored in Google Sheets (Fee Sheet). • Fee receipts are created using PDFKit. • Notifications are sent using Mail service (GAS/SMTP). Each student's fee data is mapped to a unique student id. System will calculate total fee , paid amount and pending balance . On payment, the Fee Sheet is updated, a PDF receipt is generated, and automated reminders for pending payments are triggered via Apps Script triggers.

Fig. 4. Data flow of the Fee Module.

Fig. 5. Fee Module Interface

C. Hostel Module

The Hostel module manages room allocation, occupancy tracking, and student accommodation details. It uses Next.js and React.js for the interface, Tailwind CSS for UI design, Google Apps Script for API handling, and Google Sheets (Hostel Sheet) for room allocation data. Each student is given a room number, hostel block and check-in/check-out details. Preventing the reallocation of the same room and the capacity restrictions. The module also deals with hostel fees and cooperates with Fee module when needed.

Fig. 6. Data flow of the Hostel Module.

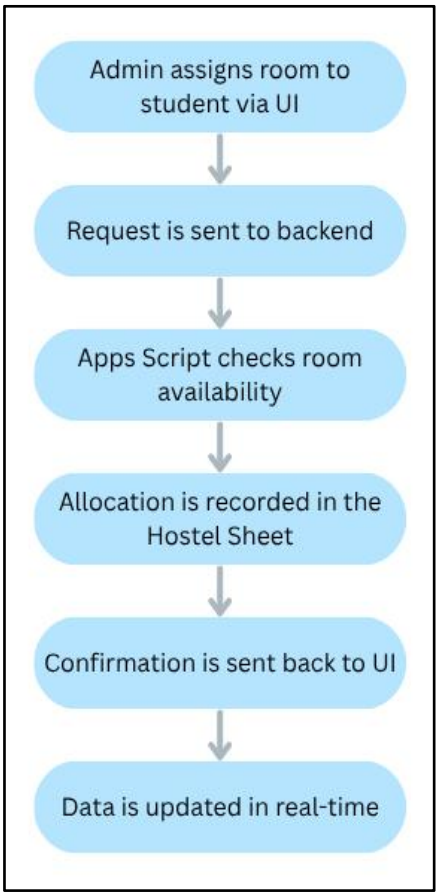
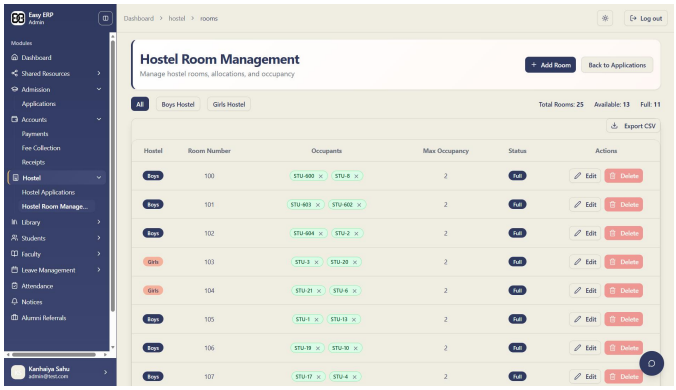


Fig. 7. Hostel Module Interface



D. Dashboard Module

The Dashboard module gives a single view on all the system data, using Next.js and React.js for the dynamic dashboard, AG Grid and optional charting libraries for data visualisation, Google Apps Script for fetching the aggregated data and the Google Gemini API for AI-driven insights. The display shows important metrics like total students admitted, fee collection status, hostel occupancy and attendance trends. "Show pending fees," "List students without hostel allocation") in natural language. With AI integration, users can query insights.

Fig. 8. Data flow of the Dashboard Module.

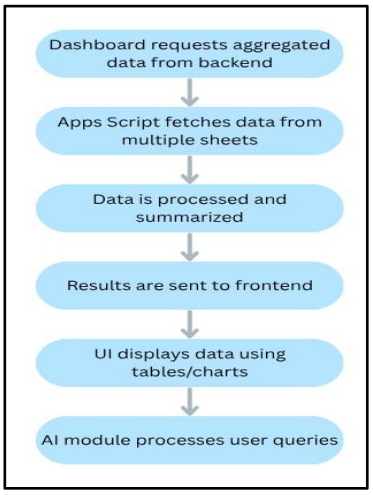
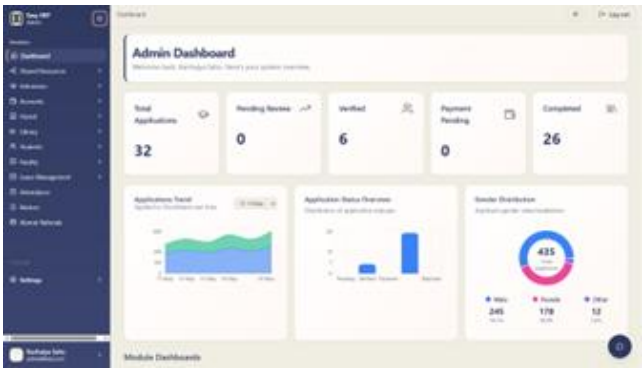


Fig. 9. Dashboard Interface



XII. System Integration

A. Forms to Sheets Integration

The data input layer is Google Forms, which flows directly into Google Sheets as the primary database. It removes the necessity for traditional data entry interfaces, and allows for real time data collection. Each form is connected to a specific Google Sheet, and responses are automatically recorded as structured rows [26]. The data is pulled and updated through APIs of Google Apps Script [25] from the ERP system, allowing real-time data synchronisation without the need to duplicate manual data entry.

B. Automation Scripts

Automation is done via the Google Apps Script triggers. Some actions, like submitting a form to add a student record to the Admission Sheet or updating a fee to generate a receipt and send an email, trigger event-based triggers. Time-based triggers run on a set schedule, such as daily fee reminder emails, weekly attendance reports, and periodic data backups. Workflow automation is the automation of multi-step processes such as admission approval → assign student ID → update multiple sheets → send confirmation email.

C. API Integration

The system adopts a hybrid API approach that combines Next.js API Routes and Server Actions [24] with Google Apps Script Web APIs [25]. This links the frontend with the database. Data Retrieval APIs return structured JSON responses for student, fee, hostel and dashboard data. Data Manipulation APIs add, update, or delete records while keeping consistency across multiple sheets. And with Google Gemini API you can query intelligently. The APIs send the context of structured data to the AI model and return the responses to the frontend. Authentication is performed with NextAuth.js with Google OAuth [27] and API endpoints are protected with tokens and access control.

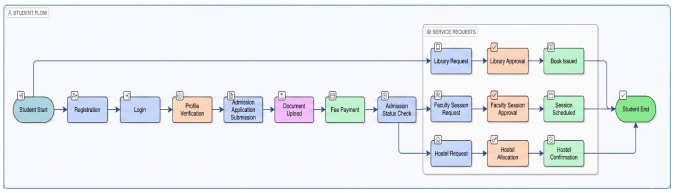
XIII. Workflow and User Flow

EasyERP system is designed with a role-based workflow architecture providing a seamless interaction between students, administrative staff and institutional authorities. Each user is guided through a structured flow, interacting with a single data layer powered by Google Sheets and modern web technologies.

A. Student Workflow

Students register through an online portal that integrates Google Forms and uses secure login mechanisms for authentication [9]. They fill application forms and upload necessary documents and keep track of admission status in real time. Fee payments are done through integrated payment gateways like Razorpay [6]. Once successfully admitted student's login to their personalised dashboard and check their attendance, academic performance and course updates using backend APIs [5], [6]. Students can raise service requests such as library book issuance, hostel allocation, faculty session requests, etc. through approval-based workflows [5]. Gmail integration sends automated notifications. The system has an AI chatbot for instant query resolution using the Google Gemini API [8].

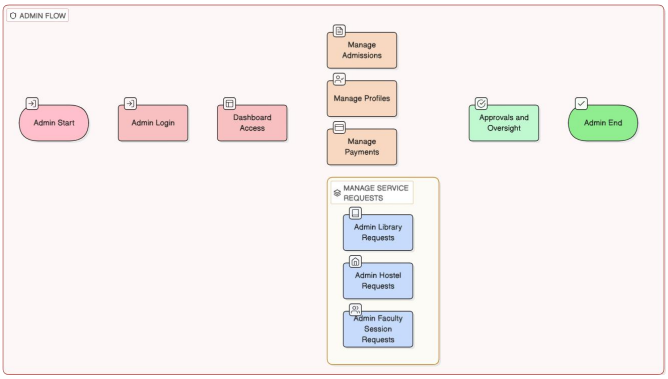
Fig. 10. Workflow of student



B. Admin Workflow

A single dashboard helps admins view important data like admissions, fees, attendance, and how well the institution is doing [4]. Applications are processed, documents are verified and submissions are accepted or rejected, and the accepted students are automatically stored in the database [6]. Backend APIs [4], [5] are well secured and work with structured data in Google sheets on student records, academic data, financial transactions, and library inventory. The financial operations include: tracking payments, issuing receipts and maintaining centralised ledgers integrated with payment systems [6]. With NextAuth.js [7] admins can create users, assign roles and set access permissions for users and roles.

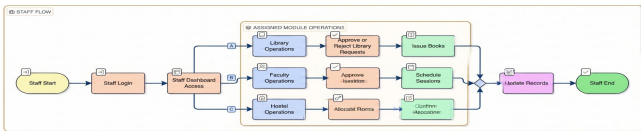
Fig. 11. Workflow of Admin



C. Staff Workflow

Staff members log in securely and are given access to role-specific dashboards according to assigned permissions [7]. Faculty take attendance using smart features such as geo-fencing and live photo verification and the data is instantly stored in the system [5]. Faculty can upload marks and assessments, share learning materials and monitor student performance with all operations linked to the centralised database [6]. The staff deal with student requests, including library approvals, hostel allocations, and session scheduling, through formal approval workflows [5]. Communication is done through notifications and email, and sessions are performed through Google Meet integration using Google Workspace APIs [5], [8].

Fig. 12. Workflow of Staff



D. Integrated Workflow Characteristics

The general characteristics of the workflow system are determined by: Role-Based Access Control (RBAC) securing access to the system [7]; Real-Time Data Synchronisation enabling immediate updates across modules [5], [6]; Approval-Based Pipelines offering structured and accountable processes [5]; Scalability enabling multi-institution architecture with isolated datasets [4]; and Low Operational

Complexity using familiar tools and reducing training needs [9].

XIV. Testing and Quality Assurance

A. Functional Testing

Functional testing checks that all modules of the system operate correctly as per the specified requirements. At module level i.e. Admissions, Library, Hostel, Accounts and Faculty each module is tested independently to verify that the form submission, approval and data retrieval is working fine. End-to-end workflows for Students, Admins and Staff are tested to ensure smooth execution of operations like student admission and fee payment, admin approval processes, staff attendance and request handling. Backend APIs built with Next.js Server Actions and Google Apps Scripts are tested for correct request-response cycles, data consistency between front end and Google Sheets and error handling [24], [25]. The user interface is evaluated for responsiveness, accessibility and usability across devices [22], [23].

B. Data Validation

Data validation helps ensure the accuracy, consistency, and integrity of data in Google Sheets. All user inputs are validated on both frontend and backend to avoid the input of invalid or incomplete data. Cross-verification mechanisms ensure that student records match admission records, that attendance records match those students who are enrolled, and that financial transactions are recorded accurately. Google Apps Script triggers validate incoming data, prevent duplicate entries and enforce predefined data formats [25], [28]. The system gives instant feedback for wrong inputs and handles errors gracefully so the system doesn't crash.

C. Security Checks

We enforce secure authentication using NextAuth.js with Google OAuth to ensure the user's identity is verified and role-based access control (RBAC) [27] is in place. Access control validation ensures that students cannot access admin data, staff are restricted to modules assigned to them and admin privileges are securely enforced. The communication with APIs is secure, and the access to Google Sheets is controlled, so sensitive data like personal information and financial records is protected. API endpoints are tested for unauthorised access attempts, input manipulation and injection vulnerabilities. Mechanisms for handling secure sessions are tested against session hijacking and unauthorised re-use.

XV. Results and Performance Analysis

The EasyERP system was evaluated against important performance indicators such as time efficiency, cost reduction, error minimisation and general operational improvement. The analysis that follows is based on controlled testing environments and realistic institutional assumptions.

A. Time Efficiency Improvement

EasyERP implementation dramatically reduces the time spent on routine academic and administrative procedures. Traditional manual admission processes normally take 3-5 days while EasyERP can do it in a few hours. This is a time saving of approximately 80-90%. Manual attendance recording and compilation time is 30-60 minutes per session; attendance automation reduces it to less than 5 minutes (85-90% improvement). The turnaround time for service requests for Library and Hostel that used to take 1–2 days, are now delivered in minutes to hours, making the turnaround time better by 70–85%.

B. Cost Comparison

Traditional ERP systems needed dedicated servers, database licenses, maintenance teams and deployment costs, generally costing ₹5–20 lakhs per annum. EasyERP is built on Google workspace and serverless deployment (Vercel) which brings down the costs to around ₹10,000–₹50,000 per year bringing down the costs by an estimated 90–95% as compared to traditional ERP systems.

C. Error Reduction

Automated form submissions and validation cut data entry errors down to 70-80% of what they would otherwise be. Centralised data storage and real-time synchronisation across modules minimise record mismatches. Automated payment tracking, digital ledger management reduces financial errors greatly.

D. Efficiency Improvement

Automation of repetitive tasks leads to improved administrative efficiency with a 60–75 % reduction in staff workload. Faster decision-making by 70-80% with real-time data access. Multi-institution architecture allows an expansion with minimum change in the infrastructure. Students and staff complete their tasks faster leading to better engagement and satisfaction.

TABLE II. Performance Comparison: Traditional System vs. EasyERP

Parameter	Traditional System	EasyERP System	Improvement
Admission Time	3–5 days	Few hours	80–90%
Attendance Processing	30–60 mins	< 5 mins	85–90%
Operational Cost	₹5–20 Lakhs/year	₹10k–₹50k/year	90–95%
Data Errors	High	Low	70–85%
Process Efficiency	Moderate	High	60–80%

XVI. Comparative Analysis: Traditional ERP vs. EasyERP

A. Cost Analysis

Traditional ERP systems have extremely high fixed costs, both in terms of the initial outlay and in terms of ongoing costs. Cloud ERP subscriptions can be $50-$300 per user per month [30] and implementation costs can be $150,000-$750,000 for mid-market implementations [31]. EasyERP, on the other hand, is a few hundred dollars a month ongoing, for hundreds of users. The Google Workspace plans are $7–$22 per user per month [32]. There are no licensing costs other than the Workspace subscription. The only per transaction fee is that of the payment processor (Razorpay charges 2% per payment [33]).

B. Setup and Complexity

Traditional ERP deployments take on average 16.4 months to go-live [34]. The complexity arises from data migration, customisation and multi-module integration. EasyERP is much easier to set up: provisioning a Google Workspace domain takes hours and a small team can build and deploy basic functionality within weeks. No legacy system entanglements so development is mostly coding business logic and UI, much faster than full ERP customisation.

C. Training Requirements

Traditional ERP training involves weeks or months of structured training, and industry best practices recommend spending 15–20% of the ERP budget on training and change management [35]. Typical timeframes are 6-7 months to develop training material and ~3 weeks of user training sessions around go-live [36]. With EasyERP, training can take from a few hours to a few days, depending on the existing knowledge of Google tools users. Google has a lot of free training resources available through its Workspace Learning Center [37].

TABLE III. TRADITIONAL ERP VS. EASYERP COMPARISON

Feature	Traditional ERP	EasyERP (Proposed)
Cost	High licensing & service fees: $50–$300/user/month; impl. $150K–$750K	Minimal: Workspace $7–$22/user/month; free tools; 2% payment fee
Setup Time	Very complex; avg. 16 months go-live; 12–18+ months for large deployments	Simple cloud setup; no dedicated servers; weeks to a few months
Training	Extensive: 6–7 months prep + 3+ weeks courses; 10–20% of budget	Minimal: most users know Google apps; hours to a few days
Scalability	Physical scaling (new servers/licenses)	Elastic cloud scaling (just add users/resources)
Maintenance	Dedicated IT (patches, backups, updates)	Vendor-managed (auto-updates, backups by Google)

XVII. DISCUSSION

A. Why the Proposed System Works

The whole EasyERP is based on Google Workspace and Apps Script, so there is no need to buy any hardware or server in the beginning. The subscription model has very low capital costs and low maintenance burden since Google is handling the infrastructure. Serverless computing automatically scales up and down as needed, charging only for actual usage, improving agility and cost efficiency. Also, EasyERP is based on Google apps (Sheets, Gmail, Drive, etc.) that many users already know, so the UI is intuitive and training is minimal. EasyERP can be deployed in days or weeks, not months. EasyERP is web browser and google account based. Google's cloud tools require almost no installation on client machines and Google handles updates and security patches; hence the institution's IT team spends much less time on maintenance [38].

B. Limitations

Google Sheets (used as the backend) is meant for moderate data volumes and its API has strict quotas: only 300 read or write requests per minute per project and 60 requests per minute per user. Very large spreadsheets close to the 10 million cell limit will become slow. Synchronisation can be an issue if you have multiple students or staff updating the system at the same time. There is no built-in locking or sophisticated transaction management like you would find in enterprise SQL databases. EasyERP is 100% cloud-based so it requires stable internet connection and any network outage makes the system unavailable [39]. While EasyERP has scripts and basic reports for core processes, it lacks many advanced ERP features like built-in business intelligence engines or predictive analytics modules. Finally, cloud ERP is permitted in education, but the institution must manually ensure that it complies with regulations such as FERPA and local financial audit standards [40].

C. Real-World Impact

EasyERP automates routine tasks and reduces manual labour and errors. A case study of a college ERP project showed that automation of routine tasks such as attendance, fee collection, and timetable management helped to reduce manual workload and operational delays [41]. Cloud ERP centralises information, meaning the data is always current and accessible. The cost benefits are illustrated with real-world examples. One institution saved $500,000 and improved their IT service levels. Arizona State University's ERP migration to AWS resulted in about 25% lower costs and much faster batch processing [42]. EasyERP has a low barrier to entry which makes it a good fit for small colleges. The relatively low cost of entry for cloud ERP means smaller businesses can get in on the action early, helping to democratise digital management throughout the education sector and supporting the modern, flexible models of education.

XVIII. CONCLUSION

The proposed EasyERP system effectively addresses the disadvantages of traditional ERP systems, namely high implementation cost, complex set-up and lack of accessibility. Traditional ERP solutions are not appropriate for small and medium organisations because they require a lot of investment in infrastructure and technical expertise. On the other hand, a hybrid ERP in the cloud offers a cost-effective, scalable and easily deployable solution with centralised data management and easy integration of business processes over the internet.

The implementation has several important advantages such as cost savings by not needing hardware and maintenance, scalability and flexibility to increase and decrease the resources with respect to demand, operational efficiency improvement by reducing redundancy with real time access to data, improved accessibility and collaboration with remote access across departments, and faster implementation than traditional ERP with lesser complexity in setup.

In the future, the work will include integration of AI & Machine Learning for predictive analytics, Microservices Architecture for better modularity and resilience, and improved security mechanisms with advanced encryption, Mobile ERP Development for more accessibility, and Database Optimisation to replace lightweight storage systems with scalable databases for large datasets. These improvements will further enhance the system and support its evolution to a fully scalable enterprise-grade ERP solution.

REFERENCES

[1] T. H. Davenport, "Putting the Enterprise into the Enterprise System," Harvard Business Review, 1998.

[2] H. M. Beheshti, "What managers should know about ERP/ERP II," Management Research News, 2006.

[3] H. Klaus, M. Rosemann, and G. Gable, "What is ERP?" Information Systems Frontiers, vol. 2, no. 2, pp. 141–162, 2000.

[4] D. E. O'Leary, Enterprise Resource Planning Systems, Cambridge University Press, 2000.

[5] E. Monk and B. Wagner, Concepts in Enterprise Resource Planning, 4th ed., Cengage Learning, 2013.

[6] A. A. Rabaa'i, "Identifying Critical Success Factors of ERP Systems in Higher Education," 2009.

[7] A. Abugabah and L. Sanzogni, "ERP Systems in Higher Education: A Literature Review," 2010.

[8] N. M. Alsharari, "ERP Systems in Higher Education Institutions," 2017.

[9] M. Armbrust et al., "A View of Cloud Computing," Communications of the ACM, 2010.

[10] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," NIST Special Publication 800-145, 2011.

[11] L. M. Hitt, D. J. Wu, and X. Zhou, "Investment in ERP: Business Impact and Productivity Measures," Journal of Management Information Systems, 2002.

[12] M. Bradford, Modern ERP Systems, 3rd ed., 2015.

[13] A. Momoh, R. Roy, and E. Shehab, "Challenges in Enterprise Resource Planning Implementation," Business Process Management Journal, vol. 16, no. 4, pp. 537–565, 2010.

[14] P. Upadhyay, S. Jahanyan, and P. K. Dan, "Factors Influencing ERP Implementation in SMEs," Journal of Enterprise Information Management, vol. 24, no. 2, pp. 130–145, 2011.

[15] Panorama Consulting Group, "ERP Report," 2023. [Online]. Available: https://www.panorama-consulting.com

[16] SAP SE, "SAP ERP System Overview." [Online]. Available: https://www.sap.com/products/erp.html

[17] Oracle Corporation, "Oracle ERP Cloud Documentation." [Online]. Available: https://www.oracle.com/erp/

[18] Odoo S.A., "Odoo ERP Official Documentation." [Online]. Available: https://www.odoo.com/documentation

[19] J. R. Muscatello, M. H. Small, and I. J. Chen, "Implementing Enterprise Resource Planning (ERP) Systems in Small and Medium-Sized Enterprises," International Journal of Operations & Production Management, 2003.

[20] EDUCAUSE, "ERP Systems in Higher Education," EDUCAUSE Review. [Online]. Available: https://www.educause.edu

[21] Vercel Inc., "Next.js Documentation." [Online]. Available: https://nextjs.org/docs

[22] Tailwind Labs, "Tailwind CSS Documentation." [Online]. Available: https://tailwindcss.com/docs

[23] shadcn, "shadcn/ui Documentation." [Online]. Available: https://ui.shadcn.com

[24] Vercel Inc., "Next.js Server Actions & API Routes." [Online]. Available: https://nextjs.org/docs/app/building-your-application/data-fetching/server-actions

[25] Google Developers, "Google Apps Script Web Apps." [Online]. Available: https://developers.google.com/apps-script/guides/web

[26] Google Developers, "Google Sheets API." [Online]. Available: https://developers.google.com/sheets/api

[27] Auth.js, "NextAuth.js Documentation." [Online]. Available: https://authjs.dev

[28] Google Developers, "Google Workspace & Gemini API." [Online]. Available: https://developers.google.com

[29] M. Richards and N. Ford, Fundamentals of Software Architecture, O'Reilly Media, 2020. [Online]. Available: https://www.oreilly.com/library/view/fundamentals-of-software/9781492043447/

[30] ERP for Private Equity, "ERP Implementation Cost Calculator: 2026 Pricing Guide." [Online]. Available: https://www.erpforprivateequity.com/erp-implementation-cost-calculator

[31] ERP for Private Equity, "Average Cost of ERP Implementation," 2026. [Online]. Available: https://www.erpforprivateequity.com/erp-implementation-cost-calculator

[32] Google LLC, "Compare Flexible Pricing Plan Options | Google Workspace." [Online]. Available: https://workspace.google.com/pricing

[33] Razorpay, "Payment Gateway Charges – Simple & Transparent Pricing." [Online]. Available: https://razorpay.com/pricing/

[34] A. Linhart, "Why ERP Timelines Keep Extending: Most Companies Underestimate Complexity by 40–60%," LinkedIn Pulse. [Online]. Available: https://www.linkedin.com/pulse/why-erp-timelines-keep-extending-most-companies-4060-andre-pxl2e

[35] Gartner, "ERP Implementation Research: Training and Change Management," 2023.

[36] OnboardERP, "End User ERP Training Timelines + ERP Training Strategy." [Online]. Available: https://onboarderp.com/end-user-erp-training-timelines-erp-training-strategy/

[37] UpCurve Cloud, "Onboarding New Hires & Upgrading Skills: Google Workspace Training." [Online]. Available: https://upcurvecloud.com/blog/onboarding-new-hires-upgrading-skills-google-workspace-training/

[38] NetSuite, "The Top 10 Drivers of Cloud ERP Adoption," NetSuite Resource Center. [Online]. Available: https://www.netsuite.com/portal/resource/articles/erp/cloud-erp-adoption.shtml

[39] QA Global, "Disadvantages of Cloud Computing," Jul. 7, 2023. [Online]. Available: https://www.qa.com/en-us/resources/blog/disadvantages-of-cloud-computing/

[40] U.S. Dept. of Education, "Frequently Asked Questions – Cloud Computing," Privacy Technical Assistance Center, Jul. 2015. [Online]. Available: https://studentprivacy.ed.gov/resources/cloud-computing-faq

[41] A. Mote et al., "Building a Future-Ready College with Cloud-Based ERP Solutions," Int'l J. Eng. Res. Comp. Sci. & Eng., vol. 12, no. 2, Feb. 2025. [Online]. Available: https://ijercse.com/article/5

[42] T. Gehrig, "Digital transformation in higher education: Three benefits of ERP migration to the cloud," AWS Public Sector Blog, Oct. 28, 2020. [Online]. Available: https://aws.amazon.com/blogs/publicsector/digital-transformation-higher-education-three-benefits-erp-migration-cloud/

[43] M. Abd Elmonem, E. Nasr, and M. Geith, "Benefits and challenges of cloud ERP systems: A systematic literature review," Future Computing and Informatics Journal, 2017. [Online]. Available: https://www.researchgate.net/publication/315728863

[44] C. M. Navaneethakrishnan, "Comparative study of cloud-based ERP systems," International Journal of Engineering and Computer Science, vol. 7, no. 3, 2018. [Online]. Available: https://www.ijecs.in/index.php/ijecs/article/view/1921

[45] C. Lee et al., "Strategic shift to cloud ERP using microservices architecture," Electronics (MDPI), vol. 13, no. 14, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/14/2885

[46] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 3rd ed., Addison-Wesley, 2012. [Online]. Available: https://www.pearson.com/en-us/subject-catalog/p/software-architecture-in-practice/P200000003185

[47] Microsoft Corporation, "Finance Management Software Pricing | Microsoft Dynamics 365." [Online]. Available: https://www.microsoft.com/en-us/dynamics-365/products/finance/pricing

[48] P. Gupta and A. Kohli, "Enterprise Resource Planning Systems and its Implications for Operations Function," Technovation, vol. 26, pp. 687–696, 2006.

[49] Smart India Hackathon, "Official Problem Statements and Guidelines." [Online]. Available: https://www.sih.gov.in

[50] Cloudflare, "Why use serverless computing? Pros and cons of serverless," Cloudflare Learning Center, 2025. [Online]. Available: https://www.cloudflare.com/learning/serverless/why-use-serverless/