Design and Development of a Student Information System

A Project Report submitted to the Department of Computer Science and Engineering, Jahangirnagar University in partial fulfillment of the requirements for the degree of Professional Master of Science in Computer Science.

Submitted By

Md. Touhid Iqbal Sagar

ID: CSE202401058

Batch 34

Supervised by

Dr. Md. Imdadul Islam

Professor

Department of Computer Science and Engineering

Jahangirnagar University

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JAHANGIRNAGAR UNIVERSITY

July, 2025

ABSTRACT

The Student Information System creates one unified and automated platform for universities to handle and view student information safely. This platform targets to fix how people used to keep records by streamlining student data in one secure system that everyone can access.

The system is structured around several core modules that handle critical aspects of student management. These include student registration and enrolment, academic performance tracking, attendance monitoring, timetable scheduling, fee management, and communication between all parties involved. A secure role-based access mechanism is implemented to protect sensitive information and ensure data privacy.

I used modern technology to build this project for maximum performance and growth potential. Our PHP backend handles data quickly and securely while the JavaScript CSS HTML libraries frameworks on the frontend create smooth user experiences. The MySQL database system stores the data securely and handles big amounts of information correctly. The system provides platform-free service and lets users view it through web browsers and mobile devices for maximum accessibility.

The Student Information System project provides advanced functions beyond basic record keeping through performance tracking tools that create instant reports for both student progress and university operations. The information helps teachers and office staff plan educational efforts and use their resources more effectively. Using the system lets us work digitally which saves natural resources we would use for paper documents.

Through automation of routine work the project enables educators to dedicate more time to their essential duties. The platform adapts to educational institutions of different sizes helping university teams work more efficiently.

DECLARATION

The project work entitled "**Design and Development of a Student Information System**" has been carried out in the Department of Computer Science and Engineering, Jahangirnagar University is original and conforms the regulations of this University.

I understand the University's policy on plagiarism and declare that no part of this project has been copied from other sources or been previously submitted elsewhere for the award of any degree or diploma.

Submitted by

____

Md. Touhid Iqbal Sagar

ID: CSE202401058

Batch: 34th

Department of Computer Science and Engineering

Jahangirnagar University

Supervised by

____

Dr. Md. Imdadul Islam

Supervisor & Professor

Department of Computer Science and Engineering

Jahangirnagar University

ACKNOWLEDGEMENT

To begin, I would want to convey my deepest gratitude and appreciation towards the Glory of God for the magnificent grace that he has bestowed upon me, which has enabled me to successfully finish the project for the senior year.

I would like to express my gratitude and my deep indebtedness to my Supervisor Dr. Md. Imdadul Islam, Professor in the Department of Computer Science and Engineering at Jahangirnagar University, for his extensive concern and inspiration throughout the entirety of the project work. I am grateful to him and wish my profound indebtedness to him.

My supervisor has extensive experience and a genuine interest in the topic of effect of usability design, which is a factor that impacted our decision to carry out this project. His limitless patience, scholarly guidance, continuous encouragement, constant and energizing supervisory, constructive comments, valuable suggestions, reading numerous inferior drafts and attempting to correct them at all stages, and making it possible to finish this project are all reasons why it was possible to finish it.

I would like to extend my deepest appreciation to all of the faculty members in the Department of CSE for their insightful recommendations, which have inspired me to complete the task inside the allotted amount of time. I would like to extend my most sincere gratitude to those unnamed reviewers who have provided some feedback that may be used to enhance the quality of the article even more. I would want to extend my gratitude to all of my classmates from Jahangirnagar University, particularly those individuals who participated in this conversation while they were working on their assignments.

In closing, but certainly not least, I feel it is important to recognize and give proper credit to my parents for their unwavering support and endless patience.

APPROVAL

The project, which was given the title “Design and Development of a Student Information System”, was presented to the following respected people on the board of examination of Computer Science & Engineering as a part of the requirements for the degree of Professional Master of Science in Computer Science & Engineering, and it was deemed to be satisfactory by those members. The board's decision was to award the student the degree.

Dr. Md. Imdadul Islam

Supervisor & Professor

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka, Bangladesh

Accpected by:

Abu Sayed Md. Mostafizur Rahaman

Coordinator, PMSCS Program 2025

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka, Bangladesh

CONTENTS

Chapter 1

Introduction

1.1 OVERVIEW

In today's rapidly evolving academic landscape, modern educational institutions are increasingly relying on digital solutions to enhance operational efficiency, student engagement, and overall institutional performance. A **Student Information System (SIS)** stands at the forefront of this transformation, serving as a comprehensive digital backbone that centralizes, automates, and streamlines a wide array of academic and administrative functions.

More than just a data repository, the SIS acts as an intelligent interface between students, faculty, and administrative staff. It facilitates real-time access to critical information such as enrollment records, grades, attendance, scheduling, course registration, and more—ensuring transparency, accountability, and efficiency across all levels of the academic structure.

By integrating diverse operations into a unified platform, the system not only reduces manual workload but also enables data-driven decision-making and seamless communication. This results in a more connected and responsive academic environment where students can track their academic progress, faculty can manage their courses more effectively, and administrators can oversee institutional operations with greater clarity and control.

Ultimately, the Student Information System empowers educational institutions to focus more on delivering quality education and less on managing paperwork—fostering a smarter, more agile, and future-ready learning ecosystem.

1.2 Literature Review

Effective management of student academic records has long posed a significant challenge for educational institutions, particularly in regions where legacy systems or manual processes still dominate. Common problems include inaccurate or incomplete records, delayed release of examination results, tedious manual calculations, and inefficient data retrieval processes. These issues are often compounded by departmental silos, where multiple versions of the same student data are stored in non-standardized formats across various units, leading to redundancy and data inconsistencies. For example, a simple change of address might need to be updated in multiple locations, increasing the risk of error and administrative burden.

The emergence of database technologies has played a pivotal role in addressing these limitations. Centralized databases significantly reduce redundancy and ensure that data is updated and synchronized across departments, thereby improving accuracy and accessibility. As described in an early implementation study, a student registration and course management system developed with Microsoft Access 2003 illustrated how a well-structured database application could streamline institutional workflows and enhance data reliability. This study also highlighted the importance of selecting an appropriate database model, designing user-friendly interfaces, forecasting future system demands, and establishing long-term maintenance protocols [1].

Traditionally, academic institutions have relied on paper-based record-keeping systems, storing information as hardcopies in physical file cabinets. While organized, these methods are time-consuming, insecure, and prone to misplacement or unauthorized access. Retrieving records could take hours or even days, depending on the complexity of the request. With the advent of Information Technology, however, digital systems have become the new standard. Computerized record management systems offer faster access, enhanced security, and ease of use, thus mitigating the limitations of traditional file systems. A case study conducted at Salem University, Lokoja, demonstrated how a Microsoft Access-based prototype system could improve record-keeping processes and serve as a model for other institutions seeking digital transformation [2].

The development and deployment of such systems follow the principles of the **System Development Life Cycle (SDLC)**, which includes key phases such as planning, analysis, design, implementation, testing, deployment, and maintenance. SDLC provides a structured approach to building software that meets user expectations and functional requirements. While the concept of a "system" is broad—encompassing hardware, software, and human elements—SDLC specifically focuses on the creation of software systems that integrate seamlessly with existing institutional infrastructures [3][4].

As education becomes increasingly digitized, the cybersecurity of student information systems cannot be overlooked. Recent findings by BitSight Technologies reveal that higher education institutions face heightened cybersecurity threats, particularly at the beginning of academic sessions. Alarmingly, many universities still rank lower in cybersecurity readiness compared to other vulnerable sectors like retail and healthcare. This calls for robust security protocols and proactive IT governance in any student information system to prevent unauthorized access, data breaches, and service disruptions [5].

In global contexts, especially in countries like China, educational institutions are progressively adopting integrated digital management platforms. These platforms support comprehensive, interconnected student lifecycle management—spanning academic registration, performance tracking, administrative services, and beyond. Through continuous development of digital resources and infrastructural investments, universities aim to build intelligent, responsive campus environments that improve both student experience and institutional performance [6][7][8].

As societal expectations for educational quality continue to rise, institutions are required to meet higher standards of student oversight and academic accountability. Scholars such as Bizarria et al. [9] and Santana et al. [10] emphasize that effective student management is not merely about data handling; it encompasses academic preparedness, faculty development, and the integration of experiential knowledge. A modern student information system must, therefore, be holistic—serving as both a technological solution and a strategic tool for enhancing the educational mission of institutions.

In summary, the literature underscores the urgent need for scalable, secure, and intelligent student information systems. By drawing insights from existing implementations and aligning with best practices in system design and development, educational institutions can overcome traditional inefficiencies and build agile systems that support academic excellence and operational effectiveness.

1.3 Contribution of project work

The **Student Management System** project was conceptualized and developed as a robust, secure, and scalable digital platform aimed at transforming how academic institutions handle student-related data and operations. At its core, the system was designed to centralize and streamline the management of student information—ranging from personal records and course registrations to academic performance and attendance logs—within a unified, easily accessible database environment.

A key objective of the project was **data centralization**, ensuring that all student records are consolidated into a single source of truth. This not only reduced redundancy and discrepancies across departments but also enhanced data integrity and made administrative processes more efficient. By eliminating fragmented data storage systems, the platform improved inter-departmental collaboration and enabled faster, more accurate decision-making.

To further optimize operations, the project replaced outdated manual procedures with **automated workflows**. Core academic tasks such as attendance monitoring, grade computation, report generation, and course enrollment were digitized, significantly reducing the potential for human error and administrative delays. These automation features contributed to better time management and improved productivity for academic and administrative staff alike.

A major design consideration was **usability**. The system featured a clean, intuitive user interface tailored to accommodate users with varying levels of technical expertise. Whether accessed by administrators, teachers, or authorized staff, the platform provided a seamless experience that minimized the learning curve and encouraged widespread adoption.

Real-time accessibility was also a central feature of the system architecture. With cloud-based deployment and responsive design, the platform allowed stakeholders to access, update, or review data from any internet-enabled device, supporting both on-campus and remote academic management.

Given the sensitivity of academic records, **data security and user privacy** were prioritized throughout the system's development. Multi-layered authentication mechanisms, role-based access controls, and end-to-end data encryption were implemented to ensure that only authorized personnel could access or modify specific information. These security features not only safeguarded confidential student data but also ensured compliance with institutional policies and data protection standards.

Finally, the system was built with a forward-looking approach, emphasizing **scalability and interoperability**. Its modular design allows it to grow alongside institutional needs—whether that means handling a rising student population or incorporating additional academic modules. Furthermore, the system supports integration with other digital education tools and platforms, enabling institutions to build a connected, future-proof academic ecosystem.

In summary, the Student Management System project was not just a digital upgrade—it was a strategic initiative aimed at redefining how educational institutions manage, protect, and utilize student information. Every component of the system was purposefully designed to deliver efficiency, accuracy, user-friendliness, and long-term adaptability.

1.4 Organization of project work

This report is structured into five comprehensive chapters, each addressing a key component of the system development process and offering insights into the methodology, design, implementation, and future outlook of the Student Management System.

- **Chapter 2** presents a detailed overview of the **Software Development Life Cycle (SDLC)**, with a particular focus on the **Waterfall model** as the chosen development approach. The chapter outlines each phase of the model—requirements gathering, system design, implementation, testing, deployment, and maintenance—and explains its relevance to the linear nature of this project. The advantages and limitations of the Waterfall model are critically discussed, along with the rationale behind selecting it for this system's development from initiation to completion.

- **Chapter 3** delves into the **requirement analysis and system design procedures**. It includes various technical diagrams such as the **System Architecture Diagram**, **Use Case Diagram**, **Entity-Relationship (ER) Diagram**, **Database Schema**, and other visual representations that illustrate the system’s structure and behavior. These models serve to clarify the roles of different users (actors) and the interactions between system components, making the functional and non-functional requirements easy to comprehend.
- **Chapter 4** focuses on the **Results and Discussions**, highlighting key elements such as the target **users of the system**, the **development environment**, and the **tools and technologies** utilized throughout the project. This chapter also showcases the complete set of **User Interface (UI) designs**, providing screenshots and descriptions to demonstrate how users interact with the system in real-world scenarios.
- **Chapter 5** addresses the **Limitations and Future Enhancements** of the project. It identifies areas where the system could be improved or extended—either due to technological constraints, time limitations, or resource availability during development. Suggestions for future improvements are also presented, outlining features and integrations that could further enhance the system’s performance, scalability, and user experience.

Each chapter contributes to building a complete understanding of how the Student Management System was conceived, developed, and evaluated, while also setting the stage for potential future development and academic exploration.

Chapter 2

Basic Theory

2.1 Overview

The **Student Management System** has been thoughtfully designed to offer a comprehensive set of features that cater to the administrative and academic needs of modern educational institutions. Below is an outline of the system's core functionalities:

- **Student Registration**
  New students can seamlessly register by submitting personal, academic, and contact details through a guided digital form. This data is stored securely and becomes the foundation for future academic activities such as course enrollment, grade tracking, and departmental assignment.

- **Course Management**
  The system empowers administrators and faculty to create, edit, and manage course offerings. Instructors can assign themselves to courses, set prerequisites, and monitor student enrollment. Students, in turn, can browse available courses and register with ease based on eligibility.

- **Departmental Oversight**
  Each academic department is managed through a dedicated module that tracks associated faculty members, course inventories, and student enrollments. This feature ensures organized department-level administration and easy access to relevant academic data.

- **Grading & Assessment Module**
  This component allows for the efficient entry, computation, and management of students' academic results. Grades from assignments, tests, and examinations are compiled

automatically, and result summaries or transcripts can be generated instantly with minimal manual intervention.

• Reports & Analytics
Administrators and academic staff can access a wide range of dynamic reports, including student performance trends, attendance overviews, and course enrollment statistics.

• User Access & Authentication
Security is maintained through role-based login credentials, ensuring users only access functionalities appropriate to their roles—be it administrator, faculty member, or student. This mechanism enhances data security and user accountability within the system.

2.2 Comparison with Existing Solutions

Feature	Student Management System (Proposed Solution)	Conventional / Existing Systems
Customization	Designed to be easily customized based on institutional policies, structure, and needs.	Often inflexible, offering limited customization options out of the box.
Cost	Budget-friendly, particularly ideal for small or mid-sized institutions.	Requires high investment in licensing, upgrades, and ongoing maintenance.
Ease of Use	User-friendly interface requiring minimal training for staff and students.	Interfaces are often complex and may need formal user training sessions.
Integration	Built to support easy integration with third-party systems like e-learning tools.	Integration is available, but usually expensive and tied to vendor ecosystems.
Security & Privacy	Implements role-based access and customizable security protocols tailored to users.	High-level security, but rigid and less adaptable to internal policies.
Report Generation	Offers flexible, institution-specific report templates and analytics.	Typically comes with fixed, predefined reports with limited filtering options.
Technical Support	Support can be managed internally or outsourced affordably.	Vendor-provided support is reliable but comes at a significant cost.

2.3 Software Development Life Cycle:

The **Software Development Life Cycle (SDLC)** serves as a structured methodology that guides developers and software engineers through the systematic creation, deployment, and ongoing support of software applications. It provides a disciplined framework that ensures each stage of the development process is well-defined, efficient, and geared toward producing reliable, high-quality software products.

The SDLC is composed of several interdependent phases, each contributing to the overall success of the software project:

• Requirement Analysis

This is the foundational phase where developers work closely with stakeholders to identify and understand the functional and non-functional needs of the intended system. Information is gathered through various methods such as interviews, surveys, and stakeholder consultations. Additionally, feasibility studies—covering technical, financial, and operational aspects—are conducted to validate the viability of the project. The outcome of this phase is a comprehensive Software Requirements Specification (SRS) document.

• System Design

Once requirements are clearly defined, the next phase involves translating them into a detailed technical plan. This includes designing system architecture, defining data models, constructing user interface mockups, and identifying the interaction between system components. The deliverable from this stage is the Design Document Specification (DDS), which acts as a blueprint for development.

• Implementation (Coding)

Based on the DDS, developers begin writing source code using appropriate programming languages and tools. The focus here is on creating functional modules that align with the defined specifications. Good coding practices, adherence to design standards, and modular development are emphasized to ensure maintainability and scalability.

• Testing

After implementation, the software undergoes rigorous testing to validate its performance, reliability, and functionality. Testing is performed at multiple levels, including Unit Testing, Integration Testing, System Testing, and User Acceptance Testing (UAT). This phase ensures that the application meets the specified requirements and is free from critical defects.

• Deployment

Upon successful testing, the software is released into a production or live environment. This phase includes configuration, installation, and final verification to ensure the system operates as intended in a real-world setting. Deployment can be phased or full-scale depending on organizational strategy.

• Maintenance

Even after deployment, software requires ongoing support to address bugs, implement enhancements, and adapt to changing user needs. This phase ensures long-term stability, security, and performance through regular updates, patches, and user feedback integration.

2.4 Agile Model

The **Agile Model** represents a contemporary and dynamic approach to software development, centered around delivering functional components of a project in small, manageable increments. It emerged as a solution to the shortcomings of traditional methodologies like the Waterfall model, which tend to be linear, inflexible, and less responsive to evolving requirements. Agile prioritizes adaptive planning, continuous delivery, iterative improvement, and the ability to swiftly respond to changes, all with the ultimate aim of maximizing customer satisfaction through the consistent delivery of valuable software.

In practice, Agile breaks down a project into multiple short development cycles called **iterations** or **sprints**, usually lasting anywhere from one to four weeks. Each sprint encompasses a full development workflow—planning, design, coding, testing, and review—focusing on a limited set of features or tasks. Upon completion of a sprint, the team produces a functional, potentially deployable version of the software that stakeholders can assess. This cycle enables frequent feedback loops, ensuring that the evolving product aligns closely with user expectations and allowing for adjustments based on real-world input.

Within the Agile framework, several methodologies have gained prominence, each with distinct philosophies and structures. Scrum, perhaps the most popular, organizes work into fixed-length sprints and assigns clear roles such as Scrum Master and Product Owner to facilitate efficient team collaboration. Kanban offers a visual approach to workflow management, emphasizing transparency and limiting the amount of work in progress to enhance focus and throughput. Extreme Programming (XP) advocates for technical excellence, frequent releases, and strong communication practices. Meanwhile, Lean methodology seeks to optimize value by minimizing waste throughout the development lifecycle.

Overall, Agile has become the methodology of choice for many development teams due to its flexibility, strong customer involvement, and ability to deliver high-quality software rapidly. It is particularly effective in environments where requirements are fluid or not entirely defined from the outset, providing a responsive framework that embraces change rather than resisting it.

The Agile methodology is particularly well-suited for developing the Student Management System due to its emphasis on adaptability, continuous refinement, and active stakeholder participation—qualities that are crucial for a project within the dynamic environment of educational institutions. The needs surrounding student data management are often fluid, influenced by changes in academic policies, administrative requirements, and user feedback from teachers, students, and support staff. Agile's framework allows the development process to begin with a core set of essential features—such as student enrollment, attendance monitoring, and grading—while progressively expanding and refining the system based on iterative feedback, rather than deferring all value delivery until project completion.

A significant advantage of Agile lies in its iterative and incremental nature, which provides stakeholders with regular, tangible progress updates. For a Student Management System, this means that functional components like timetable management or fee processing can be developed and demonstrated at the end of each sprint. Early and frequent reviews facilitate the prompt identification and resolution of misunderstandings or requirement changes, ensuring the final product aligns closely with the institution's evolving expectations.

Moreover, Agile fosters continuous collaboration between the development team and end-users—a critical factor in educational contexts. Teachers, administrators, and students can provide ongoing insights that help prioritize features and enhance usability. This engagement helps guarantee that vital system functionalities are addressed and that the overall user experience remains intuitive and efficient.

The inherent flexibility of the Agile approach also supports projects where objectives may shift or expand over time. Should the institution later wish to introduce advanced capabilities such as online examination modules, student performance dashboards, or integration with external learning management systems, Agile allows these enhancements to be incorporated smoothly without disrupting the project's momentum or overarching structure.

In summary, adopting the Agile model for the Student Management System project facilitates a development process that is highly adaptive, cooperative, and responsive to change. This approach promotes a streamlined workflow, effective risk mitigation, and ultimately results in a more robust, user-centered system tailored to the actual needs of the educational community.

Chapter 3

Design Procedure

3.1 Introduction

The design phase in software development is a critical step that transforms the gathered system requirements into a comprehensive blueprint for building the software solution. Serving as a crucial link between the requirement analysis and the actual coding stages, this phase ensures that the envisioned system is both functional and aligned with user expectations. The main goal of the design process is to produce a clear, well-structured plan that guarantees efficient operation, ease of maintenance, and future scalability.

The design process typically starts with the high-level design or architectural design, where the overall system structure is outlined. This includes defining the core modules, their interactions, and the flow of data between components. Following this, the focus shifts to low-level design, which dives deeper into the specifics of each module—detailing algorithms, data management techniques, user interfaces, and database schemas. Throughout this stage, key design principles such as modularity, abstraction, encapsulation, and reusability are carefully applied to promote robustness and maintainability.

An effective design process not only simplifies the system's complexity but also enhances understanding among the development team. It enables early detection of potential flaws, which reduces development risks and costs down the line. Ultimately, a thoughtfully crafted design provides a solid foundation for creating software that is dependable, scalable, and user-friendly.

3.2 System Architecture Diagram

A System Architecture Diagram serves as a graphical blueprint that outlines the fundamental framework of a software application. It visually depicts how the system's various components or modules interconnect and how data is exchanged among them. This diagram is an essential communication tool, helping developers, project stakeholders, and end-users grasp the system's structural layout, the interplay between different elements, and the overall functionality. Key components typically illustrated include user interfaces, core application logic, databases, and external integrations, providing a comprehensive snapshot of how the system operates and is organized.

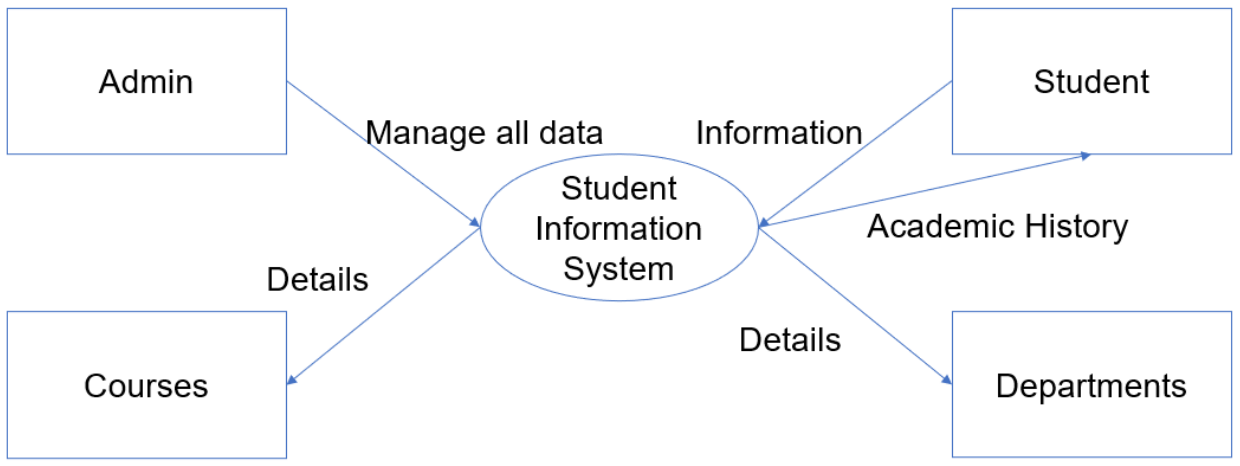


Figure 3.2.1 System Architecture Diagram of Student Information System

3.3 Entity- Relationship Diagram

An Entity-Relationship Diagram (ERD) serves as a crucial graphical representation used in the design and modeling of databases. It visually depicts the various entities within a system and the associations that exist between them, helping to conceptualize the data structure effectively.

- **Entities** represent distinct objects or concepts relevant to the system, such as Student, Course, or Instructor.
- **Attributes** provide descriptive details about these entities, including specifics like a student's identification number, full name, or enrollment date.
- **Relationships** define how entities interact with one another—for example, a student registers for multiple courses, or a teacher is assigned to several classes.

By utilizing ERDs, database architects and developers can map out the organization of data in a clear and systematic way. This planning stage ensures the database is well-structured, promotes data integrity, reduces redundancy, and supports efficient data retrieval once implemented.

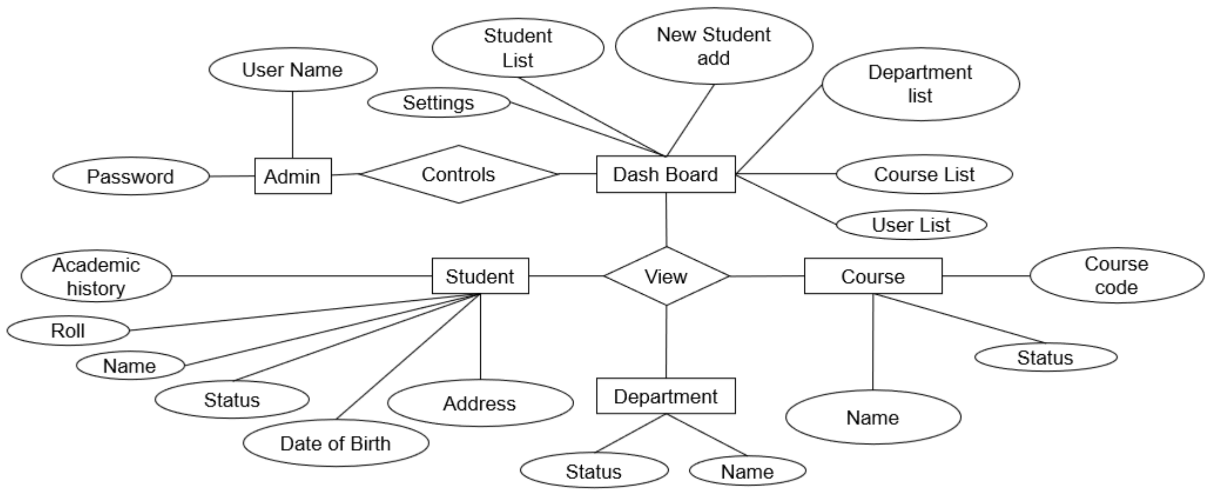


Figure 3.3.1 Entity- Relationship Diagram of Student Information System

3.4 Use Case Diagram

A Use Case Diagram is a graphical tool in software engineering that captures the interactions between external users (known as actors) and the system itself. It highlights the system's functional requirements by outlining the different use cases, which represent the distinct tasks or services the system offers to its users.

These diagrams serve as an essential communication bridge, enabling developers and stakeholders to visualize how users engage with the system. By focusing on user goals and actions, Use Case Diagrams provide valuable insights that inform system design, development, and ensure alignment with user needs.

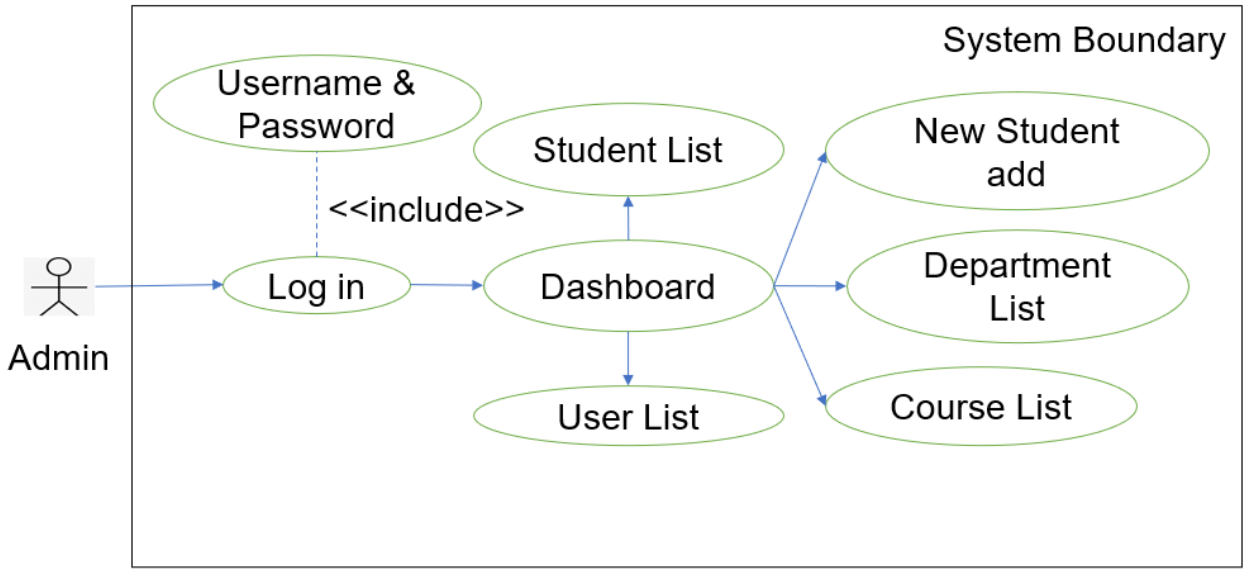


Figure 3.4.1 Use Case Diagram of Student Information System

3.5 Database Schema Overview

A **database schema** is the foundational design or framework of a database system that defines how data is logically structured and how various elements within the database are interconnected. It specifies the organization of tables, the fields (or columns) within those tables, the data types assigned to each field, and any constraints—such as primary keys, foreign keys, or rules—that govern how data is entered, related, and maintained.

In essence, the schema acts as a roadmap for how information is stored and accessed, ensuring consistency, integrity, and efficiency in data handling. It plays a vital role for developers and database administrators by offering a clear reference for developing, managing, and scaling the database system in a structured and reliable manner.

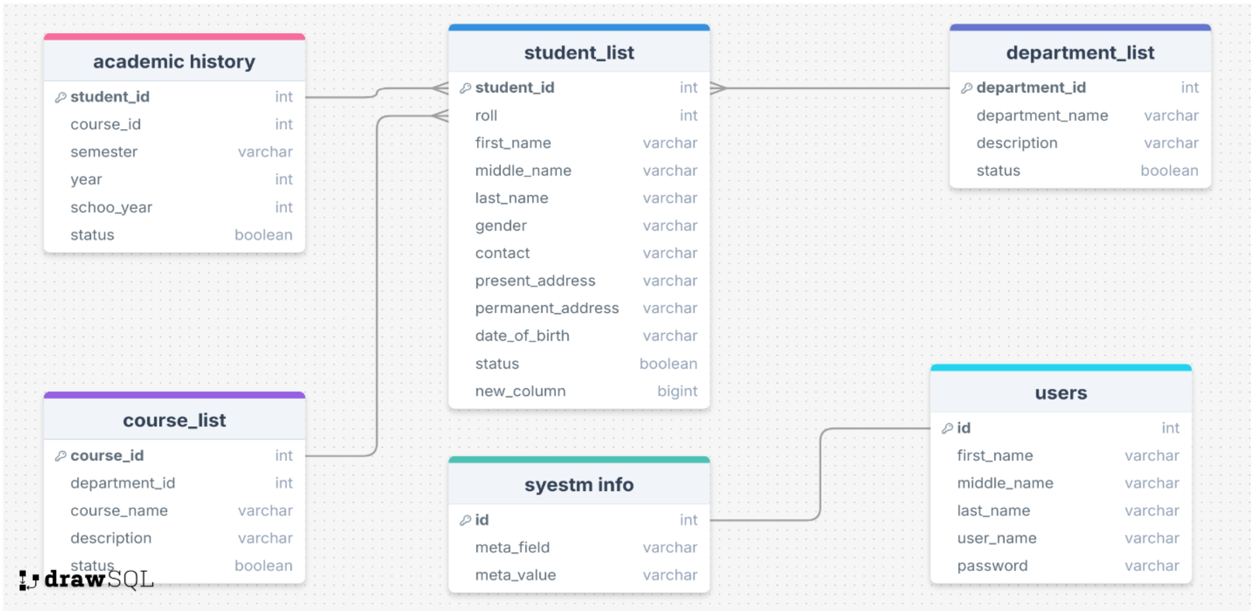


Figure 3.5.1 Database Schema Overview of Student Management System

Chapter 4

Results and Discussion

4.1 Introduction

Results present the direct outcomes of the project, such as system performance, user feedback, and functional test results. These are shown through tables, charts, or summaries without interpretation.

Discussion interprets these findings, analyzing how they align with project goals, highlighting insights, and identifying any unexpected issues or improvements.

4.2 Dashboard

Figure 4.2.1 is the dashboard page of the project where all the options are visible.

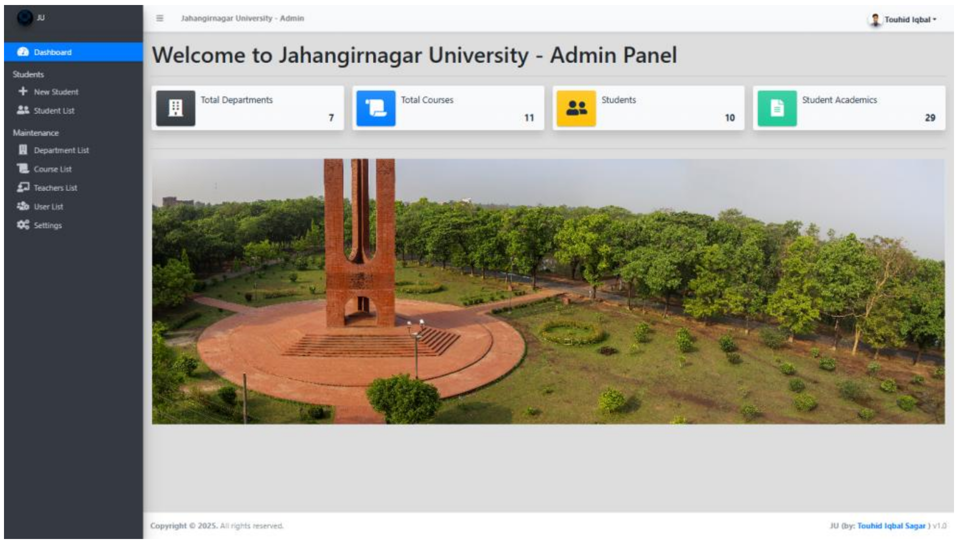
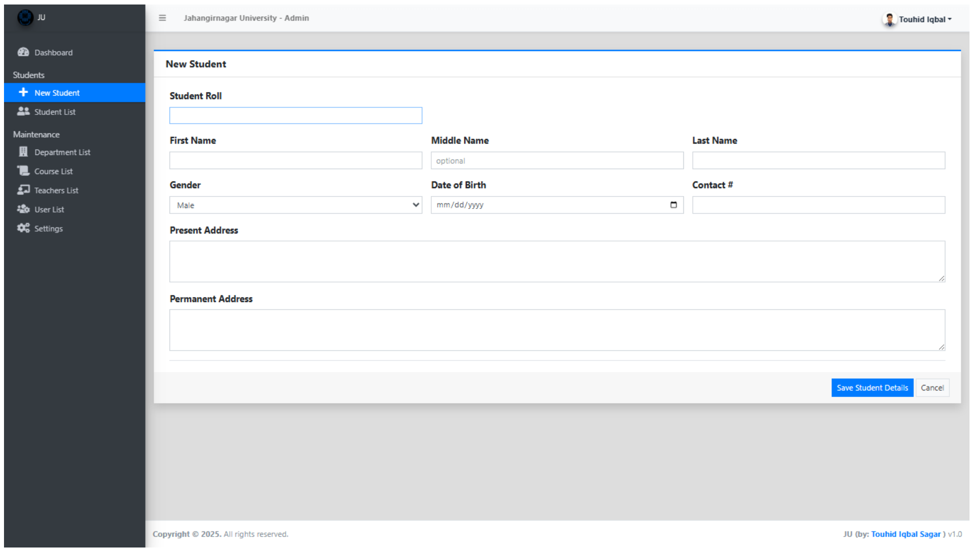


Fig 4.2.1: Dashboard of Student Information System

4.3 New student add page:

From this page we can add new student's information in Fig 4.3.1.



New Student

Student Roll

First Name | Middle Name: optional | Last Name

Gender: Male | Date of Birth: mm/dd/yyyy | Contact #

Present Address

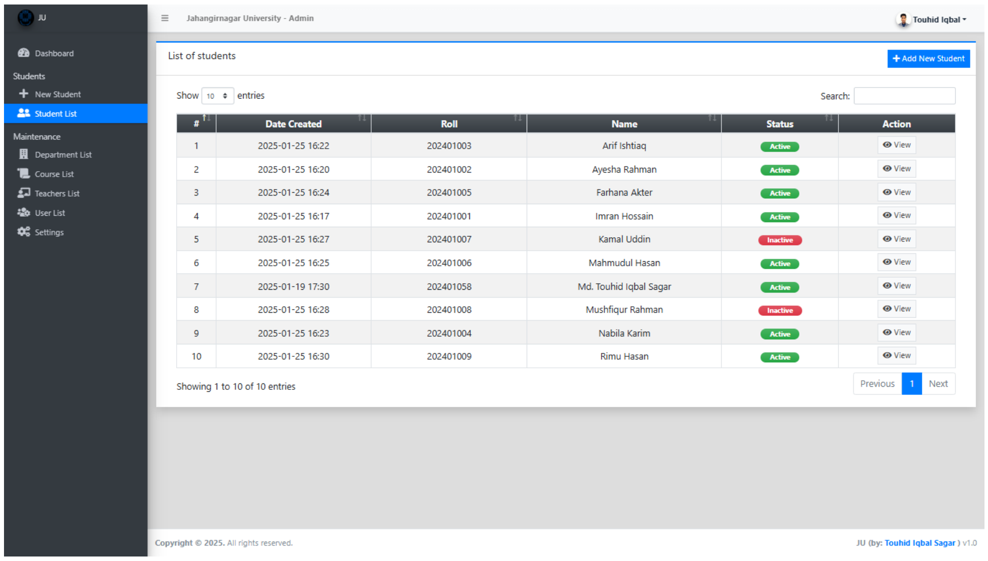
Permanent Address

Copyright © 2025. All rights reserved. JU (by: Touhid Iqbal Sagar) v1.0

Fig 4.3.1: New student add page of Student Information System.

4.4 Student list:

This page shows the information of the existing students in Fig 4.4.1.



List of students

Show 10 entries Search:

#	Date Created	Roll	Name	Status	Action
1	2025-01-25 16:22	202401003	Arif Ishtiaq	Active	View
2	2025-01-25 16:20	202401002	Ayesha Rahman	Active	View
3	2025-01-25 16:24	202401005	Farhana Akter	Active	View
4	2025-01-25 16:17	202401001	Imran Hossain	Active	View
5	2025-01-25 16:27	202401007	Kamal Uddin	Inactive	View
6	2025-01-25 16:25	202401006	Mahmudul Hasan	Active	View
7	2025-01-19 17:30	202401058	Md. Touhid Iqbal Sagar	Active	View
8	2025-01-25 16:28	202401008	Mushfiqur Rahman	Inactive	View
9	2025-01-25 16:23	202401004	Nabila Karim	Active	View
10	2025-01-25 16:30	202401009	Rimu Hasan	Active	View

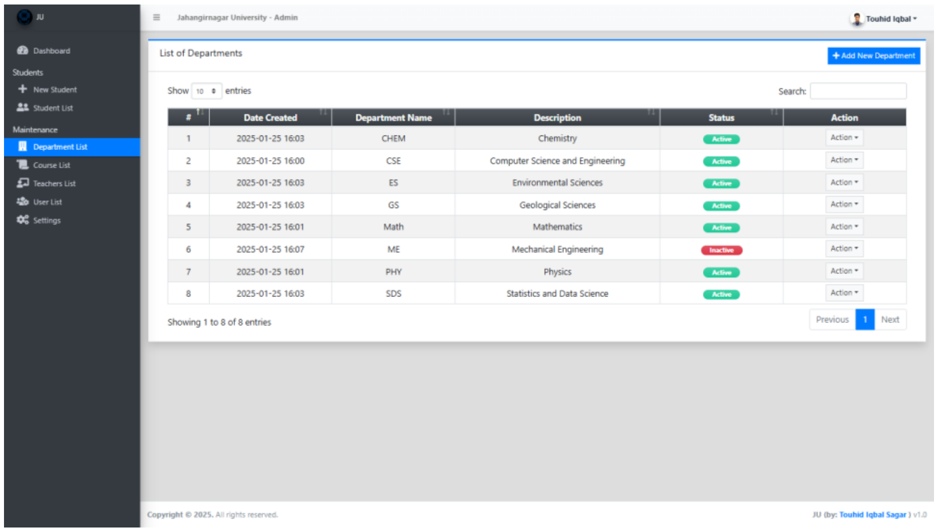
Showing 1 to 10 of 10 entries

Copyright © 2025. All rights reserved. JU (by: Touhid Iqbal Sagar) v1.0

Fig 4.4.1: Student list page of Student Information System

4.5 Department List:

This page shows the existing department's list as well as we can add new list from the top right button in Figure 4.5.1.



List of Departments

Show 10 entries Search:

#	Date Created	Department Name	Description	Status	Action
1	2025-01-25 16:03	CHEM	Chemistry	Active	Action
2	2025-01-25 16:00	CSE	Computer Science and Engineering	Active	Action
3	2025-01-25 16:03	ES	Environmental Sciences	Active	Action
4	2025-01-25 16:03	GS	Geological Sciences	Active	Action
5	2025-01-25 16:01	Math	Mathematics	Active	Action
6	2025-01-25 16:07	ME	Mechanical Engineering	Inactive	Action
7	2025-01-25 16:01	PHY	Physics	Active	Action
8	2025-01-25 16:03	SDS	Statistics and Data Science	Active	Action

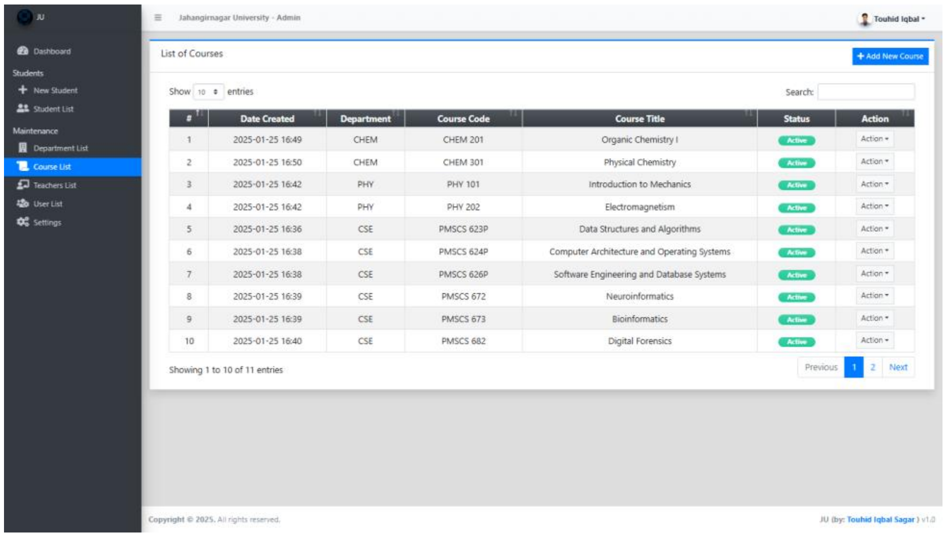
Showing 1 to 8 of 8 entries

Copyright © 2025. All rights reserved.   JU (by: Touhid Iqbal Sagar) v1.0

Figure 4.5.1: Department list page of Student Information System

4.6: Course list

This page shows the existing course's list as well as we can add new list from the top right button in Figure 4.6.1.



List of Courses

Show 10 entries Search:

#	Date Created	Department	Course Code	Course Title	Status	Action
1	2025-01-25 16:49	CHEM	CHEM 201	Organic Chemistry I	Active	Action
2	2025-01-25 16:50	CHEM	CHEM 301	Physical Chemistry	Active	Action
3	2025-01-25 16:42	PHY	PHY 101	Introduction to Mechanics	Active	Action
4	2025-01-25 16:42	PHY	PHY 202	Electromagnetism	Active	Action
5	2025-01-25 16:36	CSE	PMSCS 623P	Data Structures and Algorithms	Active	Action
6	2025-01-25 16:38	CSE	PMSCS 624P	Computer Architecture and Operating Systems	Active	Action
7	2025-01-25 16:38	CSE	PMSCS 626P	Software Engineering and Database Systems	Active	Action
8	2025-01-25 16:39	CSE	PMSCS 672	Neuroinformatics	Active	Action
9	2025-01-25 16:39	CSE	PMSCS 673	Bioinformatics	Active	Action
10	2025-01-25 16:40	CSE	PMSCS 682	Digital Forensics	Active	Action

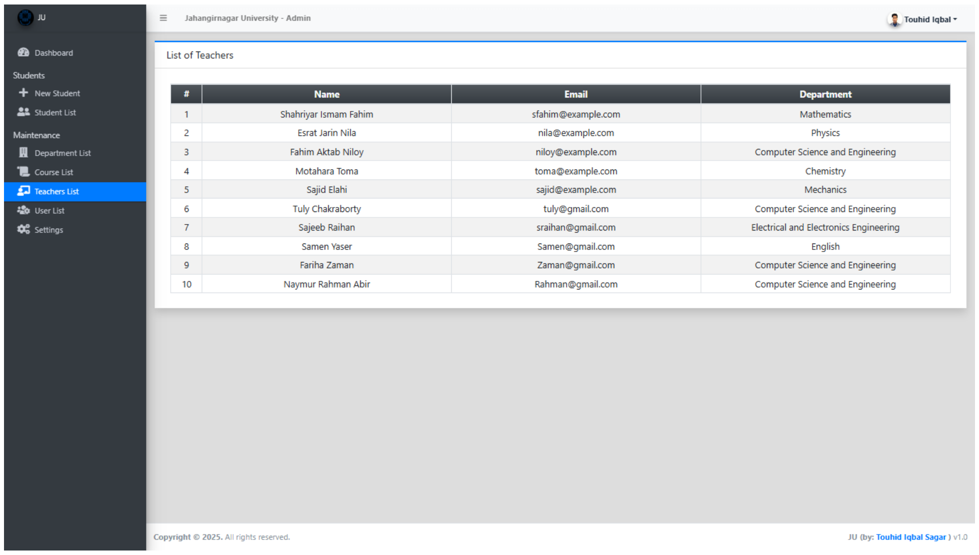
Showing 1 to 10 of 11 entries

Copyright © 2025. All rights reserved.   JU (by: Touhid Iqbal Sagar) v1.0

Figure 4.6.1: Course list page of Student Information System

4.7 Teacher list page

This page shows the existing list of teachers in Figure 4.7.1.



List of Teachers

#	Name	Email	Department
1	Shahriyar Ismam Fahim	sfahim@example.com	Mathematics
2	Esrat Jarin Nila	nila@example.com	Physics
3	Fahim Aktab Niloy	niloy@example.com	Computer Science and Engineering
4	Motahara Toma	toma@example.com	Chemistry
5	Sajid Elahi	sajid@example.com	Mechanics
6	Tuly Chakraborty	tuly@gmail.com	Computer Science and Engineering
7	Sajeeb Raihan	sraihan@gmail.com	Electrical and Electronics Engineering
8	Samen Yaser	Samen@gmail.com	English
9	Fariha Zaman	Zaman@gmail.com	Computer Science and Engineering
10	Naymur Rahman Abir	Rahman@gmail.com	Computer Science and Engineering

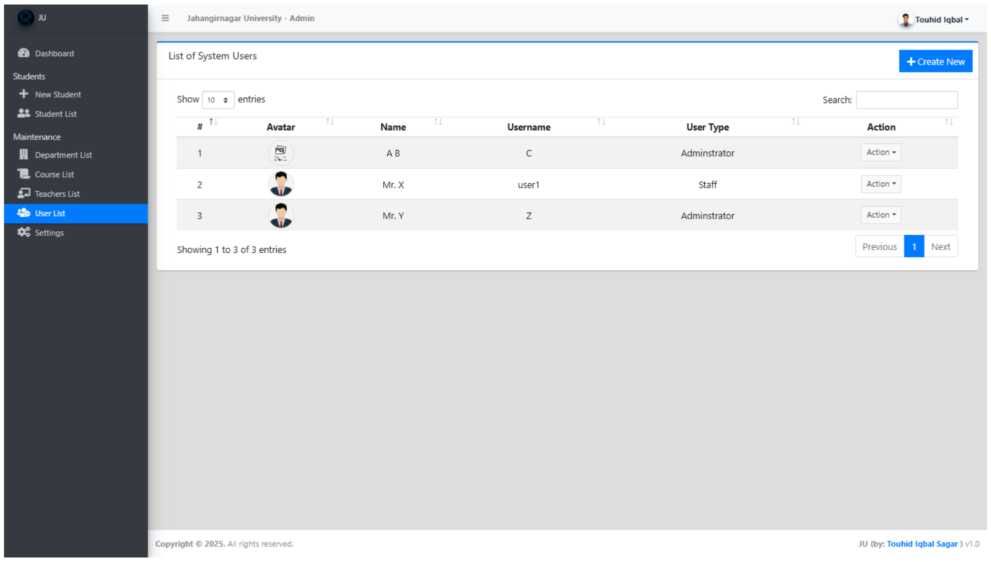
Copyright © 2025. All rights reserved.

JU (by: Touhid Iqbal Sagar) v1.0

Figure 4.7.1: Teachers list page of Student Information System

4.8 User list page:

This page shows the existing list of users in Figure 4,8.1.



List of System Users

#	Avatar	Name	Username	User Type	Action
1		A B	C	Adminstrator	Action
2		Mr. X	user1	Staff	Action
3		Mr. Y	Z	Adminstrator	Action

Showing 1 to 3 of 3 entries

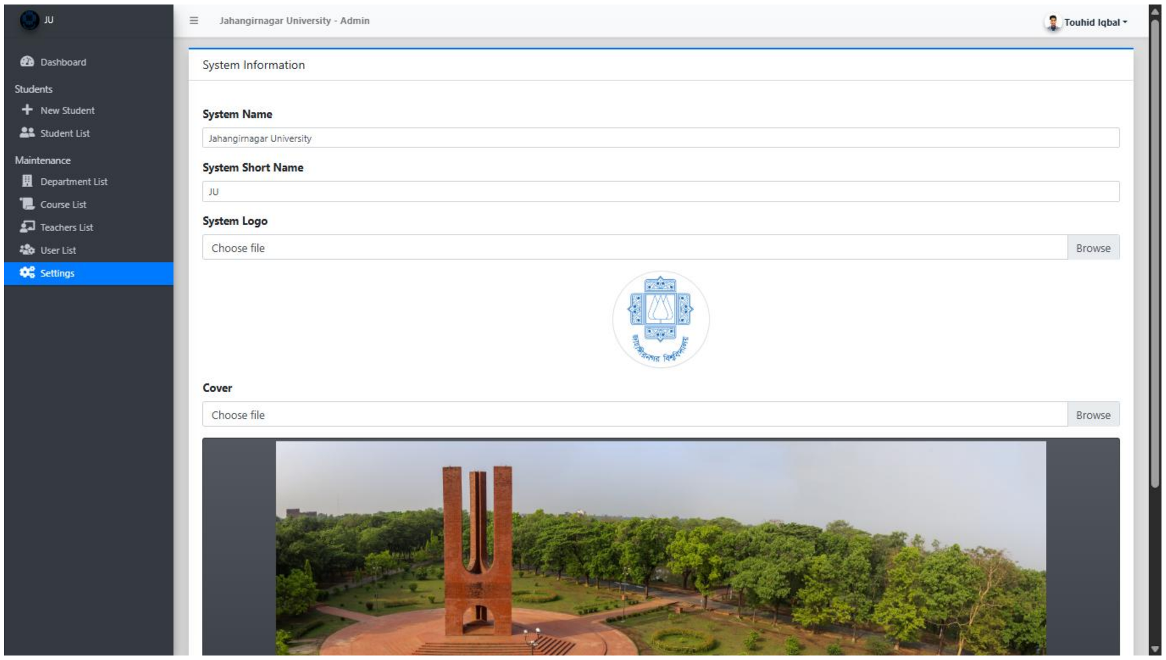
Copyright © 2025. All rights reserved.

JU (by: Touhid Iqbal Sagar) v1.0

Figure 4.8.1: User list page of Student Information System

4.9 Settings page

This page is used for settings purpose of the website in Figure 4.9.1.



System Information

System Name

Jahangirnagar University

System Short Name

JU

System Logo

Choose file

Cover

Choose file

Figure 4.9.1: Settings page of Student Information System

4.10 Testing Implementation

✓ Development methodology:

- **Requirements Gathering & Analysis:** Initial phase focused on identifying user needs and system expectations through stakeholder input and research.
- **System Planning & Design:** Outlining the system architecture, user interfaces, database structure, and component interactions.
- **Agile Development Cycle:** Leveraged an iterative, sprint-based workflow that allowed continuous refinement and integration of user feedback.
- **Testing & Quality Assurance:** Comprehensive testing was conducted at each stage to validate functionality, performance, and security.
- **Deployment & Ongoing Maintenance:** Final implementation into a live environment, with routine updates, bug fixes, and feature enhancements as needed.

✓ Programming language and framework used:

- **Programming Language:**
    - Core logic and backend functionality developed using **PHP**.
- **Frontend Development:**
    - Designed using **HTML** and **CSS** for structure and styling.
    - **Bootstrap** was used to ensure responsive and consistent UI/UX across devices.
- **Backend Framework:**
    - Implemented with **Laravel**, a powerful PHP framework offering modularity, security, and built-in tools.
- **Database Management:**
    - **MySQL** and **SQL Server** were used to handle data storage, retrieval, and relational integrity.
- **Version Control & Collaboration:**

    - **Git** for local version tracking.
    - **GitHub** for remote repository hosting and team collaboration.

Chapter 5

CONCLUSION AND FUTURE SCOPE

The creation of the **Student Management System** was driven by the need to streamline, digitize, and centralize student-related operations—ranging from enrollment and academic tracking to departmental coordination and administrative oversight. This system successfully brought together essential functionalities such as student registration, course allocation, grading, attendance management, and report generation, all integrated within a secure, role-specific interface accessible by administrators, instructors, and students alike.

Throughout the development journey, several complexities surfaced. A primary concern was ensuring **real-time data consistency** across interconnected modules—particularly when updates occurred simultaneously from different user roles. Safeguarding student data also became a top priority, prompting the integration of **role-based authentication** and **secure data transmission protocols** to prevent unauthorized access.

Architecting the system for future scalability posed another significant challenge. The design required a flexible **database schema** and a modular codebase that could accommodate institutional-specific expansions, such as additional departments or evolving grading policies. This required thoughtful abstraction and modularity to ensure future development wouldn't disrupt existing features.

User testing played a pivotal role in refining the system. Initial feedback pointed to the need for a more intuitive interface and clearer navigation paths. As a result, enhancements were made to improve usability, including the use of icons, tooltips, and color-coded indicators. A highly requested feature—the ability to generate **exportable and printable reports**—was later integrated, boosting administrative efficiency.

In conclusion, the project not only fulfilled its initial objectives but also evolved through feedback and iterative improvements. The final product is a responsive, secure, and scalable Student Management System capable of adapting to the dynamic needs of educational institutions, improving productivity and ensuring accurate academic record-keeping.

5.2 Future Work

- ✓ Mobile application support (Android, IOS)
- ✓ Online fee management.
- ✓ Timetable management.
- ✓ Automated Notifications.
- ✓ Integrated Learning Management System (LMS).
- ✓ Chat and Discussion Forum.
- ✓ Attendance Tracking.
- ✓ Parent and Student Portals.

References

[1] A.A. Eludire (August 30, 2015). Design and Implementation of Student Academic Record Management System. Retrieved from Academia: https://www.academia.edu/20105861/The_Design_and_Implementation_of_Student_Academic_Record_Management_System

[2] Abuka V. (May 2019). Design and Implementation of Student Information System. Retrieved from Academia: https://www.academia.edu/41948679/Design_and_Implementation_of_a_Student_Information_System

[3] Alwan, M. (2015, January 9). What is System Development Life Cycle? Retrieved from Airbrake: https://airbrake.io/blog/sdlc/what-is-system-development-life-cycle

[4] Didacus, (2018). What Does System Design Mean?. Retrieved from techopedia: https://www.techopedia.com/definition/29998/system-design

[5] Gagliordi, N. (2014, August 21). US universities at greater risk for security breaches than retail and healthcare: BitSight. Retrieved from ZDNet: https://www.zdnet.com/article/us-universities-at-greater-risk-for-security-breaches-than-retail-and-healthcare-bitsight/

[6] Al-Shaikhli, D. The effect of the tracking technology on students' perceptions of their continuing intention to use a learning management system. *Educ. Inf. Technol.* **28**(1), 343–371 (2023).

[7] Rajmane, S. S., Mathpati, S. R. & Dawle, J. K. Digitalization of management system for college and student information. *Res. J. Sci. Technol.* **8**(4), 179–184 (2016).

[8] Bessadok, A., Abouzinadah, E. & Rabie, O. Exploring students' digital activities and performances through their activities logged in learning management system using educational data mining approach. *Interact. Technol. Smart Educ.* **20**(1), 58–72 (2023).

[9] de Almeida Bizarria, F. P., Tassigny, M. M., & Barbosa, F. L. Decoloniality and university management: Unveiling knowledge in managerial narratives. **8**(11): 246–264 (2020).

[10] Santana, L. F. *et al.* Scrum as a platform to manage students in projects of technological development and scientific initiation: a study case realized at UNIT/SE. *J. Inform. Syst. Eng. Manag.* https://doi.org/10.20897/jisem.201707 (2017).