|  |  |
| --- | --- |
|  |  |
| **High School Management System** |  |
|  |  |
|  | ***Documentation***  *Submitted to Mr. Megersa Daraje*  ***FSE Project*** |

Table of Contents

* **Chapter One:**

1.1 Introduction ………………………………………………………………….1

1.2 Background of the Project …………………………………………..………2  
1.3 Statement of the problem ………………………………………………..….2  
1.4 Justification of the project …………………………………………………..3  
1.5 Objective of the project ………………………………………………….….4

1.5.1 General objective …………………………………………………4  
 1.5.2 Specific objective …………………………………………………4  
1.6 Scope and Limitation………………………………………………………..4

1.6.1 Scope of the study…………………………………………………4  
 1.6.2 Limitation of the Project…………………………………………..5  
1.7 Feasibility Study…………………………………………………………….5

1.7.1 Technical Feasibility………………………………………………5  
 1.7.2 Operational Feasibility………………………………………….…5   
 1.7.3 Economical Feasibility…………………………………………….5   
1.8 Significance of the Project……………………………………………..……6

1.9 Beneficiaries of the project………………………………………….7  
1.10 Methodology……………………………………………………………….9  
1.11 Development tools………………………………………………………...11

1.11.1 Hardware and Software tools……………………………………11

1.11.1.1 Software tools…………………………………............11  
 1.11.1.2 Hardware tools………………………………………..12

1.12 Required Resource………………………………………………………..12   
 1.13 Schedule…………………………………………………………………..13  
 1.14 Team Composition………………………………………………………..15

**Chapter Two:** Description of the Existing System

2.1 Major Function of the Existing System…………………………………….18  
 2.2 Users of current system…………………………………………………….18  
 2.3 Drawback of current system………………………………………………..19  
 2.4 Business rule……………………………………………………………..…21

* **Chapter Three:** Proposed System

3.1 Overview…………………………………………………………………23  
3.2 Functional Requirement………………………………………………….23  
3.3 Non-Functional Requirement…………………………………………….24

3.3.1 User Interface…………………………………………………...24  
 3.3.2 Performance Consideration……………………………………..25  
 3.3.3 Reliability……………………………………………………….25  
 3.3.4 Error Handling………………………………………………….25 3.3.5 Security…………………………………………………………25  
3.4 System Model

3.4.1 Scenarios………………………………………………………..25  
 3.4.2 Use Case Model………………………………………………...27  
3.5 Object Model

3.5.1 Data Dictionary. ………………………………………………..28 3.5.2 Class Diagram…………………………………………………..29  
3.6 Dynamic Model  
 3.6.1 Sequence Diagram………………………………………………31  
 3.6.2 Activity Diagram………………………………………………..37 3.6.3 State Chart Diagram…………………………………………….41

* **Chapter 4:** System Design

4.1 Overview………………………………………………………………….43  
4.2 Proposed System Architecture

4.2.2 Subsystem Decomposition……………………………………....45  
 4.2.3 Hardware and Software Mapping……………………………….45  
 4.2.4 Persistent Data Management……………………………………46  
 4.2.5 Component Diagram……………………………………………46 4.2.6 Database Design………………………………………………...48 4.2.7 Access Control…………………………………………………..49  
 4.2.8 User interface Design……………………………………………49

4.2.9 Boundary Conditions………………………………..…………..51

* **Chapter Five:** Implementation

5.1 Overview…………..……………………………………………………….54   
 5.2 Coding standard…….……………………………………………………...55  
 5.3 Development Tools….……………………………………………………..55  
 5.4 Prototype……………………………………………………………………55 5.5 Implementation Detail………………………………………………………56

* **Chapter six:** System Testing

6.1 Introduction…………………………………………………………………57 6.2 Scope…………………………….………………………………………….57 6.3 Resources…………………………...………………………………………57  
 6.4 Schedule……………………………...……………………………………..57  
 6.5 Test case scenario…………………….…………………………………….58  
 6.6 Pass/fail criteria……………………………………………………………..60  
 6.9 Test case specification………………………………………………………61  
 6.9 Estimated risk and con-tangency plan………………………………………62

**References** …………………………………………………...…………………………………63

**High School Management System**

**Chapter One**

**1.1 Introduction**

Education, a cornerstone of society, plays a pivotal role in shaping the future of individuals and, consequently, the nation. In the face of technological advancements, it has become crucial to integrate these developments into the educational system to enhance efficiency and effectiveness.

This document introduces the High School Management System project, designed to streamline the operations of high schools. Developed using the Scrum methodology, this agile framework fosters iterative progress and rapid delivery.

In Ethiopia, high schools are integral in guiding the educational journey of students. Recognizing their significance, we initiated the High School Management System project. This endeavor aims to simplify the operational aspects of these institutions by developing a digital platform that proficiently manages student information, schedules, academic records, and communication channels.

Our goal is to establish a user-friendly system that caters to all stakeholders - teachers, students, parents, and administrative staff. This platform centralizes essential school functionalities, optimizing administrative tasks and fostering improved communication and organization across various school departments. The objective is to enhance overall efficiency and facilitate a seamless educational experience for all parties involved.

The project incorporates both functional and non-functional requirements to create a robust system. We are committed to ensuring secure data management, high performance, scalability, usability, and reliability. The system will be web-based, with various components including a detailed plan for construction, a clear structure for information storage, different interfaces for different users, communication channels, and security measures.

The project adheres to the Scrum methodology, an Agile-based development approach. This approach facilitates iterative and incremental development, continuous feedback, and adaptations, ensuring the delivery of a system that meets the evolving needs of the school stakeholders.

Our process model provides a comprehensive guide, covering project initiation, sprint planning, development sprints, testing, documentation, deployment, post-deployment activities, and ongoing maintenance. It emphasizes the importance of iterative development, regular feedback collection, testing, and continuous improvement to ensure the system’s quality, reliability, and adaptability.

Ultimately, our aim is to deliver a high school management system that effectively meets the specific requirements and objectives set forth, while adhering to industry best practices and standards. This project is not merely about integrating technology into education; it is about revolutionizing the way high schools operate, leading to improved efficiency, reduced workload, and enhanced educational outcomes.

In the subsequent sections, we will delve deeper into the specifics of the project, including its background, objectives, feasibility, significance, beneficiaries, and the resources required for its implementation. This introduction sets the stage for an exciting journey into the realm of digital transformation in education.

**1.2 Background**

In the Ethiopian education scenario, high schools have traditionally managed their operations manually, using paper records and basic communication methods. This approach often led to challenges in maintaining student information, organizing schedules, and facilitating effective communication among teachers, students, and parents. The existing systems lacked integration and efficiency, making it difficult to manage the growing needs of a modern educational setup.

Over the years, as Ethiopia aims for educational advancement, the need for a more direct and digitized management system has become evident. Modernizing administrative tasks within high schools has become crucial to keep pace with evolving educational standards and to ensure a more effective learning environment. The implementation of technology in education has been recognized as a key step towards enhancing efficiency, accuracy, and communication within high school administrative processes.

In this context, the introduction of High School Management Systems is seen as a significant step forward. These systems are designed to alleviate the limitations of manual processes, offering a centralized and organized platform to manage student records, class schedules, and academic progress. By transitioning from conventional methods to a digital framework, there's an opportunity to improve the overall management and communication within Ethiopian high schools, aligning with the nation's aspirations for a more efficient and effective educational system.

**1.3 Statement of the problem**

In the context of high schools in Ethiopia, the current administrative processes predominantly rely on manual and paper-based systems. This traditional approach poses several challenges, including inefficient record-keeping, disorganized scheduling, and limited communication avenues among stakeholders, such as teachers, students, and parents. The lack of a centralized and integrated system leads to difficulties in managing student information, academic progress, and class schedules effectively. Additionally, the absence of a digital platform slows down the optimization of administrative tasks and inhibits the potential for streamlined communication within the educational institution.

Furthermore, as education changes and tries new ways of teaching, absence of a digital system is a big problem. Using old methods stops schools from keeping up with new ways of learning and adapting quickly to what's needed. So, there's a real need for a new High School Management System that keeps everything in one place, makes schedules better, improves how everyone talks, and makes things easier for everyone involved in the school.

As we know in Ethiopia, Grade 12 students must be well prepared for their Entrance exams. However, the current lack of an efficient high school management system poses an added obstacle for these students. Without a centralized system to organize previous exam sheets, and take mock models using PC’s, it will be difficult for them to adapt to the environment of the exams. This absence of a comprehensive system affects their academic journey, slowing their ability to access vital resources and support required for excelling in these critical exams. This highlights the need for a robust High School Management System to assist and support Grade 12 students preparing for university entrance.

**1.4 Justification**

Implementing a High School Management System serves as a crucial step towards modernizing the educational infrastructure in Ethiopia. The project's justification lies in its potential to reduce long-standing administrative challenges that are frequent in high schools across the country. By introducing a digital platform, the project aims to address the existing inefficiencies caused from manual record-keeping, disorganized scheduling, and limited communication among stakeholders, including teachers, students, and parents. This initiative aligns with the nation's commitment to advancing educational methodologies, fostering an environment that embraces technological innovations for enhanced efficiency and accessibility within the education sector.

In addition, the High School Management System responds to the evolving needs of a competitive academic landscape, preparing students for higher education. The system's implementation intends to streamline administrative processes, optimizing resource utilization, and enabling comprehensive student data management. By facilitating a centralized platform, the project fosters better communication between educators, students, and parents, enhancing academic performance, improving student support systems, and increasing the overall effectiveness of high school administration. This project's justification is rooted in the transformative potential it holds to uplift the quality and effectiveness of the educational experience while aligning with global advancements in educational technology.

**1.5 Objective of the project**

**1.5.1 General Objectives**

The main aim of the project is to develop and implement a user-friendly High School Management System that efficiently automates administrative processes within Ethiopian high schools. This system seeks to efficiently manage data, improve communication channels, and enhance overall operational efficiency without requiring specialized technical expertise from its users.

**1.5.2 Specific Objectives**

The specific objectives include designing an intuitive digital platform capable of managing student information, schedules, and academic records. Additionally, the system aims to facilitate easy access to essential functionalities for teachers, students, and parents, fostering improved communication and resource allocation within the educational institution. This specific objective also encompasses the development of a secure and accessible system that caters to the diverse needs of high school stakeholders while promoting a more organized and efficient educational environment.

    1, To automate student record management, enabling easy access, update, and organization of student information.

   2, To create an integrated scheduling system that efficiently manages class timetables and assigns teachers based on subject expertise and grade levels.

   3, To establish a robust communication module facilitating effective interactions among teachers, students, and parents.

  4, To implement a secure user authentication mechanism ensuring data privacy and confidentiality.

  5, To develop a user-centric interface, promoting ease of use and accessibility for all stakeholders involved in high school administration.

These objectives collectively aim to propel high school administration into the digital era, fostering an environment of enhanced efficiency, transparency, and collaborative engagement among all stakeholders in the educational process.

**1.6 Scope and Limitation**

**1.6.1 Scope of the study**

The scope of this project encompasses **the development and implementation of a digital High School Management System tailored to Ethiopian high schools’** xspecific needs. It includes functionalities for student data management, class scheduling, teacher assignments, attendance tracking, and a communication platform. The system's scope extends to facilitating smooth administrative operations and fostering improved communication between teachers, students, and parents within the high school ecosystem.

**1.6.2 Limitation of the Project**

One limitation of this project lies in the initial implementation focusing on core functionalities, potentially lacking advanced features that might evolve with future requirements.

*Time* constraints could also limit the depth of integration and extensive testing of the system.

Additionally, dependencies on available resources and infrastructure might influence the project's scale and depth of implementation within high schools.

These limitations will be considered in the project's development to ensure a balanced approach and efficient delivery of the proposed system.

**1.7 Feasibility Study**

**1.7.1 Technical Feasibility**

The technical feasibility of this project revolves around the availability of resources, expertise, and technology required for system development. Utilizing prevalent web development technologies, frameworks, and database systems, the project aims for compatibility and accessibility across various devices and network environments. By leveraging common programming languages and frameworks, alongside accessible web hosting services, the technical aspects of this system development are within reach and align with the prevalent technology landscape.

**1.7.2 Operational Feasibility**

The operational feasibility of implementing a High School Management System considers the adaptability and ease of integrating the system into the existing operational environment of Ethiopian high schools. This involves assessing the potential resistance to change, training requirements, and acceptance of the proposed system by end-users. Strategies for user training and change management will be integral to ensuring smooth adoption and minimizing disruptions during the system's implementation and subsequent operation within the school environment.

**1.7.3 Economic Feasibility**

The economic feasibility analysis of the proposed high school management system underscores its financial viability, evaluating development, implementation, and maintenance costs against anticipated benefits. A meticulous cost-benefit analysis affirms the system's cost-effectiveness, encompassing expenses for development, training, and infrastructure. Ongoing operational costs cover maintenance, support services, and training updates. The system is expected to yield substantial cost savings and revenue through enhanced efficiency, with a positive return on investment (ROI). Despite higher deployment and training expenses, the system's low development cost ensures long-term benefits, making it an unquestionably economically viable and cost-effective solution.

When we consider the users, which include students, families, and school staff, they won’t require much additional hardware to use the system beyond their existing devices (like phones, tablets, or personal computers). Therefore, the cost required by the user is low, further affirming that the system is economically feasible.

**1.8 Significance of the Project**

The significance of the high school management system project is multifaceted and holds paramount importance within the educational context. Firstly, it addresses the pressing need for efficient and streamlined administrative processes in high schools, aiming to automate routine tasks such as attendance tracking, grade management, and scheduling. This automation not only reduces the burden on administrative staff but also enhances accuracy and eliminates manual errors.

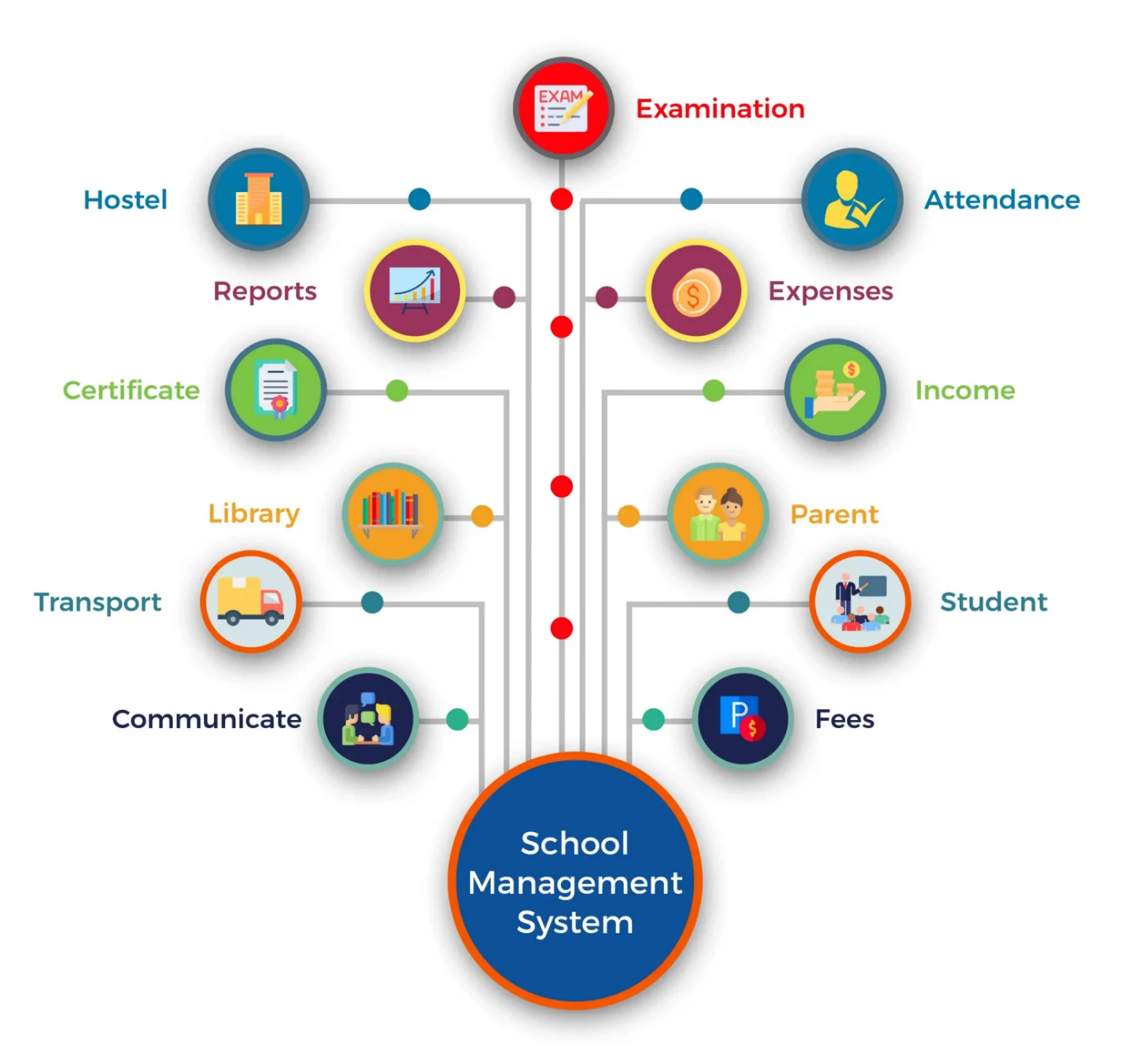
Secondly, the project contributes to improved communication and collaboration among stakeholders, including students, teachers, and parents. The implementation of the system allows for real-time access to academic information, updates, and announcements, fostering a transparent and collaborative educational environment.

 Furthermore, the high school management system enhances data management and reporting capabilities, providing valuable insights into student performance, attendance trends, and resource utilization. This data-driven approach empowers educators and administrators to make informed decisions, implement targeted interventions, and continually enhance the overall educational experience.

The project's significance extends to resource optimization, where efficient allocation of classrooms, teaching staff, and other resources ensures optimal utilization and, consequently, an improved learning environment. Additionally, the system facilitates effective communication with parents, keeping them informed about their child's academic progress, attendance, and school activities.

In summary, the high school management system project is of paramount significance as it addresses administrative inefficiencies, promotes transparency and collaboration, enables data-driven decision-making, and optimizes resource utilization. Ultimately, its successful implementation is poised to positively impact the overall educational experience for students, teachers, and parents alike.

**1.9** **Beneficiaries of the Project**



**Figure: 1 Beneficiaries of the project and the uses they get.**

The implementation of the high school management system is poised to bring about transformative benefits for various stakeholders within the educational ecosystem. The primary beneficiaries include:

**1. Administrative Staff:**

-Streamlined processes and automated tasks reduce the workload, allowing administrative staff to focus on more strategic aspects of school management.

-Enhanced data management capabilities facilitate quicker decision-making and improved efficiency.

**2. Teachers:**

-Access to real-time student data and automated grading systems simplify administrative tasks, enabling teachers to dedicate more time to instructional activities.

-Improved communication tools foster better collaboration among teaching staff and facilitate parent-teacher communication.

**3. Students:**

-Enhanced learning experiences through optimized resource allocation, improved scheduling, and efficient classroom management.

-Increased transparency into academic progress and streamlined communication contribute to a more informed and engaged student body.

**4. Parents/Guardians:**

-Access to real-time updates on their child's academic performance, attendance, and school activities.

-Improved communication channels with teachers and school administrators for a more holistic understanding of their child's educational journey.

**5.School Management:**

-Comprehensive data insights support informed decision-making, enabling strategic planning and resource optimization.

-Improved operational efficiency and transparency contribute to the overall effectiveness of school management.

**6. IT Personnel:**

-Skill enhancement through the implementation and maintenance of the high school management system.

-Involvement in a project that aligns with modern technological trends and advances.

**7. Education Regulatory Bodies:**

-Access to standardized and streamlined data for regulatory reporting and compliance purposes.

-Increased efficiency in monitoring and evaluating the educational institution's performance.

**8. Future Students:**

-The establishment of a technologically advanced and efficient educational environment sets the foundation for a positive and conducive learning experience for incoming students.

In essence, the high school management system seeks to empower and benefit a diverse range of stakeholders, fostering a more collaborative, informed, and efficient educational community.

**1.10 Methodology**

1 Data Source

Our methodology for data collection involved a multifaceted approach:

In-Person Surveys: Conducted surveys within the high school community to gather firsthand insights and requirements directly from stakeholders, including administrators, teachers, and students.

Recent Research: Reviewed and analyzed recent research papers and studies related to high school management systems, ensuring our approach aligns with current trends and best practices.

2 Fact Finding Methodology

Observation: Visited high school campuses to observe existing administrative processes and gather real-time data on the challenges faced by the staff.

Document Analysis: Scrutinized documentation on high school management and augmented our understanding with recent research papers, ensuring our findings are informed and up to date.

3 Software Development Methodology

For the development of the High School Management System, we adopted the Agile Scrum methodology, offering an iterative and collaborative approach. The reasons for selecting Agile Scrum include:

Iterative Development: Features are developed incrementally in short cycles, allowing for quick adaptability to changing requirements.

Cross-Functional Collaboration: Collaborative efforts of cross-functional teams, including developers, designers, and stakeholders, ensure a holistic approach to system development.

Client Involvement: Regular client involvement and feedback loops allow for real-time adjustments based on evolving needs.

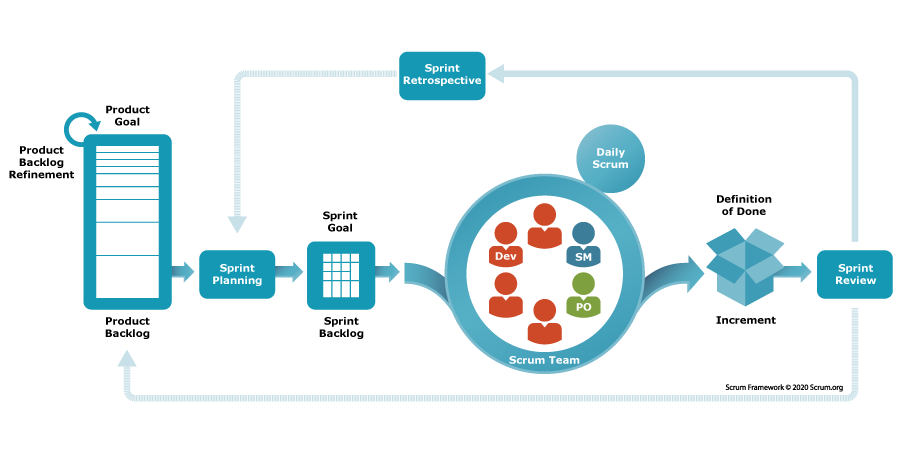
4 Technology Stack

Our technological choices were driven by the need for a robust, interactive, and scalable system:

HTML, CSS, JavaScript: Utilized for the frontend development to create an intuitive and user-friendly interface for both administrators and end-users.

Node.js: Employed for server-side development, ensuring a fast and scalable backend infrastructure.

5 Agile Scrum Framework



**Figure 2: Agile Scrum Framework Lifecycle**

The Agile Scrum framework guided our development process through the following key components:

**Sprints:** Divided the project into time-boxed iterations, or sprints, each lasting two weeks, allowing for frequent releases and continuous improvement.

**Scrum Team:** Formed a cross-functional team, including developers, designers, and a product manager, to collaboratively work on backlog items.

**Daily Stand-ups:** Conducted daily stand-up meetings to discuss progress, challenges, and plan for the day's work.

**Backlog Prioritization:** Maintained a prioritized backlog of features and functionalities, ensuring the team focused on delivering high-value items first.

By combining in-person surveys, recent research findings, and an Agile Scrum approach with HTML, CSS, JavaScript, and Node.js technologies, our methodology aimed to create a high school management system that is both user-centric and technologically advanced.

**1.11 Development Tools for High School Management System**

The development of the High School Management System involved a strategic selection of software and hardware tools to ensure a streamlined and effective development process.

**1 Software Tools:**

**🖥 Integrated Development Environment (IDE):**

- Visual Studio Code (VS Code): A versatile and collaborative IDE that supported the coding process with features like syntax highlighting and integrated version control.

- Studio 3T: A comprehensive IDE tailored for MongoDB, equipped with a range of collaborative tools, and features essential for database development, including intuitive query building, robust data visualization, and seamless integration with version control systems.

**🔄 Version Control System (VCS):**

- Git: The fundamental version control system that facilitated collaborative coding efforts and version tracking.

- GitHub: The platform for hosting Git repositories, enhancing code management and collaboration.

**📝 Project Documentation and Collaboration:**

- Notion and Google Docs: Platforms used for project documentation, task management, and collaborative writing and for version control when writing the documentation.

**🌐 Web Browsing:**

- Google Chrome: The web browser of choice for testing and debugging web-based functionalities.

- Microsoft edge: some member of the group chose to use this application instead of chrome for the same purpose.

💻 **Operating Systems**:

- Linux Mint Mate and Windows 11: Operating systems providing diverse development environments.

**🎨 Web Development Frameworks:**

- Bootstrap: A front-end framework used for creating responsive and visually appealing user interfaces.

- Node.js: A runtime environment used for building scalable and efficient server-side applications on this project.

-Express: A minimalist and flexible Node.js web application framework leveraged for crafting scalable and high-performance server-side applications on this project.

**🗃 Backend Database:**

- MongoDB: The chosen NoSQL database management system for backend data storage.

**2 Hardware Tools:**

**💻 Personal Devices:**

- Personal Computer: A high-performance workstation serving for coding, compilation, and testing.

- Personal Phone: Various devices, including smartphones, were employed for comprehensive testing to ensure cross-platform compatibility.

**📡 Connectivity:**

- Wi-Fi Dongle: Enhancing wireless connectivity for seamless collaboration.

- Flash Drive: Used for data storage and transfer during development.

By integrating tools such as VS Code, Git, GitHub, Notion, Google Docs, Google Chrome, Linux Mint Cinnamon, Windows 11, Bootstrap, MySQL, and various hardware devices, the development team ensured a comprehensive toolkit for creating a robust High School Management System. This selection of tools facilitated collaborative coding, version control, project documentation, and efficient development across different platforms.

**1.12 Required Resources**

Ensuring the successful development and implementation of the High School Management System involves careful consideration of the required resources, both human and material, accompanied by associated costs.

**1.12.1 Human Resources**

- Development Team:

  - Project Manager: Responsible for overall project coordination and management.

  - Software Developers: Engaged in coding, testing, and ensuring the functionality of the system.

  - Database Administrator: Manages and optimizes the backend database structure.

  - UI/UX Designers: Focus on creating an intuitive and visually appealing user interface.

  - Quality Assurance (QA) Testers: Conduct thorough testing to identify and rectify any system issues.

**1.12.2 Material Resources**

- **Hardware:**

  - Personal Computers: Utilized for coding, testing, and other development tasks.

  - Personal Phones: Used for comprehensive testing to ensure cross-platform compatibility.

  - Wi-Fi Dongle: Enhances wireless connectivity for seamless collaboration.

  - Flash Drives: Employed for data storage and transfer during development.

- **Software:**

  - Integrated Development Environment (IDE): Utilized tools such as Visual Studio Code.

  - Version Control System (VCS): Employed Git for efficient version tracking and collaboration.

  - Project Management Tools: Utilized Notion and Google Docs for project coordination.

  - Web Browsing: Used Google Chrome for testing and debugging.

  - Operating Systems: Developed on Linux Mint Cinnamon and Windows 11 environments.

  - Web Development Frameworks: Utilized Bootstrap for front-end development.

  - Backend Database: Implemented MongoDB for efficient data storage.

**1.13 Schedule (Modified for 2-Month Timeline)**

Given the condensed time frame of 2 months, the development schedule for the High School Management System has been adjusted for efficiency while maintaining key milestones.

**1.13.1 Project Phases**

**Phase 1: Planning and Analysis (Weeks 1-2)**

- Detailed project planning, goal setting, and scope finalization.

- Rapid analysis of high-level requirements and risk identification.

**Phase 2: Design and Documentation (Weeks 3-4)**

- Swift creation of wireframes and design prototypes by UI/UX designers.

- Expedited documentation of system specifications and design choices.

**Phase 3: Development (Weeks 5-8)**

- Agile coding by software developers, emphasizing key functionalities.

- Continuous integration of components with regular sprint reviews.

**Phase 4: Testing and Quality Assurance (Weeks 9-10)**

- Intensive testing by QA testers to identify and rectify bugs promptly.

- Ensuring system reliability and performance under time constraints.

**Phase 5: Deployment and Training (Weeks 11-12)**

- Expedited deployment of the system in a controlled environment.

- Fast-tracked training sessions for end-users and administrators.

**1.13.2 Project Milestones**

**1. Project Kickoff (Week 1):**

   - Quick introduction, alignment on objectives, and assignment of roles.

**2. Design Approval (Week 4):**

   - Swift finalization of UI/UX design and system architecture.

   - Client approval to proceed with development.

**3. Mid-Development Review (Week 7):**

   - Rapid evaluation of development progress and goal alignment.

   - Immediate identification and resolution of any issues.

**4. Quality Assurance Completion (Week 10):**

   - Swift completion of rigorous testing and bug resolution.

   - Assurance of system reliability within the condensed timeframe.

**5. Deployment and Training (Week 12):**

   - Expedited system deployment in the live environment.

   - Efficient training sessions for end-users and administrators.

**1.13.3 Contingency Planning**

Given the accelerated timeline, a proactive approach to contingency planning is essential. Regular check-ins and quick decision-making will be crucial to addressing any unexpected challenges without compromising the overall project delivery.

This adjusted schedule prioritizes efficiency and fast-tracked development while maintaining a structured approach to ensure the successful and timely completion of the High School Management System.

**1.14 Team Composition**

Our team is structured into two primary subgroups to maximize our human resources effectively, in line with an agile development plan.

The **first subgroup** is primarily responsible for documentation, which encompasses:

* Requirement engineering (functional, nonfunctional, performance)
* System models, which include each scenario and use case.
* Analysis models such as functional models (use case and scenario), analysis object model (class and object diagram), and dynamic model (state chart and sequence diagrams)
* Activity diagram, collaboration diagram, component diagram, and deployment diagram

In summary, this subgroup is dedicated to the comprehensive detailing of documentation.

The **second subgroup** is focused on the system’s development, including the front end, backend, testing, and validation.

While these groups have their primary focus areas, they are not strictly confined to these tasks. They will collaborate and contribute to other areas as needed. This structure is designed to assign clear responsibilities to each team member and ensure the timely completion of the project.

|  |  |  |
| --- | --- | --- |
| Role | Responsible | Sub-Group |
| Team leader | Nebiyu Musbah | **Group 2** |
| Requirement analysis, design, architecture, and documentation. | Yohannes Gezachew | **Group 1** |
| Muhammed Samson |
| Abel Gezu |
| Requirement, design, architecture, implementation. | Nebiyu Musbah | **Group 2** |
| Fahmi Dinsefa |
| System architecture analysis, analysis, design, and documentation | Yohannes Gezhacew | **Group 1** |
| Mohammod samson |
| UI design, implementation. | Fahmi Dinsefa | **Group 2** |
| Nebiyu Musbah |
| Abel Gezu |
| Database design, architecture, analysis | Yohannes Gezhacew  Muhammed Samson | **Group 1** |
| Database implementation. | Nebiyu Musbah  Fahmi Dinsefa | **Group 2** |

**Chapter Two**

**Description of the Existing System**



**Figure 3: Description of Existing System**

In the current educational landscape of Ethiopian high schools, administrative processes primarily rely on traditional manual methods and basic tools for managing student information, academic records, and communication. The existing system involves cumbersome paperwork, decentralized record-keeping, and rudimentary communication channels among stakeholders. Administrativem, tasks, such as scheduling and class management, are executed through manual processes, leading to challenges in organization and efficiency. The current record-keeping mechanisms, while functional, lack the optimization needed for a rapidly evolving educational environment. Additionally, communication between teachers, students, and parents relies on conventional methods, limiting the potential for streamlined interaction. Recognizing the limitations of the existing system, this project aims to address these inefficiencies by proposing a comprehensive High School Management System that centralizes data management, enhances scheduling, improves communication channels, and optimizes the overall educational experience.

**2.1 Major Function of the existing system**

Most of the high schools we contacted (through telegram, phone, and some in person) and managed to collect information use high school management systems that operate predominantly with manual processes, incorporating traditional paper-based methods and limited utilization of Excel. These practices present challenges in terms of efficiency, accuracy, and accessibility. In the subsequent sections, we will delve into the drawbacks of this manual system and propose a more streamlined and automated approach in the proposed system.

* **Student Management:**
  + Collects and stores student information using traditional paper records, with basic Excel usage for specific tasks like enrollment.
* **Staff Management:**
  + Handles staff-related activities, such as recruitment and payroll, through paper-based documentation with minimal use of Excel.
* **Course Management:**
  + Manages courses, registrations, and schedules with traditional paperwork, supplemented by basic Excel usage.
* **Attendance Tracking:**
  + Records daily attendance manually on paper, occasionally utilizing Excel for tracking.
* **Examination System:**
  + Conducts exams manually from creating timetables to result generation, with limited integration of Excel.
* **Communication:**
  + Relies on traditional communication methods, occasionally using Excel for basic communications.
* **Administrative Tasks:**
  + Manages administrative tasks manually, using traditional methods with basic support from Excel.
* **Reporting:**
  + Generates and distributes reports manually, occasionally utilizing Excel, requiring considerable time and effort.

**2.2 User of Current System**

The current high school management system involves various users who interact with the system in different capacities. These users play distinct roles in the manual processes. Below are the primary users of the existing system:

1. **Administrators:**
   * Responsible for overseeing and managing overall system operations, including administrative tasks, resource allocation, and policy implementation.
2. **Teachers:**
   * Engaged in student management, attendance tracking, grade recording, and other academic activities.
3. **Students:**
   * Interact with the system for enrollment, course registration, attendance monitoring, and accessing academic information.
4. **Staff (non-teaching):**
   * Includes non-teaching staff responsible for recruitment, payroll management, performance evaluation, and financial transactions, as well as managing communication within the school community, handling announcements, and basic communication tasks, and managing libraries books.
5. **Parents/Guardians:**
   * Access information related to their child's academic progress, attendance, and communicate with teachers and administrators.
6. **Accountant:**
   * Manages financial transactions, budgeting, and payroll activities within the school.

**2.3 Drawback of current system**

Despite serving the essential functions of a high school management system, the current manual system presents several drawbacks that hinder its effectiveness and efficiency. Below are the key drawbacks associated with the existing system we observed:

1. **Inefficiency:**
   * The reliance on manual processes leads to inefficiencies in various tasks, such as enrollment, attendance tracking, and report generation, consuming more time and resources than necessary.
2. **Error-Prone Processes:**
   * Manual data entry and handling increase the risk of errors in student records, attendance registers, and financial transactions, potentially impacting the accuracy of information.
3. **Limited Accessibility:**
   * The paper-based nature of the system restricts accessibility to information, making it challenging for stakeholders to retrieve real-time data and communicate effectively.
4. **Communication Challenges:**
   * Lack of an integrated communication system results in difficulties in disseminating information promptly, affecting communication between teachers, administrators, and parents.
5. **Scalability Issues:**
   * Manual systems may face challenges in handling the increasing volume of data and processes as the school grows, potentially limiting scalability.
6. **Data Redundancy:**
   * The use of paper records and manual data handling can lead to redundancy and inconsistency in information across different branches of the school.
7. **Limited Reporting Capabilities:**
   * Generating reports manually, with occasional use of Excel, is time-consuming and may not provide the comprehensive insights needed for effective decision-making.
8. **Dependency on Individuals:**
   * Certain processes, especially those managed by specific individuals (e.g., a single communication coordinator), may be heavily dependent on individual skills and availability, posing a risk in case of personnel changes.
9. **Security Concerns:**
   * The manual system may lack robust security measures, making it susceptible to unauthorized access and potential data breaches.
10. **Lack of Integration:**
    * The absence of an integrated system makes it challenging to streamline processes and share information seamlessly across different departments.

Understanding these drawbacks is crucial for proposing improvements and enhancements in the upcoming sections of the documentation, where the focus will be on the development of a more efficient and automated high school management system.

**2.4 Business Rule**

In the context of building the new high school management system, various business rules will govern the processes and interactions within the system. These rules are essential for maintaining order, consistency, and compliance with organizational policies. Below are some proposed business rules for the new system:

1. **Enrollment Eligibility:**
   * Only students who meet the school's admission criteria, including academic qualifications and relevant documentation, are eligible for enrollment.
2. **Attendance Tracking:**
   * Students must adhere to the prescribed attendance policies. A minimum attendance percentage may be required for eligibility in examinations and other academic activities.
3. **Grade Calculation:**
   * The system will calculate grades based on a predefined grading system, incorporating factors such as class participation, assignments, and examination performance.
4. **Financial Transactions:**
   * Financial transactions, including fee payments and other monetary activities, must follow the school's established financial policies. The system will enforce secure and transparent financial processes.
5. **Communication Protocols:**
   * Official communications within the system, including notifications, announcements, and messages, will follow standardized communication protocols to ensure clarity and consistency.
6. **Performance Evaluation:**
   * The system will facilitate the performance evaluation of staff and teachers based on predefined criteria and performance metrics.
7. **Data Privacy and Security:**
   * The system will adhere to robust data privacy and security measures, ensuring compliance with relevant data protection regulations and safeguarding the confidentiality of student and staff information.
8. **Resource Allocation:**
   * The system will automate the allocation of resources, including classrooms, materials, and facilities, based on predefined schedules and allocation criteria.
9. **Curriculum Management:**
   * The system will support the management of course offerings and curriculum development, ensuring compliance with educational standards and guidelines.
10. **Timely Reporting:**
    * The system will automate the generation and distribution of academic and administrative reports within specified periods.

These proposed business rules will guide the development of the new high school management system, ensuring that it aligns with organizational policies, enhances operational efficiency, and provides a secure and user-friendly environment for all stakeholders.

**Chapter 3**

**Proposed System**

**3.1 Overview**

The proposed high school management system, built on internet-based architecture, aims to enhance operational efficiency by automating key functions, relieving teachers and staff of manual workloads. Core features include student and staff management, course administration, attendance tracking, and examinations, accessible through a user-friendly interface for all stakeholders.

The system integrates a web-based communication platform, fostering seamless interactions and collaboration. Robust reporting capabilities provide real-time insights into student performance, attendance, and financial transactions, facilitating informed decision-making.

Prioritizing security, the system employs encryption and access controls. Its scalable and flexible architecture, coupled with a modular structure, allows for continuous improvement based on user feedback and evolving educational needs.

A comprehensive training program and ongoing support mechanisms ensure a smooth transition. The proposed system stands as an efficient, user-friendly solution, easing the workload on teachers and staff. Subsequent sections detail specific functionalities and requirements, providing a roadmap for development and implementation.

**3.2 Functional Requirements**

In outlining the functional requirements for our envisioned high school management system, we have focused on crafting features tailored to the diverse needs of administrators, teachers, students, and parents. Here is an overview of the key functionalities:

1. **Student Management:**
   * Automate student enrollment, track academic progress, and manage personal details.
   * Features include registration, profile management, and academic history tracking.
2. **Staff Management:**
   * Streamline staff-related processes, covering recruitment, payroll, and performance evaluation.
   * Features encompass staff profiles, a recruitment module, and payroll management.
3. **Course Administration:**
   * Efficiently manage course offerings, registrations, and schedules.
   * Features involve course creation, enrollment, and timetable generation.
4. **Attendance Tracking:**
   * Automate attendance recording and monitoring.
   * Features provide real-time attendance tracking and automated notifications.
5. **Examinations:**
   * Automate exam scheduling, grading, and result generation.
   * Features include an exam timetable, automated grading, and result publication.
6. **Communication Platform:**
   * Facilitate seamless communication among stakeholders.
   * Features encompass a messaging system, announcements, and event notifications.
7. **Reporting:**
   * Provide comprehensive and real-time reports on various metrics.
   * Features offer customizable reports on student performance, attendance, and financial transactions.
8. **Security Measures:**
   * Ensure data privacy and system security.
   * Features include encryption, access controls, and regular security audits.
9. **Scalability and Flexibility:**
   * Design the system to adapt and grow with changing needs.
   * Features involve a modular architecture and scalability for increased data volume.
10. **User Training and Support:**
    * Facilitate a smooth transition and ongoing system use.
    * Features include comprehensive training modules and user support channels.

These functional requirements serve as the backbone of our proposed high school management system, addressing the unique needs of each user group and contributing to a more effective and efficient educational environment. In the next section, we will delve into non-functional requirements, system models, and design considerations.

**3.3 Non-Functional Requirements**

In addition to the specific functionalities, our high school management system needs to meet certain non-functional requirements—qualities that define its performance, usability, and overall effectiveness. Here is a closer look at these key non-functional requirements:

**3.3.1 User Interface**

The User Interface (UI) is a pivotal element of the proposed High School Management System, designed with simplicity and user-friendliness in mind. The interface caters to diverse users, including administrators, teachers, students, and parents, ensuring an intuitive and accessible interaction. A responsive web application fosters seamless navigation across devices, offering an inclusive experience for individuals with varying levels of technical expertise.

**3.3.2 Performance Consideration**

Performance is a key non-functional requirement, emphasizing swift response times even during periods of high usage. The system is meticulously engineered to maintain optimal performance, with continuous monitoring and optimization strategies in place. Regular database maintenance ensures sustained efficiency, contributing to an uninterrupted and responsive user experience.

**3.3.3 Reliability**

Reliability is paramount in ensuring the consistent availability and functionality of the High School Management System. The system incorporates a robust backup and recovery mechanism to safeguard against data loss or system failures. This commitment to reliability instills confidence among users, assuring them of the system's dependability for critical educational functions.

**3.3.4 Error Handling**

The proposed system prioritizes effective error handling mechanisms to enhance user experience and system stability. Comprehensive error detection, notification, and resolution protocols are integrated to identify and address issues promptly. User-friendly error messages are implemented to guide users in troubleshooting and resolving any encountered challenges, ensuring a smooth and error-tolerant operational environment.

**3.3.5 Security**

Security is a foundational aspect of the High School Management System, with stringent measures in place to safeguard sensitive information. The system implements secure authentication mechanisms for administrators, teachers, students, and parents. Encryption protocols are applied to protect confidential data, ensuring that the system adheres to industry best practices for information security.

The non-functional requirements outlined above collectively contribute to a high-quality, reliable, and secure High School Management System. By focusing on these aspects, the system aims to provide a seamless and trustworthy educational management solution for all stakeholders involved.

**3.4 System Model**

**3.4.1 Scenarios**

**Scenario 1: Student Enrollment** *Description:* A new student enrolls in the high school. *Steps:*

1. Parents access the online enrollment portal.
2. Provide student details, upload necessary documents.
3. Select desired courses and extracurricular activities.
4. Submit the enrollment form. *Expected Result:* Student information is stored in the system, and the enrollment status is confirmed.

**Scenario 2: Exam Result Publication** *Description:* Teachers publish exam results. *Steps:*

1. Teachers access the grading system.
2. Enter and validate students' scores.
3. The system calculates grades and publishes results.
4. Automated notifications are sent to students and parents. *Expected Result:* Students and parents access individual results through the student portal.

**Scenario 3: Class Scheduling** *Description:* Administrators create and manage class schedules. *Steps:*

1. Administrators access the scheduling module.
2. Input class details, assign teachers, and allocate students.
3. System optimizes distribution considering capacity and subject requirements.
4. Resulting class schedule is organized and balanced.

**Scenario 4: Attendance Tracking** *Description:* Teachers systematically monitor student attendance. *Steps:*

1. Teachers mark daily attendance using the system.
2. Attendance data is recorded in the database.
3. System generates reports on daily, weekly, and monthly attendance.
4. Reports assist in monitoring and analyzing student attendance trends.

**Scenario 5: Online Resource Access** *Description:* Students access educational resources online. *Steps:*

1. Students log in to the student portal.
2. Navigate to the resource section.
3. Access digital textbooks, study materials, and other educational resources.
4. Utilize online resources to enhance learning.

**Scenario 6: Fee Management** *Description:* The accountant manages fee-related transactions. *Steps:*

1. Access the fee management module.
2. Record incoming fee payments.
3. Generate fee receipts for students and parents.
4. Provide financial reports to administrators.
5. Manage Student Finance.

**Scenario 7: Manage Teacher/Staff Finance** *Description:* The accountant manages financial transactions related to teachers and staff. *Steps:*

1. Access the staff finance module.
2. Record salary payments and other financial transactions for teachers and staff.
3. Generate financial statements.
4. Provide financial reports to administrators.
5. Manage Teacher/Staff Finance.

**3.4.2 Use Case Model**

**Actor: Student**

* **Use Case 1: View Class Schedule**
  + Students can view their class schedules through the student portal.
* **Use Case 2: Access Exam Results**
  + Students access and review their exam results through the student portal.
* **Use Case 3: Pay Fees Online**
  + Students can pay fees online through the system.
* **Use Case 4: Online Resource Access**

Students can access digital textbooks, study materials, and other educational resources through the student portal.

**Actor: Teacher**

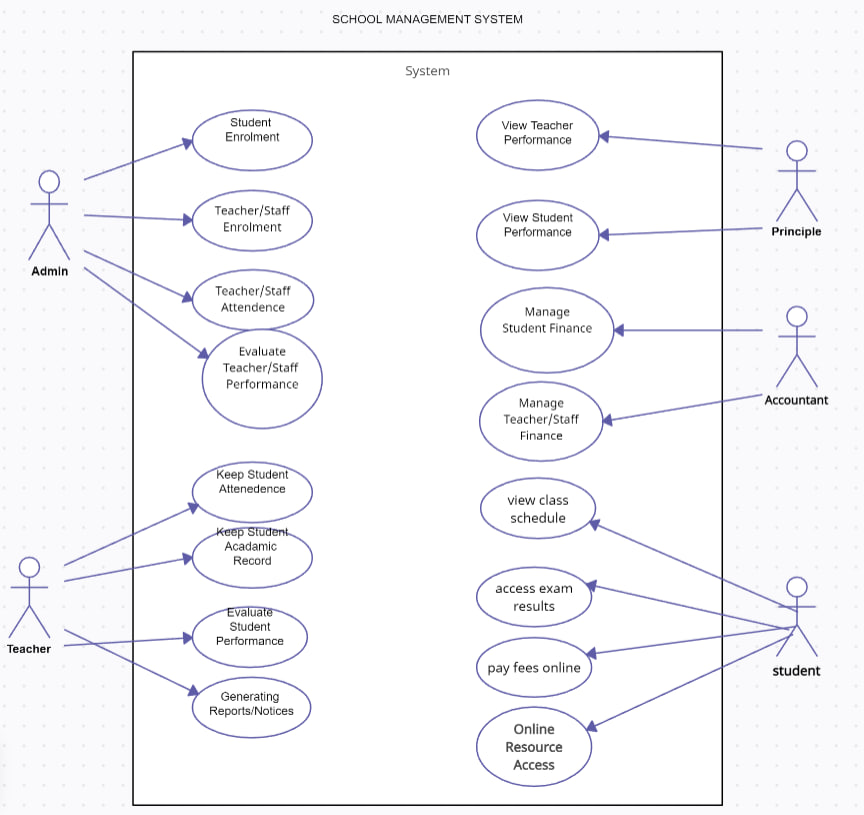
* **Use Case 5: Enter Grades**
  + Teachers enter and submit students' grades into the grading system.
* **Use Case 6: Schedule Parent-Teacher Meetings**
  + Teachers can schedule and manage parent-teacher meetings through the system.

**Actor: Administrator**

* **Use Case 7: Generate School Reports**
  + Administrators generate various school reports using the system.
* **Use Case 8: Manage Teacher Assignments**
  + Administrators assign teachers to specific classes and subjects using the system.

**Actor: Accountant**

* **Use Case 9: Fee Management**
  + The accountant manages fee-related transactions, records payments, generates fee receipts, and provides financial reports to administrators.
* **Use Case 10: Manage Teacher/Staff Finance**
  + *Description:* The accountant manages financial transactions related to teachers and staff, records salary payments, generates financial statements, and provides financial reports to administrators.



**Figure 4: Use Case Model**

**3.5 Object Model**

**3.5.1 Data Dictionary**

The data dictionary for our High School Management System outlines the key classes, attributes, operations, and descriptions within the system. Here is an overview:

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Attributes | Operations | Description |
| Register | First\_name, Last\_Name, student\_ID, Department, School, Year, user\_name, password | Register () - Used to register new users to the system. | This class manages the registration process for new users, capturing essential details such as names, student ID, department, and login credentials. |
| Course | Course\_code, course\_title, course\_credit\_hour, course\_department, course\_instructor, course\_fee | create Course (), take Course (), browse Category () | The Course class handles various aspects related to courses, including creation by instructors, enrollment by students, and browsing available courses based on categories. |
| Quiz | Quiz\_code, course\_code, question, answer | take Quiz (), add Quiz () | This class deals with quizzes associated with courses. Students can take quizzes, and instructors can prepare and add quiz questions for evaluation. |
| Support | Course\_code, question\_id, question\_body | get Support () | This class facilitates a support system where users can seek assistance by accessing a community forum. It includes attributes related to the course and questions for support. |
| User | User\_name, password | Login () | The User class manages user authentication, allowing users to log in to the system using their usernames and passwords. |

**3.5.2 Class Diagram**

1. **Classes:**

   - **Student**: Enrolls and views grades.

   - **Teacher**: Creates classes and assigns grades.

   - **Administrator**: Adds students and teachers.

   - **Course**: Manages students, teachers, and exams.

   - **Class**: Manages student enrollment.

   - **Exam**: Records grades.

   - **Grade**: Enumerates grades.

2. **Associations**:

   - **Inheritance**: Student, Teacher, and Administrator inherit from User.

   - **Student**-**Course**: Enrollment.

   - **Teacher-Course**: Teaching.

   - **Administrator-Student/Teacher**: Adding users.

   - **Course-Student/Teacher**: Association.

   - **Class-Student**: Enrollment.

   - **Exam-Course/Student**: Record keeping.

3. **Actions:**

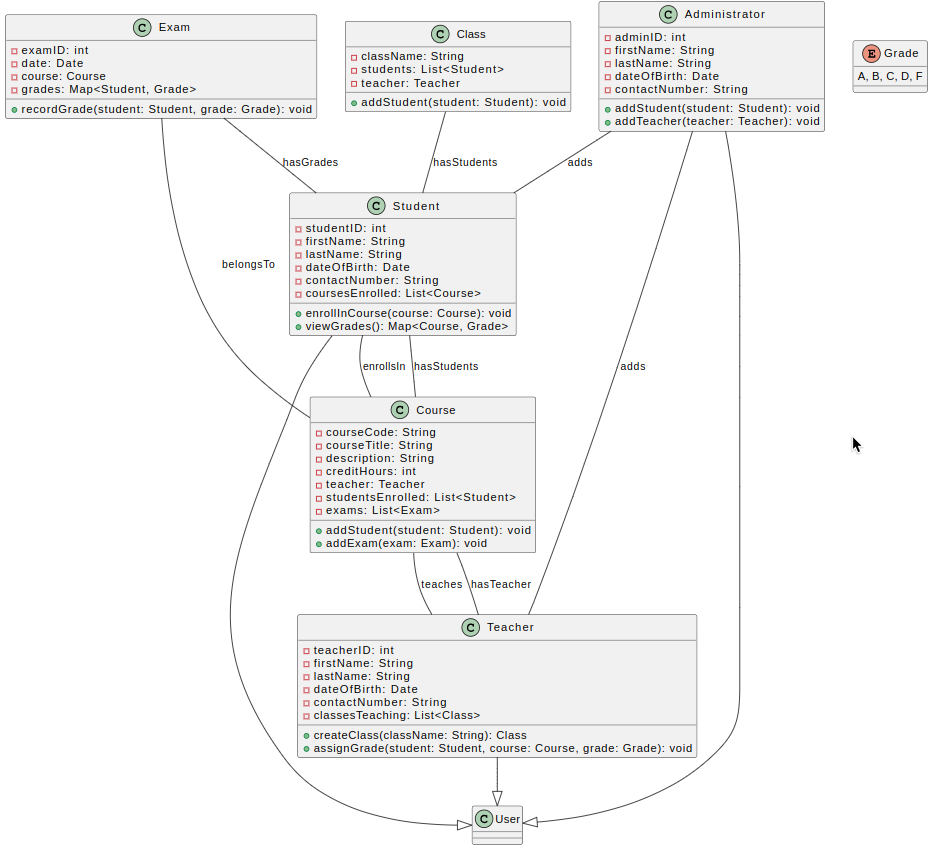
   - **Enrolling/Viewing Grades**: Students.

   - **Teaching/Assigning Grades**: Teachers.

   - **Admin Tasks:** Administrators.

   - **Course/Class Management**: Administrators.

   - **Exam/Grade Management:** Administrators.

****

**Figure 5: Class Diagram**

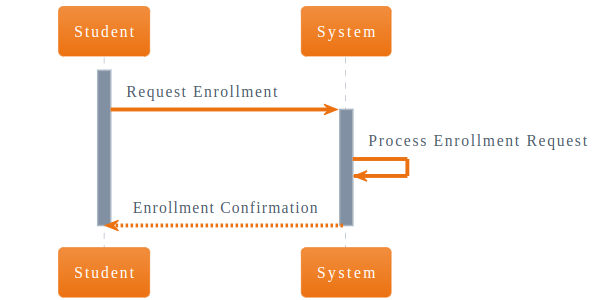
**3.6 Dynamic Model**

**3.6.1 Sequence Diagram**

The sequence diagrams will show how different parts of the high school management system interact. They'll start with student enrollment, showing how students sign up for classes. Then, they'll cover publishing exam results, scheduling classes, and tracking attendance. Students accessing online resources and managing fees and teacher finances will also be explained. These diagrams will give a clear picture of how the system works, helping understand its inner workings and processes.

1. Student enrollment Sequence diagram

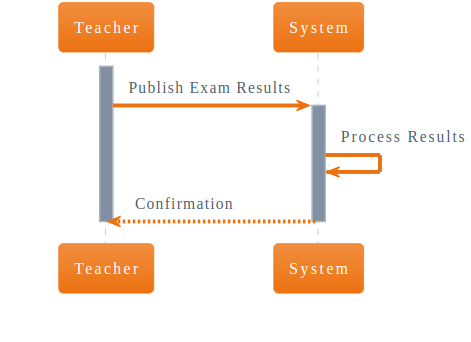
The student enrollment sequence diagram is a graphical representation of the interactions between the student, the enrollment system, and the database during the enrollment process. It shows the messages exchanged between the objects in a chronological order, using vertical lifelines and horizontal arrows.



**Figure 6: Student Enrollment Sequence Diagram**

1. Exam Result Publication Sequence Diagram

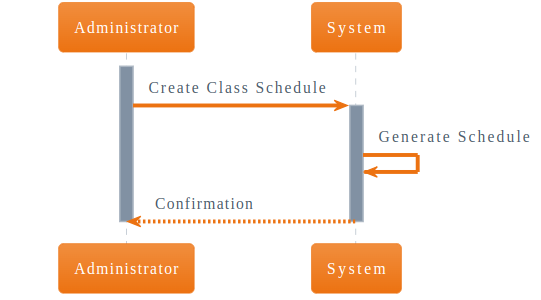
The exam result publication sequence diagram is a graphical representation of the interactions between the student, the exam system, and the database during the exam result publication process. It shows how the teacher publishes exam results, how the exam system verifies the Teacher’s identity and retrieves the results from the database.



**Figure 7: Exam Result Publication Sequence Diagram**

1. Class Scheduling Sequence Diagram

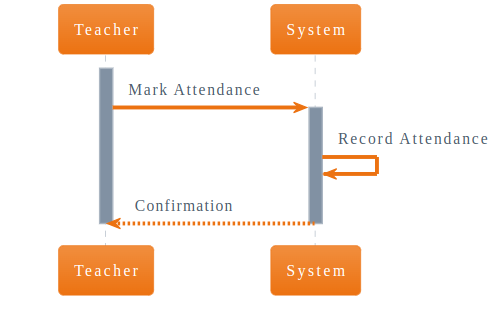
The class scheduling sequence diagram is a graphical representation of the interactions between the administrator and the system during the class scheduling process. It shows how the administrator requests to view, add, edit, or delete classes, how the system validates the request and updates the database, and how the system generates and confirms the changes to the administrator.



**Figure 8: Class Scheduling Sequence Diagram**

1. Attendance Tracking Sequence Diagram

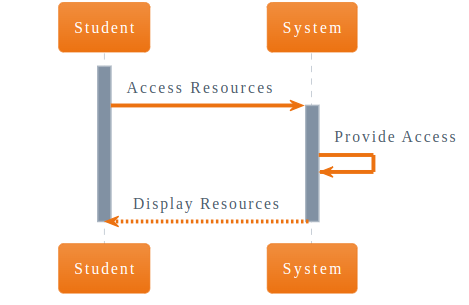
The attendance tracking sequence diagram is a graphical representation of the interactions between the teacher and the system during the attendance tracking process. It shows how the teacher logs in to the system, selects a class, marks the attendance of the students, and submits the attendance report to the system.



**Figure 9: Attendance Tracking Sequence Diagram**

1. Online Resource Access Sequence Diagram

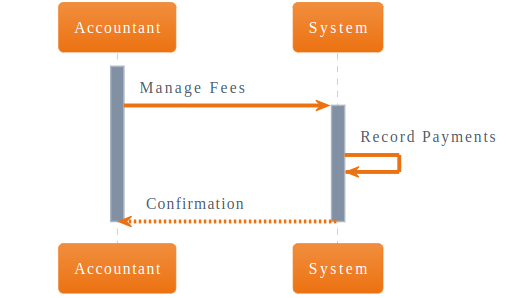
The online resource access sequence diagram is a graphical representation of the interactions between the student and the system during the online resource access process. It shows how the student logs in to the system, selects a course, views the available resources, access a resource, and the system provides access and eventually the system display the resources.



**Figure 10: Online Resource Access Sequence Diagram**

1. Fee Management Sequence Diagram

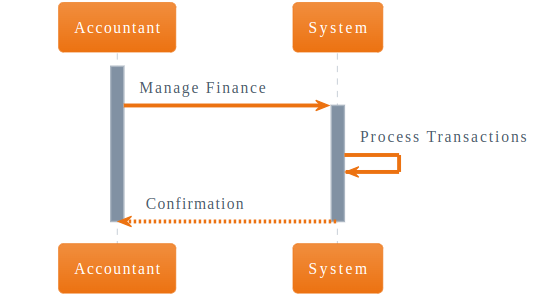
The fee management sequence diagram is a graphical representation of the interactions between the accountant and the system during the fee management process. It shows how the accountant logs in to the system, views the fee details of the students, generates invoices, collects payments, and updates the database while the system records payments and confirms to the accountant.



**Figure 11: Fee Management Sequence Diagram**

1. Manage Teacher/ Staff Finance Sequence Diagram

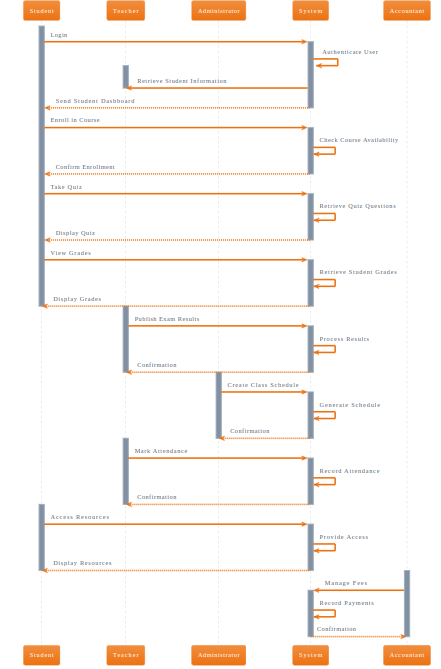
The manage teacher/staff finance sequence diagram is a graphical representation of the interactions between the accountant and the system during the finance management process. It shows how the accountant logs in to the system, views the salary details of the teachers and staff, generates pay slips, transfers payments, and updates the database.



**Figure 12: Manage Teacher/Staff Finance Sequence Diagram**

The overarching sequence diagram for high school system management would encompass a comprehensive array of sequences, illustrating various events and interactions that occur within the application. This diagram serves as a visual representation of the flow of actions and communications between different components or actors involved in the system, providing a clear understanding of how the system operates under different scenarios.

The general Sequence diagram altogether would look like:



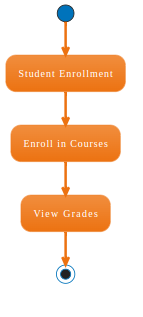
**Figure13:General Sequence Diagram**

**3.6.2 Activity Diagram**

The activity diagram offers a clear picture of how things work in our educational system. It covers student enrollment, what teachers do, actions by administrators, managing courses and classes, and handling exams and grades. It helps us see the steps involved in each task, making it easier to understand how our system operates and making things more efficient for everyone involved.

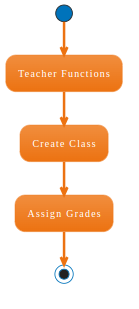
1. Student Enrollment Activity Diagram

The student enrollment activity diagram is a graphical representation of the actions that a student performs during the enrollment process. It shows how the student starts the enrollment, chooses a course, enrolls for the course, views the grades.

**Figure14: Student Enrollment Activity Diagram**

1. Teacher Functions Activity Diagram

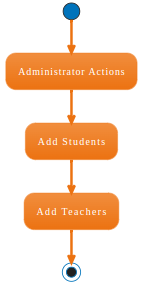
The teacher functions activity diagram is a graphical representation of the actions that a teacher performs during the teaching process. It shows how the teacher logs in to the system, prepares the lesson plan, delivers the lecture, assigns Grades, and eventually logs out of the system.



**Figure 15: Teacher Functions Activity Diagram**

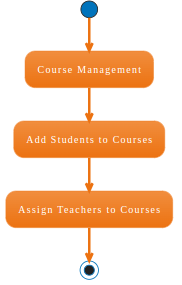
1. Administrator Actions Activity Diagram

The administrator actions activity diagram is a graphical representation of the actions that an administrator performs during the administration process. It shows how the administrator logs in to the system, manages the students, the courses that the student takes, and logs out of the system.



**Figure 16: Administrator Actions Activity Diagram**

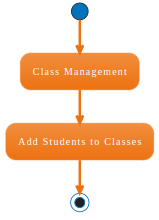
1. Course Management Activity Diagram

The course management activity diagram is a graphical representation of the actions that a course performs during the course management process. It adds students and teachers to the course.

**Figure17: Course Management Activity Diagram**

1. Class Management Activity Diagram

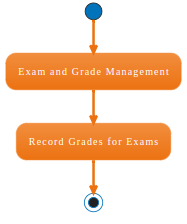
The class management activity diagram is a graphical representation of the actions that a class performs during the class management process. It shows how the class is created, updated, deleted, or viewed by the administrator or the teacher, and adds Students to classes.



**Figure18: Class Management Activity Diagram**

1. Exam and Grade Management Activity Diagram

The exam and grade management activity diagram are a graphical representation of the actions that a teacher or an administrator performs during the exam and grade management process. It shows how the teacher, or the administrator logs in to the system, creates, updates, deletes, or views the exams and grades of the students, and eventually records grades for exams.



**Figure 19: Exam and Grade Management Activity Diagram**

This is the general overall Activity Diagram focusing on student interactions. Here is a breakdown of the key elements:

**1. Start:** The process begins with the student or teachers logging into the system.

**2. Login Check**: The system checks if the login was successful. If yes, it proceeds to authenticate the user and retrieve student or teacher information.

**3. User Type Check:** The system verifies if the authenticated user is a student or a teacher. The administrator also verifies the student and teacher status.

**4. Student or Teacher Dashboard:** If the user is identified as a student, the system displays the Student Dashboard. If the user is identified as a teacher, the system displays the Teacher Dashboard. These serve as central points for interactions.

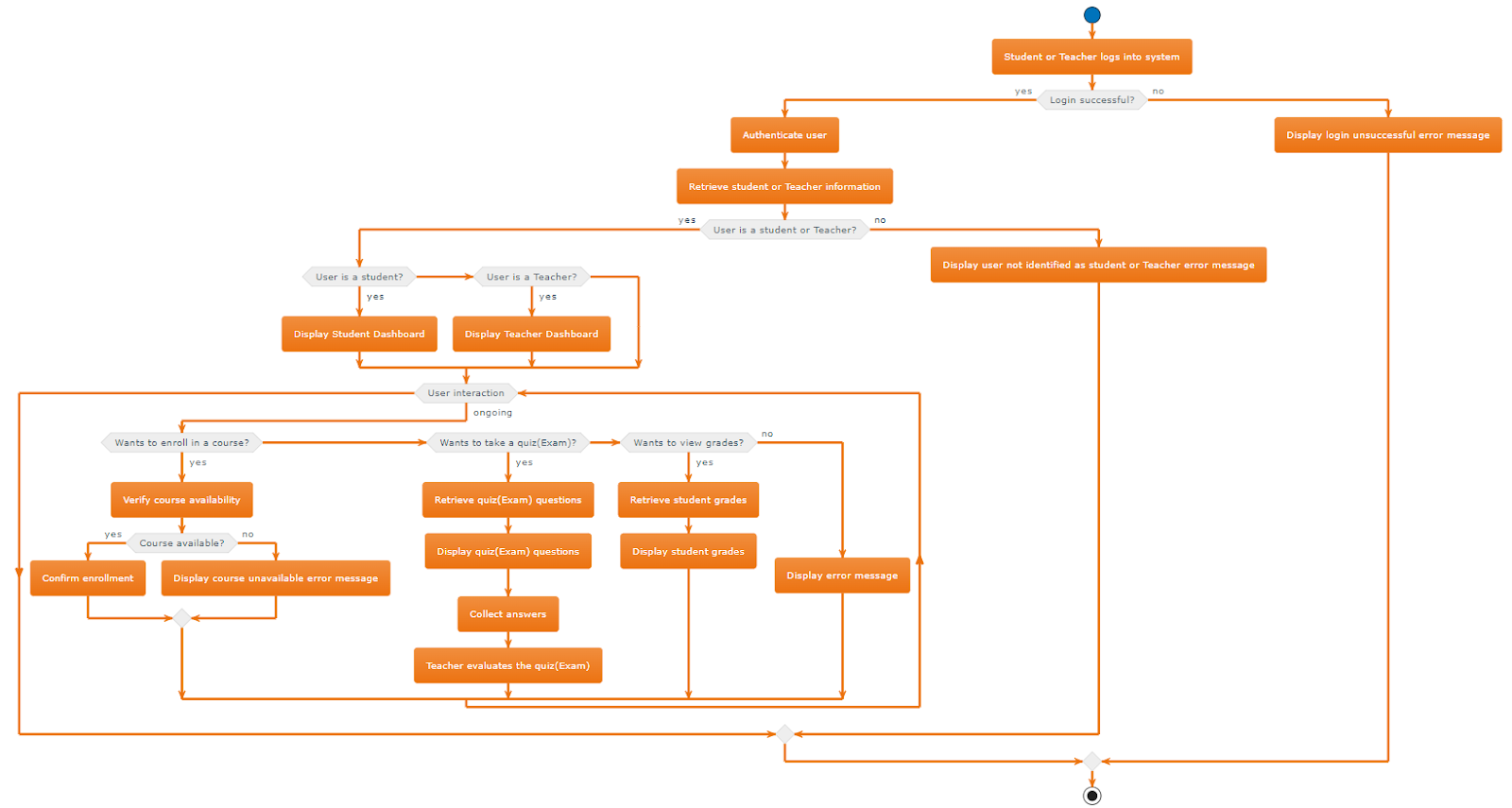
**5. Enrollment Process:** The diagram includes a decision point where the system checks if the student wants to enroll in a course. If yes, the system verifies course availability and confirms enrollment. If not, it proceeds to check if the student wants to take a quiz or exam prepared by the teacher.

**6. Quiz or Exam Process:** If the student chooses to take a quiz or exam, the system retrieves questions, displays them, collects answers, and the teacher evaluates the quiz or exam.

**7. Grade Viewing:** If the student does not want to enroll or take a quiz/exam, the system checks if the student wants to view grades. If yes, it retrieves student grades and displays them.

**8. Error Handling:** Throughout the process, error messages are displayed if certain conditions are not met, such as unsuccessful login or course unavailability.

**9. Stop:** The activity diagram concludes with a stop symbol, indicating the end of the process.

****

**Figure 20: General Activity Diagram**

**3.6.3 State Diagram**

The State Chart Diagram outlines how a High School Management System operates. It starts with system startup, moves to initialization, and then into the 'Ready' state. From there, users can enroll students in courses and classes. The system updates information accordingly. If maintenance is needed, it switches to maintenance mode for tasks like data backup before returning to 'Ready.' Eventually, it shuts down to end operations.

The State Chart Diagram illustrates the different states and transitions within the high school management system. Let's break down the key components:

**1. System Startup**: The diagram begins with the system startup, represented by the initial state. This signifies the moment when the system is powered on or initialized.

**2. Initializing:** After startup, the system enters the "Initializing" state, where it performs initialization tasks such as loading configurations and setting up internal components.

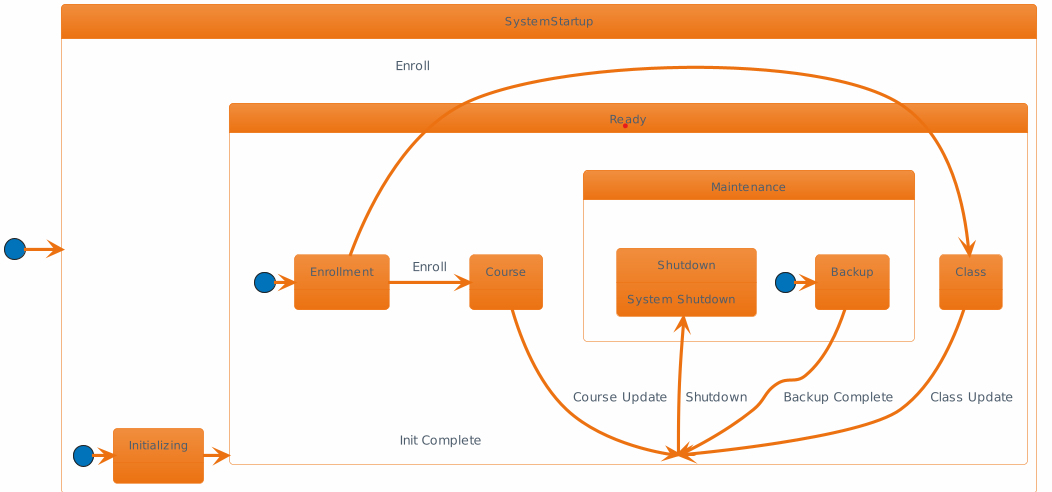
**3. Ready State:** Once initialization is complete, the system transitions to the "Ready" state. In this state, the system is fully initialized and ready to perform various tasks.

**4. Enrollment**: From the "Ready" state, users can initiate the enrollment process, where students are enrolled in courses and classes. This involves transitions to the "Enrollment" state.

**5. Course and Class Updates:** As enrollments occur, the system updates course and class information accordingly. This is represented by transitions from the "Enrollment" state to the "Course" and "Class" states.

**6. Maintenance:** If maintenance is required, the system transitions to the "Maintenance" state. Here, tasks such as data backup and system updates are performed.

**7. Shutdown:** Finally, when the system needs to shut down, it transitions to the "Shutdown" state, marking the end of its operation.

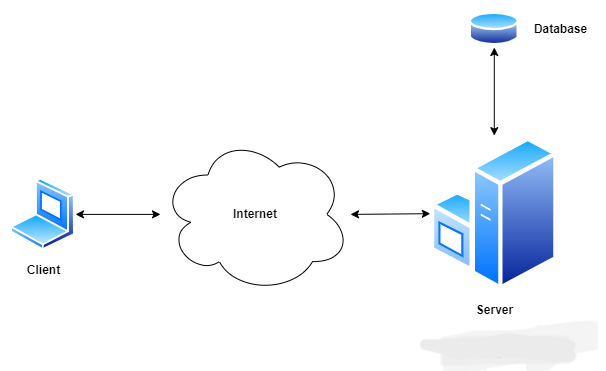
****

**Figure 21: State Diagram**

**Chapter 4**

**System Design**

This chapter presents the design aspects of the High School Management System. It includes the design objectives, the proposed system design, and the object design.



**Figure 22: Simple System Design Representation**

**4.1 Overview of the System**

This document outlines the design considerations of the overall system. It offers a comprehensive architectural overview of the proposed system. The aim is to capture and convey the significant architectural decisions made on the system.

**4.1.1 Design Goals**

The Design Goals define the qualities that the system should achieve and address during its design. The design goals for the system are categorized into four groups:

**Performance:** For the High School Management System to deliver efficient service, it should meet the following performance criteria:

* Response Time: The system should interact and respond to user requests within a maximum of 15 seconds.
* Memory: The system should require an average of 10-15 megabits of free memory to load on a user’s device.

**Dependability:** The High School Management System should attain these dependability characteristics to withstand crashes:

* Robustness: The system should continue to operate despite errors during the data update process from a web-based database server, providing error messages and continuing without crashing.

**Maintenance:** The system should be easily maintainable and extendable:

* Extensibility: Adding new functionality should only necessitate the creation of the system’s next version.
* Modifiability: Modifications to functionality should be confined to specific functions without impacting the overall system functionality. The system should automatically update when an internet connection is available.

**End User:** The system should prioritize end user satisfaction:

* Utility: The system language should be straightforward, clearly expressing the function of each activity.
* Usability: The system should encourage interaction and ease of use, considering principles such as learnability, clear interfaces, and flexibility.
* Robustness: The system should assist users in achieving their goals without ambiguity, supporting tasks such as course searching or quiz taking.

**4.1.2 Priority Design Goal**

1. End User
2. Performance
3. Usability
4. Maintenance

**4.2 Proposed System Architecture**

The proposed system architecture serves as the foundational structure upon which the High School Management System is built. It encompasses various architectural elements, including:

Client-Server Model:

The system adopts a client-server architecture, where clients, such as web browsers or mobile devices, interact with a central server to access and manipulate data. This model facilitates centralized data management, scalability, and efficient communication between clients and the server.

**4.2.2 Subsystem Decomposition**

The system is decomposed into several subsystems to enhance manageability and modularity. Each subsystem focuses on specific functionality, allowing for easier maintenance and development. Examples of subsystems include:

* **Student Management**: Handles student-related functionalities such as enrollment, attendance tracking, and academic performance monitoring.
* **Teacher Management**: Manages teacher-related tasks including scheduling, grading, and communication with students and parents.
* **Administrative Management**: Facilitates administrative processes such as staff management, resource allocation, and reporting.
* **Course Management**: Handles the creation, scheduling, and management of courses offered by the school. This includes defining course curriculum, assigning teachers, managing enrollment, and tracking student progress.

A white circle with black text

Description automatically generated

**Figure 23: Subsystem Decomposition Representation**

**4.2.3 Hardware and Software Mapping**

The system's architecture maps hardware and software components to their respective functionalities. Key mappings include:

* **Hardware**: so, on our system the Cloud-based services are utilized for hosting the system and databases, providing scalability, reliability, and cost-effectiveness. And the Networking equipment ensures efficient communication between components, while end-user devices such as computers, tablets, and smartphones provide access to the system.

* **Software**: on our system we used Database management systems for data storage, web servers for hosting web-based interfaces, application servers for business logic processing, and client-side technologies for user interaction.

**4.2.4 Persistent Data Management**

Persistent Data Management refers to the methods and processes used to store and maintain data that exists for prolonged periods, even beyond the duration of the program that created the data. This is crucial for a high school management system as our system needs to maintain records over an extended period.

Here are some key aspects of Persistent Data Management:

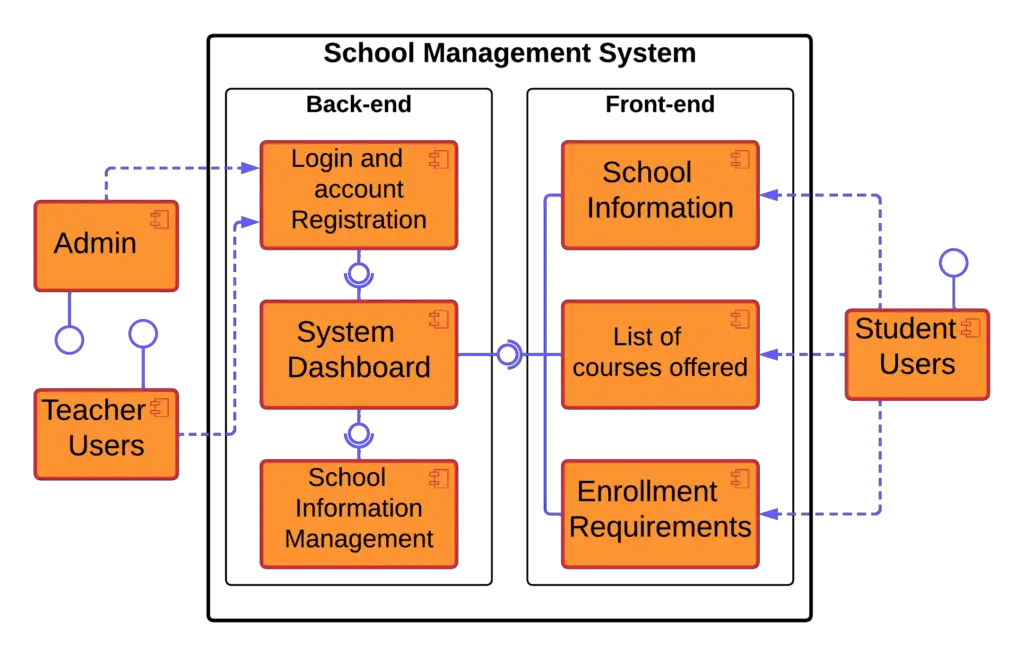
1. **Database Management**: our system uses a reliable database management system (DBMS) to store and manage data. The choice of DBMS (MongoDB) to ensure that data about students like their marks and certification will stay in the database for a longer time.
2. **Data Security**: our system implements robust security measures to protect the data. This includes encryption, access controls, and regular audits.
3. **Backup and Recovery**: we take Regular backups to prevent data loss. A recovery plan is also in place in case of data corruption or loss.
4. **Data Consistency**: our system ensures data consistency. This means that the data be accurate and identical, no matter where it’s accessed from within the system.
5. **Scalability**: The systems data management is scalable. As the number of students increases, the system should be able to handle the increased data load.
6. **Data Archiving**: Old data that’s not frequently accessed is send to archive. This helps improve system performance.

**4.2.5 Component Diagram**

The component diagram for the School Management System is divided into two main sections: Back-end and Front-end.

1. **Back-end Components**:
   * **Login and Account Registration**: This component is connected to the admin user. It handles the authentication and account creation for the system.
   * **System Dashboard**: This component is linked to the Login component. It provides an overview of the entire system.
   * **School Information Management**: This component is connected to both the Dashboard and Teacher users. It manages all the information related to the school.
2. **Front-end Components**:
   * **School Information**: This component is accessible by Student users. It displays all the necessary information about the school.
   * **List of Courses Offered**: This component is linked to the School Information component. It provides a list of all the courses offered by the school.
   * **Enrollment Requirements**: This component is connected to the List of Courses Offered component. It outlines the requirements for enrollment in the various courses.

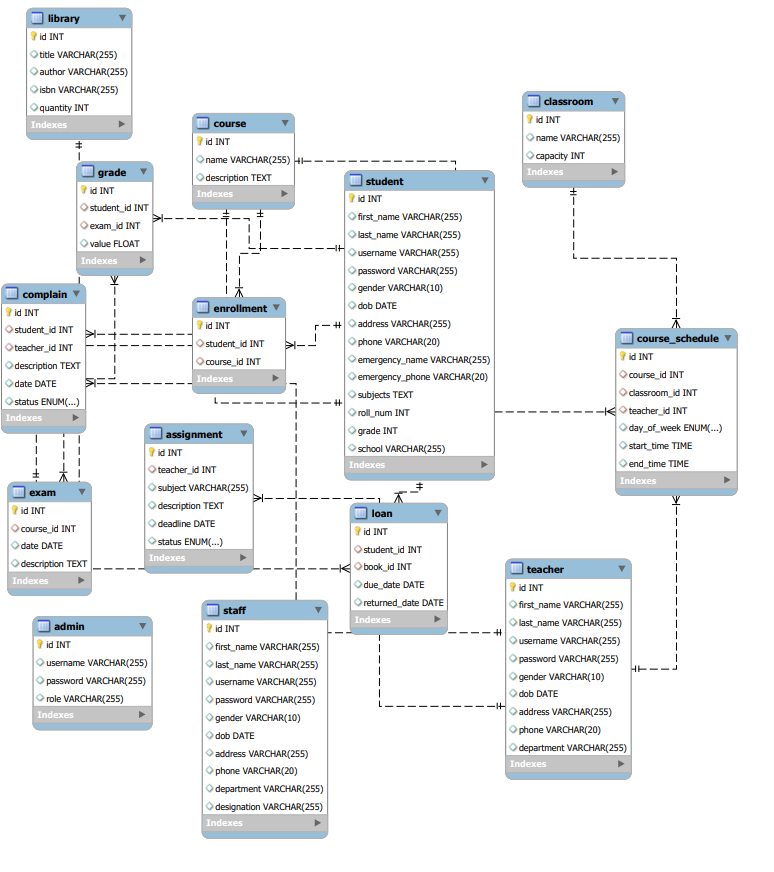
The diagram shows the interaction between the different components and how they serve the Admin, Teacher, and Student users. Each component has a specific role, and they work together to ensure the smooth operation of the School Management System.



**Figure 24: Component Diagram**

**4.2.6 Database Design**

The database design encompasses the schema and relationships between different data entities. It defines the structure of the database tables, attributes, keys, and constraints. Normalization techniques are employed to minimize redundancy and maintain data consistency. Additionally, appropriate indexing strategies are implemented to optimize query performance.



**Figure 25: Database Design**

**4.2.7 Access Control**

We implemented Access control mechanisms to regulate user access to system resources and ensure data security. We employed Role-based access control (RBAC), where users are assigned roles with specific privileges. Authentication and authorization mechanisms authenticate users' identities and enforce access policies based on their roles and permissions.

**4.2.8 User Interface Design**

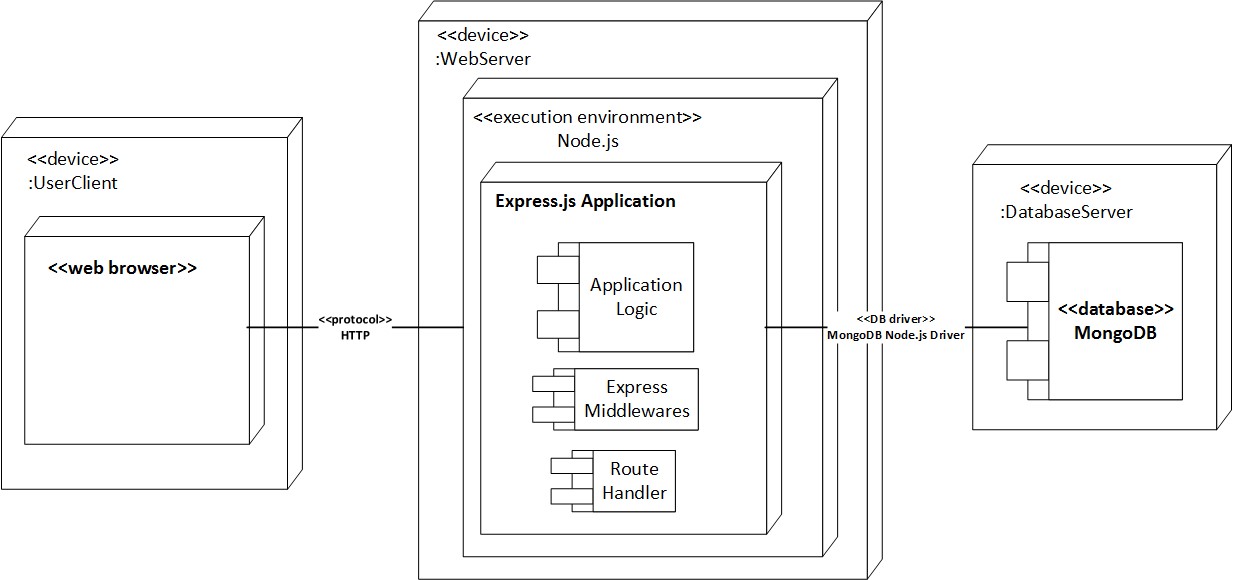
When we worked on User interface design, we mainly focused on creating intuitive and user-friendly interfaces for interacting with the system. Our Considerations include layout, navigation, visual design, and usability. Different user roles may have customized interfaces tailored to their specific needs and preferences. Modern design principles and technologies are utilized to enhance the user experience and optimize workflow efficiency.

By meticulously addressing each aspect of the proposed system architecture, the High School Management System aims to deliver a robust, scalable, and user-centric solution to effectively meet the needs of stakeholders involved in school management, including course management.

**4.2.8 Deployment Diagram**

The deployment diagram illustrates the physical deployment architecture of the high school management system. It showcases how different components of the system are distributed across various nodes, servers, and environments to ensure scalability, reliability, and performance.

1. Client Devices:
   * These represent the end-user devices such as desktop computers, laptops, tablets, and smartphones used by administrators, teachers, students, and parents to interact with the system.
   * The system's user interface is accessed through web browsers installed on these devices.
2. Web Server:
   * The web server node hosts the Node.js runtime environment along with the Express framework.
   * Node.js facilitates the execution of server-side JavaScript code, while Express provides a robust and efficient framework for building web applications.
   * This node serves as the primary entry point for handling incoming HTTP requests from client devices.
3. Load Balancer:
   * In a scalable deployment setup, a load balancer is employed to distribute incoming traffic across multiple instances of the web server for better performance and fault tolerance.
   * The load balancer node ensures that each request is forwarded to an available and healthy web server instance, thereby optimizing resource utilization, and preventing overload on any single server.
4. Database Server:
   * This node represents the backend database server where data related to students, teachers, courses, classes, grades, schedules, and other administrative information is stored.
   * MongoDB or any other suitable database management system is used to manage and store the relational data efficiently.
   * Node.js and Express interact with the database server to perform CRUD (Create, Read, Update, Delete) operations, ensuring data integrity and persistence.
5. External Services:
   * This node encompasses any external services or APIs utilized by the high school management system, such as authentication services, payment gateways, email services, etc.
   * Node.js and Express communicate with these external services to incorporate additional functionalities into the system, such as user authentication, online payments, and communication features.
6. Internet:
   * Represents the network infrastructure that facilitates communication between different nodes and allows client devices to connect to the web server over the internet.
   * The internet acts as the communication medium through which HTTP requests and responses are exchanged between client devices and the web server.



**Figure 26: Deployment Diagram**

**4.2.9 Boundary Conditions**

In this section, we will discuss the steady-state, initialization, and termination states of the high school management system.

A. Initialization

When the high school management system is starting up, it needs to initialize its components and data. This includes:

* Setting up initial configurations such as user roles (admin, teacher, student), academic year, and school details.
* Loading required data like student records, course details, and staff information from the database.
* Establishing connections with external services if needed, such as email servers for notifications or third-party APIs for additional functionalities.

B. Steady-State

During the steady state, the system performs its regular functions such as:

* Allowing users to log in and access their respective modules.
* Teachers can manage courses, input grades, and communicate with students.
* Students can enroll in courses, view their grades, and interact with teachers.
* Admin can oversee the entire system, manage users, and generate reports.

C. Termination

When the system is shutting down or being terminated, it should:

* Save any unsaved data to prevent data loss.
* Release acquired resources such as database connections.
* Perform necessary cleanup tasks like clearing temporary files or caches.

D. Potential Failures and Solutions

1. Hardware Failure
   * Potential Failure: Server crashes, disk failures, or network equipment malfunctions.
   * Solution: Implement redundancy and failover mechanisms. Regularly monitor hardware health and implement proactive maintenance practices.
2. Software Error
   * Potential Failure: Bugs, crashes, or unexpected behavior in the system software.
   * Solution: Conduct thorough testing during development. Implement proper exception handling and error logging mechanisms. Regularly update and patch software components to address known issues.
3. Network Issues
   * Potential Failure: Network outages, slow connections, or packet loss.
   * Solution: Implement redundancy and load balancing for network connections. Use caching mechanisms to reduce reliance on network requests.
4. External Service Failure
   * Potential Failure: Third-party services or APIs become unavailable or experience performance issues.
   * Solution: Implement timeouts and error handling mechanisms when interacting with external services. Consider implementing backup or alternative service providers.
5. Data Corruption
   * Potential Failure: Data becomes corrupted or inconsistent due to software or hardware issues.
   * Solution: Implement regular backups and disaster recovery mechanisms to restore data in case of corruption.
6. Security Breach
   * Potential Failure: Unauthorized access, data breaches, or vulnerabilities in the system.
   * Solution: Implement robust authentication and authorization mechanisms. Use encryption for sensitive data. Conduct regular security audits and penetration testing.

**Chapter 5**

**Implementation**

**5.1 Overview**

The implementation phase of the High School Management System marks the transition from design to actual development, where the envisioned system begins to take shape. This phase is characterized by the translation of system designs and specifications into functional software components tailored to the unique requirements of high school management.

In our project, we've embarked on a journey of exploration, leveraging cutting-edge technologies to ensure the success of our High School Management System. Our implementation strategy encompasses a blend of front-end and backend technologies, along with an Agile approach to development, aimed at delivering a robust and user-centric solution.

Frontend Technologies:

We've harnessed the power of HTML, CSS, JavaScript, and Bootstrap to craft dynamic and visually captivating interfaces for our High School Management System. HTML provides the structural foundation, while CSS and JavaScript enable us to enhance the presentation and interactivity of our web application. Bootstrap, a front-end framework, has been instrumental in facilitating responsive design, ensuring seamless accessibility across devices and screen sizes. This integration of frontend technologies empowers us to create intuitive and visually appealing user interfaces that prioritize user engagement and ease of navigation.

Backend Technologies:

At the core of our backend architecture lies MongoDB, a NoSQL database solution renowned for its flexibility and scalability. MongoDB's document-oriented approach aligns seamlessly with the dynamic nature of educational data, allowing us to efficiently manage diverse datasets within the high school environment. Complementing MongoDB, Node.js serves as our backend runtime environment, offering a robust foundation for server-side logic and API development. Node.js's event-driven architecture enables asynchronous processing, enhancing the scalability and performance of our HSMS backend. Together, MongoDB and Node.js form a formidable backend stack that empowers us to handle complex data operations with agility and efficiency.

Agile Development Methodology:

Embracing the Agile methodology, we prioritize collaboration, adaptability, and incremental development in our project workflow. By breaking down the development process into iterative sprints, we facilitate continuous feedback loops and stakeholder engagement, ensuring that our HSMS evolves in tandem with evolving requirements and user needs. Agile principles guide our development practices, emphasizing transparency, flexibility, and customer-centricity. This iterative approach allows us to respond swiftly to change, incorporate stakeholder feedback, and deliver tangible value to end-users at each iteration.

In the upcoming subtopics, we'll delve deeper into the implementation and prototype of our High School Management System, exploring coding standards, development tools, and the intricate details of our system architecture. Through a combination of innovative technologies and Agile methodologies, we're poised to revolutionize high school management and empower educational institutions with a dynamic and efficient management solution.

**5.2 Coding Standard**

In our project, we prioritize maintaining a meticulous coding standard to uphold clarity, consistency, and longevity within our codebase. We believe in crafting code that is not only easy to read and understand but also simple to maintain and build upon in the future. To achieve this, we follow a set of guidelines and practices that govern everything from naming conventions to documentation strategies.

One key aspect of our coding standard is our approach to naming conventions. We ensure that all variables, functions, classes, and other elements are named in a descriptive and meaningful manner, facilitating quick comprehension of the code's purpose and functionality. Additionally, we emphasize organizing our code logically, breaking it down into modular components and adhering to best practices for file organization and project structure. This structured approach enhances navigability and maintenance of our codebase as it evolves over time, promoting collaboration and reducing the risk of errors.

**5.3 Development Tools**

In our project, we make use of some handy tools to help us work better and faster. First off, we use GitHub, a platform where we can store our code, keep track of changes, and work together with our team. Then, there's Visual Studio Code, or VSCode for short, which is like our digital workspace where we write and edit our code. It's got lots of helpful features that make coding easier, like highlighting mistakes and suggesting fixes.

On the creative side, we use front-end technologies such as HTML, CSS, JavaScript, and Bootstrap. These are what we use to build the parts of our project that people see and interact with, like buttons, menus, and forms. For the behind-the-scenes stuff, we've got Node.js and MongoDB. They help us manage data and make sure everything runs smoothly behind the scenes. By using these tools together, we can work together better, write better code, and make sure our project turns out just right.

**5.4 Prototype**

In the Prototype phase of our High School Management System project, we developed an initial version of the system to demonstrate key features and functionalities. Our prototype focused on providing a user-friendly interface for managing student information, schedules, and academic records.

The prototype showcased the main dashboard, where users could access different modules such as student profiles, class schedules, and assignment submissions. We implemented basic CRUD (Create, Read, Update, Delete) functionality for managing student records, allowing administrators to add new students, update their information, and delete records as needed.

Additionally, we integrated features for managing class schedules, allowing administrators to create and modify schedules for various classes and subjects. The prototype also included a feature for teachers to upload and distribute assignments to students, with options for students to submit completed assignments electronically.

While the prototype provided a basic framework for the High School Management System, further development and refinement were needed to fully implement all planned features and functionalities. Feedback from stakeholders and users during the prototype testing phase was invaluable in guiding our development process and prioritizing features for future iterations of the system.  
  
**5.5 Implementation Details**

We provide a comprehensive overview of the development and deployment process of our High School Management System. Our team utilized a variety of technologies and tools to bring the project to fruition, ensuring a seamless and efficient implementation.

Our development process followed agile methodologies, allowing for iterative development and continuous feedback loops. We adopted a collaborative approach, utilizing communication tools such as Slack and Trello to facilitate coordination among team members and track project progress.

Technically, our system architecture involved the integration of front-end technologies such as HTML, CSS, JavaScript, and Bootstrap for the user interface, coupled with back-end technologies including Node.js and MongoDB for server-side logic and data management. This architecture provided a robust foundation for our system, enabling scalability, performance, and flexibility.

Throughout the implementation phase, we prioritized testing methodologies to ensure the quality and reliability of the system. This included rigorous unit testing, integration testing, and user acceptance testing to identify and address any issues or bugs.

Additionally, we implemented various performance optimizations and security measures to enhance the system's efficiency and protect sensitive data. This included measures such as caching, compression, encryption, and role-based access control.

Overall, the Implementation Details section provides a detailed breakdown of the technical aspects of our High School Management System's development and deployment, highlighting our team's expertise, strategic decision-making, and commitment to delivering a high-quality solution.

**Chapter 6**

**System Testing**

**6.1 Introduction**

Software testing is a critical element of software quality assurance and represents the ultimate review of system requirement specification document, system design and programming aspects.

**6.2 Scope**

The scope of this project is to develop a comprehensive High School Management System that facilitates various administrative tasks, student management, teacher management, and academic operations. The system will consist of a web application accessible to administrators, teachers, and students, providing features for enrollment, attendance tracking, grading, scheduling, and generating reports.

**6.3 Resources**

System testing relies on various resources, including documentation such as the Software Requirements Specification (SRS) document, design specifications, and test plans. These resources outline the expected behavior of the system and serve as a reference for creating test cases and evaluating the system's performance.

**6.4 Schedule**

Scheduling is the time arrangements given by the responsible body of the system to perform whatever activities designed.

|  |  |  |
| --- | --- | --- |
| Types of Testing | Date of Testing | Tested by |
|  |  |  |
| Unit testing | pending | Fahmi Dinsefa |
|  |  |  |
| Integration testing | Pending | Nebiyu Musbah |
|  |  |  |
| System Testing (Installation  Testing) | Pending | Abel Gezu |

**6.4.1 Features to Be Tested (Prototype Stage)**

As a prototype development stage, and considering the time constraints, certain unimplemented features and non-functional requirements will not undergo testing.

User Registration and Authentication: Test the functionality of user registration and authentication processes to ensure secure access to the system.

Student Enrollment: Test the system's ability to enroll students into courses and manage their information accurately.

Attendance Tracking: Verify the system's capability to track and manage student attendance records efficiently.

Grading: Test the automated grading system to ensure accurate and reliable grading of student assignments and exams.

Class Scheduling: Validate the scheduling feature's ability to create and manage class schedules for teachers and students effectively.

Report Generation: Test the system's capability to generate reports such as student progress reports, attendance reports, and grade reports accurately.

# 6.5 TESTING SCENARIO

There are several techniques for testing, but the two key techniques will be evaluated which comprises of white- box and black-box testing.

# 6.5.1 TESTING TECHNIQUES

# a) BLACK BOX TESTING

While performing black box testing, a tester must know the system design and will not have access to the source code. Black box testing also referred to as functional or behavioral testing, it is used to test the functionalities of the system focusing on its functional requirements. It derives some inputs that will exercise the functional requirements of the system. This type of testing is termed black box testing because no knowledge of the workings of the program is used as part of the testing.

Specification

Desired Result

Test Data

Execution

Output

Match?

Yes, Goto next Test.

No

Implementation

Debug

**Figure 27: Black Box Testing Approach**

# 

# b). WHITE BOX TESTING

we used white box testing also referred to as **structural or glass box testing** to test methods and structures of a program, in this technique we tested case design that uses the control structure of the procedural design. Using this test case we tested entities like:

* + Test all independent paths.
  + Test logical decisions (true/false)
  + Test loops (boundaries and operational bounds)

Validate internal data structures.

Program Source

Insufficient

Test Data

Coverage?

Sufficient, Done

Implementation

**Figure 28: White Box Testing Approach**

# JUSTIFICATION OF TESTING TECHNIQUE

**The technique chosen for testing our High School Management System is the black box testing because.**

* Black Box can be done by someone with little or no experience in programming.
* The software developed has a collaborative interface and all the bugs detected by the compiler within the source program are debugged and a thorough compilation process has been conducted. As such, **White Box Testing is Irrelevant.**
* Black box testing’s main efforts are to do the following key activities:
* Find errors in an incorrect or missing function.
* Interface errors
* Errors in data structures or external database access
* Behavior or performance errors
* Initialization and termination errors.

.

**6.6 Pass/fail criteria.**

Is the situation in which the pass and fail criteria of the system are determined using standard terminologies of pass/fail.

**6.7 Test case specification**

Table below shows the test specifications of functional requirements used to write the test cases along with Black Box Testing Information with test case, result and log**.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S/N | Function (Test  Case) | Input (Test Data) | Output  (Expected result) | (Test log) |
| 1. | Login using valid.  password | Type username Type password Press the Sign in button | Login successful, proceed to Dashboard | Successful |
| 2. | Login using invalid password | Wrong email Wrong password | Incorrect Sign in | Successful |

**6.9 Estimated Risk and Contingency Plan**

**6.9.1 Estimated Risks**

- Lack of experience in testing

- Underestimation of the time required for software development

**6.9.2 Contingency Plan**

- Allocate sufficient time for testing activities to ensure thorough evaluation of the system

- Identify and acquire necessary testing tools and resources

- Regularly analyze and reassess project timelines to mitigate potential delay

**References**

<https://humansofdata.atlan.com/2018/11/myths-paper-based-data-collection/>

<https://www.google.com/search?q=traditional+manual+methods+and+basic+tools+for+managing+student+information>

<https://iq.opengenus.org/system-design-of-school-management-software/>

<https://www.google.com/search?sca_esv=6df0982588bcee2b&sxsrf=ACQVn0_4bJJP5ctEYVuUJHdQPQx92fmCSA:1708435869027&q=Scrum+methodology&tbm=isch&source=lnms&sa=X&ved=2ahUKEwje4ODIg7qEAxULV6QEHUppAJQQ0pQJegQIDBAB&biw=922&bih=665&dpr=1.5#imgrc=528ezLqsaC8qIM>

<https://www.matellio.com/blog/develop-school-management-system-software/>

<https://www.lucidchart.com/pages/>

<https://creately.com/diagram/example/ik43pck9/simple-client-server-communication-classic>

<https://stackoverflow.com/questions/35834262/having-trouble-with-deployment-diagram>

<https://app.creately.com/d/start/dashboard>

<https://creately.com/diagram-community/popular/t/state-chart>

<http://www.plantuml.com/plantuml/uml/ZPHTJiCm3CVVSueyRaAzG1-cZJ4u0700qvWkKkEW96L8mDt9HpOvsbJnOAbnVlxx8rQtFe0Bivc4cF0zo62bq-CnIAKT3a5R4cAOm7lvwj79Ro7btHo_20psqWUdQKo-ZuYSh5FSDzbHqshTopTh9mGIlsUjvp0hf53aa9oTfjKwVNzgF3qvKEZZ-OA_80p7S_QG3eMETtgap7Dyfumc7Uk0O6iGA9NgBs4624RSszbvR3mfcZiENs2uS0d9YfMeZJHHjTehfQQbd8ih7V8KJ1JZM8ehbCF1EiLdWpGRcHsPs4MxZxQFzX3jCJ7vqRhkPtCUEVFKEfcd6PmGDIJUvPNeDdK0lSHBiQAgt83oGwLLuB89LFP4tQMYNe9I2rXDi8YnhE2z7JJapRsDbjd_XypZRq5HybntmIaEjJrmA8alrv-9vKovbsk6lr0hKpQcbuEb09gy45iabNu0Vm00>

<https://online.visual-paradigm.com/diagrams/templates/deployment-diagram/>

<https://www.lucidchart.com/pages/how-to-draw-component-diagram-in-UML>

<https://www.google.com/search?q=Scrum+methodology&oq=Scrum+methodology&gs_lcrp=EgZjaHJvbWUyCQgAEEUYORiABDIMCAEQIxgnGIAEGIoFMgwIAhAAGBQYhwIYgAQyBwgDEAAYgAQyBwgEEAAYgAQyBwgFEAAYgAQyBwgGEAAYgAQyBwgHEAAYgAQyBwgIEAAYgAQyBwgJEAAYgATSAQc3OTRqMGo3qAIAsAIA&sourceid=chrome&ie=UTF-8>

<https://humansofdata.atlan.com/2018/11/myths-paper-based-data-collection/>