Tribhuvan University

Faculty of Humanities and Social Sciences

"AcademiaPlus"-Academic Management System

A PROJECT REPORT

Submitted to

Department of Computer Application

Lumbini City College

In partial fulfillment of the requirements for the Bachelors in Computer Application

Submitted by

Suman Khatri

(T.U.Reg.No: 6-2-1134-60-2020)

Chitra Bahadur Thapa

(T.U.Reg.No: 6-2-1134-42-2020)

Under the Supervision of

Dinesh Neupane

SUPERVISOR'S RECOMMENDATION

I hereby recommend that this project prepared under my supervision by Suman Khatri and Chitra Bahadur Thapa entitled "**AcademiaPlus**" in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

……………………

Dinesh Neupane

SUPERVISOR

Faculty of Humanities and Social Sciences

Lumbini City College

Tilottama-4, Drivertole, Rupandehi

Tribhuvan University

Faculty of Humanities and

Social Sciences

Lumbini City College

LETTER OF APPROVAL

This is to certify that this project prepared by Suman Khatri and Chitra Bahadur Thapa entitled "**AcademiaPlus**" in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

Mr. Dinesh Neupane Supervisor Lumbini City Collage	Mr. Bishal Kandel Program Coordinator Lumbini City Collage
Internal Examiner Lumbini City Collage	External Examiner FOHSS, Tribhuvan University

ABSTRACT

The AcademiaPlus application is a comprehensive platform designed to streamline and enhance the management of educational institutions, including schools, colleges, and universities. By integrating essential functions such as student registration, class management, grading, attendance tracking, and student management, AcademiaPlus centralizes academic workflows into a unified system. This integration provides real-time access to academic records and fosters effective communication between students, faculty, and administrators. The system's ability to automate administrative tasks improves efficiency, reduces workload, and enables more accurate data-driven decision-making.

AcademiaPlus supports institutions by offering data analytics for performance tracking, which assists in monitoring academic progress and informs decision-making processes at various levels. Its scalability and adaptability make it a valuable solution for institutions of different sizes and complexities. By promoting a more organized and efficient academic environment, AcademiaPlus enhances both institutional governance and student outcomes.

As educational institutions evolve, the implementation of AcademiaPlus becomes increasingly critical in addressing the dynamic needs of modern academia. By automating routine processes and centralizing critical data, AcademiaPlus reduces administrative burdens, improves communication, and contributes to the growth and success of educational institutions.

Keywords: *attendance tracking, data analytics*

ACKNOWLEDGEMENT

We are obliged to a number of people who helped organize this project, and we are thankful for their kind opinions, suggestions, and appropriate guidance. We have received endless support and encouragement in finalizing our project, and we would like to take this opportunity to thank them all.

First and foremost, we would like to express our special thanks of gratitude to our supervisor, Dinesh Neupane, for giving us the honor to work on this wonderful project on the topic of the Academic Management System Application, which has greatly enhanced our knowledge. We are also thankful for his continuous feedback and encouragement throughout the project. Without his guidance, this project would not have been accomplished. We have learned many new things while working on this project, and for that, we are deeply thankful to him.

Secondly, we are overwhelmed with all the support and guidance we received from our parents and friends. Their assistance and innovative ideas have been invaluable in helping us complete this project satisfactorily. We are truly grateful for their help in making this project a success.

Lastly, we would like to extend our gratitude to everyone who has encouraged and supported us through the challenges we faced during the completion of this project.

Suman Khatri and Chitra Bahadur Thapa

TABLE OF CONTENTS

SUPERVISOR’S RECOMMENDATION i

LETTER OF APPROVAL 2

ABSTRACT.................... iii

ACKNOWLEDGEMENT iv

TABLE OF FIGURES viii

LIST OF TABLES ix

CHAPTER 1: INTRODUCTION 1

1.2 Problem Statement 1

1.3 Objective 2

1.4 Scope and Limitation 2

1.5 Development Methodology.................... 4

1.6 Report Organization 5

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW 6

2.1 Background Study:.................... 6

2.2 Literature Review:.................... 7

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN.................... 9

3.1 System Analysis 9

B. Non-Functional Requirements: 10

3.1.2 Feasibility Analysis 10

3.1.3 Data Modelling: ER Diagram 11

3.1.4 Process Modelling: DFD.................... 12

B. Level 1 Data Flow Diagram 13

C. Level 2 Data Flow Diagram 13

3.2 System Design.................... 14

3.2.1 Architectural Design 14

3.2.2 Database Schema Design 15

3.2.3 Algorithms 16

Chapter 4: Implementation and Testing.................... 17

4.1 Implementation 17

4.1.1 Tools Used.................... 17

4.1.2 Implementation Details of Modules 17
4.2 Testing 18
4.2.1 Test Cases for Unit Testing 18
4.2.2 Test Cases for System Testing 19
Chapter 5: Conclusion and Future Recommendations 20
5.1 Conclusion 20
5.2 Future Recommendations 20
REFERENCES 22
APPENDICES 23

LIST OF ABBREVIATIONS

• API:	Application Programming Interface
• CSS:	Cascading Style Sheets
• DBMS:	Database Management System
• HTML:	Hypertext Markup Language
• JS:	JavaScript
• JWT:	JSON Web Token
• UI:	User Interface
• UX:	User Experience

TABLE OF FIGURES

Figure 1 Waterfall Methodology....5
Figure 2 Use Case of Academic Management System Application....9
Figure 3 Gantt Chart....11
Figure 4 ER-Diagram of Academic Management System Application....11
Figure 5 Zero Level DFD....12
Figure 6 First Level DFD....13
Figure 7 Second Level DFD....13
Figure 8 Architecture Design....14
Figure 9 Database Schema Design....15

LIST OF TABLES

Table 1 Test Cases for Unit Testing 18
Table 2 Test Cases for System Testing..19

CHAPTER 1: INTRODUCTION

1.1 Introduction

An Academic Management System is a digital platform designed to optimize the administrative and academic functions of educational institutions. With the increasing complexity of managing academic workflows in schools, colleges, and universities, AMS serves as a vital tool to efficient processes such as student enrollment, course management, attendance tracking, grading, faculty coordination, and exam scheduling. By consolidating these tasks into a centralized system, AMS enhances efficiency, reduces manual errors, and ensures that students, faculty, and administrators have real-time access to important academic information. As education continues to embrace digital transformation, AMS plays a crucial role in facilitating a more organized, data-driven, and collaborative academic environment. The development of these applications often leverages frameworks such as React, Material-UI for the frontend. Node.js and Express.js for backend while using databases like MongoDB to store histories and user data.

1.2 Problem Statement

Educational institutions face numerous challenges in managing their academic processes efficiently. Traditional methods often rely on manual record-keeping and disjointed systems, leading to several key issues:

i. Attendance Management**:** Automate attendance tracking to reduce errors and enhance student access to records.

ii. Marks Management: Provide transparency in grades and feedback for students to track their academic progress.

iii. User Authentication: Implement a secure, role-based authentication system to protect sensitive data.

iv. Feedback System: Establish a formal mechanism for students to provide feedback on courses and teaching.

1.3 Objective

The objectives of an Academic Management System are designed to enhance the efficiency and effectiveness of educational institutions.

i. To centralize and streamline academic data management.
ii. To enable seamless communication among stakeholders.
iii. To automate routine administrative tasks.
iv. To track and support student performance.

By achieving these objectives, an Academic Management System aims to enhance the overall educational experience, improve institutional efficiency, and contribute to the academic success of students.

1.4 Scope and Limitation

Scope of an Academic Management System

i. Student Management: AMS facilitates the management of student information, including personal details, enrollment, grades, attendance, and academic history.
ii. Course and Curriculum Management: It allows administrators to create, modify, and manage course offerings, syllabi, and class schedules.
iii. Faculty Management: AMS enables institutions to manage faculty profiles, assign teaching roles, track attendance, and monitor performance.
iv. Examination and Grading: The system automates exam scheduling, grading, and result generation, reducing manual errors and delays.
v. Attendance Tracking: AMS tracks student attendance and provides automated reports to faculty and administrators.

vi. Communication and Collaboration: It provides communication tools for students, faculty, and administrators to share announcements, assignments, and feedback.
vii. Fee Management: The system helps track tuition and other fees, enabling efficient financial management and payment processing.
viii. Analytics and Reporting: AMS generates real-time data analytics and reports to support decision-making and track institutional performance.

Limitations of an Academic Management System

i. Initial Setup Costs: Implementing an AMS can be expensive, requiring investments in software, hardware, and training.
ii. Customization Complexity Institutions may have unique requirements, and the system may require extensive customization to meet those needs.
iii. Data Security and Privacy: AMS handles sensitive academic and personal data, making it vulnerable to cyber threats if not properly secured.
iv. Dependency on Internet Connectivity: Many AMS platforms are cloud-based, which may limit functionality in regions with poor internet access.
v. Technical Support: Continuous technical support is necessary for system maintenance, troubleshooting, and upgrades, which can be resource-intensive.
vi. User Adaptability: Faculty, staff, and students may face a learning curve when transitioning to a new system, which could initially impact productivity.
vii. Integration with Other Systems: Difficulty in integrating AMS with existing systems (such as Learning Management Systems, ERP software) may pose challenges for institutions with pre-established technology infrastructure.
viii. The scope and limitations of AMS must be carefully considered to ensure it meets the specific needs of the institution while addressing any potential drawbacks.

1.5 Development Methodology

We have used the Waterfall Methodology to build the Academic Management System application. This approach is structured and linear, meaning we complete each development phase before moving to the next. It involves detailed planning and documentation at each step, ensuring all requirements are clearly defined upfront. The Waterfall model includes sequential phases: requirements gathering, system design, implementation, integration and testing, deployment, and maintenance. This method helps keep the project organized and focused, with clear goals for each stage.

i. Requirement: In this phase, we have collected and document all the key requirements for the Academic Management System application. This includes features like real- time messaging, user authentication, and security. We have also analyzed any specificuser needs and define a clear project scope before proceeding to the next phase.

ii. System Design: After finalizing the requirements, we move to system design. This involves creating detailed plans for the application's architecture, database structure, user interfaces, and workflows. Key components like the messaging system, authentication, and user interface is carefully designed to meet the project's requirements.

iii. Implementation: Once the design is finalized, developers start coding the Academic Management System application. They build the frontend and backend based on the design documents, focusing on creating the user interface and adding real-time messaging features.

iv. Integration and Testing: After building the application, we have integrated its components and conduct thorough testing for functionality, security, and performance. Any bugs or issues is identified and fixed to ensure the application meets the defined requirements.

v. Deployment: After successful testing, the Academic Management System application is deployed to the production environment. At this stage, the application is released to the users, and final configurations are made to ensure smooth operation.

vi. Maintenance: After deployment, the application enters the maintenance phase. Any issues or bugs that arise be fixed, and minor updates or enhancements can be made based on user feedback. Major changes usually require starting a new project cycle.

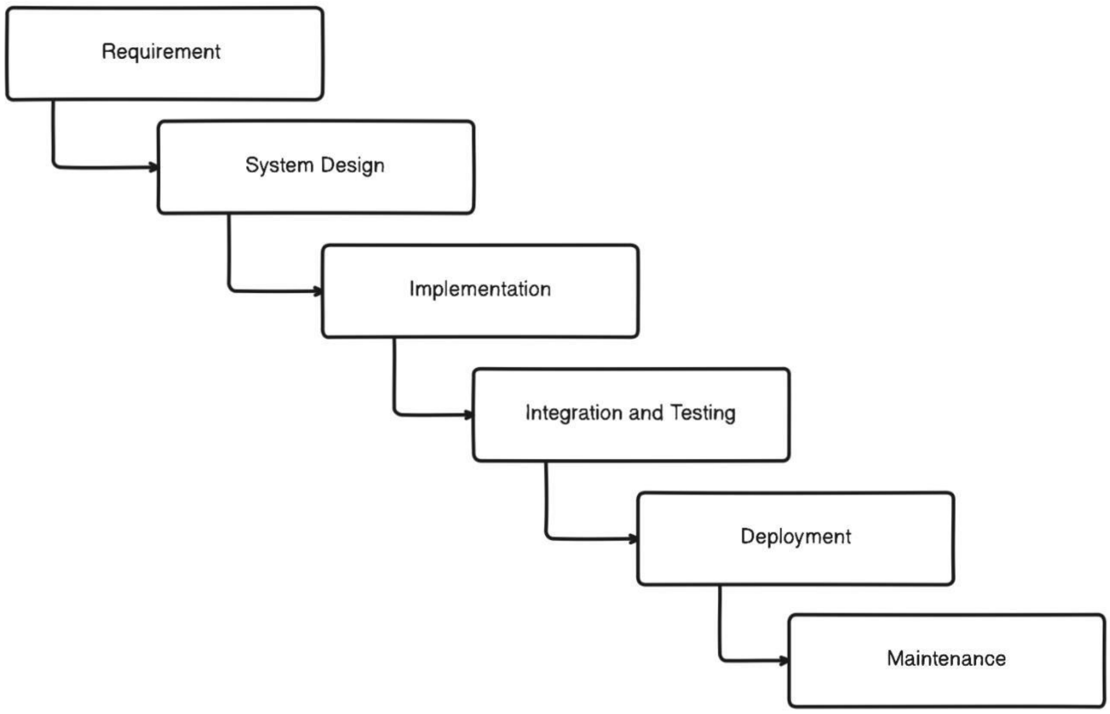


Figure 1 Waterfall Methodology

1.6 Report Organization

This report is divided into five chapters, each with specific headings. The preliminary section includes general project information such as the abstract, table of contents, list of figures, and abbreviations.

- **Chapter 1** introduces the Academic Management System application, detailing the problem definition, objectives, scope, and limitations of the system.
- **Chapter 2** provides a background study and literature review, summarizing existing research related to Academic Management System applications and relevant technologies.
- **Chapter 3** covers the analysis and design of the system, including information about the current systems, data collection methods, system analysis, feasibility study, and system configuration. It also presents the overall system architecture, class diagrams, sequence diagrams, and database diagrams.
- **Chapter 4** details the development models, implementation techniques, tools used, and test cases for system testing.
- **Chapter 5** discusses the future scope of the project, recommendations for future improvements, and provides a conclusion.

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study:

Academic Management System is a software solution designed to streamline and automate the administration of educational institutions, encompassing key activities such as student enrollment, course management, faculty coordination, grading, and attendance tracking. Historically, academic institutions relied on manual processes for these operations, which were often time-consuming, error-prone, and inefficient. With the growing scale and complexity of modern educational systems, the need for a digital transformation became imperative. The development of an Academic Management System leveraging modern web technologies is a strategic approach to creating a robust, efficient, and scalable platform for educational institutions. In this system, React is used for the frontend, providing a dynamic and responsive user interface that enhances user experience through its component-based architecture. Material-UI, a popular React framework, is integrated for pre-built and customizable UI components, ensuring a visually appealing and consistent design.

On the backend, Node.js and Express form a powerful combination for handling server-side logic and APIs, offering fast, event-driven performance. This enables the AMS to process requests and manage operations such as student records, course data, and faculty management efficiently. The system's database is powered by MongoDB, a NoSQL database that offers flexibility in storing unstructured data, allowing the system to manage complex datasets like course structures, attendance logs, and grading records with ease.

For secure authentication, JSON Web Token (JWT) is implemented, ensuring that user sessions are safely managed and protected from unauthorized access. This is particularly crucial when handling sensitive information such as student data and academic results. Additionally, Axios is used for handling HTTP requests, enabling smooth communication between the frontend and backend, while Mongoose, an Object Data Modeling (ODM) library, simplifies interaction with MongoDB by providing schema-based models for data handling.

Together, these technologies create a comprehensive AMS that is scalable, secure, and responsive, meeting the growing needs of modern educational institutions.

2.2 Literature Review:

The College Management System (CMS) is an advanced web-based application designed to streamline administrative and academic processes within educational institutions. Developed using the Django framework for backend functionalities and HTML, CSS, and JavaScript for frontend presentation, the CMS offers a comprehensive solution for managing various tasks efficiently. Leveraging SQLite as the databadminisratiase management system ensures robust data management capabilities, scalability, and ease of maintenance. The CMS provides a wide array of features to automate routine tasks and enhance collaboration among students, faculty, and administrators. Student admission and enrollment management, course scheduling, curriculum planning, and faculty assignment are seamlessly handled by the system, reducing manual workload and minimizing errors. Integrated messaging systems and announcement boards facilitate communication, fostering a collaborative learning environment [1].

This project is aimed at developing a College Management Information System (CMIS) that is of importance to either an educational institution or a college. It is difficult to prepare the manual work to store the information about the all students, teachers as well as about workers. This system can be used as a knowledge/information management system for the college. So this project helps to store those type of information using computerized system. The title of the project is “COLLEGE MANAGEMENT INFORMATION SYSTEM” (CMIS). CMIS is an Intranet based application that aims at providing information to all the levels of management within an organization. This system can be used as a information management system for the college. The front-end will be HTML pages with Java Script for client side validation where as all business logics will be in Java reside at middle layer. And these layers will interact with third layer of database, which will be MS-access database. The web server will be Glassfish. To start working on this project environment required is a server having Glassfish as web server, MS-access as database and Java Runtime Environment (JRE) as development environment [2].

College Data Management System gives a straightforward interface to support of understudy data, staff information,attendance,fee record. It very well may be utilized by instructive establishments or schools to keep up the records of understudies without any problem. The creation and the executives of exact, modern data with respect to an understudies' scholastic profession is fundamentally significant in the college just as universities. Understudy data framework manages all sort of understudy subtleties, scholarly related reports, school subtleties, course subtleties, educational program, cluster subtleties, position subtleties and other asset related subtleties as well [3].

The college management system based an the particular College related and information about the college. The college Management system in focused an the college history and Events programs, placement of the information .The programs module can be subdivided an the two part UG, PG. The UG means undergraduate related an the course can be divided in four .The each and ever course can be included an the Subject name of the courses . the PG course means PostGraduate it can be subdivided in three parts .The PG also have each course and subject name also .The Placement means who are recruiters in our college recruiter in details available here .News and Events module focused an the college related information [4].

This research paper introduces a novel approach to School Management Systems (SMS), addressing existing gaps in current systems. The proposed SMS emphasizes a comprehensive solution, focusing on student attendance and performance beyond traditional marks and assignments. By incorporating a seamless user interface and providing access to video lectures for each subject, the SMS aims to enhance the learning experience. Furthermore, the system includes intuitive attendance tracking and detailed performance assessment features, ensuring a holistic approach to academic operations. This research highlights the potential for SMSs to improve educational institutions' administrative tasks, communication, and academic performance, while also addressing the need for tracking missed lectures. The proposed SMS offers a more robust and comprehensive solution for managing educational institutions' administrative and academic needs. Keywords— School Management System (SMS), Academic Management, Performance Evaluation, Missed Lectures, Attendance Tracking, Technology Acceptance Model (TAM) [5].

University students can have low academic flow when using a Learning Management System (LMS). Three variables are predicted to correlate with the academic flow (FA) of students who use LMS: academic self-efficacy (ASE), academic resilience (AR), and LMS use intensity (LMSI). This study looks at the link between academic self-efficacy, academic resilience, LMS use intensity, and academic flow among university students who use LMS. This study employs a quantitative approach, using correlational approaches and path analysis [6].

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

3.1 System Analysis

3.1.1 Requirement Analysis

A. Functional Requirements:

Functional requirements for Academic Management System Application might include:

- User Management: Role based access for different users.
- Attendance Tracking: Monitoring and reporting attendance.
- Analytics and Reporting: Dashboards for performance tracking and customizable reports.
- Subject Management: Creation and management of subjects.

Functional Requirement can be expressed in Use Case for

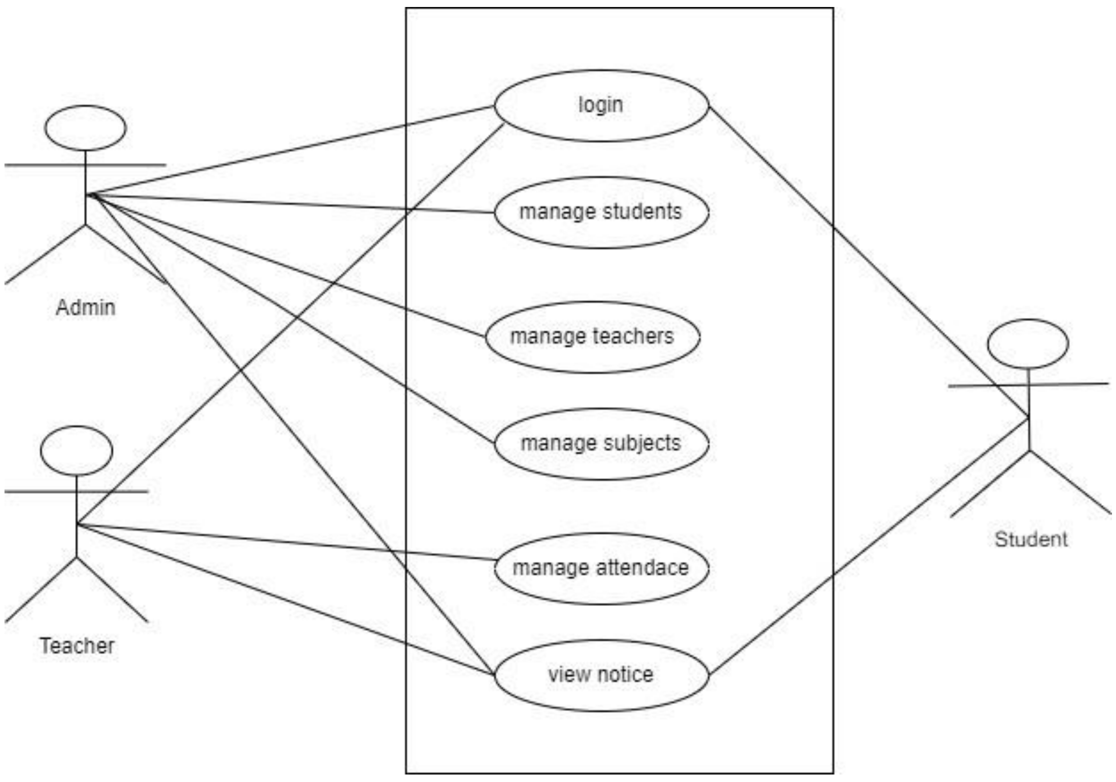


Figure 3. 1 Use-Case

B. Non-Functional Requirements:

Non-functional requirements are the characteristics or qualities of a Academic Management System that cannot be directly observed by end-users but are still crucial for the application's overall performance and user experience. Here are some non-functional requirements for a Academic Management System application:

i. Security measures for data protection: Implement encryption and secure authentication protocols to protect user data and conversations from unauthorized access.

ii. User-friendly and responsive design: Ensure the academic management system is intuitive, easy to navigate, and responsive across various screen sizes and devices for a seamless user experience.

iii. Availability, usability, and scalability: The Academic management system must be available at all times, easy to use, and able to scale efficiently as the number of users grows.

iv. High-performance system with low latency: Messages should be delivered and received instantly with minimal delay, ensuring smooth and fast real-time communication.

v. Compatibility with different browsers and devices: The application should work consistently across major browsers (e.g., Chrome, Firefox) and on various devices, including desktops, tablets, and smartphones.

3.1.2 Feasibility Analysis

i. Technical Feasibility:

Technical feasibility assesses whether the current infrastructure can support the academic management system application. It identifies the necessary software tools, procedures, and inputs required for development. The academic management system is designed to be user-friendly and fast, reducing maintenance costs. With its efficient processing speed, the system is considered technically and operationally feasible.

i. Operational Feasibility: Operational feasibility for an Academic Management System refers to its ability to function effectively and efficiently in the current technological environment. A clear operational plan is essential for smooth operation and user satisfaction, ensuring the system meets user needs efficiently.

ii. Economic Feasibility: Economic analysis is used to evaluate the effectiveness of the Academic Management System application by comparing its costs and benefits.

The application is economically feasible as it doesn't require additional hardware resources and saves significant time, making it a cost-effective solution

iii. Schedule Feasibility:

This includes the project schedule and all time allocated for their completion. The Gantt chart is as follow:

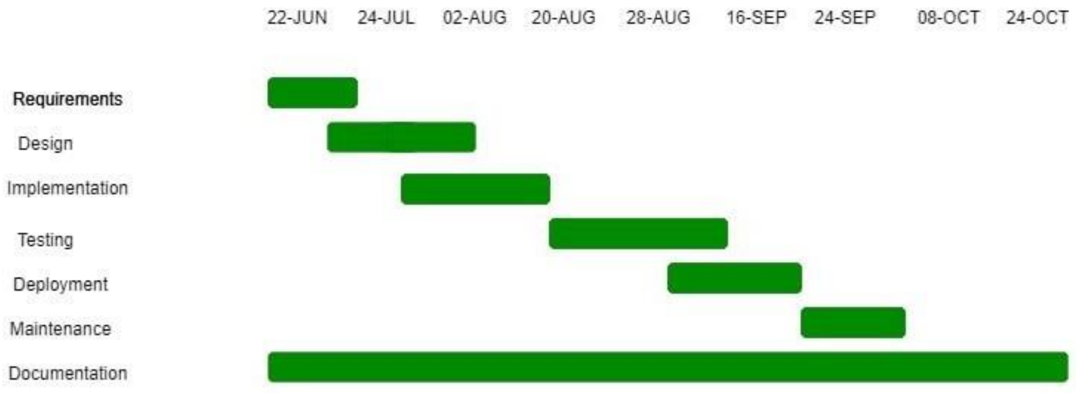


Figure 2 Gantt Chart

3.1.3 Data Modelling: ER Diagram

An ER diagram, also known as ER model, is a graphical representation of entities and their relationships to each other. Typically, it's used in computing regarding the organization of data within database or information system. The basic components of an ER diagram are entities, attributes, and relationship between and among those entities.The ER diagram for the AcademiaPlus is as follows.

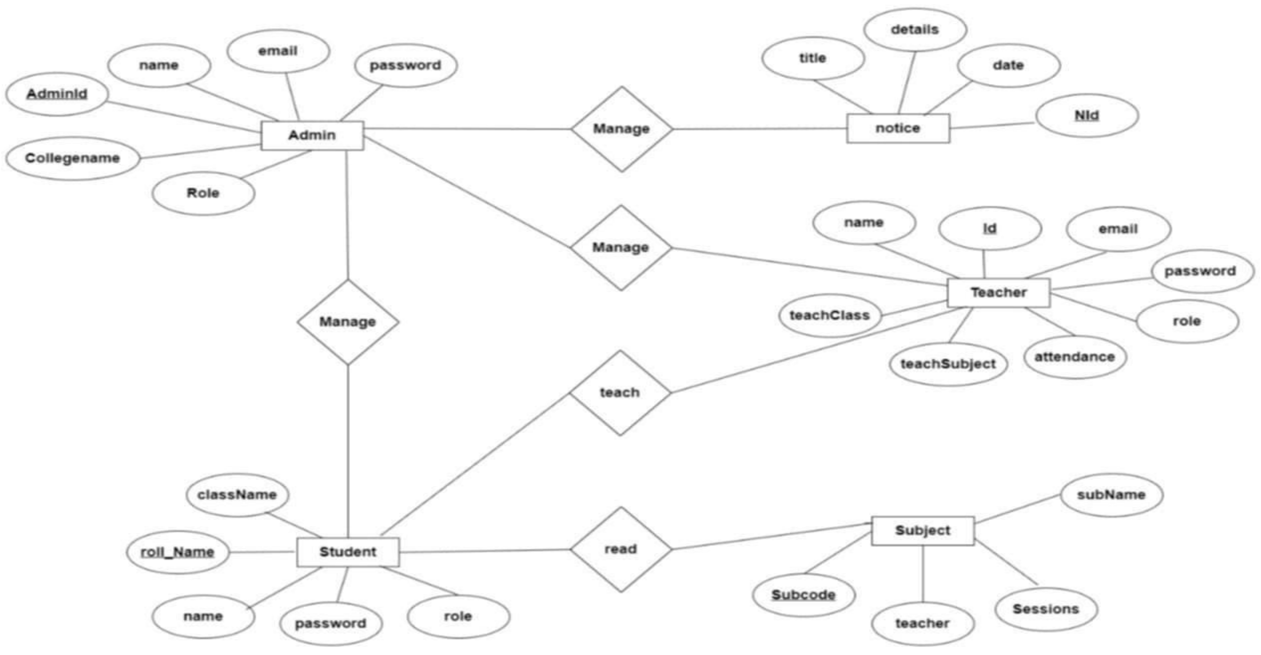


Figure 3. 2 ER-Diagram

3.1.4 Process Modelling: DFD

A. Zero level DFD (Context Diagram)

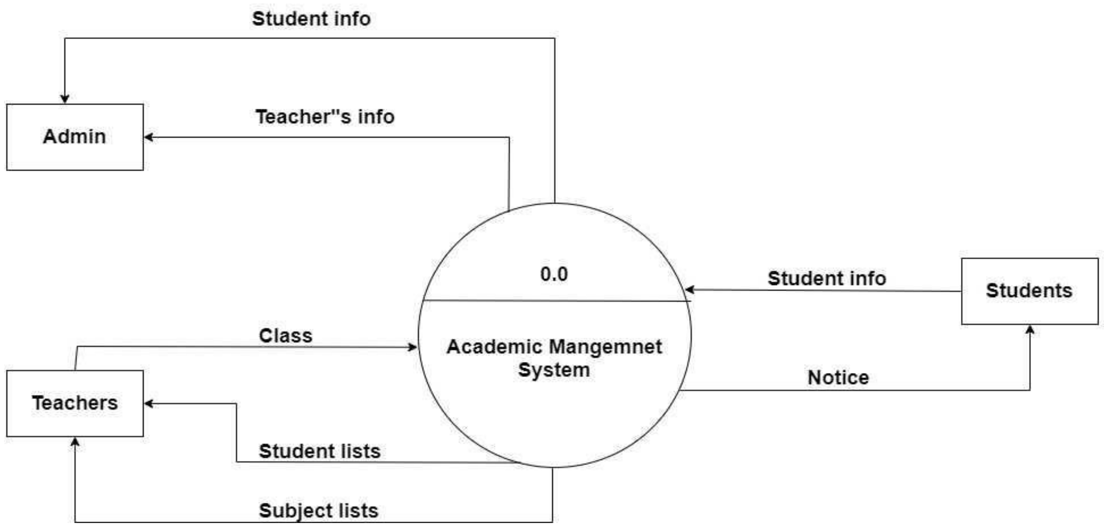


Figure 3. 3 Zero Level of DFD

B. Level 1 Data Flow Diagram

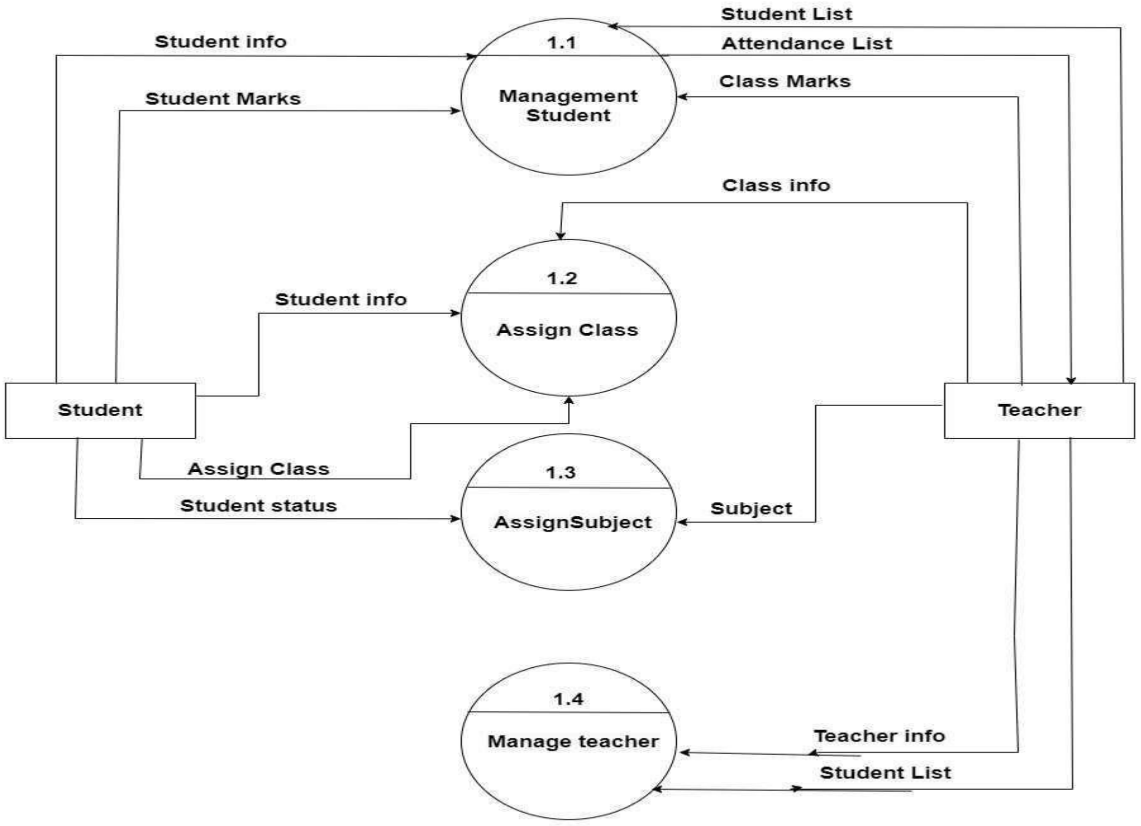


Figure 3. 4 Level 1 DFD

C. Level 2 Data Flow Diagram

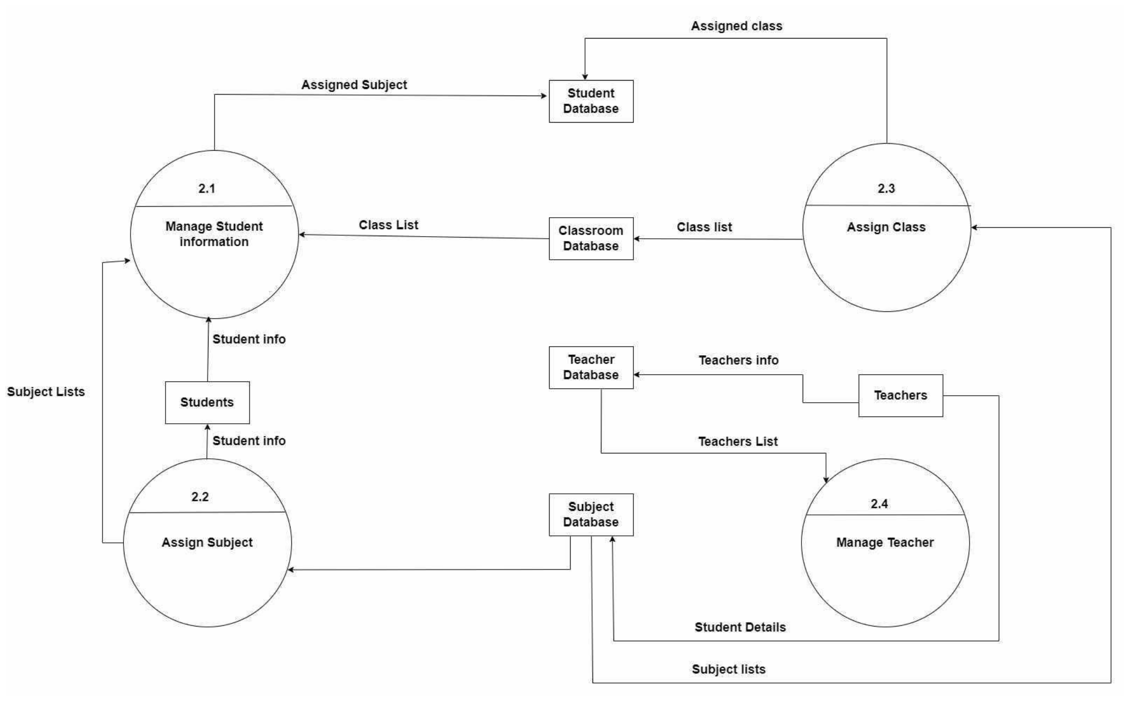


Figure 3. 5 Level 2 of DFD

3.2 System Design

3.2.1 Architectural Design

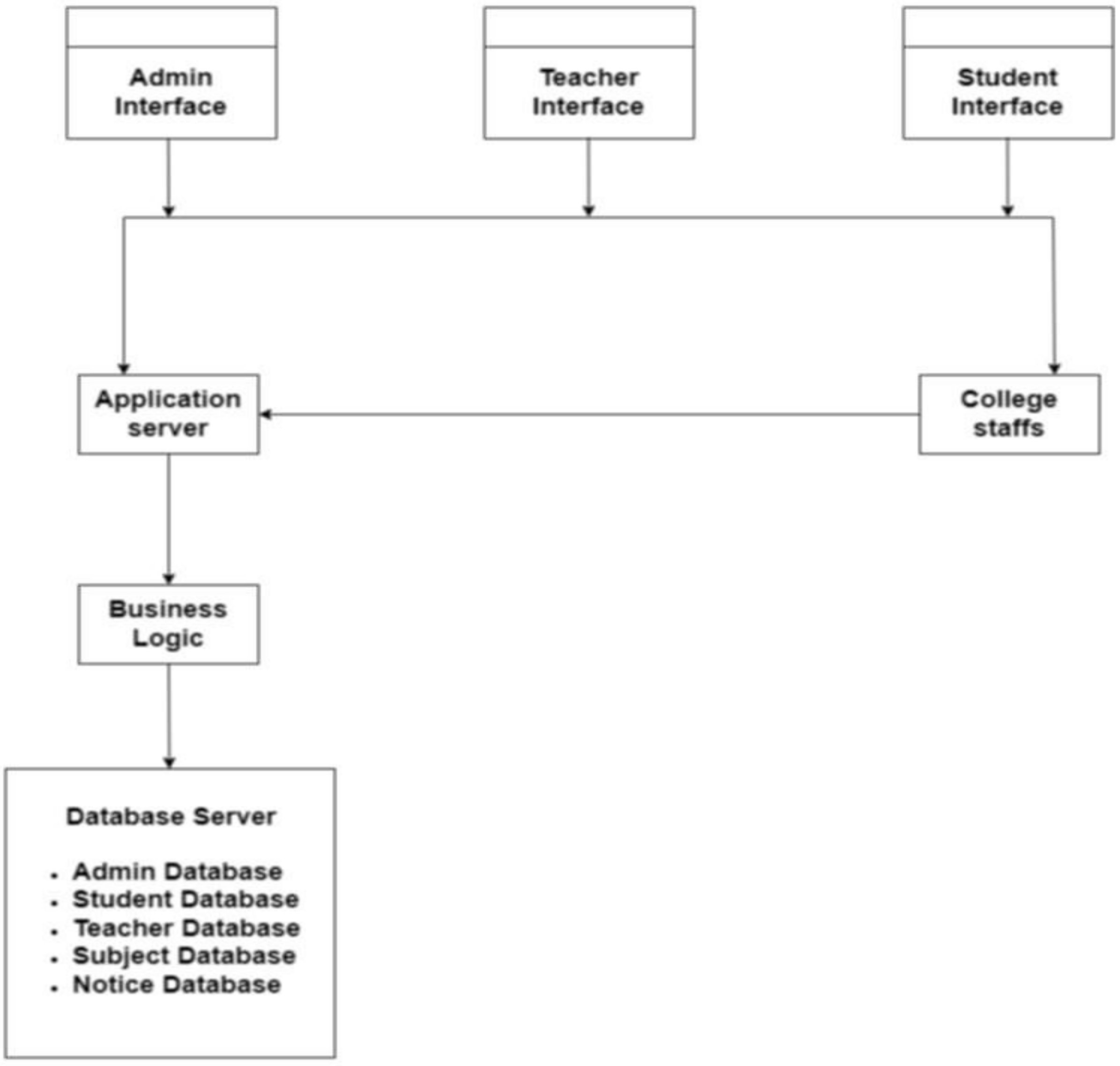
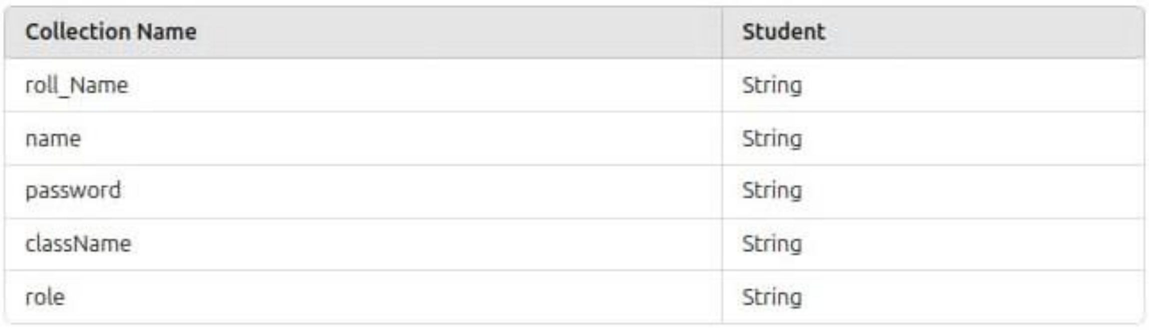
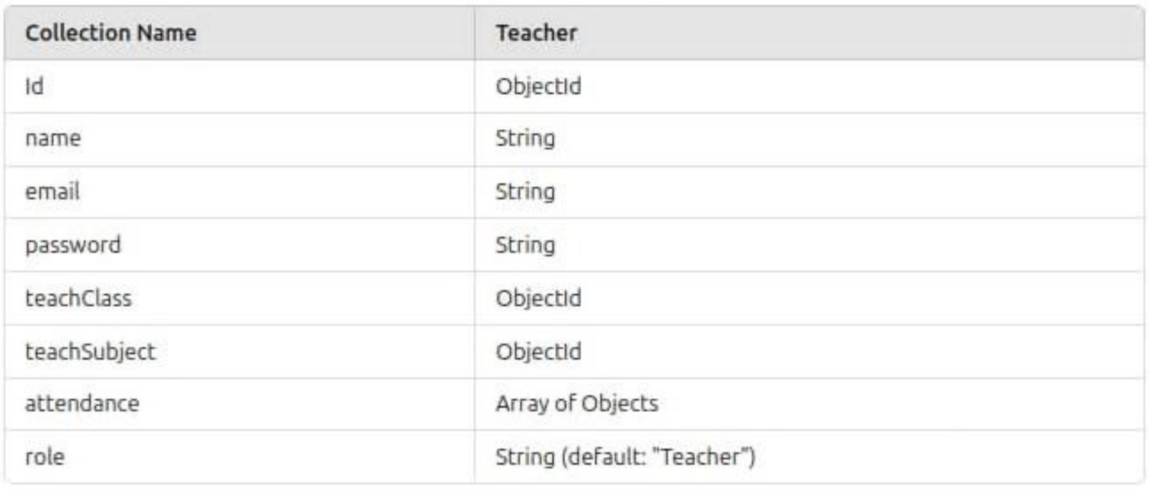


Figure 3. 6 Architecture Diagram

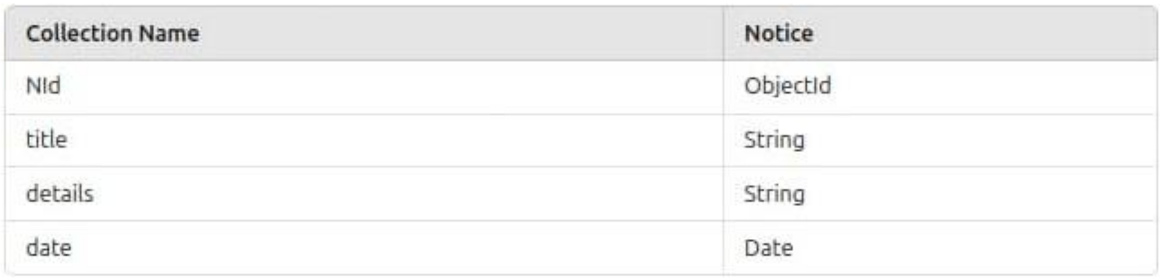
3.2.2 Database Schema Design



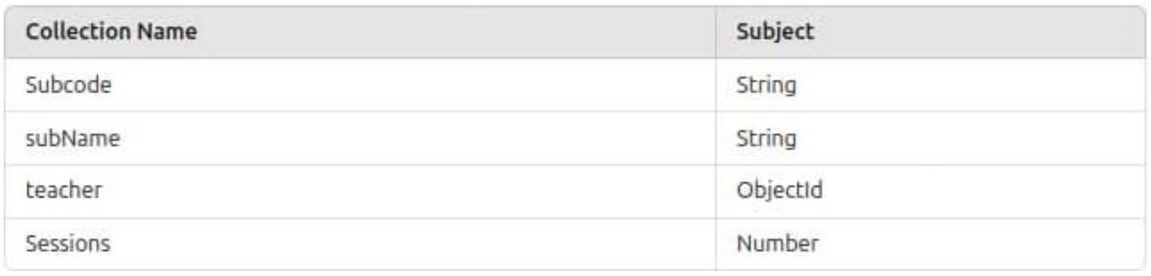
Collection Name	Student
roll_Name	String
name	String
password	String
className	String
role	String



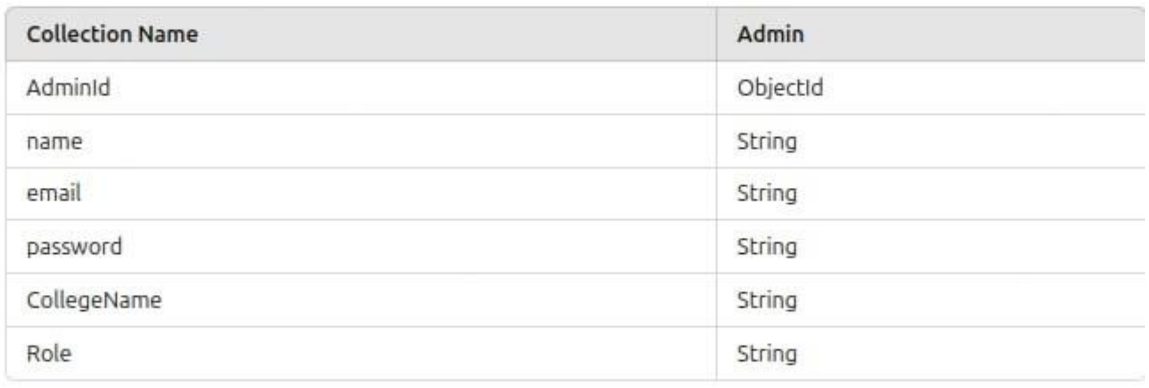
Collection Name	Teacher
Id	ObjectId
name	String
email	String
password	String
teachClass	ObjectId
teachSubject	ObjectId
attendance	Array of Objects
role	String (default: "Teacher")



Collection Name	Notice
NId	ObjectId
title	String
details	String
date	Date



Collection Name	Subject
Subcode	String
subName	String
teacher	ObjectId
Sessions	Number



Collection Name	Admin
AdminId	ObjectId
name	String
email	String
password	String
CollegeName	String
Role	String

Figure 3. 7 Database Schema

3.2.3 Algorithms

Algorithm 1: Linear Search for Student Records

Purpose: Filter student records based on user input, such as student ID or name.

Steps:

1. Accept user-provided search query (e.g., student ID, name, or grade).
2. Iterate through the list of student records: Retrieve each student's details (ID, name, grade, etc.).
3. Check if the student's attribute matches or includes the search query: If true, add the student record to the result set.
4. Return the filtered list of student records.

Algorithm 2: Set-Based Filtering for Unique Courses

Purpose: Extract unique courses offered by the institution.

Steps:

1. Fetch the list of courses with attributes like course name and department.
2. For each course, retrieve the course name.
3. Exclude entries without a valid course name.
4. Add course names to a Set to ensure uniqueness.
5. Convert the Set into an array.
6. Return the list of unique courses for filtering or display.

Chapter 4: Implementation and Testing

4.1 Implementation

4.1.1 Tools Used

- IDE: Visual Studio Code (VS Code) – A user-friendly integrated development environment (IDE) used for coding and development tasks.
- Documentation: MS Office – Tools for creating and editing project-related documents and presentations.
- Front-end: HTML, CSS, JavaScript – Languages used for the development of the graphical user interface of the Academic Management System application.
- Back-end: Node.js, Express.js – Node.js is a JavaScript runtime used for backend operations, while Express.js is a web framework to handle server-side logic and manage communication between the front-end and database.
- Database Management: MongoDB – A database used to store user information and chat messages in a flexible, document-oriented format.
- Localhost: Node.js with Express.js – A web server framework that runs the application locally during development and testing.
- Diagramming: Draw.io – An online diagramming tool used for creating visual representations, such as system architecture diagrams.

4.1.2 Implementation Details of Modules

i. User Registration and Authentication: This module allows users to register, log in, and log out of the chat application. It securely stores user credentials, enabling each user to access their chat history and profile settings.

ii. Search Functionality: The search module enables users to quickly find specific messages or conversations by searching keywords within their chat history.

iii. Profile Management: Users can update their personal information, such as username and profile picture, through this module. It allows users to customize their profile and control their visibility in the chat.

iv. Private Messaging (Direct Messages): This module allows users to initiate one-on-one private conversations with specific users, ensuring that messages are visible only to the participants involved.

4.2 Testing

4.2.1 Test Cases for Unit Testing

Table 4. 1 Test Cases for Unit Testing

Test No.	Module	Test Description	Steps	Expected Result	Result
TC _01	Admin Authentication	Verify valid admin allows successful login	Input valid admin details	admin is successfu llylogin	Pass
TC _02	Add Class	Update class name	Input the relevant class name and Save	Class added Successfully	Pass
TC _03	Add Student	Verify that a studentcan join the specific class	Fill the student roll number ,name and password	Student added Successful ly	Pass
TC _04	Add Subject	Verify the specificclass to add subject	Fill the subject name andsubject code	Subject added successful ly	Pass

4.2.2 Test Cases for System Testing

Table 4. 2 Test Cases for System Testing

Test No	Module	Test Description	Steps	Expected Result	Result
TC_05	User Role Access Control	Verify role based permissions	Test action for each role	Admin full access,teacher manages classes,student read only	pass
TC_06	Role data consistency	Ensure roles are correctly stored in MongoDb	Check MongoDb collection for role consistency	Role correctly assigned and consistent	pass
TC_07	Authentication security	Test accessed control for unauthorized users	Attempt login with invalid credentials	Unauthorized access blocked	pass
TC_08	NoSQL injection security	Test for NoSQL injection vulnerabilities	Inject malicious queries in inputs	System prevents injections	pass

Chapter 5: Conclusion and Future Recommendations

5.1 Conclusion

The development of an Academic Management System using modern technologies like React, Node.js, Express, MongoDB, and JWT authentication provides an efficient, scalable, and secure solution for managing the various aspects of academic institutions. By leveraging React and Material-UI on the frontend, the system offers an intuitive and responsive user interface, ensuring an enhanced user experience for students, faculty, and administrators. The backend, built with Node.js and Express, facilitates smooth handling of data and operations through well-defined APIs, while MongoDB offers a flexible and scalable database solution to manage diverse datasets such as student records, courses, and grading. JWT authentication ensures secure access to sensitive data, enhancing the system's overall security. Together, these technologies streamline the administration of academic operations, reducing paperwork, minimizing human error, and improving efficiency.

5.2 Future Recommendations

This project is currently limited to basic management of academic system. The following modules are recommended to be included in the Academic Management System in the future:

1. AI and Machine Learning Integration: Future versions of the AMS could incorporate AI-powered analytics to track student performance trends, predict outcomes, and offer personalized learning recommendations.
2. Cloud Integration: To enhance scalability and accessibility, the AMS could be deployed in a cloud environment, offering institutions the flexibility to manage resources efficiently and provide access from anywhere.
3. Mobile App Development: Developing a mobile application alongside the web platform would improve accessibility for students and faculty, allowing them to engage with the system from any device.
4. Enhanced Security Features: In addition to JWT, implementing features like multi-factor authentication (MFA) and end-to-end encryption would strengthen the system’s security, especially for handling sensitive academic data.
5. Role-based Access and Permissions: A more granular role-based access control mechanism could be developed to allow different levels of permission based on roles (e.g.,

administrators, teachers, students, and parents) to manage the scope of data access and operations.

By adopting these recommendations, the AMS can evolve into a more sophisticated, accessible, and future-proof system, continuing to meet the evolving demands of modern educational institutions.

REFERENCES

[1] J. Thilagavathi, S. Yasaswini, E. Aiswarya and G. P. Narasimha, "College Management System using Python & Django," *Shanlax International Journal of Arts Science and Humanities*, no. July, July2024.

[2] K. Acharya, "College information management system," no. March, March 2024.

[3] U. "College Database Management System (DBMS) UDDIT," *Applied Stochastic Models and Data Analysis*, no. August, 2024.

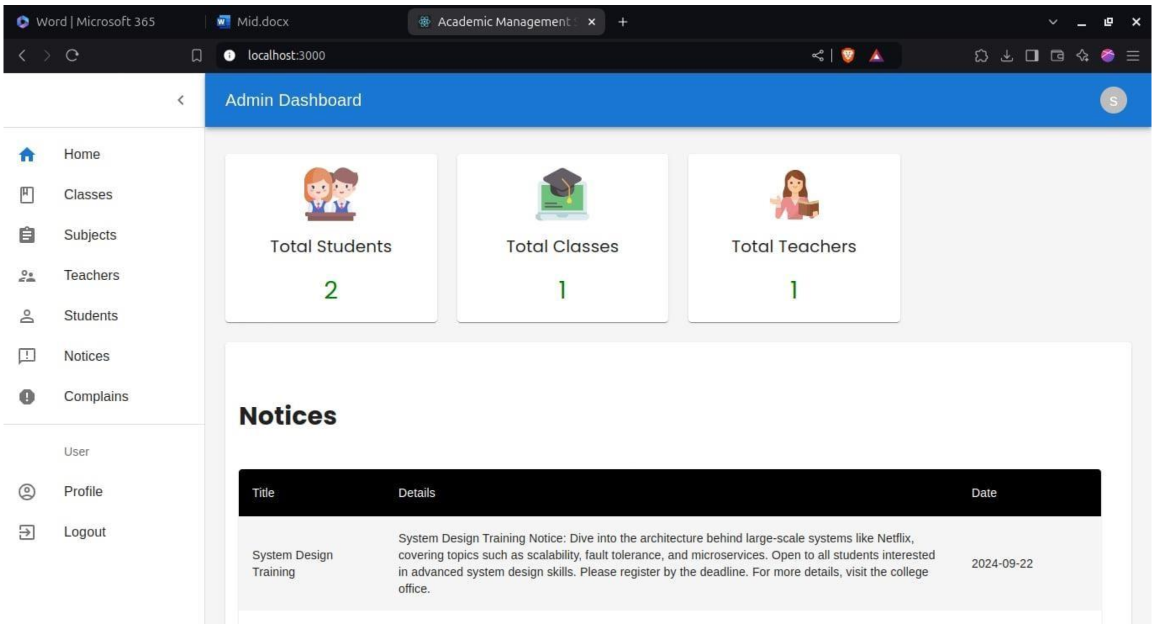
[4] a. chaurasiya, "College Management System," no. JULY, p. 4, 2022.

[5] A. Jain, "Enhancing Educational Administration and Academic Management: A Study of the School Management System," *International Scientific Journal of Engineering and Management*, no. May, 2024.

[6] Y. Syukur, A. H. Putra, Z. Ardi and V. Mardian, "The Relationship among Academic Self-Efficacy, Academic Resilience, and Academic Flow: The Mediating Effect of Intensity Using Learning Management System," *JOIV International Journal on Informatics Visualization*, no. May, 2024.

APPENDICES

- Dashboard (Admin)

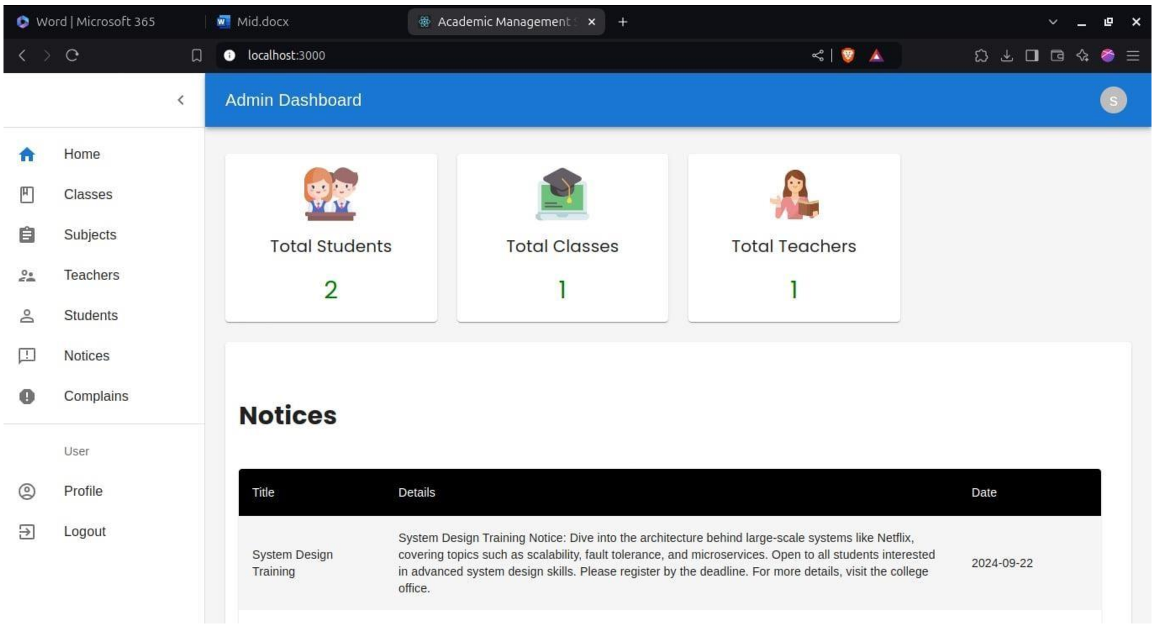


Notices

Title	Details	Date
System Design Training	System Design Training Notice: Dive into the architecture behind large-scale systems like Netflix, covering topics such as scalability, fault tolerance, and microservices. Open to all students interested in advanced system design skills. Please register by the deadline. For more details, visit the college office.	2024-09-22

Figure 3 Index Page

- Dashboard (Student)

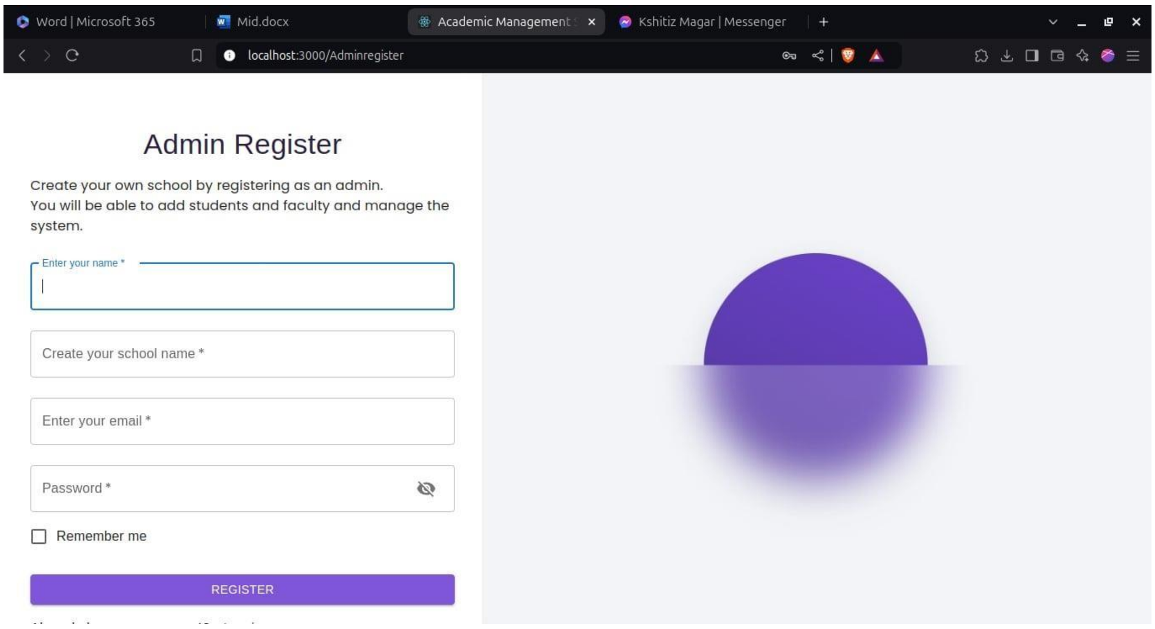


Notices

Title	Details	Date
System Design Training	System Design Training Notice: Dive into the architecture behind large-scale systems like Netflix, covering topics such as scalability, fault tolerance, and microservices. Open to all students interested in advanced system design skills. Please register by the deadline. For more details, visit the college office.	2024-09-22

Figure 4 Profile Update

- Register



Admin Register

Create your own school by registering as an admin.
You will be able to add students and faculty and manage the system.

Enter your name *

Create your school name *

Enter your email *

Password *

Remember me

REGISTER

Figure 5 Register

- Login

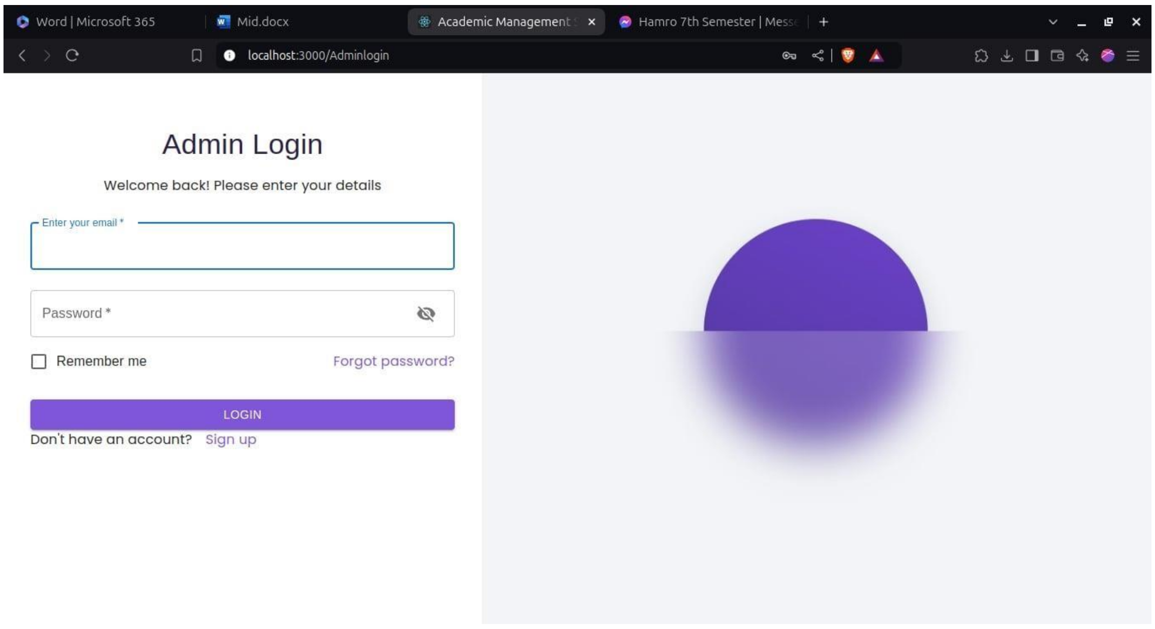


Figure 6 Login