RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

STUDENT RESULT MANAGEMENT SYSTEM

Submitted by:

BENJAMIN NICOLAS S (221801005)
CHARLESS BINNY K (221801007)
MADHUMITHA G (221801030)

AD19541 SOFTWARE ENGINEERING METHODOLOGY

Department of Artificial Intelligence and Data Science

Rajalakshmi Engineering College, Thandalam

BONAFIDE CERTIFICATE

Certified that this project report "**STUDENT RESULT MANAGEMENT SYSTEM**" is the bonafide work of **"BENJAMIN NICOLAS S (221801005), CHARLESS BINNY K(221801007), MADHUMITHA G (221801030) "**who carried out the project work under my supervision.

Submitted for the Practical Examination held on _____

SIGNATURE

DR. J M GNANASEKAR
Professor & Head,
Dept. of Artificial Intelligence and Data Science,
Rajalakshmi Engineering College

SIGNATURE

Dr. MANORANJINI J
Associate Professor,
Dept. of Artificial Intelligence and Data Science,
Rajalakshmi Engineering College

INTERNAL EXAMINER

EXTERNAL EXAMINER

TABLE OF CONTENT

TABLE OF FIGURES

ABSTRACT

The Student Result Management System is a streamlined, secure digital platform designed to handle the management, storage, and dissemination of academic results for educational institutions. As institutions increasingly move away from manual, paper-based result systems, this solution leverages modern technology to simplify result processing, improve data accessibility, and ensure secure handling of student information. This system enables students to view their results online through a dedicated student portal, while administrators can easily input and update grades via an admin portal. The system architecture integrates key components such as an API Gateway for secure communication, an Authentication Service for user verification, and a Result Processing Service for managing data processing and storage. Two primary databases, Student Data DB and Results DB, are used to store student information and academic records securely, ensuring data integrity and privacy.

Developed using the waterfall model, the project follows a structured approach with distinct phases, including requirements analysis, design, implementation, testing, and deployment. Each stage is completed before moving on to the next, which ensures thorough testing and a clear progression from system requirements to a functional, deployed solution. The system also incorporates automation features, including notifications and email confirmations, to enhance user experience and streamline communication. By using secure encryption and authentication protocols, the system ensures that sensitive student data is protected against unauthorized access. This project is aimed at creating an efficient, reliable, and scalable solution for managing academic results, offering benefits in terms of accuracy, security, and convenience for both students and staff. Overall, the Student Result Management System addresses key challenges in academic data management, offering a practical and robust solution that meets the demands of modern educational institutions.

CHAPTER 1

INTRODUCTION

1.1 General

In educational institutions, the process of managing student results is fundamental yet often labor-intensive, involving data collection, grade calculation, and result distribution. Traditionally, result management relied on manual or semi-digital methods, which were prone to errors, time-consuming, and resource-intensive. With the growing number of students and courses, handling this data accurately and efficiently has become a challenge for administrators and educators alike. These issues highlight the need for a streamlined, automated, and secure system that can enhance the accuracy and efficiency of result management in educational settings.

The Student Result Management System addresses these needs by offering a comprehensive digital platform that automates the calculation, storage, and dissemination of student results. Through an intuitive interface, the system allows authorized users, such as administrators and teachers, to securely input, process, and publish grades with minimal manual intervention. By leveraging modern technology, this system reduces the likelihood of calculation errors, increases the speed of report generation, and minimizes the administrative workload associated with traditional result management processes.

One of the key features of the system is its ability to ensure data security and privacy, which are critical when dealing with sensitive academic information. The Student Result Management System employs encrypted databases and role-based access control to protect student records from unauthorized access. This helps institutions comply with data protection regulations and builds trust among students, parents, and faculty by

safeguarding personal and academic information. Only authorized users can access or modify data, ensuring that student records remain accurate and confidential.

In addition to improving data accuracy and security, the system enhances accessibility for students and faculty. Students can view their results in real-time once grades are finalized, removing the need for printed report cards and on-site visits. Faculty and administrators benefit from a user-friendly dashboard that enables them to quickly generate and manage individual or class-wide reports. This digital accessibility not only makes result retrieval more convenient but also promotes transparency in academic evaluations, fostering a better educational experience for students.

Overall, the Student Result Management System is a scalable, reliable solution designed to meet the evolving needs of modern educational institutions. By integrating secure, automated, and efficient result management processes, this system minimizes administrative effort, enhances data integrity, and supports better decision-making based on accurate academic records. As educational institutions continue to embrace digital transformation, systems like this will play a vital role in streamlining operations, promoting accessibility, and improving academic administration.

1.2 Need for the Study

The need for an efficient, secure, and accessible **Student Result Management System** has become increasingly urgent in today's educational landscape. Traditional methods of managing student results often involve manual entry, calculation, and physical record-keeping, all of which are prone to human error, time delays, and resource constraints. As institutions grow in size and the number of enrolled students increases, managing this data manually becomes a logistical challenge that can negatively impact both administrative efficiency and student satisfaction. A digital result management system addresses these challenges by providing automated calculation, secure data storage, and quick access to results, saving time and reducing the administrative burden on staff.

Beyond improving operational efficiency, the need for data security in educational institutions has grown, especially with rising concerns around data privacy and protection. Student records include sensitive information, from personal details to academic performance, which must be stored and accessed securely to prevent unauthorized access. Traditional storage methods, such as physical files or basic unencrypted digital systems, lack the necessary security measures to protect student data. Implementing a digital Student Result Management System with encrypted databases and role-based access control allows institutions to safeguard data and comply with privacy standards, giving students, parents, and faculty peace of mind regarding data confidentiality.

Furthermore, students today expect more accessibility and transparency in academic processes, including the ability to view their results in real-time. Traditional methods often require students to wait for physical report cards, which can delay feedback on their performance and hinder their ability to make timely academic decisions. With an online result management system, students can access their grades immediately after publication, regardless of location, fostering a sense of transparency and responsiveness in academic evaluations. This level of accessibility not only enhances the student experience but also aligns with the broader trend of digital transformation in education, making it a necessary evolution for institutions aiming to keep pace with modern expectations and technological advancements.

1.3 Objectives of the Study

The objectives of the Student Result Management System project are designed to create an efficient, accurate, and secure platform for managing and distributing student results**.**

Automate Grade Calculation and Result Processing: The system will automate the calculation of final grades based on predefined criteria, eliminating the need for manual calculations and reducing the potential for errors. This ensures results are processed quickly and accurately, allowing staff to focus on other critical academic tasks.

Enhance Data Security and Privacy: The system will protect sensitive student data through encrypted storage and role-based access control, ensuring only authorized users can access or modify student records. This maintains confidentiality and integrity of academic data, in compliance with data protection regulations and institutional privacy policies.

Streamline Report Generation and Accessibility: The project aims to enable easy and automated generation of student report cards that can be viewed, downloaded, or printed by authorized users. This simplifies the reporting process for administrators and teachers, allowing them to generate comprehensive grade reports quickly and distribute them efficiently.

Provide Real-Time Access to Results: The system will allow students and faculty to access results as soon as grades are finalized, enhancing accessibility and minimizing wait times. This ensures that students have immediate access to their academic records, supporting informed decision-making and academic planning.

Create a User-Friendly Interface: This objective focuses on designing an intuitive, easy-to-navigate interface for administrators, teachers, and students, reducing the learning curve and improving user experience. The system will be simple for users of all technical abilities to manage and access results, fostering high engagement and satisfaction.

Increase Operational Efficiency and Reduce Workload: By automating repetitive tasks associated with managing and calculating results, the system will reduce the administrative workload, freeing up staff time and resources, and enabling the institution to operate more efficiently with fewer errors in the grading process.

Facilitate Long-Term Record Keeping and Retrieval: The system will securely store and maintain historical academic records for future reference, supporting trend analysis and accreditation requirements. This allows educational institutions to keep comprehensive, organized, and easily retrievable records of student performance over time.

Support Scalability and Future Growth: The system will be flexible and scalable, capable of handling an increasing number of students, courses, and additional features as the institution grows. This ensures that the system can evolve to meet future needs without compromising performance or data security.

1.4 Overview of the Project

The **Student Result Management System** is a comprehensive digital platform designed to streamline the process of managing, calculating, and distributing student grades in educational institutions. By automating grade calculation and report generation, the system minimizes errors and reduces administrative workload, ensuring timely and accurate results. It offers a secure, user-friendly interface with role-based access, allowing administrators, teachers, and students to view and manage academic records safely. Through encrypted data storage and real-time access, the system protects sensitive information while enhancing accessibility for users. This scalable and efficient solution is aimed at modernizing result management, increasing transparency, and supporting data-driven decision-making in education.

WORKFLOW

A student result management system can be developed in five comprehensive stages: **Planning and Requirements Analysis, System Design, Development, Testing, and Deployment and Maintenance**. Each of these sections provides a framework for an organized workflow, ensuring the final system is effective, efficient, and user-friendly.

1. Planning and Requirements Analysis

The first phase focuses on gathering and defining requirements, a foundational step in ensuring the system meets user needs. Key stakeholders, including teachers, students, and administrators, are consulted to understand the necessary features and functionalities. This phase covers identifying the scope, such as registering students, inputting exam scores, generating averages and grades, and providing secure access to results. Additionally, it's essential to select a technology stack that suits the project requirements, such as a programming language, a web framework (like Django for

Python or Laravel for PHP), and a database system. This planning lays the groundwork for the project and minimizes the likelihood of scope changes later on.

2. System Design

In the design phase, both the database and the user interface (UI) are structured to align with the functional requirements. Database design is a critical part, requiring a well-organized schema that includes tables for students, subjects, exams, scores, and roles to capture the different types of data accurately. Defining relationships among these tables (e.g., each student can have multiple scores) ensures that the system is structured for efficient data management and retrieval. On the front end, intuitive UI design is essential to ensure easy navigation for various user roles, making data input and result retrieval simple and straightforward. Tools like Figma or Adobe XD are helpful here, as they allow the team to create and visualize user interface prototypes before development begins.

3. Development

The development stage involves coding both the frontend and backend to turn design concepts into functional components. On the frontend, pages are created for activities like student registration, score entry, and result viewing, utilizing HTML, CSS, and JavaScript, or a framework like React or Vue for dynamic elements. Backend development involves creating API endpoints or functions that handle data transactions, such as adding students, updating scores, and calculating grades. This code is responsible for the application's logic, managing the secure processing and storage of data, and interacting with the database to ensure that users can perform their tasks. Additionally, implementing role-based access control (for instance, different privileges for teachers, students, and admins) ensures that data privacy and security are maintained.

4. Testing

Once development is complete, testing verifies that all components function as expected. Unit testing is conducted to test individual modules, like score calculations and database queries, to ensure they work independently without errors. Integration testing follows to verify that the interaction between the frontend and backend works seamlessly, ensuring data flows accurately through the system. User acceptance testing (UAT) involves having end-users test the system, providing feedback to improve usability or functionality. Finally, performance testing ensures that the system can handle a large volume of data and users efficiently, making it reliable under various conditions. Testing is crucial for identifying bugs, optimizing performance, and improving the system before release.

5. Deployment and Maintenance

The final stage involves deploying the system on a server or cloud platform, such as AWS or Heroku, making it accessible to users. Deployment includes configuring security protocols to protect sensitive data, setting up database backups, and creating a maintenance plan to ensure consistent performance. Continuous monitoring is essential to address any arising issues quickly, and user feedback is valuable for planning future enhancements. Documentation of the system is created, covering functionality, setup, and troubleshooting, which serves both as a reference for users and as support for future developers maintaining the system. Maintenance ensures the system remains up-to-date, functional, and adaptable to new requirements over time.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

Over the past few decades, student result management has evolved significantly, especially with technological advancements that streamline how academic results are recorded, processed, and shared. Modern result management systems aim to address both institutional efficiency and user experience, as these systems are integral to educational operations and impact students, teachers, and administrators alike. Literature surrounding student result management systems indicates a growing demand for reliable, secure, and adaptable platforms that can manage large amounts of academic data efficiently. Key themes in the literature focus on system accessibility, data security, user-friendly design, and accuracy, all of which are essential in supporting educational institutions' requirements.

One prominent area of focus is data security, especially considering the sensitive nature of academic records, which often include personal and performance-related information. Studies have emphasized the importance of employing encryption techniques and secure storage solutions, such as relational databases and scalable databases like MongoDB, to protect against unauthorized access and data breaches. Another significant area of research addresses the usability of result management systems, particularly in creating interfaces that simplify navigation for users with varying technical skills. Many studies suggest that intuitive designs help reduce errors and enhance accessibility, making it easier for students and educators to interact with the system.

Another advancement highlighted in the literature is the automation of result calculations, report generation, and notifications. Automated processes reduce administrative workload, minimize human error, and provide timely updates to students, thereby improving overall satisfaction and engagement. The use of automated

notifications, such as email or SMS alerts, is found to increase transparency and help students access their academic results in real time.

Thus, the existing literature on student result management systems lays a foundation for building a platform that integrates secure data handling, automation, and usability. This project builds on these insights, aiming to address identified gaps in security, efficiency, and user experience, ultimately providing a reliable, accessible, and student-friendly solution for modern educational institutions.

2.2 LITERATURE REVIEW

S. No	Author Name	Paper Title	Description	Journal	Volume/Year
1	D. G. Patel and H. P. Modi,	"A Review on Student Result Management System",	Reviews advancements in digital student result management systems.	International Research Journal of Engineering and Technology (IRJET)	vol. 9, no. 7, pp. 2963-2966, July 2022.
2	M. W. Bara,	"Advanced Automated Result Processing and Management System: (A Case Study of Mai Idris Alooma Polytechnic Geidam Yobe State)",	Discusses an automated result processing system at Mai Idris Alooma Polytechnic to enhance efficiency and result management.	International Journal of Advances in Engineering and Management (IJAEM)	vol. 3, no. 3, pp. 1384-1391, March 2021.

3	M. Sami and N. A. Sharma	"Learning Computer Modules Using Multimedia and Social Media Platform in a Developing Country"	Explores multimedia and social media for teaching computer science in developing countries to boost engagement.	IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)	2021
4	K. S. Kumar, K. Chandrakala, M. Manogna, M. Alekhya, T. Reshma and M. Arshiya,	"STUDENT RESULT MANAGEMENT SYSTEM Using Web Technologies",	Details a web-based student result management system using modern technologies for better accessibility.	Juni Khyat I (UGC Care Group I Listed Journal),	vol. 11, no. 1, pp. 476-480, 2021.

Table 1: Literature Review

CHAPTER 3

SYSTEM OVERVIEW

3.1 EXISTING SYSTEM

In traditional educational institutions, academic record management is often conducted through fragmented systems or even manually, leading to inefficiencies and potential data inconsistencies. Typically, schools and universities use separate tools for different tasks: one system for managing grades, another for attendance, and others for student information management and announcements. These systems are usually not integrated, which requires administrators to enter and update data across multiple platforms. This lack of centralization can lead to redundant data entry, errors, and time-consuming processes. Students, too, may need to access various platforms to view their grades, read notices, and track their academic progress, which can be confusing and inconvenient.

Another limitation of existing systems is the lack of real-time access and automation. In many cases, students can only access their results at the end of the term, and they may not receive notifications about updates unless they check manually or are informed in person. Important academic announcements, like exam schedules or result publication dates, are often communicated through physical notice boards, emails, or a dedicated web page, which may not be updated promptly or accessed easily. Additionally, the security of student data in these older systems may not be as robust, leaving sensitive information vulnerable to unauthorized access. Overall, traditional academic management systems are characterized by disjointed functionalities, manual processes, and limited accessibility, resulting in an inefficient experience for both students and administrators.

In recent years, some institutions have adopted more modern software solutions, such as learning management systems (LMS) and enterprise resource planning (ERP) platforms, which offer integrated modules for managing student data, grades, and communication. However, these systems may still lack certain specialized features,

such as predictive analytics or personalized notifications, and can sometimes be cost-prohibitive for smaller institutions.

High Error Rates: Manual calculations are susceptible to human error, which lcan lead to incorrect results.

- **Time-Intensive Processes:** Processing results for a large student body can take weeks, creating delays in grade distribution.
- **Data Insecurity:** Physical records and unsecured databases are vulnerable to loss, theft, and unauthorized access.
- **Limited Access for Students:** In manual systems, students often have to wait for printed results, which is inconvenient and inefficient.

3.2 PROPOSED SYSTEM

The proposed student result management system aims to address the modern needs of educational institutions by integrating a secure, efficient, and user-friendly platform for managing academic records. Building on the insights from existing literature, this system is designed to streamline the process of result entry, calculation, storage, and retrieval, ensuring that all stakeholders — students, teachers, and administrators — benefit from an organized and accessible platform. The primary objectives of the proposed system include enhancing data security, automating result processing, and improving the overall user experience through intuitive design and real-time communication.

To meet the demand for secure data handling, the system will employ a robust database architecture featuring encrypted data storage and access control mechanisms. Data security is a crucial aspect given the sensitivity of academic records, and the proposed system will use encryption methods and scalable databases like MongoDB to prevent unauthorized access and ensure data integrity. In addition, role-based access control will restrict system permissions based on user roles, allowing administrators to manage user accounts securely. This approach will create a trustworthy environment that safeguards

both student information and institutional data, aligning with privacy regulations and addressing concerns around data breaches.

Another key component of the proposed system is the automation of repetitive and time-consuming processes. Automating calculations, grade assignments, and result notifications will significantly reduce the administrative workload, minimize human error, and enable faster processing times. For instance, once scores are entered, the system will automatically compute overall grades, percentages, and averages based on predefined criteria. This automation extends to result dissemination, where automated notifications — via email or SMS — will promptly inform students of their results, fostering transparency and immediate access to academic performance data.

Furthermore, the proposed system places a strong emphasis on usability to accommodate users with varying levels of technical expertise. The interface will be designed for easy navigation, allowing teachers and administrators to input data seamlessly and students to access their results intuitively. A dashboard will provide an overview of important information, including individual and cumulative performance metrics, which students can review without assistance. This design will create an accessible environment, promoting user engagement and reducing the learning curve associated with the system.

In conclusion, the proposed student result management system aims to create a reliable, secure, and efficient solution for educational institutions. By combining secure data handling practices, process automation, and user-friendly design, the system addresses the critical needs identified in the literature. This project aspires to bridge gaps in current systems, offering a comprehensive platform that enhances the accuracy, security, and accessibility of student result management in modern educational settings.

3.3 FEASIBILITY STUDY

The **operational feasibility** of this project is highly favorable, as it addresses key pain points in current academic result management practices by automating manual tasks and securing data storage. Institutions will benefit from streamlined workflows that

minimize human error and reduce administrative workloads. The system's user-friendly interface will allow teachers, students, and administrators to perform tasks efficiently, resulting in improved transparency and satisfaction. By aligning with institutional goals of accuracy, accessibility, and efficiency, the system will integrate seamlessly into daily academic processes, proving both manageable and valuable for educational institutions of varying sizes and capacities.

The **technical feasibility** of the project is strong, supported by the availability of reliable technologies for secure data storage, automation, and user interface design. The system will leverage scalable databases, such as MongoDB, and encryption methods for data protection, addressing privacy and security concerns effectively. Automation tools and frameworks for result calculation and notifications are well-established, making them practical and achievable within the project's scope. The system's architecture is designed for compatibility with diverse institutional IT infrastructures, ensuring it can be readily deployed and maintained with minimal technical complications or specialized hardware requirements.

The **economic feasibility** of the project is promising, as it provides long-term cost savings by reducing the time and labor spent on manual result processing. Although initial setup costs are associated with development, implementation, and training, these investments are balanced by the reduction in administrative workload and increased process efficiency. Automated notifications and secure data handling will also reduce potential costs associated with data breaches or errors. This solution thus offers high value for its investment, enabling institutions to modernize their result management processes within a cost-effective framework.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 SOFTWARE REQUIREMENTS

Operating System:

- **Server**: Windows Server, Linux (e.g., Ubuntu, CentOS) or cloud OS (if using cloud hosting).
- **Client Devices**: Windows 10 or higher, macOS, Linux, Android, or iOS.

Database Management System (DBMS):

- MongoDB (for scalability and support for encrypted data storage) or an alternative relational DBMS like MySQL/PostgreSQL if structured data is preferred.

Backend Development Framework:

- Python (Django or Flask) or PHP (Laravel) to handle server-side processing, authentication, and API creation for secure data handling.

Frontend Development Tools:

- HTML, CSS, JavaScript for basic UI.
- Optional frameworks like React, Vue, or Angular for enhanced user experience and interactivity.

Automation and Notifications:

- Email and SMS integration libraries (e.g., Twilio for SMS notifications, SMTP for emails) to automate communication.

Security Tools:

- SSL/TLS for secure data transmission.
- Encryption libraries for secure storage (AES or RSA for data encryption).

Browser Compatibility:

- The application should support popular browsers like Chrome, Firefox, Safari, and Edge for cross-device accessibility.

4.2 Hardware Requirements

Server Requirements: A dedicated or cloud server is recommended for hosting the system. Specifications should include:

- **Processor**: Multi-core CPU (e.g., Intel Xeon or AMD EPYC) for handling multiple concurrent users and data processing tasks.
- **Memory**: Minimum 8 GB RAM for small institutions; 16 GB or higher for larger institutions to handle high volumes of data.
- **Storage**: SSD storage with a minimum of 256 GB, scalable based on data size (larger storage for bigger institutions). Storage should support encryption for secure data handling.
- **Network Connectivity**: High-speed internet with a stable connection to ensure accessibility for all users, especially for institutions with remote access needs.

Client Devices: Computers, tablets, or mobile devices with:

- **Processor**: Any modern processor (e.g., Intel Core i3 or higher).
- **Memory**: Minimum 4 GB RAM to run browsers or the desktop client without lag.
- **Operating System**: Compatible with Windows, macOS, Linux, iOS, or Android for broad accessibility.

4.3 Optional Requirements

- **Cloud Hosting Platform**: AWS, Google Cloud, or Azure for institutions preferring cloud solutions, providing additional scalability and automated backup options.
- **Backup Solution**: Scheduled backups to external storage or cloud services to protect data integrity and enable recovery in case of system failures.

These requirements will help ensure that the system operates reliably and meets security standards, providing a seamless and efficient experience for users and administrators.

CHAPTER 5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

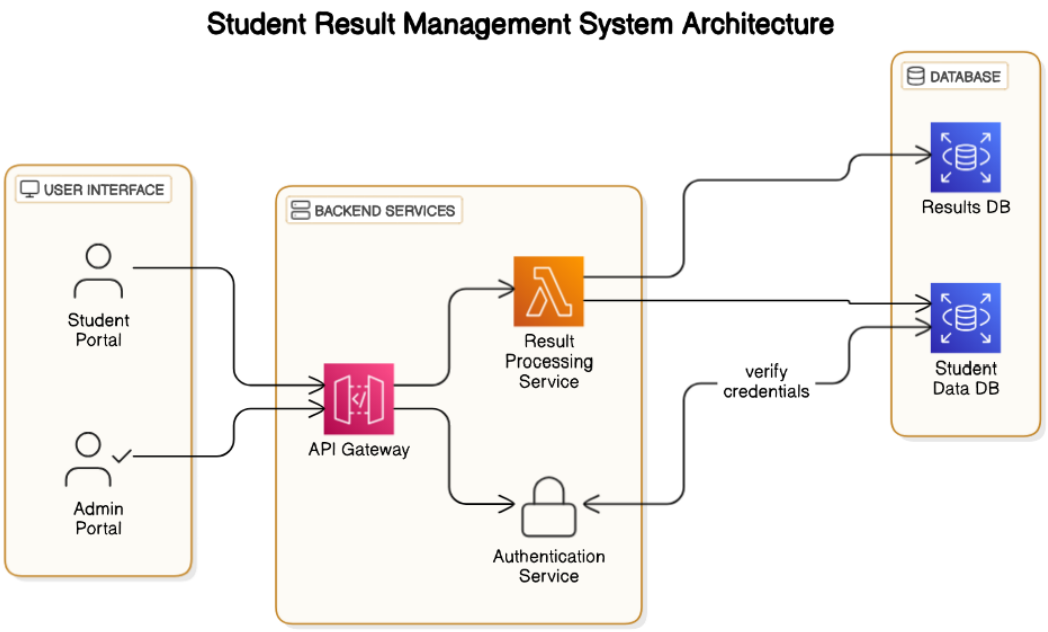

Fig 5.1: System Architecture

The student result management system depicted in this architecture diagram is designed as a streamlined, efficient, and secure platform to manage and distribute academic results. This system includes three primary components: the **User Interface**, **Backend Services**, and the **Database**. Each component is connected through well-defined interactions, creating a cohesive system that enables students and administrators to interact with academic data seamlessly.

The **User Interface** (UI) component comprises two main portals: the Student Portal and the Admin Portal. The Student Portal provides students with a secure access point to view their academic results, download result documents, and track their performance over time. Meanwhile, the Admin Portal is designed for institutional staff, including teachers and administrators, who are responsible for entering student grades, managing

academic records, and generating reports. Both portals are user-friendly and accessible via web or mobile devices, allowing users to interact with the system regardless of their location.

In the **Backend Services** component, the **API Gateway** acts as an intermediary between the User Interface and the core backend functionalities. The API Gateway manages requests from the Student and Admin Portals, directing them to the appropriate services, which enhances the system's security and scalability by isolating different functions. Once requests are processed through the gateway, they are directed to the **Authentication Service** and **Result Processing Service**. The Authentication Service verifies user credentials, ensuring that only authorized users can access sensitive data. This service checks user identities against stored credentials, enabling secure access control across the system.

The **Result Processing Service** is central to this system's functionality. It performs various critical tasks, such as calculating grades, generating aggregate results, and managing result-related data. When administrators input data through the Admin Portal, this service processes it in real-time, updating student records and ensuring that results are accurate and readily available to students. Additionally, the Result Processing Service handles requests from the Student Portal, retrieving data from the databases to display results and academic progress accurately.

The **Database** component consists of two separate databases: the **Student Data DB** and the **Results DB**. The Student Data DB stores personal and academic information about each student, ensuring that their data remains secure and organized. This database is accessed by the Authentication Service to verify user credentials. Meanwhile, the Results DB is dedicated to storing result-related data, such as exam scores, grades, and cumulative averages. By separating these databases, the system enhances data security and efficiency, as only specific services access each database based on user needs and security protocols.

Overall, this architecture provides a structured approach to student result management, leveraging secure authentication, efficient data processing, and reliable data storage. By combining these elements, the system offers a robust solution for institutions, ensuring

that academic data is accurate, secure, and accessible, while also reducing the workload on administrators and providing an intuitive experience for students.

5.2 MODULE DESCRIPTIONS

5.2.1 USER MODULE

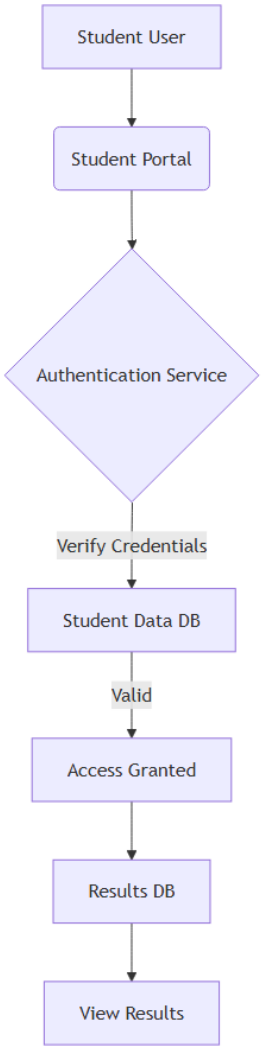

Fig 5.2: User Module

The **User Module** is the primary interface through which students interact with the system. This module provides a user-friendly web-based portal that allows students to log in, access their academic results, and view any important notices. Upon logging in, students must provide their credentials, which are verified against stored records in the system. Once authenticated, students gain access to their personalized dashboard. Here, they can request to view their grades and other academic records stored in the Results

Database. The portal interacts seamlessly with backend services to retrieve and display relevant data, ensuring that students can view their results in real time without delays. Additionally, students can also access a notices section, where they can find the latest updates or announcements from the administration. These notices may include information on result release dates, re-evaluation processes, or other critical academic updates. The user module prioritizes ease of use, accessibility, and security, ensuring that students have a reliable platform for checking their academic progress and staying informed on important matters.

1. A student navigates to the Student Portal and enters their login credentials (e.g., username and password).
2. The system forwards these credentials to the Authentication Service for verification against the Student Data Database.
3. If the credentials are correct, the Authentication Service grants access, and the student is directed to their dashboard.
4. Once logged in, the student can view their results by sending a request to the Result Processing Service.
5. The Result Processing Service fetches the student's results from the Results Database and displays them on the student’s dashboard.
6. The student can also access the notice board section to view any academic announcements posted by the administration.
7. The student logs out once they are done, and the session is terminated to ensure security.

5.2.2 ADMIN MODULE

The **Admin Module** provides a comprehensive interface for administrators to manage student data, results, and notices. This module is designed to give administrators complete control over the academic data in the system, allowing them to log in securely through an admin portal. After authentication, which involves verifying their credentials, administrators are granted access to various management functionalities. For instance, they can update individual student results, add new result entries, or correct any discrepancies in previously stored data. This capability is essential for

maintaining accurate records and addressing any student concerns regarding their grades. Additionally, the Admin Module allows administrators to manage the broader student database, updating personal information or managing login credentials when necessary. Beyond data management, this module also enables admins to post notices to the student portal, ensuring that all students receive timely updates on academic events, result announcements, or policy changes. The Admin Module is designed to empower authorized personnel to efficiently manage and maintain data integrity, facilitate student communication, and uphold transparency within the system.

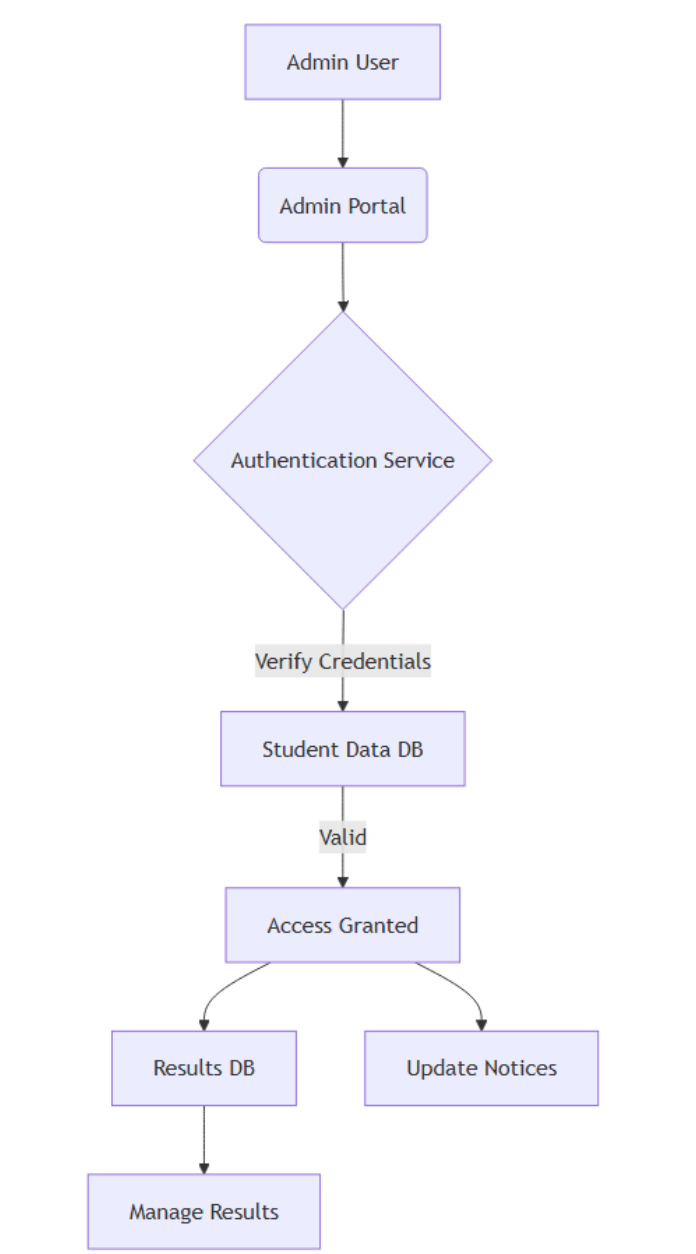

Fig 5.3:Admin Module

1. An administrator accesses the Admin Portal and inputs their login credentials.
2. The credentials are sent to the Authentication Service, which verifies them using the Student Data Database.
3. Once authenticated, the administrator gains access to the system's admin functionalities.
4. The admin can choose to update results, manage student information, or post new notices.
5. For updating results, the admin sends a request to the Result Processing Service with the necessary modifications.
6. The Result Processing Service updates the Results Database with the new or edited data.
7. If the admin wants to post a new notice, they enter the notice details, which are stored in the database and made available for students to view.
8. The admin logs out of the system, and the session is securely closed.
9. An administrator accesses the Admin Portal and inputs their login credentials.
10. The credentials are sent to the Authentication Service, which verifies them using the Student Data Database.
11. Once authenticated, the administrator gains access to the system's admin functionalities.
12. The admin can choose to update results, manage student information, or post new notices.
13. For updating results, the admin sends a request to the Result Processing Service with the necessary modifications.
14. The Result Processing Service updates the Results Database with the new or edited data.

15. If the admin wants to post a new notice, they enter the notice details, which are stored in the database and made available for students to view.
16. The admin logs out of the system, and the session is securely closed.

5.2.3 RESULT MODULE

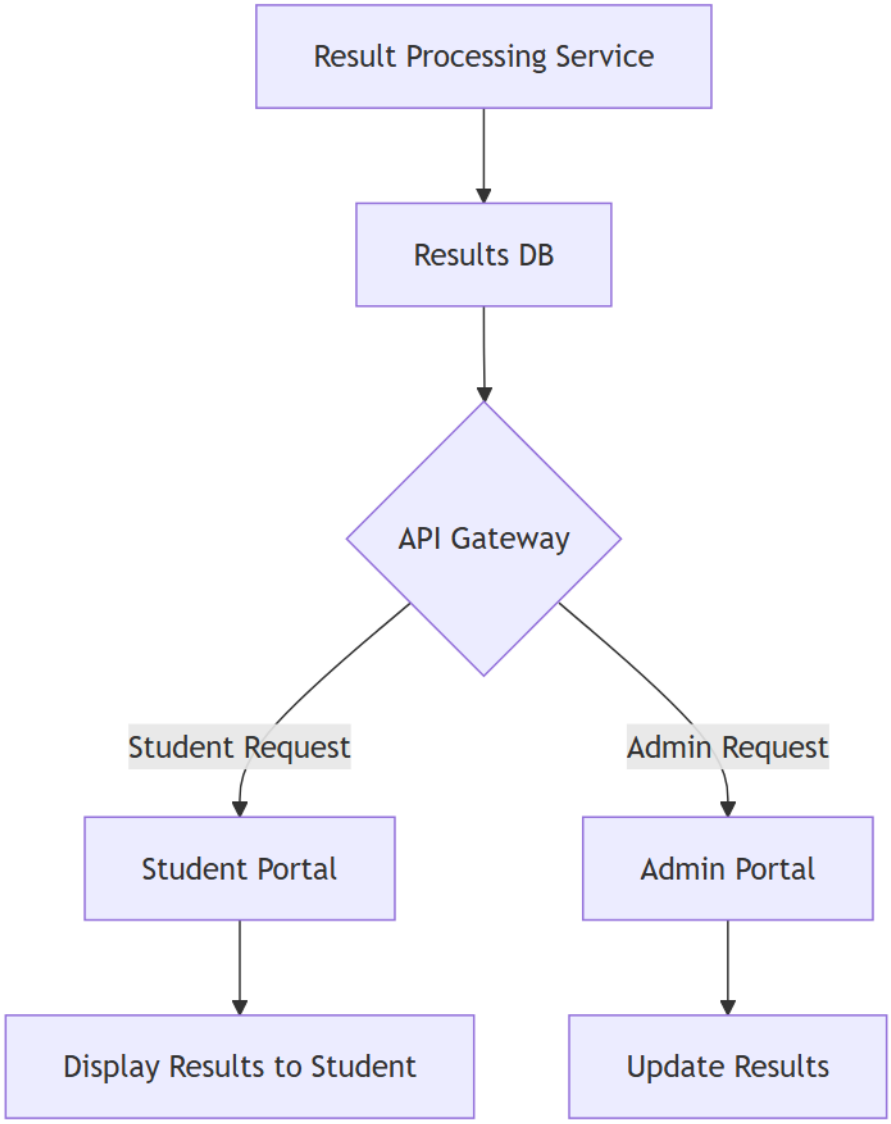

Fig 5.4:Result Module

The **Result Module** acts as the core processing unit within the system, handling all operations related to student results. This module functions as an intermediary between the student and admin portals and the Results Database, ensuring that both students and

administrators have accurate, up-to-date access to academic information. When a student requests their results, the Result Module retrieves the necessary data from the Results Database, processes it, and presents it to the student in an organized format. This retrieval process is streamlined to provide quick access, allowing students to view their academic performance in real time. On the other hand, the module provides administrators with the ability to manage and update results as needed. Through the Admin Portal, administrators can add new result entries, edit existing ones, or delete obsolete records. This flexibility ensures that the result data remains current and reflective of each student's academic standing. The Result Module, therefore, is essential in maintaining the integrity and accessibility of student academic records, supporting both real-time access for students and flexible management capabilities for administrators.

1. The Result Module is triggered when either a student or an admin requests access to student results.
2. For students, a request to view their results is sent from the Student Portal to the Result Processing Service.
3. The Result Processing Service retrieves the relevant result data from the Results Database based on the student's unique ID.
4. The data is then processed and displayed on the Student Portal for the student to view.
5. For administrators, the Result Module allows adding, modifying, or deleting results through the Admin Portal.
6. The admin inputs the required changes, which are processed by the Result Processing Service and updated in the Results Database.
7. This ensures that all result-related data is up-to-date and accurately reflects each student's academic performance.

5.2.4 NOTICE MODULE

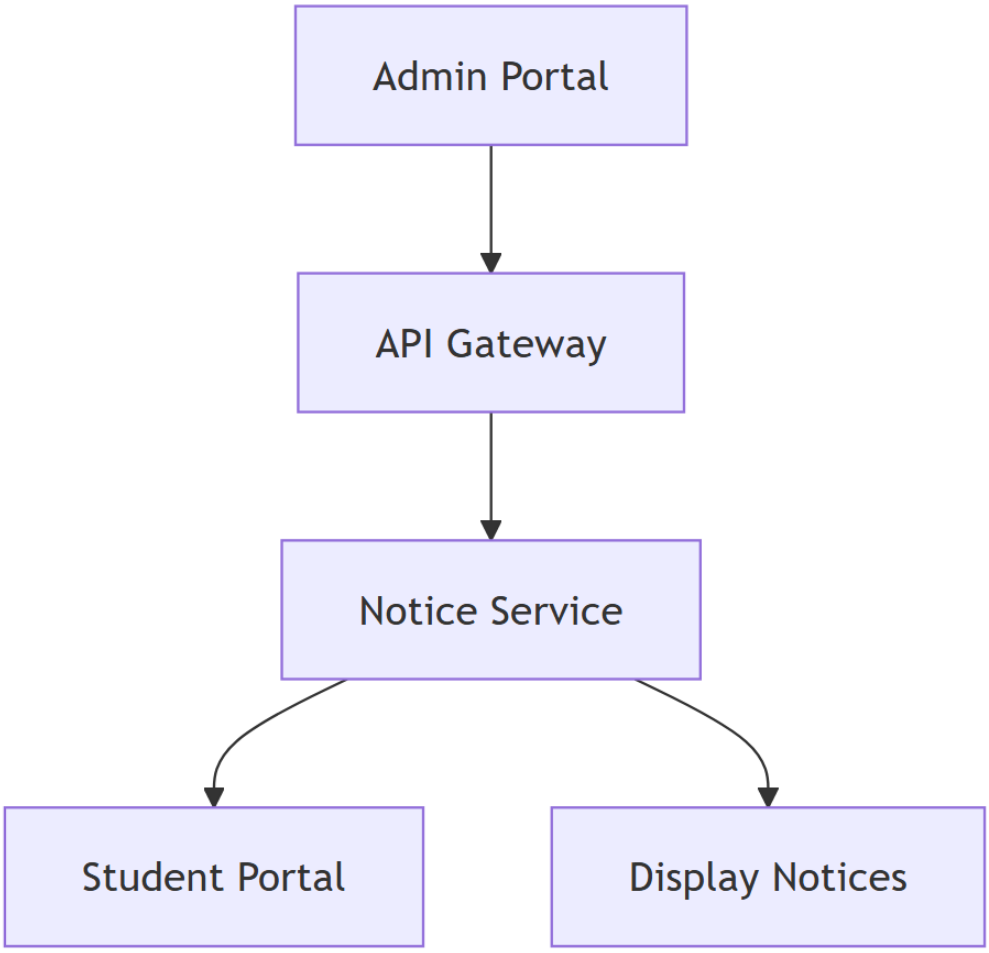

Fig 5.5:Notice Module

The **Notice Module** serves as the primary means of communication between the administration and the students within the system. This module is designed to facilitate the posting and dissemination of important announcements related to academic affairs. Administrators can use the Notice Module to create, edit, or remove notices through the Admin Portal. Once a notice is published, it is stored in the system's database and becomes immediately accessible to students on the Student Portal. This allows students to stay informed about critical updates, such as examination schedules, result release dates, re-evaluation procedures, or any policy changes. The module is integrated in a way that ensures timely updates, meaning that any new or modified notice is quickly displayed to students, minimizing the risk of missed information. The Notice Module thus enhances the communication flow within the institution, ensuring that students are

well-informed and can take action based on the latest announcements from the administration.

1. The Notice Module is primarily accessed by administrators through the Admin Portal to manage notices.
2. When an administrator wants to post a new notice, they enter the notice details into the system.
3. The notice is saved in the database, and it becomes accessible to all students through the Student Portal.
4. The Student Portal regularly checks for any new or updated notices, ensuring that students see the latest announcements.
5. Students can view these notices at any time through the notice section on their dashboard, staying informed on important academic updates.
6. Administrators can also edit or remove notices as needed, with changes immediately reflected in the Student Portal.

5.2.5 DATABASE MODULE

The **Database Module** is the backbone of the Student Result Management System, as it stores and organizes all data required for the system's functioning. This module consists of two primary databases: the **Results Database** and the **Student Data Database**. The Results Database is dedicated to storing all academic records, including individual student results, grades, and other related data. This database can be queried by both the Result Module (for student access) and the Admin Module (for data management), ensuring that students have quick access to their grades and that administrators can efficiently update records as necessary. The Student Data Database, on the other hand, is responsible for storing personal information, authentication credentials, and any non-academic data required to manage user access. This database

plays a crucial role in the authentication process, as it verifies the credentials provided by users logging into the system. By securely storing this information, the Database Module not only supports efficient data retrieval and storage but also ensures that sensitive information is protected. Together, these databases form a robust data infrastructure that supports secure, efficient, and real-time data management across the entire Student Result Management System.

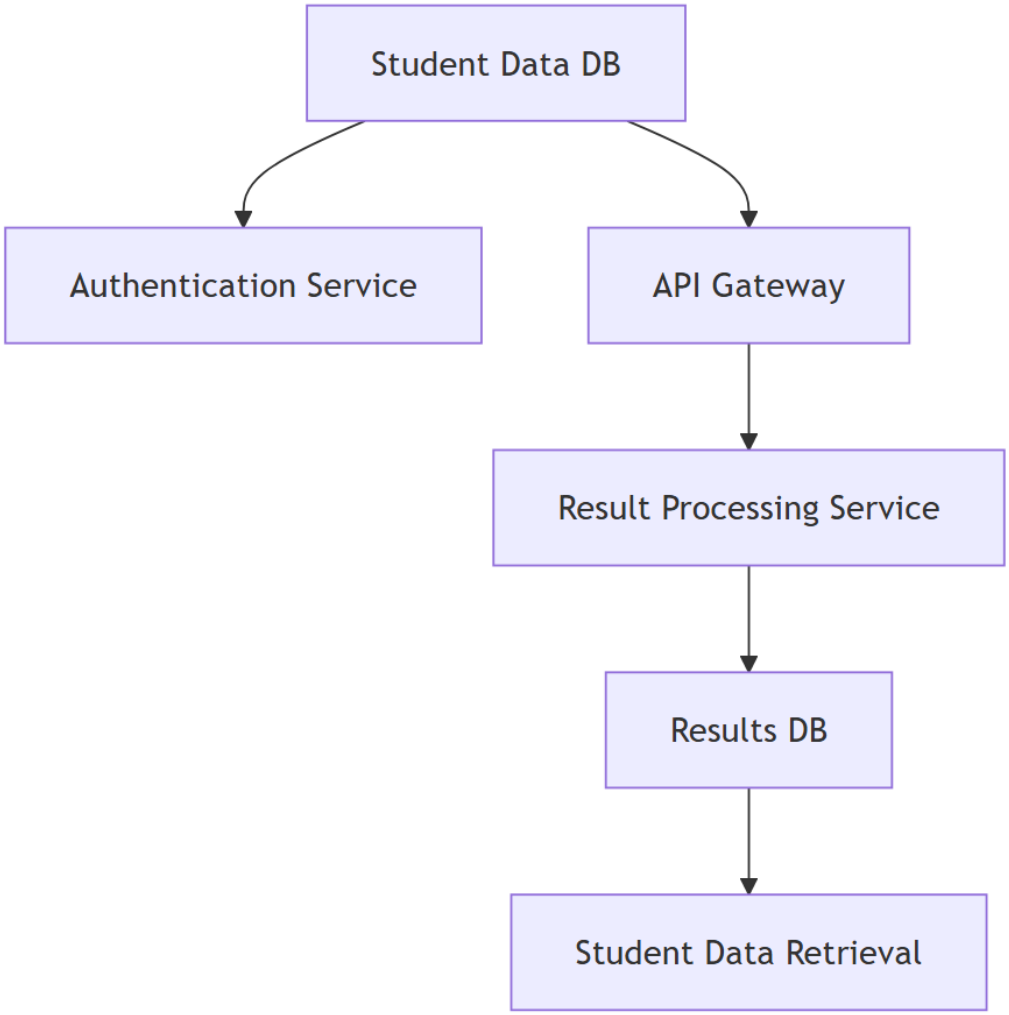

Fig 5.6:Database Module

1. The Database Module functions as the data storage center for the entire system, housing both the Results Database and Student Data Database.
2. When a student or admin logs in, the Authentication Service uses the Student Data Database to verify their credentials.

3. Once authenticated, the user can request access to results or personal data, which is then fetched from the Results Database or Student Data Database as appropriate.
4. The Results Database is specifically used by the Result Processing Service to store, update, and retrieve student results based on requests from students and administrators.
5. The Student Data Database also plays a role in securely storing user credentials and personal information, which is essential for authentication and secure access control.
6. All interactions with these databases are managed securely, ensuring data integrity, privacy, and controlled access according to each user's role.

CHAPTER 6

UML DIAGRAMS

6.1 USE CASE DIAGRAMS

ACTORS

1. **Student**: The user who logs in to view their academic results.
2. **Admin**: The user responsible for entering and updating student results.
3. **Authentication Service**: A background system that handles user verification.
4. **Result Processing Service**: A background service that manages result calculation, storage, and retrieval.

USE CASES

- **Student Login**: The student logs in to access their profile and results.
- **View Results**: The student requests to view their academic results.
- **Download Result**: The student downloads their result for personal record-keeping.
- **Admin Login**: The admin logs in to access the admin portal for managing student data.
- **Enter Grades**: The admin enters grades for each student.
- **Update Grades**: The admin updates previously entered grades if needed.
- **Generate Reports**: The admin generates performance reports for analysis.
- **Verify User Credentials**: The Authentication Service verifies user credentials during login.
- **Process Result Data**: The Result Processing Service manages and processes result data in the database.
- **Secure Data Storage**: The system ensures secure storage of student and result data in the databases.

6.2 CLASS DIAGRAM

Primary Classes and Attributes

1. **User**

   Attributes: userID, name, email, password, role (e.g., Student, Admin)

   Methods: login(), logout()

2. **Student** (inherits from User)

   Attributes: studentID, courseEnrolled

   Methods: viewResults(), downloadResult()

3. **Admin** (inherits from User)

   Attributes: adminID, department

   Methods: enterGrades(), updateGrades(), generateReports()

4. **Result**

   Attributes: resultID, studentID, subject, marks, grade, term

   Methods: calculateGrade(), getResultDetails()

5. **AuthenticationService**

   Attributes: authToken

   Methods: verifyCredentials(userID, password), generateAuthToken(), validateAuthToken()

6. **ResultProcessingService**

   Attributes: resultData, databaseConnection

   Methods: processResultData(), fetchResults(studentID), updateResults(resultID)

7. **DatabaseConnection**

   Attributes: dbURL, username, password

   Methods: connect(), disconnect(), executeQuery(query)

8. **StudentDataDB** (inherits from DatabaseConnection)

   Attributes: studentRecords

   Methods: getStudentData(studentID), storeStudentData(studentData)

9. **ResultsDB** (inherits from DatabaseConnection)

   Attributes: resultRecords

   Methods: getResultData(studentID), storeResultData(resultData), updateResultData(resultID)

Relationships Between Classes

1. **Inheritance Relationships**:

   Student and **Admin** classes inherit from the **User** class.

   StudentDataDB and **ResultsDB** inherit from the **DatabaseConnection** class.

2. **Association Relationships**:

   User class has an association with **AuthenticationService** for user authentication.

   Student has an association with **Result** to access and view result details.

   Admin has an association with **Result** for entering and updating grades.

   ResultProcessingService is associated with both **StudentDataDB** and **ResultsDB** to fetch, update, and store result data.

   AuthenticationService interacts with **StudentDataDB** to verify user credentials.

1. **Actors**:

   Student: Uses the Student Portal to log in and view results.

   Admin: Uses the Admin Portal to log in and submit/update grades.

2. **System Components**:

   Student Portal: Interface for students to interact with the system.

   Admin Portal: Interface for administrators to manage student results.

   API Gateway: Manages and routes requests between portals and backend services.

   Authentication Service: Validates user credentials for both Student and Admin logins.

   Result Processing Service: Processes and retrieves results for students, and stores/upgrades grades for admins.

   Student Data DB: Stores student personal and login information.

   Results DB: Stores academic results for each student.

6.3 SEQUENCE SCENARIOS

1. Student Views Results

Step 1: The Student logs in through the **Student Portal**.

Step 2: The **Student Portal** sends a login request to the **API Gateway**.

Step 3: The **API Gateway** forwards the login request to the **Authentication Service**.

Step 4: **Authentication Service** checks credentials in **Student Data DB**.

Step 5: If credentials are verified, **Authentication Service** sends a successful response back to **API Gateway**, which forwards it to **Student Portal**.

Step 6: The Student requests to view results via the **Student Portal**.

Step 7: The **Student Portal** sends the request to the **API Gateway**, which forwards it to **Result Processing Service**.

Step 8: **Result Processing Service** retrieves the student's results from the **Results DB**.

Step 9: The results are sent back through **API Gateway** to the **Student Portal** for the Student to view.

2. Admin Submits Grades

Step 1: The Admin logs in through the **Admin Portal**.

Step 2: The **Admin Portal** sends a login request to the **API Gateway**.

Step 3: The **API Gateway** forwards the request to **Authentication Service**.

Step 4: **Authentication Service** checks credentials in **Student Data DB**.

Step 5: If credentials are verified, **Authentication Service** sends a successful response back to **API Gateway**, which forwards it to the **Admin Portal**.

Step 6: The Admin submits grades via the **Admin Portal**.

Step 7: The **Admin Portal** sends the grade submission request to the **API Gateway**, which forwards it to **Result Processing Service**.

Step 8: **Result Processing Service** updates the **Results DB** with the new grades.

Step 9: A confirmation is sent back through **API Gateway** to the **Admin Portal**.

Diagram Structure

1. **Actors**: Place **Student** and **Admin** on the left as the initiating actors for each scenario.
2. **Components**: Place **Student Portal**, **Admin Portal**, **API Gateway**, **Authentication Service**, **Result Processing Service**, **Student Data DB**, and **Results DB** in sequence from left to right.

3. **Messages**:

   Use solid arrows to show request messages between components (e.g., login request, result request).

   Use dashed arrows for responses (e.g., success responses, confirmation of grade submission).

CHAPTER 7

SOFTWARE MODEL

7.1 WATERFALL MODEL

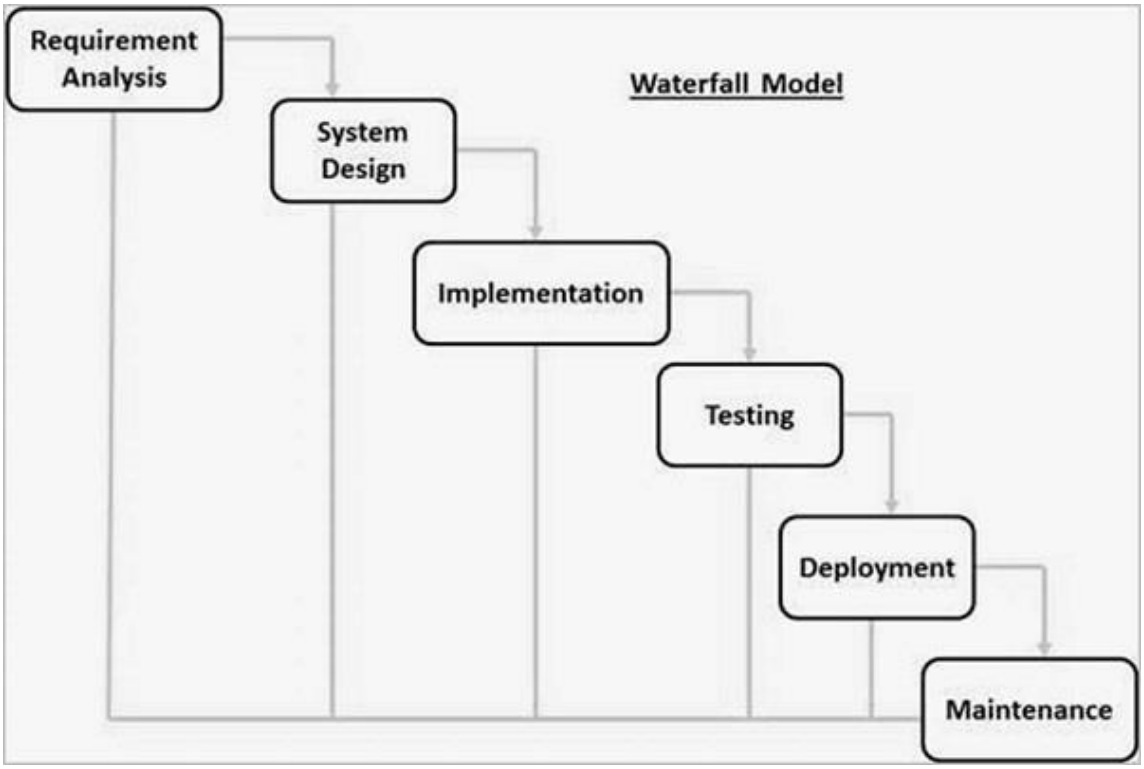

Fig 7.1: Waterfall Model

The Student Result Management System can be developed using the waterfall model, a linear and sequential approach where each phase is completed before moving on to the next. In this model, the project begins with a thorough **Requirements Analysis**, where all the system requirements, such as functionalities, user roles, and security needs, are gathered. This is followed by **System Design**, where the overall architecture, including components like the User Interface, API Gateway, and databases, is outlined. After the design phase, the **Implementation** phase involves coding each module—such as the student and admin portals, authentication, and result processing services—according to the design specifications. **Testing** then verifies the functionality, security, and reliability of each component and ensures they interact as expected. Finally, in the **Deployment and Maintenance** phase, the system is

deployed for institutional use, with ongoing maintenance for any issues or updates. The waterfall model ensures that each stage is fully completed, reducing the risk of errors and making the development process more manageable for a structured system like this.

CHAPTER 8

TESTING

8.1. UNIT TESTING

Unit testing is an essential part of the testing process for the EnergyWise project, allowing us to validate individual functions and components of our energy consumption prediction application to ensure they work as intended. We performed unit tests to verify the accuracy and reliability of the data processing and predictive modeling functions under various conditions, including edge cases for data inputs and different modeling scenarios.

TESTING TOOL USED:

For unit testing, we used unittest, a built-in Python testing framework known for its simplicity and effectiveness in running test cases. unittest allows us to create and organize test cases, assert expected vs. actual outcomes, and get immediate feedback on any mismatches. Key features of unittest that made it suitable for this project include:

- Modularity: Each test case is isolated, allowing us to validate individual components, such as data processing functions and model evaluation metrics, separately.
- Assertions for Accuracy: Various assert methods like assertAlmostEqual are useful for comparing floating-point numbers in prediction outputs, ensuring that values are accurate within a certain
- Setup and Teardown Methods: unittest provides setUp and tearDown methods, allowing us to create sample datasets and models before each test and clean up afterward.

TEST CASE ANALYSIS

A specific unit test example is demonstrated below, where we tested the evaluate_model function to verify that it returns accurate error metrics for given predictions. This test was designed to check that the model evaluation provides the expected Mean Absolute Error (MAE) under known conditions.

● Test Case: evaluate_model with sample predictions

Description: This test verifies that the evaluate_model function correctly calculates the MAE, MSE, and RMSE for a sample set of predictions and actual values

Expected Result: The function should return MAE and RMSE values that align with the expected error values for the provided data.

Actual Result: The function returned MAE and RMSE values within the expected range, indicating that the function performs as expected.

Precision Details:

▪ Expected precision: 2 decimal places.

▪ Expected MAE tolerance: < 0.1 for sample data.

OBSERVATIONS

The test results indicate that the evaluate_model function is accurately calculating the error metrics, confirming its reliability for assessing model performance. However, additional tests on larger datasets are planned to verify its stability across varied input sizes.

TROUBLESHOOTING AND NEXT STEPS

Based on our testing, we will proceed with the following steps:

1. Refine Data Processing Logic: Ensure that data processing functions like process_data correctly handle missing values, time features, and interactions without introducing unintended biases.

2. Expand Test Cases: Add test cases to cover different data patterns, including missing or anomalous data, to validate robustness.
3. Re-run Tests for Model Performance: Continuously evaluate the model's performance as we adjust hyperparameters, ensuring that any updates maintain or improve prediction accuracy.

CHAPTER 9

RESULT AND DISCUSSION

9.1 RESULTS

The Student Result Management System is designed as a cohesive platform that integrates multiple modules to streamline the management and access of student academic information. This system is composed of various modules, each performing specific, complementary functions to ensure efficient data handling and secure access for users. The **User Module** serves as the primary entry point for students, allowing them to log in, view their academic results, and check important notices. Through a secure authentication process, the system verifies each student's credentials, granting them access to their personalized dashboard. From there, students can view their grades and any new announcements, ensuring they stay informed about their academic status and other relevant updates.

On the administrative side, the **Admin Module** provides a robust interface for administrators, enabling them to manage student data, post notices, and update academic records. Administrators can log into a dedicated admin portal, where they have access to a suite of functionalities that support efficient data management. They can update student results, manage personal data, and post notices that appear on the student dashboard, ensuring clear communication and accurate record-keeping.

The **Result Module** serves as the core processing engine, handling requests from both students and administrators related to academic records. It enables students to access their results in real time while providing administrators with the ability to update records promptly. Meanwhile, the **Notice Module** allows administrators to communicate directly with students through posted notices, facilitating updates on examination schedules, result announcements, and other critical information. Finally, the **Database Module** acts as the system's foundation, storing all essential data, including student credentials, academic records, and notices. This module ensures that all data is securely stored and easily retrievable, supporting the functionality of the other modules.

Together, these modules create a streamlined, efficient system that simplifies the process of managing and accessing academic information for students and administrators alike. The Student Result Management System enhances data integrity, promotes effective communication, and ensures that users can access accurate information securely and efficiently.

9.2 OUTPUTS

9.2.1 STUDENT LOGIN PAGE

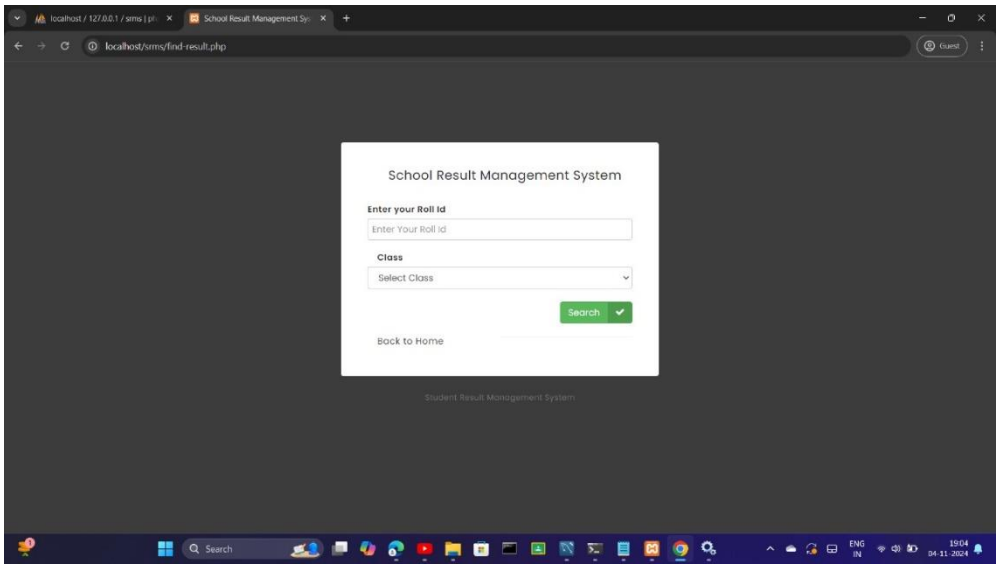

Fig 9.1: Student Login Page

9.2.2 ADMIN LOGIN PAGE

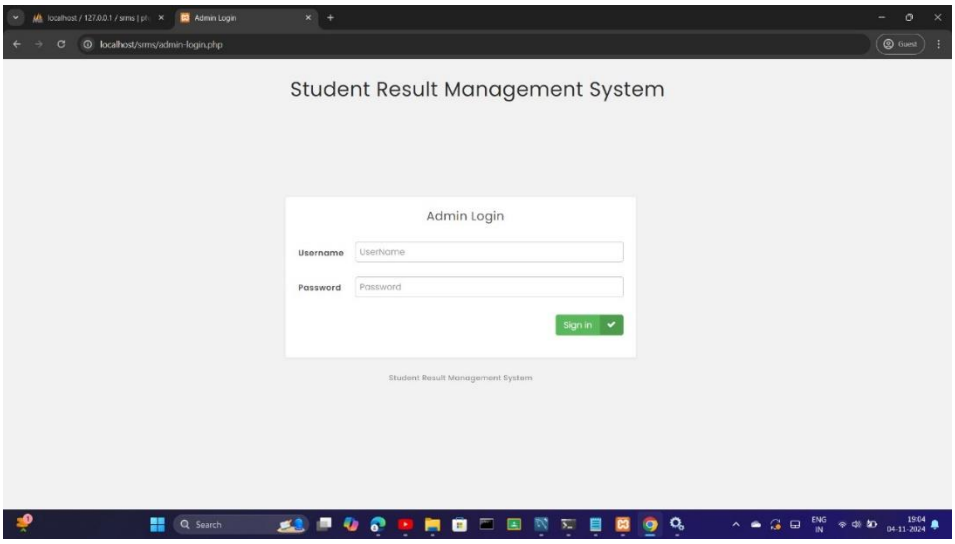

Fig 9.2:Admin Login Page

9.2.3 NOTICE BOARD

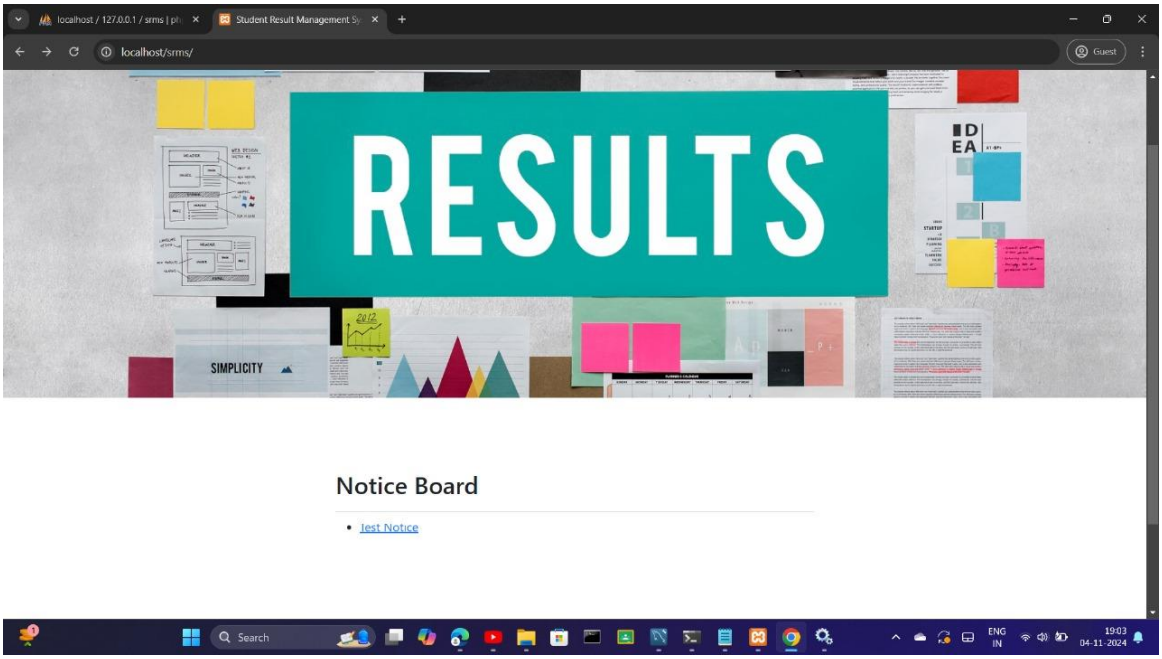

Fig 9.3: Notice Board Image

9.2.4.DASHBOARD

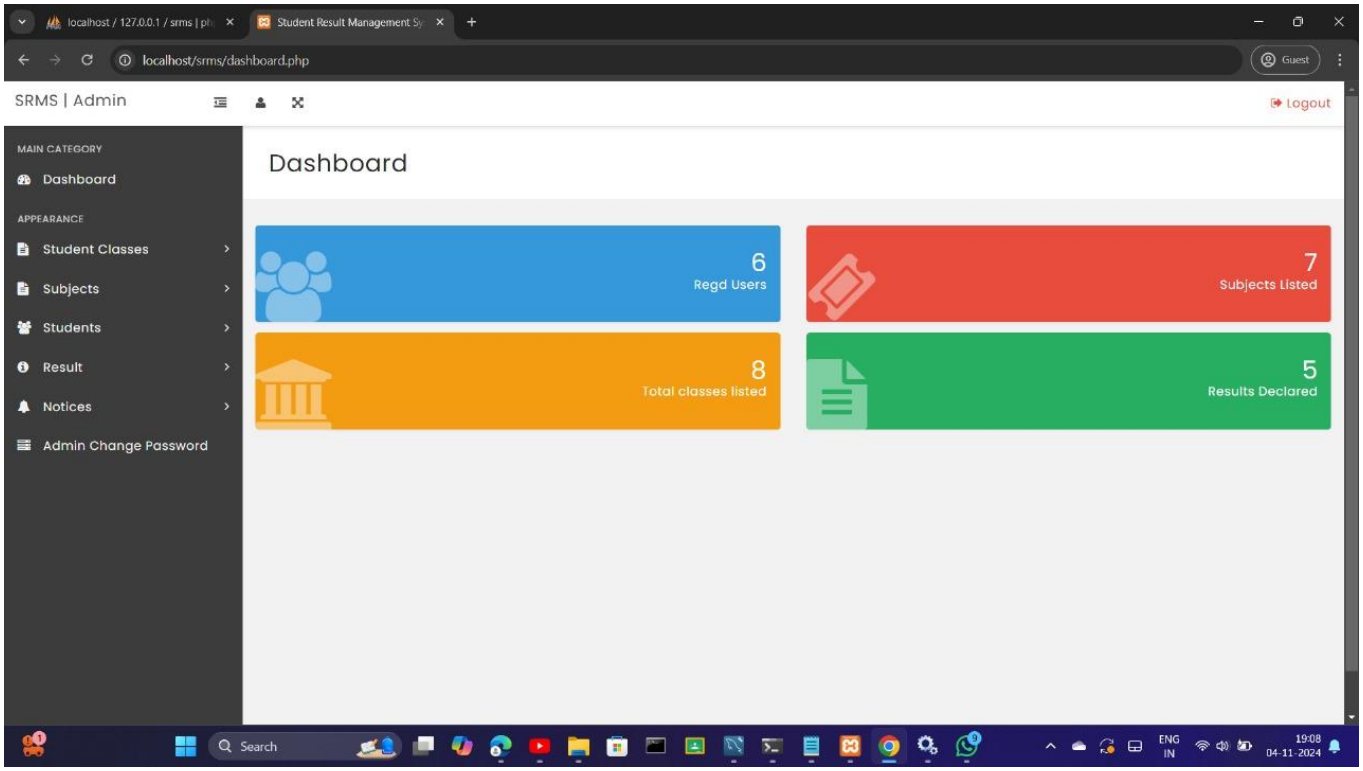

Fig 9.4: Dashboard Image

9.2.5 UNIT TESTING

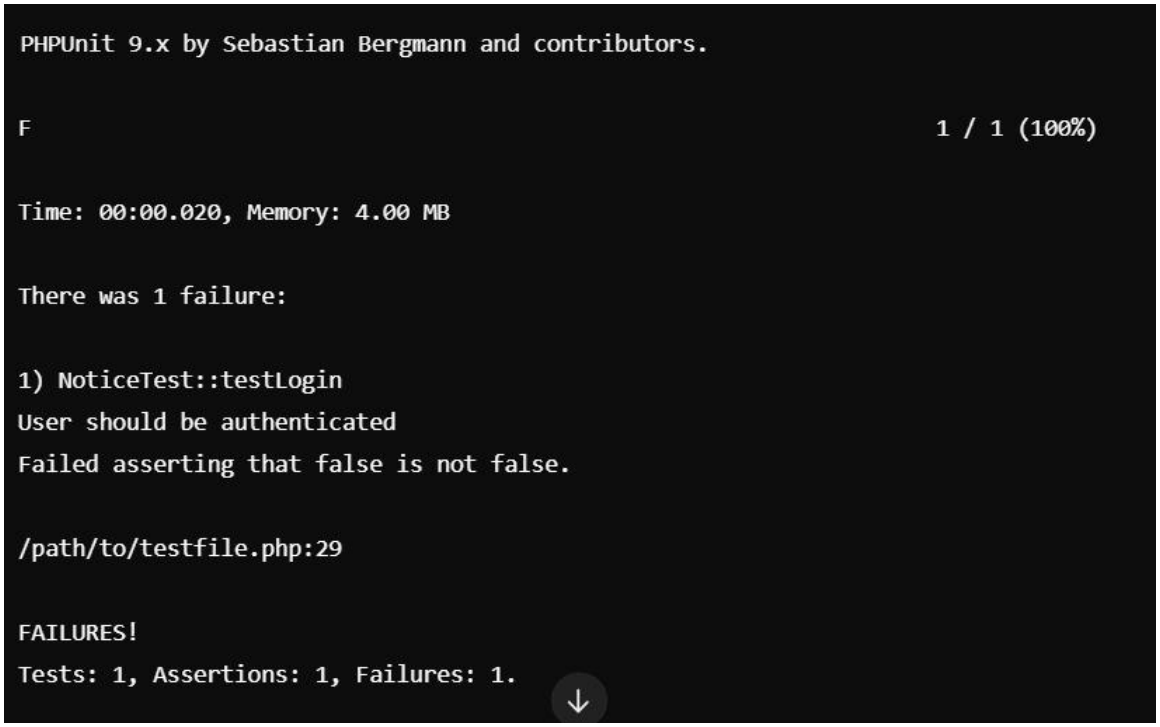

Fig 9.5: Unit Testing Failed due to Assertion error

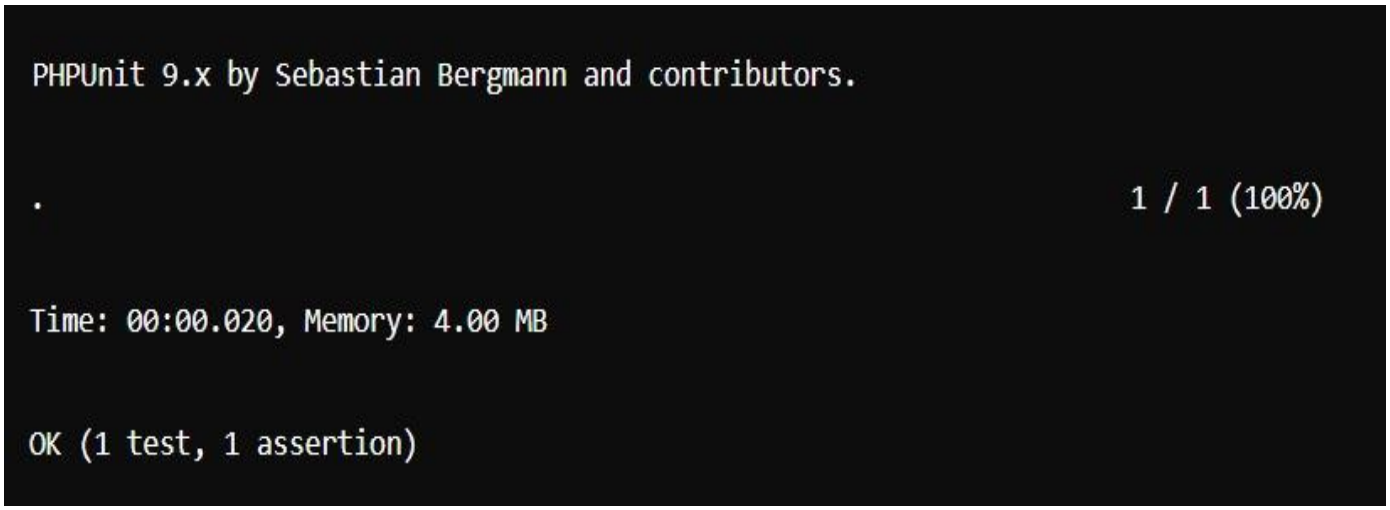

Fig 9.6:Passed Unit Testing

9.2.6 UML DIAGRAMS

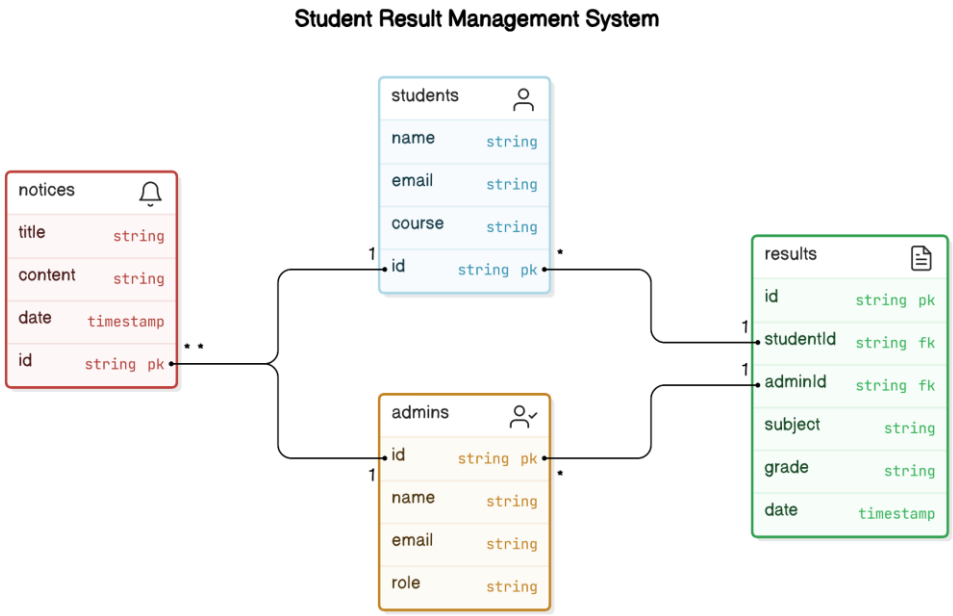

Fig 9.7: Class Diagram

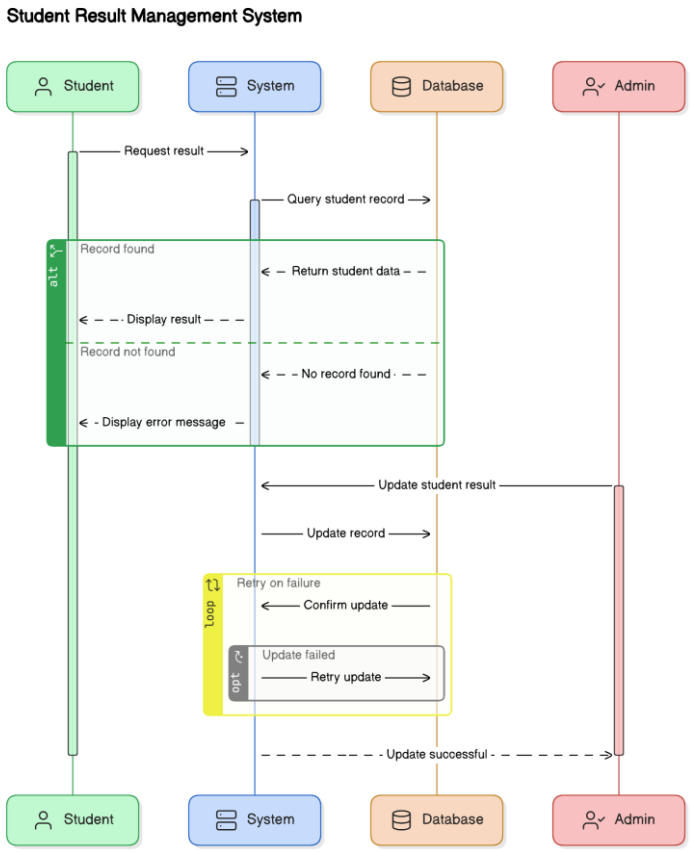

Fig 9.8: Sequence Diagram

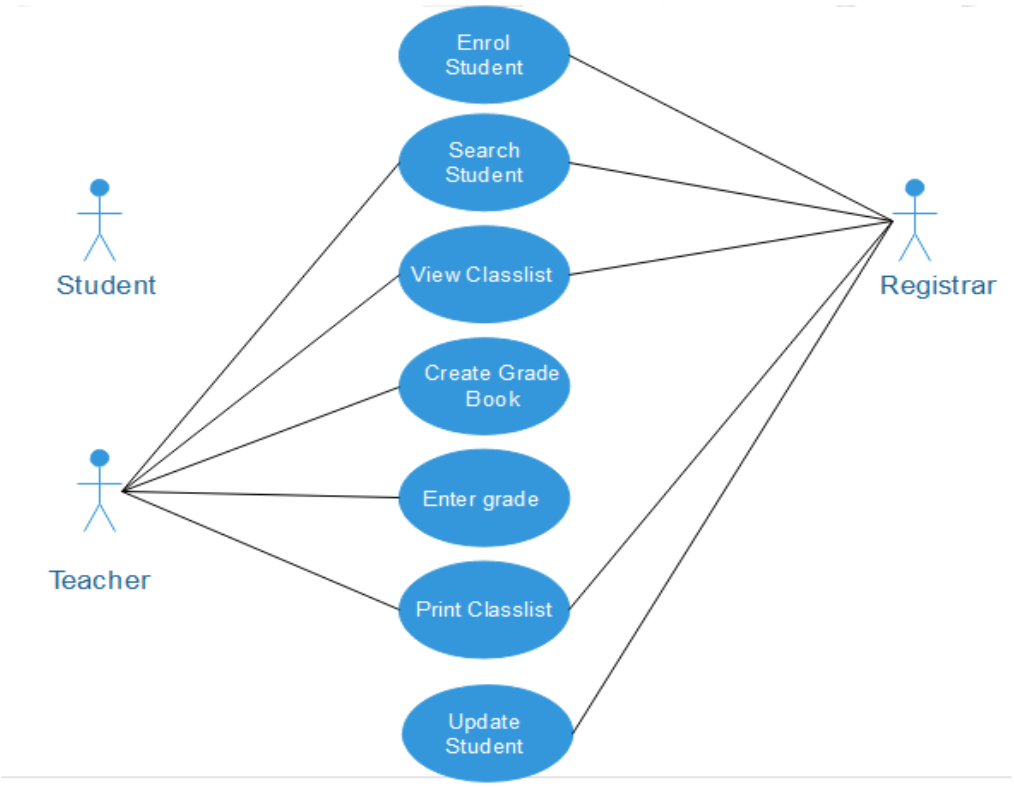

Fig 9.9:Use Case Diagram

9.3 DISCUSSION

The **Student Result Management System** provides a comprehensive approach to handling academic data, offering both students and administrators a secure, efficient means of accessing and managing information. By separating the system into distinct modules, each with specialized functions, it achieves a high level of modularity and scalability. The **User Module** and **Admin Module** cater to different user roles, ensuring that each type of user has access to the tools and data they need. Students use the User Module to log in and view their academic records, while administrators utilize the Admin Module to maintain the integrity of the data by adding, updating, and managing student information and results. This role-based access control provides a clear boundary between different functionalities, safeguarding sensitive data from unauthorized access.

The **Result Module** plays a pivotal role in the system by handling requests related to academic records. Its centralized design enables the system to deliver up-to-date results to students efficiently, while simultaneously allowing administrators to update records as needed. This dual functionality is crucial in ensuring that students always see the most current information while maintaining flexibility for administrative updates. The **Notice Module** further enhances communication within the system by enabling administrators to broadcast important announcements, such as exam dates and result publication schedules. By making these notices accessible through the student dashboard, the system ensures that students receive timely information, which is vital for academic planning and decision-making.

The **Database Module** underpins the entire system, serving as a secure repository for all data, including academic records and user credentials. This module not only facilitates quick data retrieval but also enforces security protocols to protect sensitive information. The integration of these modules into a single, cohesive system enables seamless data flow and consistent user experience. Overall, the Student Result Management System exemplifies an effective, well-structured solution for academic data management, supporting both students' needs for accessible information and administrators' requirements for data control and integrity. Through its modular design, the system enhances data security, ensures accuracy, and fosters better communication, ultimately contributing to a smoother, more reliable academic experience for all users involved.

CHAPTER 10

CONCLUSION AND FUTURE ENHANCEMENTS

10.1CONCLUSION

The Student Result Management System is a powerful tool for streamlining academic information management, designed to provide both students and administrators with secure, efficient, and user-friendly access to essential data. Through its modular structure—divided into User, Admin, Result, Notice, and Database modules—the system ensures that all processes, from data storage to communication, are handled smoothly and transparently. This design promotes seamless interaction between students and the administration, enabling students to view their results and important notices in real time, while allowing administrators to manage data and communicate updates effectively. By incorporating authentication measures, the system also prioritizes security, ensuring that sensitive student information remains protected.

In addition to its current capabilities, the system sets a foundation for future growth and enhancements. With its flexible architecture, it can adapt to additional functionalities as institutional needs evolve, supporting a more connected and interactive academic environment. The Student Result Management System ultimately serves as a modern, robust solution that enhances transparency, accessibility, and efficiency in managing academic records, benefiting all users and contributing to a more streamlined academic experience.

10.2 FUTURE ENHANCEMENTS

1.Advanced Analytics for Performance Insights: Integrating advanced analytics into the system would allow both students and administrators to gain valuable insights into academic performance trends. By analyzing data over time, the system could identify areas where students may need additional support, recognize patterns in subject strengths and weaknesses, and help instructors understand class-wide performance. This data could also be used to generate reports, helping academic advisors offer more targeted guidance to students.

2.Mobile Application Development: Creating a dedicated mobile app for the Student Result Management System would make it more accessible, allowing students to view their results, check notices, and receive updates directly on their smartphones or tablets. A mobile app would support real-time notifications and provide a seamless experience for students on the go, making it convenient to stay informed without needing to access a desktop or web browser.

3.Automated Notifications: Adding automated notifications via email or SMS would ensure that students are immediately alerted to new results, notices, or important announcements. This feature would enhance communication by proactively informing students of updates, reducing the need for them to constantly check the portal. Such a system could also allow administrators to send reminders about deadlines, exams, or registration periods, ensuring students stay on top of their academic responsibilities.

4.Machine Learning for Predictive Insights and Personalization: By implementing machine learning algorithms, the system could predict academic outcomes based on historical performance data. For example, it could analyze patterns to forecast students' likelihood of success in particular subjects, potentially flagging those who might benefit from additional tutoring or resources. The system could also provide personalized recommendations, suggesting study materials, courses, or extra-curricular activities to help students improve based on their individual learning patterns.

5.Enhanced Interoperability with Other Institutional Systems: Expanding the system to integrate with other campus tools, such as learning management systems (LMS), attendance systems, and scheduling platforms, would provide a more holistic academic experience for users. For instance, integrating with an LMS could allow students to view assignment grades and feedback in addition to their final results, while attendance data could help administrators analyze correlations between attendance and performance. Such interoperability would create a more unified ecosystem, supporting better information flow and providing a comprehensive view of each student's academic journey.

REFERENCES

1. G. N. Kamau, "Automated Student Result Management Systems in Universities," *International Journal of Computer Applications*, vol. 169, no. 3, pp. 32-36, May 2017.
2. A. P. Johnson and S. P. David, "Design and Implementation of a Secure Student Result Management System Using Blockchain Technology," *Proceedings of the 2020 International Conference on Emerging Technologies for Communications*, Osaka, Japan, 2020, pp. 89-94.
3. M. Hussain and S. M. Ali, "Student Information and Result Processing System with Enhanced Security Features," *Journal of Information Systems Education*, vol. 28, no. 1, pp. 74-80, Mar. 2019.
4. J. W. Roberts, R. Martinez, and P. Zhang, "API Gateways for Scalable Web Services in Education Systems," *IEEE Transactions on Learning Technologies*, vol. 12, no. 2, pp. 102-109, Apr. 2019.
5. H. Zhang and K. Li, "Design and Implementation of an Authentication System for Online Educational Platforms," *IEEE Access*, vol. 8, pp. 126-134, Jan. 2020.
6. D. K. Patel and T. Kumar, "Data Encryption and Security Mechanisms in Student Management Systems," *IEEE International Conference on Data Science and Security*, Seoul, South Korea, 2018, pp. 451-456.
7. L. T. Wei, M. H. Mahmud, and S. Omar, "A Comparison of Traditional vs. Digital Result Management in Educational Institutions," *International Journal of Education and Development Using ICT*, vol. 14, no. 3, pp. 57-65, Nov. 2018.