STUDENT MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

DENIS REMIJEUS A (8115U13AM014)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

in

COMPUTER SCIENCE AND ENGINEERING

(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE ,New Delhi)

SAMAYAPURAM–621112

DECEMBER - 2024

K.RAMAKRISHNANCOLLEGEOFENGINEERING (AUTONOMOUS)

SAMAYAPURAM–621112

BONAFIDE CERTIFICATE

Certified that this project report on **“STUDENT MANAGEMENT SYSTEM”** is the bonafide work of **DENIS REMIJEUS A(8115U23AM014)**who carried out the project work during the academic year 2024 - 2025 under my supervision.

Signature

Mr. P.GEETHA,

ASSISTANT PROFESSOR,

Department of Artificial Intelligence and Data Science,

K. Ramakrishnan College of Engineering,

Samayapuram, Trichy-621 112.

Signature

Dr. B.KIRAN BALA,B.Tech,M.E.,M.B.A.,Ph.d

HEAD OF THE DEPARTMENT,

Department of Artificial Intelligence and Machine Learning,

K. Ramakrishnan College of Engineering,

Samayapuram, Trichy-621 112.

.

Submitted for the End Semester Examination held on……………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

I declare that the project report on **“STUDENT MANAGEMENT SYSTEM”** is the result of original work done by me and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of BACHELOR OF ENGINEERING. This project reportis submitted on the partial fulfillment of the requirement of the award of the course **CGB1201-JAVA PROGRAMMING**

Signature

____

DENIS REMIJEUS A

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in debtedness to our institution, “**K.RAMAKRISHNAN COLLEGE OF ENGINEERING (Autonomous)**”,for providing me with the opportunity to do this project. I extend our sincere acknowledgment and appreciation to the esteemed and honorable Chairman, **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director , **Dr.S. KUPPUSAMY, MBA, Ph.D.,** for forwarding our project and offering an adequate duration to complete it.I would like to thank **Dr. D. SRINIVASAN, M.E., Ph.D., FIE., MIIW.,MISTE., MISAE., C. Engg.,** Principal, who gave the opportunity to frame the project to full satisfaction.

Dr. B. KIRAN BALA, B.Tech., M.E., M.B.A., Ph.D., M.I.S.T.E., U.A.C.E.E., IAENG, Head of the Department of Artificial Intelligence and Machine Learning, for providing his encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide**Mrs. P.GEETHA, M.E.,** Department of Artificial Intelligence and Data Science, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work .

INSTITUTE VISION AND MISSION

VISION OF THE INSTITUTE:

To achieve a prominent position among the top technical institutions.

MISSION OF THE INSTIITUTE:

M1: To be show standard technical education excellence through state of the art infrastructure, competent faculty and high ethical standards.

M2: To nurture research and entrepreneurial skills among students in cutting edge technologies.

M3: To provide education for developing high-quality professionals to transform the society.

DEPARTMENT VISION AND MISSION

DEPARTMENT OF CSE(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

Vision of the Department

To become a renowned hub for Artificial Intelligence and Machine Learning

Technologies to produce highly talented globally recognizable technocrats to meet

Industrial needs and societal expectations.

Mission of the Department

M1: To impart advanced education in Artificial Intelligence and Machine Learning,

Built upon a foundation in Computer Science and Engineering.

M2: To foster Experiential learning equips students with engineering skills to

Tackle real-world problems.

M3: To promote collaborative innovation in Artificial Intelligence, machine

Learning, and related research and development with industries.

M4: To provide an enjoyable environment for pursuing excellence while upholding

Strong personal and professional values and ethics.

Programme Educational Objectives (PEOs):

Graduates will be able to:

PEO1: Excel in technical abilities to build intelligent systems in the fields of

Artificial Intelligence and Machine Learning in order to find new opportunities.

PEO2: Embrace new technology to solve real-world problems, whether alone or

As a team, while prioritizing ethics and societal benefits.

PEO3: Accept lifelong learning to expand future opportunities in research and

Product development.

Programme Specific Outcomes (PSOs):

PSO1: Ability to create and use Artificial Intelligence and Machine Learning Algorithms, including supervised and unsupervised learning, reinforcement Learning, and deep learning models.

PSO2: Ability to collect, pre-process, and analyze large datasets, including data Cleaning, feature engineering, and data visualization..

PROGRAM OUTCOMES(POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problemanalysis:**Identify,formulate,reviewresearchliterature,andan

alyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectivelyon complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The School Management System is a Java-based application designed to simplify and streamline academic administration tasks in educational institutions. With a secure admin login feature, it ensures that only authorized personnel can access the system. Administrators can enroll students by assigning unique IDs and names, with data stored dynamically in an Array List for efficient management. The system allows for easy tracking of attendance by marking students as present or absent, while a dedicated module enables updates to fees paid by students, ensuring financial transparency. Additionally, the marks entry feature helps record and monitor academic performance, and the "View All Students" functionality provides a detailed summary of student records, including IDs, names, attendance, fees, and marks. Its intuitive, menu-driven console interface ensures ease of use, even for those with limited technical knowledge. Designed to be scalable, the system accommodates a growing number of student records, making it ideal for small to medium-sized institutions. By automating routine administrative processes, the system minimizes manual errors, saves time, and enhances efficiency, while also showcasing Java programming concepts such as object-oriented design, Array Lists, loops, and conditional statements. It is a user-friendly, reliable, and scalable solution for improving modern educational management.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The School Management System is a Java-based application that simplifies administrative tasks in educational institutions. It includes features for secure admin login, student enrollment, attendance tracking, fee updates, and marks entry. Data is dynamically managed using an Array List for flexibility and scalability. The intuitive menu-driven interface ensures ease of use, improving efficiency and reducing errors. This system is a reliable solution for modern academic management.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 Objective

The objective of the School Management System is to streamline administrative tasks in educational institutions by providing a user-friendly and efficient platform for managing student information. The system aims to automate routine processes such as student enrollment, attendance tracking, fee updates, and marks entry, reducing manual errors and saving time. It ensures secure access through an admin login, promotes transparency in financial management, and provides a clear overview of student performance. Designed for scalability, the system supports growing student records, making it suitable for small to medium-sized institutions while improving overall academic administration.

1.2 Overview

The School Management System is a Java-based application developed to simplify and automate administrative tasks in educational institutions. It offers a secure admin login feature for authorized access, ensuring data privacy. Key functionalities include student enrollment, attendance tracking, fee management, and marks entry, all of which are seamlessly managed through a menu-driven interface. The system uses an Array List to store and manage student records dynamically, making it efficient and scalable. Administrators can view comprehensive student details, including their attendance status, fees paid, and academic performance, providing an organized overview of student data. By automating routine tasks, the system reduces manual errors, enhances operational efficiency, and saves time. Its design is user-friendly and scalable, making it suitable for institutions aiming to modernize and improve their academic management processes.

1.3 Java Programming Concepts

The School Management System program is a strong demonstration of key Object-Oriented Programming (OOP) concepts, highlighting principles such as encapsulation, abstraction, inheritance, polymorphism, and modularity. Encapsulation is evident in the Student class, which securely binds student attributes like id, name, attendance, feesPaid, and marks with the relevant operations, ensuring data integrity and controlled access. The system also employs abstraction, where complex internal processes, such as managing student records, are hidden behind simple methods like markAttendance() or enterMarks() and a menu-driven interface, providing ease of use to administrators. While the current implementation does not explicitly demonstrate inheritance, it has potential for extension by introducing a parent Person class, enabling shared attributes and reusability for roles like Teacher or Admin. Polymorphism, another cornerstone of OOP, could allow methods like displayDetails() to behave differently for various user roles, making the program adaptable and scalable. The use of dynamic memory management, with an ArrayList to store student objects, ensures efficient handling of growing records. Furthermore, modularity is achieved by organizing the system into clearly defined methods, each responsible for specific tasks like enrolling students or updating fees, enhancing maintainability and debugging. These OOP principles not only make the program robust and user-friendly but also provide a foundation for future enhancements, ensuring flexibility and scalability for a wide range of educational institution management needs.These OOP principles make the program scalable, user-friendly, and maintainable, laying a strong foundation for future enhancements and adaptability.

An Array List is a dynamic data structure that allows elements to be added or removed without needing to specify a fixed size, making it more flexible than standard arrays. It supports index-based access, allowing efficient addition, removal, and retrieval of elements. Common methods like add(), remove(), and get() make it user-friendly and ideal for managing collections of data. Although access to elements by index is fast (constant time), inserting or deleting elements in the middle of the list can be slower due to the need for shifting elements. In contrast to a Linked List, which excels in frequent insertions and deletions, an ArrayList is optimized for quick access and iteration.

The Scanner class is used to handle real-time input from the console.User inputs for selecting roles (admin or user), choosing movies, entering the number of tickets, and navigating through the menu are all captured using the Scanner.This ensures an interactive experience, allowing users to perform operations without needing additional tools or interfaces.

The program uses different methods to manage student details easily. The enrollStudent() method adds new students with their ID and name. The markAttendance() method updates if a student is present or absent. The updateFees() method helps add fees paid by students. The enterMarks() method records students' marks, and the viewAllStudents() method displays all student details, including attendance, fees, and marks. The login() method ensures only authorized admins can use the system.

Exception handling is a programming technique used to manage runtime errors, ensuring the program runs smoothly even when unexpected issues occur. Java uses keywords such as try, catch, throw, throws, and finally to manage exceptions.

The try block is used to wrap the code that might cause an exception, and the catch block is used to handle the specific exceptions that occur during execution. The finally block, if present, executes regardless of whether an exception occurs, making it ideal for resource cleanup, such as closing files or database connections. The throw keyword is used to explicitly throw an exception, while throws is used in a method signature to declare exceptions that a method might throw, leaving the responsibility of handling them to the calling method. Exception handling ensures that the program remains stable and does not terminate unexpectedly due to unhandled errors, allowing developers to maintain control over error management and offer better user experiences.

The menu-driven program is a type of application where the user is presented with a list of options, typically numbered, from which they can select the desired operation to perform. In the School Management System, this approach is used to allow the administrator to manage various aspects of the system, such as enrolling students, marking attendance, updating fees, entering marks, or viewing all students. The program repeatedly displays the menu after each action, using a loop, until the user chooses to exit. The menu is implemented with simple choices, making it easy for the user to interact with the system. For example, after the user selects an option, the corresponding operation is executed through a switch statement, and then the menu reappears, ensuring a smooth flow. This method of interaction is intuitive and user-friendly, as it simplifies navigation and keeps the program organized.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work for the School Management System program aims to create a comprehensive, automated system for managing administrative tasks within educational institutions. The system will allow administrators to log in securely, ensuring only authorized access. It will provide functionalities such as student enrollment, where details like student ID and name will be stored dynamically using an Array List for easy retrieval and updates. Attendance management will be streamlined, enabling the admin to mark students as present or absent. Fee tracking will also be incorporated, allowing admins to update and view fees paid by students. The program will include a feature to enter and manage student marks, helping maintain academic records efficiently. The system's interface will be menu-driven, ensuring ease of navigation and use. It is designed to handle multiple student records, making it scalable for growing institutions. By automating common tasks, the system will reduce human errors, increase efficiency, and save time. In future enhancements, the program could integrate with a graphical user interface (GUI), support database connectivity for persistent storage, and include additional features for generating student reports and handling multiple user roles. Overall, the project will create a reliable, user-friendly, and scalable solution to improve academic management in schools.

2.2 Block Diagram

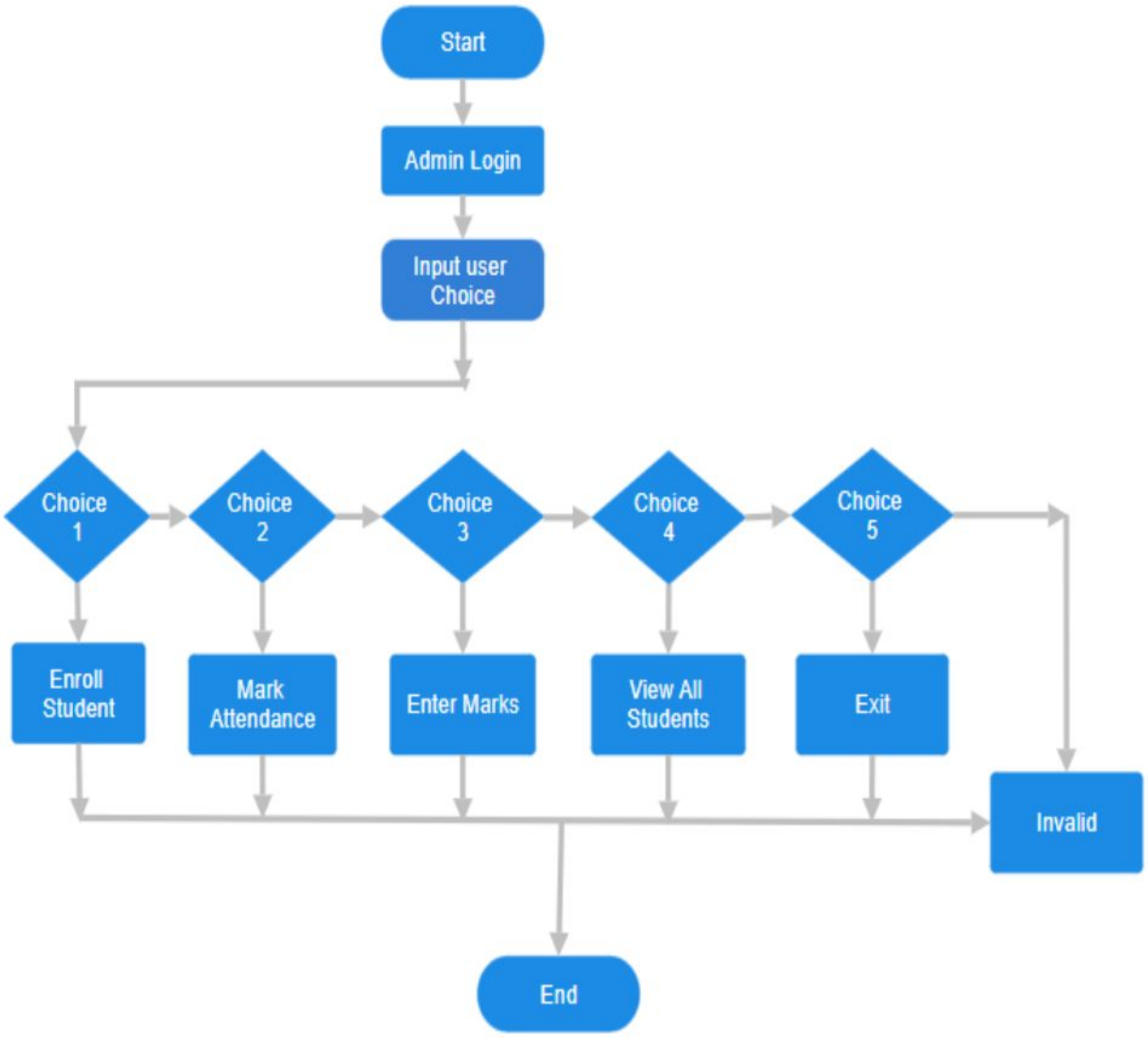


Fig. 2.1 School Management System Block Diagram

CHAPTER 3
MODULE DESCRIPTION

3.1 Admin Module

The Admin Login Module ensures secure access to the system by requiring authorized credentials. It verifies the admin's email and password before granting access to the program's functionalities. This validation prevents unauthorized users from manipulating student data. The module emphasizes system security and user accountability. Upon successful login, the admin can navigate through various system features such as enrollment, attendance, fees management, and academic performance tracking. Incorrect login attempts result in denial of access, safeguarding the system from malicious activities. This module serves as the gateway to the application, prioritizing the confidentiality and integrity of stored information. By restricting access, it ensures that sensitive data remains protected and can only be managed by authorized personnel.

3.2 Student Enrollment Module

The Student Enrollment Module is responsible for adding new students to the system. It collects essential details like student ID and name, which are stored in a dynamic data structure for efficient management. This module allows the admin to expand the database as more students join, ensuring scalability. Each student is assigned a unique identifier, simplifying record retrieval and updates. The enrollment process is user-friendly, guiding administrators through straightforward inputs to avoid errors. It forms the foundation for other modules, as attendance, fees, and marks are linked to enrolled students. The module ensures all students are systematically registered, providing a solid structure for managing academic and personal information.

3.3 Attendance Management Module

This module simplifies the process of marking and managing student attendance. The admin can record attendance status by selecting a student via their unique ID. It supports binary inputs to indicate whether a student is present or absent. The module ensures the accuracy of attendance records and reduces manual errors. These records can be used for various purposes, such as generating reports or evaluating academic engagement. The attendance data is integrated with the system's centralized student profile, providing a comprehensive overview. The module is intuitive and quick, making it an essential tool for maintaining attendance consistency and reliability.

3.4 Marks Entry Module

The Mark entry in a School Management System is a crucial feature for recording and managing students' academic performance. This module allows administrators to input marks for individual students, typically for specific subjects or overall performance. It ensures that every student's progress is accurately recorded and can be accessed for further analysis or reporting.In the system, the mark entry function prompts the admin to enter a student's ID, searches for the corresponding student record, and then requests the marks to be entered.

The system allows for flexibility in terms of entering marks for individual subjects or aggregate scores. After the marks are entered, the system updates the student's record with the new data, maintaining a seamless flow of information. This function can also include validation to check if the marks fall within acceptable ranges (e.g., 0 to 100), preventing incorrect data entry.

3.5 Fee Management Module

The Fees Management Module tracks and updates the fees paid by students. Administrators can input the amount paid by a student and view their financial status. It ensures transparency in monetary transactions and keeps records accurate and up-to-date. By linking fee details to student profiles, the module enables easy tracking of pending or completed payments. This module is essential for financial accountability and helps in generating reports for audits. It streamlines the otherwise complex process of managing school finances, ensuring clarity for both the administration and students.

3.6 View All Students Module

The View Students Module provides a comprehensive display of all enrolled student information, including IDs, names, attendance, fees paid, and academic marks. It acts as a centralized dashboard, allowing administrators to access and manage data efficiently. This module ensures that all relevant information is available in one place, simplifying decision-making and analysis. It provides a snapshot of the current state of the database, highlighting gaps or inconsistencies in the records. This centralized approach saves time and enhances the overall efficiency of the system, making it an indispensable part of the program.

These modules work together to manage the school's data effectively and efficiently, minimizing errors and streamlining administrative processes.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The School Management System is a feature-rich, Java-based application designed to simplify and automate critical administrative tasks in educational institutions. Leveraging core Java programming concepts, such as object-oriented principles, modularity, and dynamic data management, this system ensures scalability, flexibility, and reliability. The program begins with a secure admin login system, requiring authorized credentials to prevent unauthorized access, thus safeguarding sensitive institutional and student data.

The system facilitates student enrollment, allowing administrators to assign unique IDs and names to students. These records are dynamically stored using an efficient data structure, ensuring easy retrieval and updates. Attendance management is simplified through a dedicated feature that lets administrators mark students as present or absent, streamlining this routine process. The program also incorporates a fees management module, enabling administrators to update and track student payments, ensuring transparency and aiding financial oversight.

To monitor academic performance, the system includes a marks entry feature, where administrators can record and update students' scores. A comprehensive view-all module provides a detailed summary of all enrolled students, displaying their attendance, fees status, and academic performance in an organized manner. With its menu-driven console interface, the program is intuitive and user-friendly, allowing smooth navigation between features and making it accessible to non-technical users.

Error-handling mechanisms are integrated into the system to validate user inputs, ensuring that only correct and complete information is processed. For example, invalid student IDs are flagged, and incorrect menu selections prompt clear feedback, preventing runtime errors and improving user experience. This robust validation minimizes operational disruptions and contributes to the program's reliability.

Looking to the future, the system can be expanded with cloud-based storage to allow real-time access to data from remote locations, enhancing convenience and collaboration. Biometric attendance systems could be integrated for improved accuracy, while features like automated performance analytics and personalized academic reports would provide valuable insights for students and educators alike. Additionally, a mobile application extension could be developed to offer users on-the-go access to student data and administrative functions.

By automating essential processes such as enrollment, attendance, fees tracking, and performance monitoring, the School Management System saves time, reduces manual errors, and boosts overall efficiency. Its scalability ensures that it can adapt to the growing needs of small and medium-sized institutions. With the potential for continuous upgrades and advanced integrations, this system is a modern, reliable, and indispensable tool for streamlining educational administration and fostering better management practices. It not only addresses current challenges but also sets the stage for a technologically advanced future in education.

4.2 FUTURE SCOPE

The School Management System has significant potential for future enhancements to meet evolving needs in educational administration. As educational institutions embrace digital transformation, this system can be further developed to incorporate cutting-edge technologies, improving efficiency and functionality. One prominent area of improvement is the integration of cloud-based storage solutions, enabling administrators and educators to access student data in real-time from anywhere. This would also facilitate collaborative decision-making among different departments within the institution.Another important advancement could be the addition of biometric systems for attendance tracking. This feature would eliminate manual errors and ensure accurate, real-time attendance records, making the process faster and more reliable. Mobile application development is another promising avenue, allowing parents, students, and administrators to interact with the system on the go. Such an app could provide features like fee payment, viewing attendance, marks, and receiving notifications about important updates, enhancing user convenience.

The implementation of AI and machine learning algorithms could transform the way data is analyzed within the system. These technologies can generate predictive analytics to identify patterns in student performance, attendance, and behavior, helping educators provide personalized support. Automated performance reports and recommendations for academic improvement could also be introduced, saving time for teachers and adding value to the student experience.In conclusion, the future scope of the School Management System is vast, with opportunities for technological integration, enhanced security, and user-centric features. By embracing these advancements, the system can continue to support institutions in achieving operational excellence, fostering a more effective and streamlined educational environment.

APPENDIX A
(SOURCE CODE)

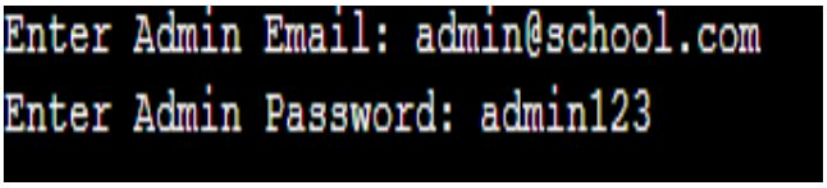
```
import java.util.Scanner;

class Student {
    int id;
    String name;
    boolean attendance;
    double marks;

    public Student(int id, String name) {
        this.id = id;
        this.name = name;
        this.attendance = false;
        this.marks = 0.0;
    }

    public void displayStudent() {
        System.out.println("ID: " + id + ", Name: " + name + ", Attendance: " +
(attendance ? "Present" : "Absent") + ", Marks: " + marks);
    }
}

public class SchoolManagementSystem {
    static final int MAX_STUDENTS = 100;
    static Student[] students = new Student[MAX_STUDENTS];
    static int studentCount = 0;
    static Scanner scanner = new Scanner(System.in);
static final String ADMIN_EMAIL = "admin@school.com";
static final String ADMIN_PASSWORD = "admin123";

public static void main(String[] args) {
    if (login()) {
        int choice;
        do {
            System.out.println("\n==== School Management System ====");
            System.out.println("1. Enroll Student");
            System.out.println("2. Mark Attendance");
            System.out.println("3. Enter Marks");
            System.out.println("4. View All Students");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1: enrollStudent(); break;
                case 2: markAttendance(); break;
                case 3: enterMarks(); break;
                case 4: viewAllStudents(); break;
                case 5: System.out.println("Exiting..."); break;
                default: System.out.println("Invalid choice. Please try again.");
            }
        } while (choice != 5);
    } else {
        System.out.println("Invalid login credentials. Exiting...");
    }
    }

    public static boolean login() {
        System.out.print("Enter Admin Email: ");
        String email = scanner.next();
        System.out.print("Enter Admin Password: ");
        String password = scanner.next();

        return ADMIN_EMAIL.equals(email) &&
ADMIN_PASSWORD.equals(password);
    }

    public static void enrollStudent() {
        if (studentCount >= MAX_STUDENTS) {
            System.out.println("Cannot enroll more students. Capacity reached.");
            return;
        }

        System.out.print("Enter Student ID: ");
        int id = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        System.out.print("Enter Student Name: ");
        String name = scanner.nextLine();

        students[studentCount++] = new Student(id, name);
        System.out.println("Student enrolled successfully!");
    }

    public static void markAttendance() {
```

```
    System.out.print("Enter Student ID to mark attendance: ");
    int id = scanner.nextInt();
    boolean found = false;

    for (int i = 0; i < studentCount; i++) {
        if (students[i].id == id) {
            System.out.print("Mark present (1) or absent (0): ");
            students[i].attendance = scanner.nextInt() == 1;
            System.out.println("Attendance updated for " + students[i].name);
            found = true;
            break;
        }
    }

    if (!found) {
        System.out.println("Student not found.");
    }
}

public static void enterMarks() {
    System.out.print("Enter Student ID to enter marks: ");
    int id = scanner.nextInt();
    boolean found = false;

    for (int i = 0; i < studentCount; i++) {
        if (students[i].id == id) {
            System.out.print("Enter marks for " + students[i].name + ": ");
            students[i].marks = scanner.nextDouble();
            System.out.println("Marks updated for " + students[i].name);
```

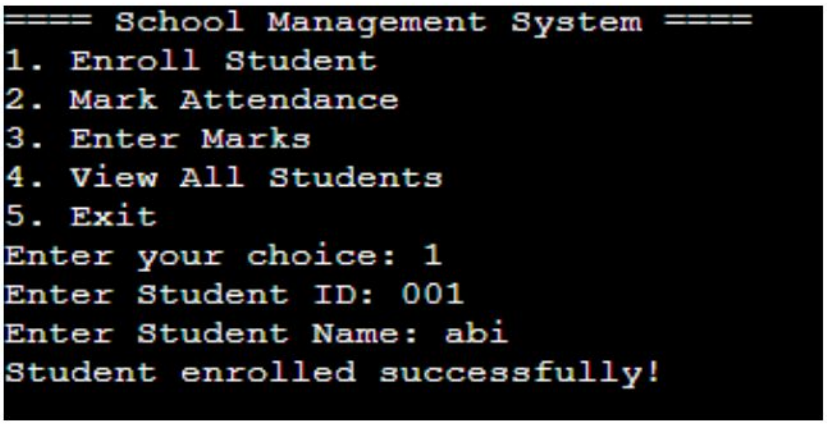
```
                found = true;
                break;
            }
        }

        if (!found) {
            System.out.println("Student not found.");
        }
    }

    public static void viewAllStudents() {
        if (studentCount == 0) {
            System.out.println("No students enrolled.");
            return;
        }

        System.out.println("\nList of Enrolled Students:");
        for (int i = 0; i < studentCount; i++) {
            students[i].displayStudent();
        }
    }
}
```

APPENDIX B
(SCREENSHOTS)

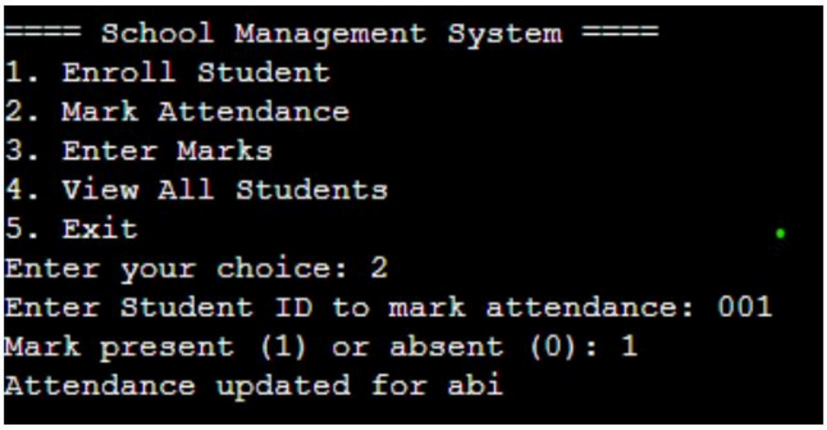
ADMIN


```
Enter Admin Email: admin@school.com
Enter Admin Password: admin123
```

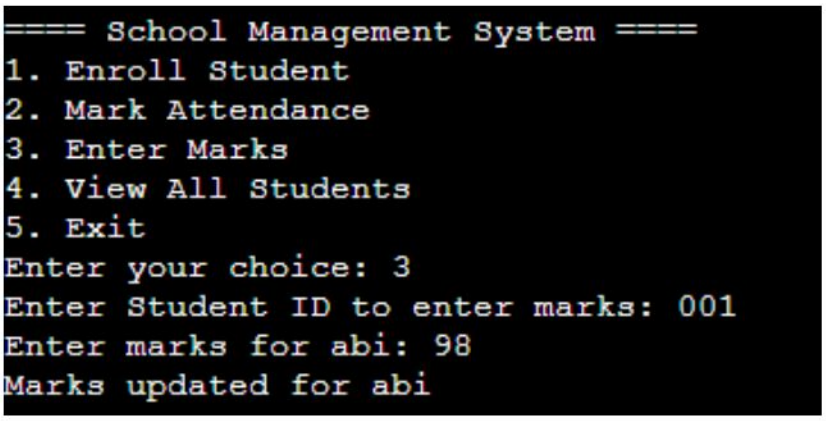
ENROLL STUDENT


```
==== School Management System ====
1. Enroll Student
2. Mark Attendance
3. Enter Marks
4. View All Students
5. Exit
Enter your choice: 1
Enter Student ID: 001
Enter Student Name: abi
Student enrolled successfully!
```

MARK ATTENDANCE

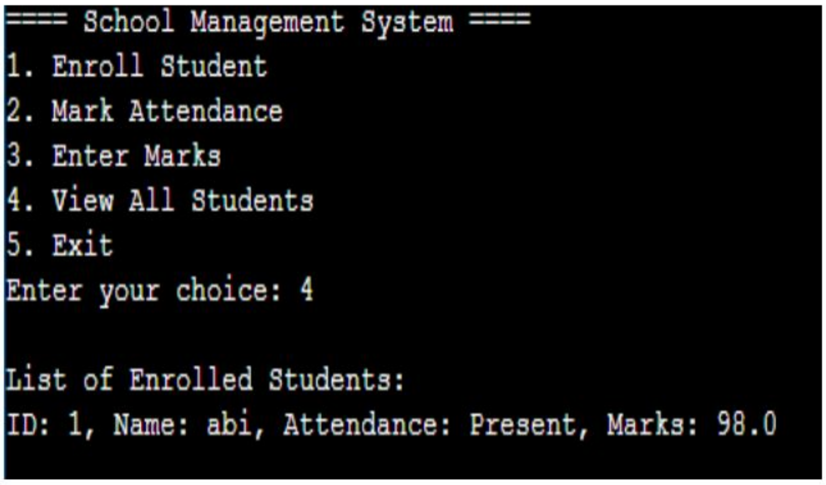

```
==== School Management System ====
1. Enroll Student
2. Mark Attendance
3. Enter Marks
4. View All Students
5. Exit
Enter your choice: 2
Enter Student ID to mark attendance: 001
Mark present (1) or absent (0): 1
Attendance updated for abi
```

ENTER MARKS



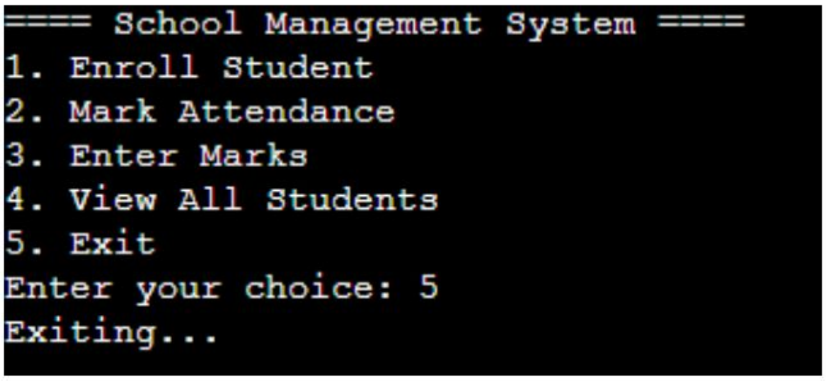
```
==== School Management System ====
1. Enroll Student
2. Mark Attendance
3. Enter Marks
4. View All Students
5. Exit
Enter your choice: 3
Enter Student ID to enter marks: 001
Enter marks for abi: 98
Marks updated for abi
```

VIEW ALL STUDENTS



```
==== School Management System ====
1. Enroll Student
2. Mark Attendance
3. Enter Marks
4. View All Students
5. Exit
Enter your choice: 4

List of Enrolled Students:
ID: 1, Name: abi, Attendance: Present, Marks: 98.0
```

EXIT



```
==== School Management System ====
1. Enroll Student
2. Mark Attendance
3. Enter Marks
4. View All Students
5. Exit
Enter your choice: 5
Exiting...
```

REFERENCES

1. Cay S. Horstmann, Core Java Volume I: Fundamentals (11th Edition), Pearson Education.
2. Herbert Schildt. "Java: The Complete Reference", 11th Edition. McGraw Hill Education, 2018.
3. https://www.w3schools.com/java