**Summer Training Project**

**SCHOOL MANAGEMENT SYSTEM**

**Submitted By**

**SAHIL MANAV**

**50128102021**

**Batch 2021-24**

Under the guidance of

**Mr. Davinder singh**



**Kamal Institute of Higher Education and Advance Technology**

**Affiliated to Guru Gobind Singh Indraprastha University**

***K-1 Extension, Mohan Garden***

***New Delhi 110059***

**CERTIFICATE**

This is to certify that the summer training project work entitled “ Report on school management system ” is Bonafide work carried out by **MR. SAHIL MANAV** in partial fulfillment of Bachelor of computer application at Kamal Institute of Higher Education and Advanced Technology affiliated to Guru Gobind Singh University, Delhi during the year 2021-2024. It is certified that all the correctness/suggestions indicated for internal assessment have been incorporated in the report submitted. The project report has been approved as if satisfys the academics requirements in respect of project of work prescribed for the said degree.

**ACKNOWLEGEMENT**

We are overwhelmed in all humbleness and gratefulness to acknowledgement my depth to all those who have helped us to put these ideas, well above the level of simplicity and into something concrete.

We would like to express our special thanks of gratitude to our esteemed guide , **MR. Davinder Singh** who gave us the golden opportunity to do this wonderfull project which also helped us in doing lots of research and we come to know about so many new things .We are thankful to them.

Any attempts at any level cannot be satisfactory completed without the support and guidance of our parents and friends.

We would like thank our parents who helped us a lot in gathering different information , collecting data abd guiding us from time to time in making this project ,despite of their busy schedules, they gave us different ideas in making this project unique.

SAHIL MANAV

50128102021

**DECLARATION**

Do hereby declare the project work entitled

**SCHOOL MANAGEMENT SYSTEM** submitted by me for the partial fulfillment of the requirement for the award of bachelor in computer applications is a record of my own research work . The report embodies the finding based on my study and observation and has not been submitted earlier for the award of any degree to any Institute or University.

Date:

Student Name: SAHIL MANAV

Enrollement No: 50128102021

## INDEX

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TOPIC** | **Pg.No** |
| **●** | **Certificate** | **2** |
| **●** | **Acknowledgment** | **3** |
| **●** | **Declaration** | **4** |
| **●** | **Abstract** | **5** |
| **1** | **PROBLEM FORMULATION** | **8** |
| **1.1** | Introduction about the Company | 8 |
| **1.2** | Introduction about the Problem | 9 |
| **1.3** | Present State Of Art | **10** |
| **1.4** | Need of Computerization | **12** |
| **1.5** | Proposed Software/Project | **13** |
| **1.6** | Importance of work | **14** |
| **2** | **SYSTEM ANALYSIS** | **15** |
| **2.1** | Feasibility Study | **15** |
| **2.1.1** | Technical Study | **16** |
| **2.1.2** | Economical Feasibility | **16-17** |
| **2.1.3** | Operational Feasibility | **17-18** |
| **2.1.4** | Other Feasibility Dimensions | **18-19** |
| **2.2** | Analysis Methodology | **20** |
| **2.3** | Choice of the Platforms | **22** |
| **2.3.1** | S/W used | **23** |
| **2.3.2** | H/W used | **23** |
| **3** | **SYSTEM DESIGN** | **24** |
| **3.1** | Design Methodology | **24** |
| **3.2.1** | ERD | **27** |

|  |  |  |
| --- | --- | --- |
| **3.2.2** | DFD | **28** |
| **3.3** | Input Design | **29** |
| **3.4** | Output Design | **32** |
| **3.5** | Code Design and Development | **35** |
| **4** | **TESTING AND IMPLEMENTATION** | **37** |
| **4.1.1** | Unit Testing | **38** |
| **4.1.2** | Module Testing | **38-39** |
| **4.1.3** | Integration Testing | **39** |
| **4.1.4** | System Testing | **39** |
| **4.1.5** | White Box Testing | **40** |
| **4.1.6** | Acceptance Testing | **40-42** |
| **4.2** | Test date test cases | **43-45** |
| **4.3** | Test Report and Debugging | **46-51** |
| **4.4** | Implementation Manual | **52-54** |
| **4.5** | Implementation | **55-56** |
| **4.5.1** | User’s Training | **57** |
| **4.6** | Post Implementation | **58** |
| **5** | **CONCLUSION AND REFERENCES** | **59** |
| **5.1** | Conclusion | **59** |
| **5.2** | System Specifications | **60** |
| **5.2.1** | H/W Requirement | **60** |
| **5.2.2** | S/W Requirement | **61** |
| **5.3** | Limitations of the System | **61** |
| **5.4** | Future Scope for Modification | **62-63** |
| **5.5** | References/Bibliography | **64** |
| **6** | **ANNEXURES** | **65** |
| **6.1** | Menu Flow Diagram | **65** |
| **6.2** | Structure Chart | **66** |
| **6.3** | Sample Inputs | 66 |
| **6.4** | Sample Outputs | **67-68** |
| **6.5** | Coding | **69-76** |

**Main Report Of School Management System**

**CHAPTER-1**

**PROBLEM FORMULATION**

**1.1 Introduction About The Company**

A school management system is a comprehensive software or digital platform designed to streamline and enhance the administrative and academic operations of educational institutions, ranging from primary schools to universities. It serves as an all-encompassing tool that unifies various aspects of school management, including student records, attendance, timetables, examination schedules, financial management, and communication between teachers, students, and parents. This technology-driven solution has revolutionized the way educational institutions operate, making processes more efficient, reducing paperwork, and allowing educators to focus on their core mission: providing high-quality education. By automating and centralizing many administrative tasks, school management systems empower schools to make data-driven decisions, improve communication, and ultimately create a more productive and organized learning environment.

**1.2 Introduction About The Problem**

School management systems, while highly beneficial, are not without their challenges. One significant problem is the initial cost of implementation, which can be a considerable burden for schools with limited budgets. Additionally, the need for consistent updates and maintenance can strain resources over time. Data security is another pressing concern, as these systems often contain sensitive student and staff information, making them potential targets for cyberattacks. User resistance can be a problem too, as not all staff may be comfortable with or accustomed to using technology, leading to adoption issues. Moreover, customizing the system to meet the unique needs of each educational institution can be complex, potentially resulting in inefficiencies and dissatisfaction. Overall, the successful implementation of a school management system requires careful planning, financial commitment, and a focus on addressing these challenges to fully realize the system's benefits.

**1.3 Present State Of Art**

The state of the art in school management systems had seen significant advancements, and it's likely that these trends have continued to evolve. Here are some aspects of the present state of school management systems:

1. **Cloud-Based Solutions:** School management systems have increasingly moved to cloud-based platforms. This provides easy accessibility, scalability, and better data security. It allows administrators, teachers, students, and parents to access information and perform tasks from any device with an internet connection.
2. **Mobile Applications:** Many school management systems now offer mobile apps, making it convenient for parents and students to stay updated on grades, attendance, and announcements. Teachers can also manage tasks on the go.
3. **Data Analytics:** Advanced analytics features are being integrated, allowing schools to gain insights from data, track student performance more effectively, and make data-driven decisions. Predictive analytics may also be used to identify students at risk of falling behind.
4. **Integration with Learning Management Systems (LMS):** To streamline the teaching and learning process, modern school management systems often integrate with learning management systems. This allows for the creation and distribution of digital content and assignments.
5. **Parent-Teacher Communication:** Enhanced communication features, including messaging, notifications, and scheduling parent-teacher conferences, are common in contemporary systems, fostering better engagement and collaboration.
6. **Financial Management:** School management systems have improved in terms of financial management, including features for fee collection, expense tracking, and budget management.
7. **Security and Privacy:** With growing concerns about data security and privacy, school management systems are emphasizing robust security measures to protect sensitive student and staff information.
8. **AI and Automation:** Some systems are starting to incorporate artificial intelligence and automation to streamline administrative tasks, reducing the workload on staff and enabling more efficient management.
9. **Customization:** Modern systems often allow schools to customize their software to meet their specific needs, whether they are primary schools, high schools, or universities.
10. **User-Friendly Interfaces:** Usability has improved, with user-friendly interfaces to cater to a wide range of users, including those who may not be tech-savvy.

**1.4 Need Of Computerization**

Computerization of a school management system offers numerous advantages that can significantly improve the efficiency and effectiveness of school operations. Here are some key points highlighting the need for computerization in school management:

* **Efficient Data Management:** Computerization allows schools to store and manage student records, attendance, academic performance, and administrative data in a digital format, reducing paperwork and manual data entry errors.
* **Streamlined Communication:** Digital systems enable easy communication between teachers, students, parents, and administrative staff through emails, messaging apps, and online portals, improving the flow of information.
* **Attendance Tracking:** Automated attendance systems can accurately monitor student attendance, reducing the chances of errors and helping in addressing truancy issues promptly.
* **Academic Performance Analysis:** Computerized systems can generate detailed reports on students' academic performance, making it easier for educators and parents to identify strengths and weaknesses.

**1.5 Proposed Software / Project**

A proposed school management software would be a comprehensive digital platform designed to streamline and enhance various aspects of school administration. This software would encompass modules for student enrollment, attendance tracking, academic performance monitoring, resource management, and financial administration. It would also facilitate communication between teachers, students through an online portal, ensuring transparency and efficiency in school operations. The software would be secure, user-friendly, and customizable to meet the specific needs of individual schools, ultimately improving overall school management and enhancing the learning experience for both students and staff.

**1.6 Importance Of Work**

Travelling The work of school management holds immense importance in the education system as it plays a pivotal role in shaping the learning environment, ensuring smooth operations, and ultimately affecting the quality of education provided. School management is responsible for a range of crucial tasks that directly impact students, teachers, and the overall institution.

First and foremost, school management is responsible for creating and maintaining a safe and conducive learning environment. This involves ensuring the physical infrastructure, security measures, and facilities are in place to support both the educational and well-being needs of students.

Additionally, effective school management oversees the recruitment and retention of qualified and motivated educators, as well as their professional development. It is vital in creating a positive work environment for teachers and staff, which directly influences the quality of education delivered.

**CHAPTER -2**

**SYSTEM ANALYSIS**

**2.1 Feasibility Study:**

A feasibility study for a school management system is a critical initial step in the process of determining the practicality and viability of implementing such a system within an educational institution. This study encompasses a comprehensive assessment of several key dimensions.

**2.1.1 Technical Study:**

 A technical study of a school management system is a critical phase in the development and implementation of such a system. This study delves into the technical intricacies and requirements necessary for a successful deployment. It involves several key components.

First, the study assesses the technological infrastructure of the school, considering factors such as hardware, network capacity, and existing software. It aims to determine if the current technology resources are adequate to support the proposed system or if upgrades are necessary.

Integration with existing systems is another vital aspect of the technical study. It involves identifying how the new school management system will interface with other systems already in use within the institution, such as attendance tracking, library management, or learning management systems. Compatibility and data exchange protocols need to be established.

Scalability is a significant consideration. The system must be capable of accommodating potential growth in the number of students, faculty, and staff. This involves assessing the system's ability to handle increased data load and user traffic.

**2.1.2 Economical Feasability**

Economic feasibility is a crucial aspect of assessing the viability of implementing a school management system. This study focuses on financial aspects to determine whether the investment is justifiable and economically beneficial.

The initial step in economic feasibility is to calculate the costs associated with the implementation of the school management system. This includes expenses like software development or licensing fees, hardware procurement or upgrades, staff training, and ongoing maintenance costs. It is essential to account for both one-time expenditures and recurring operational expenses.

In contrast to the costs, the study also assesses the potential benefits. A school management system can lead to several cost savings and revenue generation opportunities. These may include reduced administrative overhead through automation, improved resource allocation, enhanced efficiency, and the potential to increase enrollment through enhanced communication and service quality. The economic feasibility study should provide a comprehensive analysis of these benefits and their financial impact over time.

**2.1.3 Operational Feasability**

Operational feasibility is a crucial component in the assessment of whether a school management system is a viable solution for an educational institution. This aspect of the feasibility study examines whether the proposed system can be effectively integrated into the day-to-day operations of the school.

First and foremost, operational feasibility evaluates the readiness of the school staff and administration to adapt to the new system. It considers the willingness and capacity of the employees to learn, use, and effectively manage the system. Resistance to change or a lack of necessary skills may hinder the success of the implementation.

Additionally, the study takes into account the potential disruptions that may occur during the implementation phase. These disruptions can range from temporary system downtime to changes in workflow processes. Mitigation strategies should be outlined to minimize these disruptions and ensure a smooth transition.

**2.1.4 Other Feasability Dimensions**

In addition to technical, economic, and operational feasibility, several other dimensions should be considered in the feasibility study of a school management system.

Legal and Regulatory Feasibility: This dimension assesses whether the proposed system complies with all relevant laws, regulations, and industry standards. Data privacy and security regulations, in particular, are critical, as educational institutions handle sensitive student information.

Schedule Feasibility: This examines the proposed timeline for the implementation of the system. It assesses whether the project can be completed within a reasonable timeframe, factoring in any potential delays or resource constraints.

Cultural and Social Feasibility: Cultural and social factors can significantly influence the success of a school management system. This dimension evaluates the readiness of the school community, including students, teachers, and parents, to accept and adapt to the new technology.

Environmental Feasibility: Environmental considerations may involve assessing the ecological impact of implementing a new system. For example, reducing paper usage through digital processes can be more environmentally friendly.

Vendor or Solution Selection Feasibility: If opting for a commercial school management system, the feasibility study should analyze different vendors or solutions to select the one that best aligns with the school's specific requirements, budget, and support services.

Training and Capacity Building Feasibility: Assessing whether the necessary training and capacity-building programs can be established to ensure that staff and stakeholders can effectively use the system.

Maintenance and Support Feasibility: Determining whether there are adequate resources and support mechanisms in place for maintaining and troubleshooting the system over time.

Change Management Feasibility: Evaluating the strategies and plans for change management, which may include addressing staff concerns, providing ongoing support, and fostering a culture of continuous improvement.

**2.2 Analysis Methodology**

The analysis methodology for a school management system involves a systematic approach to assess the needs, requirements, and feasibility of the system. This process is critical for understanding what the system should achieve and how it will align with the school's goals and constraints.

1. **Needs Assessment:** The analysis begins with a thorough needs assessment, involving consultations with various stakeholders, including school administrators, teachers, students, and parents. Surveys, interviews, and workshops can be conducted to identify the specific challenges and opportunities that the system should address.
2. **Requirements Gathering:** Once the needs are identified, the next step is to gather detailed requirements. This involves creating a comprehensive list of functional and non-functional requirements for the system. Functional requirements outline what the system should do (e.g., attendance tracking, grade management), while non-functional requirements specify how it should perform (e.g., security, scalability).
3. **Feasibility Study:** As discussed in previous responses, a feasibility study should be conducted to determine the technical, economic, operational, legal, and other feasibility dimensions of the proposed system. This study helps assess whether the system is a practical and viable solution for the school.
4. **System Design:** Following feasibility, the system design phase involves creating a detailed design of the system architecture, data models, user interfaces, and workflows. This stage outlines how the system will be structured to meet the identified requirements.
5. **Prototype or Mockups:** Creating a prototype or mockups of the user interface can help stakeholders visualize the system's appearance and navigation. This can be a valuable step to ensure that the design aligns with user expectations and needs.
6. **Development or Vendor Selection:** Depending on the project scope and budget, the school may choose to develop the system in-house or select a commercial solution. The methodology should outline the criteria for vendor selection or the development process.
7. **Testing and Quality Assurance:** Rigorous testing, including unit testing, integration testing, and user acceptance testing, is essential to ensure that the system functions as intended and is free of errors or bugs.
8. **Deployment and Training:** Implementing the system involves deploying it in the school environment. Comprehensive training programs should be provided to familiarize staff and users with the system's features and functionalities.

**2.3 Choice Of The Platform**

* VISIUAL STUDIO CODE
* SERVER XAMP

**System Requirement Specification**

**2.3.1 Software used:**

* WINDOWS OS
* Visual Studio
* XAMP Server
* SQL server

**2.3.2 Hardware used:**

* Laptop or pc
* Ram 6GB or higher
* 100GB ROM or higher

**CHAPTER-3**

**SYSTEM DESIGN**

**3.1 Design Methodology**

**3.2 Database Design**

**3.2.1 ERD**

**3.2.2 DFD**

**3.3 Input Design**

**3.4 Output Design**

**3.5 Code Design and Development**

**3.1 Design Methodology**

Designing a school management system requires a well-structured methodology to ensure the system meets the specific needs of the educational institution. Here is a design methodology for a school management system:

1. **Project Initiation:**

Define the project scope, objectives, and goals. Determine the specific functionalities and features required, such as student enrollment, attendance tracking, grade management, resource allocation, and communication tools.

1. **Requirements Analysis:**

Collaborate with stakeholders, including school administrators, teachers, students, and parents, to gather detailed requirements.

Identify both functional and non-functional requirements, specifying what the system should do and how it should perform.

1. **System Architecture Design:**

Develop a high-level system architecture that outlines the overall structure of the system. Define the components, modules, and data flow within the system.

Select appropriate technologies and frameworks to support the architecture.

1. **Database Design:**

Create a detailed database schema to define the structure of the database, including tables, relationships, and data fields.

Choose a suitable database management system (e.g., MySQL, PostgreSQL, MongoDB) based on the data requirements.

1. **User Interface (UI) Design:**

Design the user interface to ensure it is user-friendly, intuitive, and visually appealing.

Create wireframes or prototypes to visualize the system's interface and navigation.

1. **Functional Module Design:**

Divide the system into functional modules, such as student information management, attendance tracking, academic performance, and communication.

Develop detailed specifications for each module, defining the inputs, outputs, and processing logic.

**3.2 Database Design**

Designing the database for a school management system requires careful consideration of the data that needs to be stored and how it should be structured to support the system's functionality. Here's a basic database design for a school management system:

**Entities and Tables:**

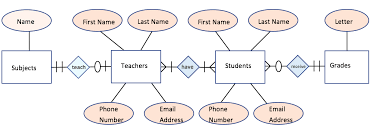
1. **Students:**

* Student ID (Primary Key)
* First Name
* Last Name

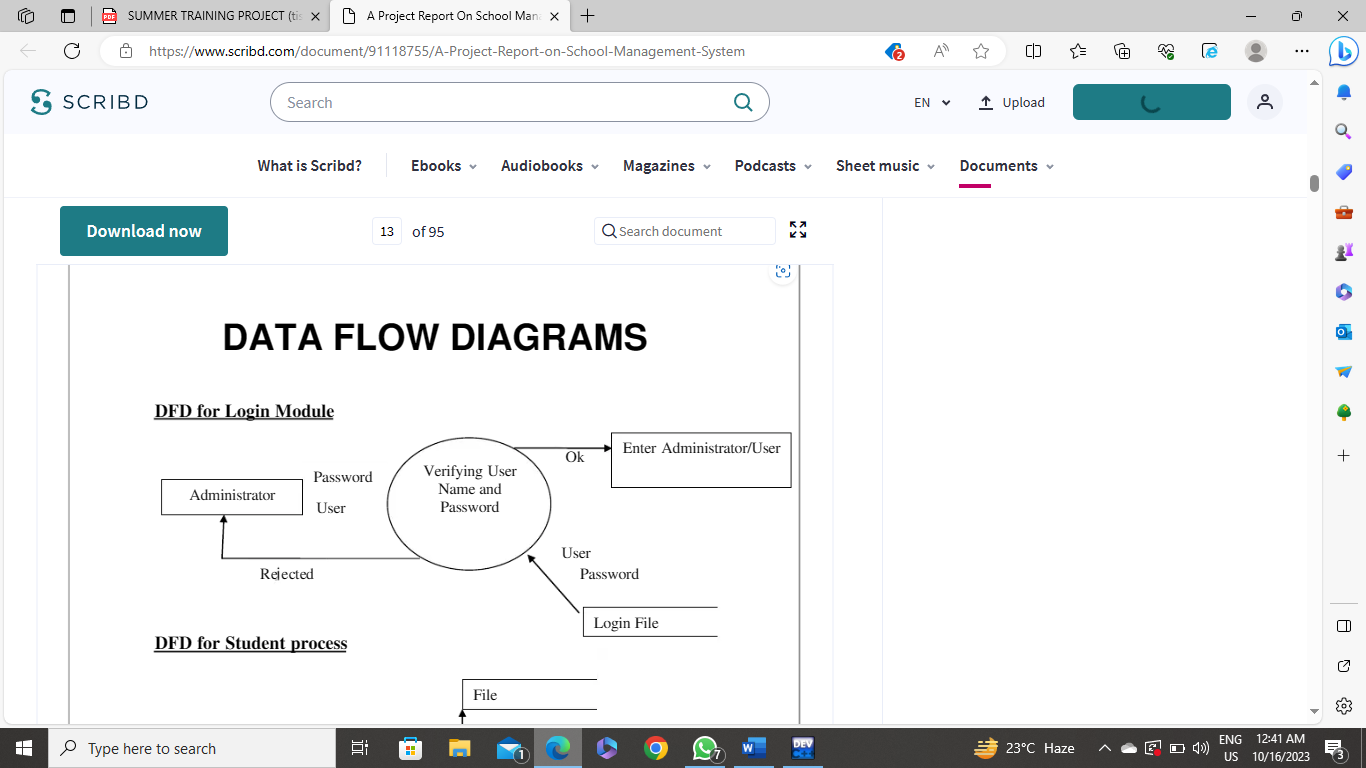
1. **Teachers:**

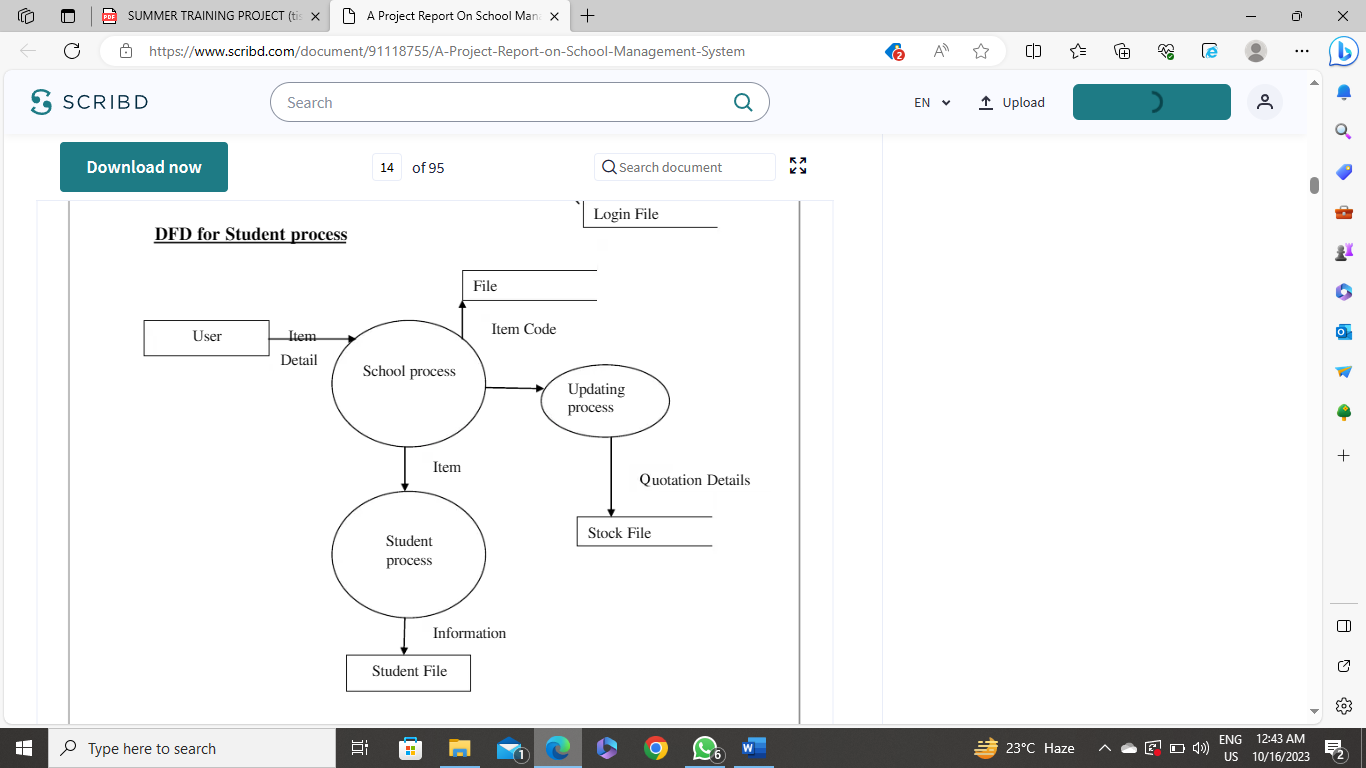
* Teacher ID (Primary Key)
* First Name
* Last Name

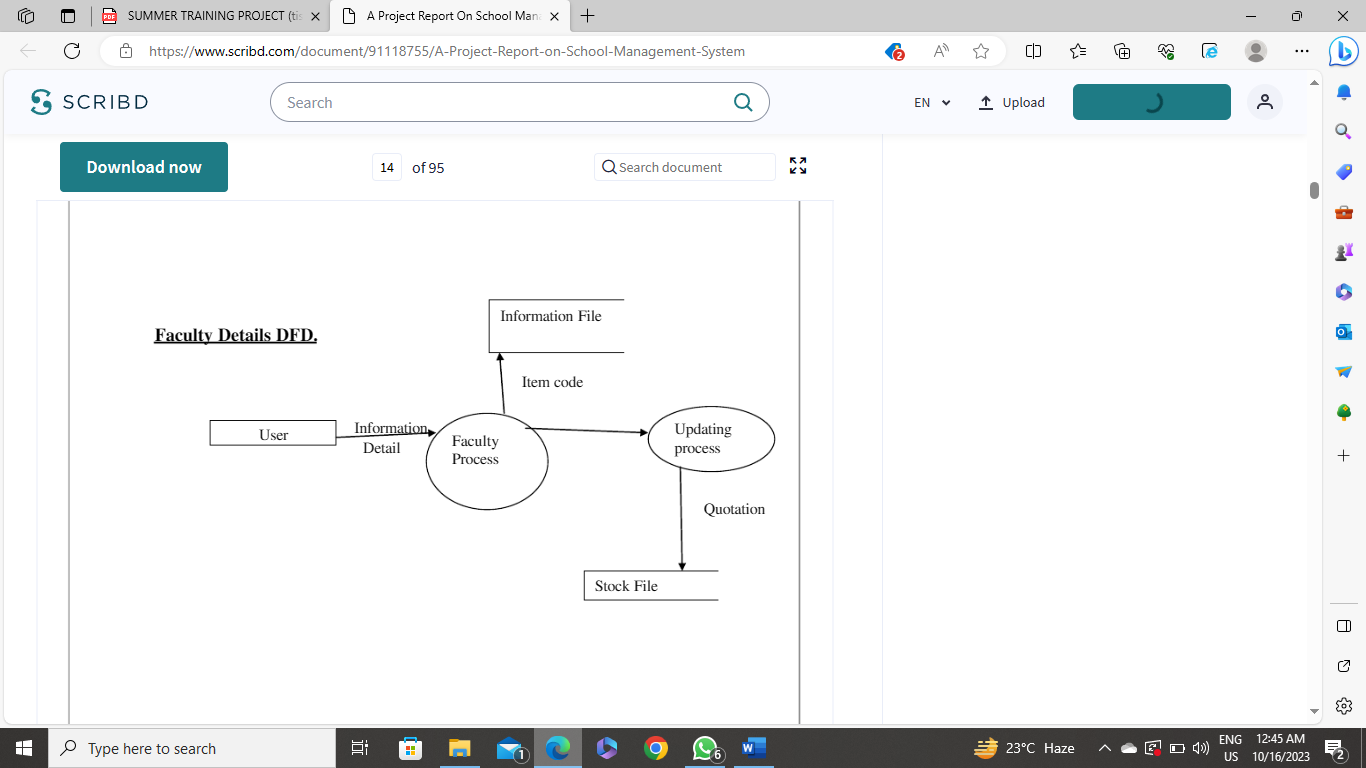
**3.2.1 ERD**

****

**3.2.2 DFD**







**3.3 Input Design**

Designing the input system for a school management system is a crucial aspect of the software development process. It involves collecting, processing, and storing data from various stakeholders, including students, teachers, parents, and administrators. Below, I'll outline the key components and considerations for designing the input system of a school management system:

1. User Authentication:

Implement a secure user authentication system to ensure that only authorized individuals can access and input data into the system.

Define user roles (e.g., student, teacher, administrator) and permissions to control access levels.

1. Student Information:

Create forms for entering and updating student information, including:

Personal details (name, date of birth, contact information)

Enrollment details (class, section, roll number)

Emergency contact information

Medical records

Previous educational history

1. Teacher Information:

Develop forms for entering and managing teacher information, including:

Personal details (name, contact information)

Qualifications and certifications

Employment history

Class assignments and subject expertise

1. Parent/Guardian Information:

Capture information about students' parents or guardians, including:

Contact details

Emergency contact information

Relationship to the student

1. Attendance:

Create a system for recording daily attendance for students and teachers.

Allow teachers to mark attendance for their classes.

Generate reports and alerts for absent students or teachers.

1. Timetables:

Implement a timetable creation system to schedule classes, exams, and other school activities.

Ensure that the system can handle conflicts and changes to the timetable.

1. Courses and Subjects:

Design forms for adding and managing courses, subjects, and curriculum.

Associate subjects with teachers and classes.

1. Exams and Grading:

Create tools for entering exam schedules, grading student performance, and calculating GPAs.

Store exam results and make them accessible to students and parents.

1. Communication:

Implement features for sending and receiving messages, notifications, and updates to and from students, teachers, and parents.

Enable real-time communication or messaging within the system.

10.Finance and Fee Management:

Develop a module for fee collection, including fee schedules, payment tracking, and receipts.

Generate financial reports for school administrators.

**3.4 Output Design**

The output design of a school management system is crucial for presenting information to users in a clear, organized, and user-friendly manner. Effective output design ensures that users can access and interpret the data and reports generated by the system easily. Here are key considerations for designing the output of a school management system:

User Roles and Access Control:

Tailor the output to the specific needs and permissions of different user roles (e.g., administrators, teachers, students, parents).

Ensure that users can only access data and reports relevant to their roles and responsibilities.

Dashboard:

Create a personalized dashboard for each user, providing an overview of their relevant information and tasks.

Include widgets or cards displaying key data, such as upcoming events, attendance summaries, and recent notifications.

Student Information:

Allow users to view and search for detailed student information, including personal details, contact information, and enrollment details.

Provide options for sorting and filtering student lists by various criteria.

Teacher Information:

Present teacher profiles, qualifications, and class assignments clearly.

Include contact information and the ability to contact teachers directly.

Parent/Guardian Information:

Enable parents to access their child's information, including attendance, performance, and school-related updates.

Provide a platform for parents to update their contact details and preferences.

Attendance Records:

Display attendance records for students and teachers in a user-friendly format, such as tables or graphs.

Highlight patterns or trends in attendance.

Timetables:

Show class schedules, exam schedules, and other school activities in a well-structured timetable format.

Allow users to filter timetables by class, teacher, or day.

Courses and Subjects:

List available courses and subjects with descriptions.

Clearly display subject-teacher-class associations.

Exams and Grading:

Present exam results, grading details, and GPA calculations in a comprehensive report format.

Include options for viewing individual student results and class performance.

Communication and Messaging:

Provide an inbox for receiving and sending messages, notifications, and updates.

Use an intuitive interface for composing and managing messages.

**3.5 Code Design and Development**

Developing a complete school management system is a complex and time-consuming project. Below, I'll provide a high-level overview of the code design and development process for a school management system, breaking it down into key components and considerations.

1. Project Setup:

Choose a programming language and technology stack that suits your team's expertise and project requirements. Common choices include Java, Python, PHP, Ruby, or JavaScript-based technologies like Node.js.

Set up version control using tools like Git to manage code changes and collaboration among developers.

Use a suitable integrated development environment (IDE) for coding and debugging.

2. Database Design:

Define the database schema for storing data such as student information, teacher details, class schedules, attendance records, exam results, and more.

Choose a relational database management system (RDBMS) like MySQL, PostgreSQL, or a NoSQL database like MongoDB based on data requirements.

Create tables, define relationships, and ensure data integrity with constraints and indexes.

3. User Authentication and Authorization:

Implement user authentication to secure access to the system. You can use frameworks like Django (Python), Laravel (PHP), or libraries like Passport.js (Node.js).

Define user roles (students, teachers, administrators, parents) and set up role-based access control (RBAC) to manage permissions.

4. User Interface (UI) Development:

Create a user-friendly web-based interface for accessing the system. You can use frontend technologies like HTML, CSS, JavaScript, and popular frontend frameworks (e.g., React, Angular, or Vue.js).

Develop responsive designs to ensure accessibility on various devices, including mobile phones and tablets.

Implement user interfaces for data input and presentation (e.g., forms for data input, dashboards, and report generation).

5. Back-End Development:

Create API endpoints to handle data processing, business logic, and communication between the front end and the database.

Implement features for managing student, teacher, and parent information, attendance tracking, timetable management, exam scheduling, and more.

Ensure data validation, error handling, and data integrity.

6. Data Validation and Security:

Validate user input to prevent data injection and security vulnerabilities. Use data validation libraries or frameworks to sanitize and validate input.

Implement security measures, such as authentication, authorization, and encryption to protect sensitive data.

7. Testing:

Perform unit testing to ensure individual components work as expected.

Conduct integration testing to verify that different parts of the system function together correctly.

Execute user acceptance testing (UAT) to validate the system's compliance with user requirements.

8. Deployment:

Deploy the system to a server or cloud infrastructure (e.g., AWS, Azure, Google Cloud) to make it accessible over the internet.

Configure domain name settings and SSL certificates for secure connections.

9. Continuous Integration/Continuous Deployment (CI/CD):

Implement CI/CD pipelines to automate code integration, testing, and deployment processes.

Ensure that updates and bug fixes can be rolled out efficiently.

10. Documentation:

Create user manuals, API documentation, and system documentation to help users and future developers understand the system's functionality and architecture.

**CHAPTER -4**

**TESTING AND IMPLIMENTATION###**

**4.1.1 Unit Testing**

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel

**4.1.2 Module Testing**

It involves testing individual components or modules of the website to ensure they function correctly. This can include testing features like user registration, flight search, hotel booking, payment processing, and more. Each module is tested independently to identify and fix any issues before integrating them into the complete website system. It helps ensure that every part of the website works as intended, providing a seamless experience for users.

**4.1.4 System Testing**

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems

**4.1.5 WHITE BOX TESTING**

This type of testing ensures that

· All independent paths have been exercised at least once

· All logical decisions have been exercised on their true and false sides

· All loops are executed at their boundaries and within their operational bounds

· All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

**4.1.6 Acceptance testing**

Acceptance testing of a school management system is a crucial phase in the software development life cycle. It involves evaluating the system to ensure that it meets the specified requirements and is ready for deployment. Acceptance testing typically includes the following key steps:

User Acceptance Testing (UAT): This is a critical phase where actual users, including school administrators, teachers, and staff, perform testing to ensure that the system aligns with their needs and expectations. UAT typically includes the following components:

* **Test Plan Creation:** Define the scope, objectives, and test scenarios for UAT. Identify the specific test cases that need to be executed.
* **Test Case Execution:** Users execute test cases based on real-world scenarios, such as enrolling students, managing attendance, inputting grades, and generating reports.
* **Issue Reporting**: Users report any issues, defects, or discrepancies they encounter during testing. These issues are then tracked and addressed by the development team.
* **Validation of User Workflows:** Ensure that the system supports essential user workflows efficiently and accurately.
* **Performance Testing:** Conduct performance testing to assess how the school management system handles a realistic load of concurrent users and data. This helps identify any performance bottlenecks or scalability issues that may need addressing.
* **Security Testing:** Verify that the system is secure and that sensitive data, such as student records, is adequately protected. Assess the system for vulnerabilities, including data encryption, user access controls, and protection against common security threats.
* **Compatibility Testing:** Ensure the system functions correctly on various devices, browsers, and operating systems commonly used by the school community. This ensures accessibility and usability for all users.
* **Integration Testing:** Confirm that the school management system seamlessly integrates with any other existing systems or third-party applications, such as learning management systems or accounting software.
* **Regression Testing:** Re-run previously conducted tests to make sure that new changes or updates have not introduced unintended issues or broken existing functionalities.
* **Accessibility Testing:** Check that the system complies with accessibility standards, ensuring it is usable by individuals with disabilities, such as those who require screen readers or keyboard navigation.
* **Data Migration Testing:** If data from previous systems needs to be migrated, test the accuracy and completeness of data transfer to the new system.
* **Documentation Review**: Validate that user manuals and documentation are accurate and comprehensive, helping users understand how to use the system effectively.
* **Stakeholder Approval:** Finally, obtain formal approval from relevant stakeholders, such as school management, before proceeding with deployment.

Remember, acceptance testing aims to simulate real user scenarios to identify any issues or areas for improvement before the website is launched.

**4.2 Test data test cases**

Certainly! Here are some test data test cases that you can consider for a traveling website:

Test data and test cases for a school management system should encompass a wide range of scenarios to ensure the system functions as intended. Here are some examples of test data and corresponding test cases:

Test Case 1: Student Enrollment

Test Data:

Student Information (First Name, Last Name, Date of Birth, Contact Details)

Class Information (Class Name, Class Teacher)

Enrollment Date

Test Steps:

Log in as an administrator.

Navigate to the student enrollment section.

Enter valid student information.

Assign the student to a class.

Verify that the student's enrollment is recorded correctly in the database.

Test Case 2: Attendance Tracking

Test Data:

Class Schedule

Student Information

Date

Attendance Status (Present, Absent, Late)

Test Steps:

Log in as a teacher.

Access the class attendance page for a specific date.

Mark students as "Present," "Absent," or "Late" based on the class schedule.

Verify that the attendance records are saved accurately and can be retrieved later.

Test Case 3: Grade Entry

Test Data:

Student Information

Course Information

Exam Type (Midterm, Final)

Exam Scores

Test Steps:

Log in as a teacher.

Navigate to the grade entry section.

Select a student and a course.

Enter exam scores for various exam types.

Verify that the scores are saved correctly and can be retrieved in grade reports.

Test Case 4: Parent-Teacher Communication

Test Data:

Parent Information

Teacher Information

Messages (Subject, Content)

Test Steps:

Log in as a parent.

Access the communication portal.

Send a message to a teacher.

Log in as a teacher.

Respond to the parent's message.

Verify that the messages are exchanged and stored in the system.

Test Case 5: User Access Control

Test Data:

User Roles (Admin, Teacher, Student, Parent)

User Permissions

Test Steps:

Log in with different user roles (admin, teacher, student, parent).

Verify that users can only access features and data allowed by their roles.

Attempt to access restricted features to ensure access control is functioning correctly.

Test Case 6: Report Generation

Test Data:

Student Information

Course Information

Exam Scores

Test Steps:

Log in as an administrator.

Navigate to the report generation section.

Select criteria (e.g., student, course, date range).

Generate various reports (e.g., student progress, course performance).

Verify that the reports are accurate and display the expected data.

These are just a few examples of test cases and associated test data. The goal is to cover various functionalities of the school management system and to ensure that it operates correctly under different scenarios. Additional test cases can be designed based on the specific features and requirements of the system.

**4.3 Test report and debugging**

A test report for a school management system is a crucial document that provides an overview of the testing process, the outcomes of the testing activities, and any issues or defects that were identified. Below is an outline for a test report for a school management system:

Test Report for School Management System

Project Name: [School Management System Name]

Test Period: [Start Date] to [End Date]

Test Team: [List of Testers]

1. Introduction

Provide a brief overview of the school management system and its purpose.

Explain the objectives of the testing phase and the scope of testing.

2. Testing Objectives

Summarize the primary goals of the testing phase, such as ensuring system functionality, security, and performance.

3. Test Environment

Describe the testing environment, including hardware, software, browsers, and any other tools or resources used during testing.

4. Test Approach

Explain the testing methodology used (e.g., manual testing, automated testing, or a combination).

Discuss the testing types performed, such as functional, security, performance, and usability testing.

Detail the testing strategies and techniques employed.

5. Test Scenarios and Test Data

List and briefly describe the test scenarios and test cases that were executed.

Mention the test data used, including sample data for student records, courses, attendance, and grades.

6. Test Execution

Describe the process of executing the test cases.

Include information on any test data preparation and setup.

7. Test Results

Summarize the outcomes of the testing activities, including both positive and negative results.

Provide pass/fail rates for different types of tests (functional, security, etc.).

Identify any issues, defects, or bugs that were discovered during testing.

8. Issue Tracking

Describe the process used for reporting and tracking issues.

List the identified issues, including their severity and status (open, closed, etc.).

Provide a summary of issue resolution.

9. Test Summary

Summarize the overall quality and readiness of the school management system based on the testing results.

Provide an assessment of whether the system meets the specified requirements and is ready for deployment.

10. Recommendations

Offer recommendations for improvements or corrective actions based on the test results.

11. Conclusion

Conclude the test report with a summary of the main findings and the readiness of the system.

12. Appendices

Include any additional information or documents relevant to the testing, such as test plans, test cases, and screenshots.

13. Sign-Off

Include signatures or approvals from the testing team and relevant stakeholders.

The test report should be a comprehensive document that helps project managers, developers, and stakeholders understand the testing process and the status of the school management system in terms of quality and readiness for implementation.

**Debugging**

1. Identify the Problem:

Debugging a school management system involves identifying and resolving issues, errors, or unexpected behaviors in the software. Here's a general process for debugging a school management system:

2. Reproduce the Issue:

The first step in debugging is to reproduce the problem. Understand the conditions under which the issue occurs and gather relevant data, such as error messages, logs, or user reports.

3. Isolate the Problem:

Determine whether the issue is a software bug or a user error. If it's a bug, identify the specific part of the system where the problem arises.

4. Review the Code:

Examine the source code in the area where the issue is occurring. Look for syntax errors, logical flaws, or unexpected conditions.

5. Use Debugging Tools:

Employ debugging tools and techniques to help identify and diagnose issues. Common tools include:

Print Statements: Insert print or log statements in the code to track the flow and values of variables.

Interactive Debuggers: Utilize integrated development environment (IDE) debuggers to step through the code, set breakpoints, and inspect variable values.

Error Logs: Check error logs or system logs for error messages and stack traces.

Profiling Tools: Use profiling tools to identify performance bottlenecks.

6. Check for Data Issues:

Examine the data being processed by the system. Ensure that data is correctly retrieved, stored, and manipulated.

7. Verify Input Validation:

Confirm that the system properly validates and sanitizes user inputs to prevent security vulnerabilities and data errors.

**4.4 Implementation manual**

Creating an implementation manual for a school management system is a comprehensive task, as it involves detailing the steps required to install, configure, and deploy the system within an educational institution. Below is a simplified outline of the contents that such a manual may include:

School Management System Implementation Manual

Table of Contents:

1. Introduction

* Brief overview of the school management system.
* Purpose and scope of the implementation manual.

1. Prerequisites

* System requirements (hardware, software, databases, etc.).
* Necessary permissions and access credentials.
* Data backup procedures.

1. Installation Instructions

Step-by-step guide to installing the system on the required server or hosting environment.

* Details on database setup and configuration.
* Verification of system dependencies and installation of any required components.

1. Configuration

* Configuration of system settings, including email servers, security settings, and user roles.
* Guidelines for customizing the system to meet the school's specific requirements.

1. Data Migration

* Procedures for importing existing student data, staff records, and other relevant information.
* Verification and validation of migrated data.

1. User Account Management

* Instructions for creating and managing user accounts, specifying roles and access levels.
* Password policies and security recommendations.

1. Training and User Guides

* Information on organizing and conducting user training sessions.
* Provision of user manuals and guides for administrators, teachers, students, and parents.

1. Testing and Quality Assurance

* Guidelines for conducting final testing before going live.
* Procedures for reporting and tracking issues, and the resolution process.

1. Data Security

* Recommendations for securing sensitive student and staff data.
* Implementation of user access controls and encryption measures.

1. Backup and Recovery

* Protocols for regular data backups and disaster recovery procedures.
* Steps for restoring the system in case of data loss or system failures.

1. Deployment and Rollout

* Strategies for a phased or full system deployment.
* Communication plans to inform stakeholders about the new system.

1. User Support

* Information on post-implementation support channels.
* Contact details for technical support and issue reporting.

**4.5 Implementation**

Implementing a school management system is a complex and crucial process. It involves several steps to ensure the successful deployment of the system within an educational institution. Here's a general guideline for implementing a school management system:

1. Needs Assessment and Planning:

* Identify the specific needs and requirements of the school, such as student data management, attendance tracking, grading, communication tools, and other features.
* Formulate a detailed project plan, including a timeline, resource allocation, and budget.

2. Select a School Management System:

* Choose a suitable school management system that aligns with the school's requirements, budget, and technical infrastructure.
* Evaluate different vendors and solutions, considering factors like scalability, support, and customization options.

3. System Configuration and Customization:

* Install and configure the school management system according to the manufacturer's guidelines.
* Customize the system to match the school's specific needs and workflows, such as class schedules and grading criteria.

4. Data Migration:

Transfer existing student records, staff information, and other relevant data to the new system. Ensure data accuracy and integrity during the migration process.

5. User Training:

* Provide comprehensive training to school staff, including administrators, teachers, and support personnel, on how to use the system effectively.
* Create user manuals and guides for ongoing reference.

6. Testing:

* Conduct thorough testing to verify that the system functions correctly and aligns with the school's requirements.
* Perform various types of testing, including functional testing, security testing, performance testing, and user acceptance testing.

7. Data Security and Privacy:

* Implement robust data security measures to protect sensitive student and staff information.
* Ensure compliance with data protection regulations, if applicable.

8. Pilot Implementation:

Run a pilot implementation with a small group of users or in a specific department to identify and address any issues or challenges before a full-scale rollout.

9. Full Deployment:

Deploy the school management system to all intended users and departments, following a phased or full-scale deployment strategy as per the project plan.

10. Data Backup and Disaster Recovery:

Implement regular data backup procedures and create a disaster recovery plan to safeguard against data loss or system failures.

**4.5 User’s Training**

User training for a school management system is essential to ensure efficient system utilization. It should encompass hands-on sessions for administrators, teachers, and staff, focusing on tasks such as data entry, reporting, and system navigation. Training materials, including user manuals, should be readily available for ongoing reference, empowering users to make the most of the system's capabilities

**4.6 Post implementation**

Post-implementation of a school management system, it's critical to continuously monitor and support the system. This includes ongoing technical support for users and addressing any technical issues or glitches promptly.

Regular user feedback collection is essential to identify any areas for improvement. This feedback loop can help enhance system usability and efficiency.

Data security and privacy remain paramount post-implementation. Ensure that data is regularly backed up, and the system's security protocols are maintained to protect sensitive student and staff information.

Lastly, consider long-term scalability and system updates. As the school's needs evolve, the system should be flexible enough to accommodate these changes. Plan for regular updates and maintenance to keep the system running smoothly and up-to-date.

**CHAPTER-5**

**CONCLUSION AND REFERENCES**

**5.1 Conclusion**

In conclusion, a well-implemented school management system is a transformative tool for educational institutions. It streamlines administrative tasks, empowers educators with valuable insights, and enhances communication between stakeholders. The system optimizes resource allocation, improves data accuracy, and fosters a more efficient learning environment. However, its success hinges on careful planning, thorough implementation, user training, and continuous support. As educational institutions embrace technology to meet the evolving needs of the 21st century, a well-designed and effectively implemented school management system can play a pivotal role in shaping the future of education.

**5.2 System Specifiaction**

The system specification of a school management system outlines the technical and functional requirements that define its capabilities and limitations. It includes details such as the supported platforms, database structure, user roles, and key features. This document serves as a blueprint for developers and stakeholders, ensuring that the system aligns with the school's specific needs. It covers areas like student data management, attendance tracking, grade recording, user access controls, reporting, and communication tools. Additionally, it addresses performance expectations, security measures, data backup procedures, and integration capabilities with other school systems. System specifications provide the foundation for the development and successful implementation of a tailored school management solution that enhances administrative efficiency and elevates the overall educational experience.

**5.2.1 Hardware Requirement**

* Laptop or pc
* Ram 6GB or higher
* 100GB ROM or higher

**5.2.2 Software Requirement**

* WINDOWS OS
* Visual Studio
* XAMP Server
* Visual Studio

**5.3 Limitation of the System**

While school management systems offer numerous benefits, they also come with limitations that need to be considered:

1. Technical Challenges: School management systems can face technical issues such as system downtime, slow performance, and software compatibility problems. These challenges can disrupt daily operations.
2. Cost: The initial setup and ongoing maintenance costs can be a significant barrier for smaller educational institutions with limited budgets.
3. Data Security and Privacy Concerns: Storing sensitive student and staff information electronically raises concerns about data security and privacy, especially with the risk of data breaches.
4. User Training Needs: Implementing a new system requires user training, which can be time-consuming and might not be well-received by all staff members. Resistance to change can also be a limitation.
5. Accessibility: Ensuring that the system is accessible to all stakeholders, including individuals with disabilities, can be challenging and may require additional development efforts.

Customization and Integration: Adapting the system to fit the school's unique needs can be limited by the system's rigidity. Integrating with other systems, such as learning management or accounting software, may also be complex.

While these limitations are worth considering, they can often be mitigated with proper planning, user training, regular maintenance, and adherence to data security best practices. Ultimately, the benefits of a well-implemented school management system usually outweigh its limitations in improving the efficiency and effectiveness of educational institutions

* 1. **Future Scope for Modification**

The future scope of school management systems is promising and will likely see several advancements and innovations:

Artificial Intelligence (AI) and Machine Learning: AI-powered features will become more common, allowing for predictive analytics to improve student performance, automated administrative tasks, and personalized learning pathways.

Blockchain Integration: Blockchain technology can enhance security, transparency, and efficiency in managing student records, diplomas, and certifications, reducing the risk of credential fraud.

Mobile Accessibility: Greater emphasis on mobile-friendly platforms, enabling students, parents, and teachers to access information and perform tasks on their smartphones and tablets.

IoT Integration: The Internet of Things (IoT) can be used to track and manage assets, monitor attendance, and enhance campus security.

Cloud-Based Systems: Widespread adoption of cloud-based solutions for easier scalability, reduced infrastructure costs, and improved accessibility.

Advanced Data Analytics: Deeper data analysis will lead to more informed decision-making in areas like resource allocation, curriculum development, and student support.

Enhanced Communication Tools: Improved communication platforms that facilitate real-time interaction between teachers, students, and parents, fostering a more connected educational community.

Gamification and Interactive Learning: Integration of gamification elements and interactive learning modules to engage students and enhance the learning experience.

The future of school management systems holds the potential to significantly improve administrative efficiency, communication, and the overall quality of education. With the integration of emerging technologies and a focus on data-driven decision-making, these systems will continue to evolve to meet the ever-changing needs of educational institutions and the students they serve.

**5.5 Bibliography**

W3Schools<https://www.w3schools.com/>

<https://www.tutorialspoint.com/index.htm>

<https://www.javatpoint.com/what-is-html>

Authors Book

Pearson

SumitaArora

**CHAPTER-6**

**ANNEXURES**

**Menu Flow Diagram**

Classes

Management

**School Management System**

Fee

Management

Student

Management

System User

Management

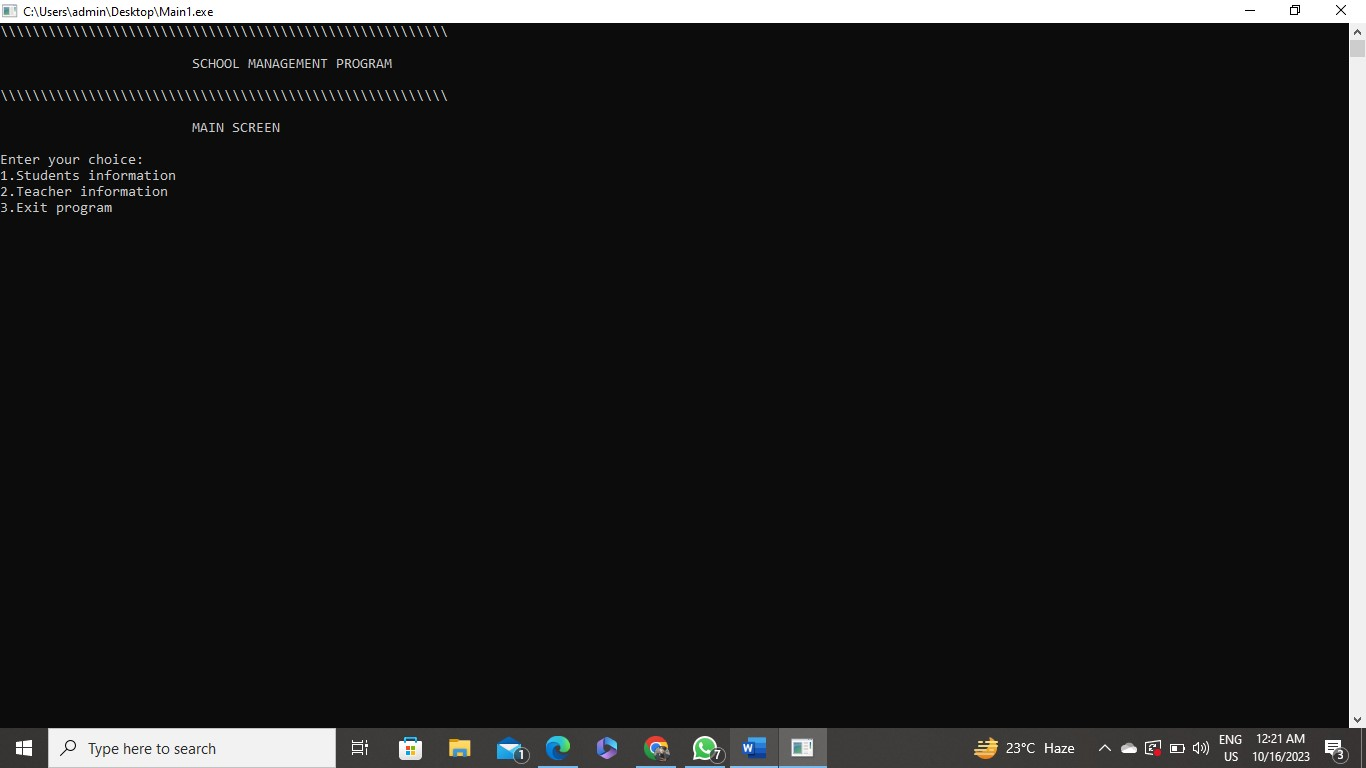
Faculty

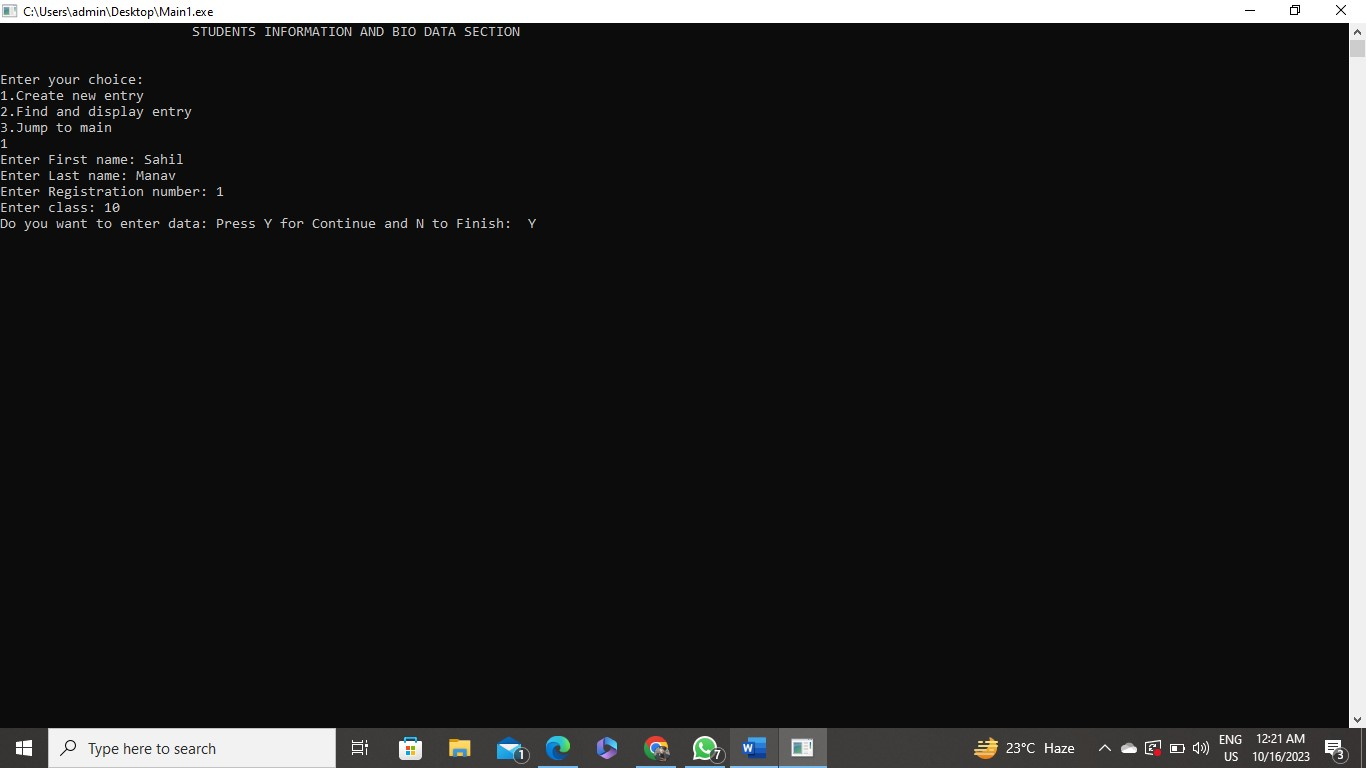
Management

Login

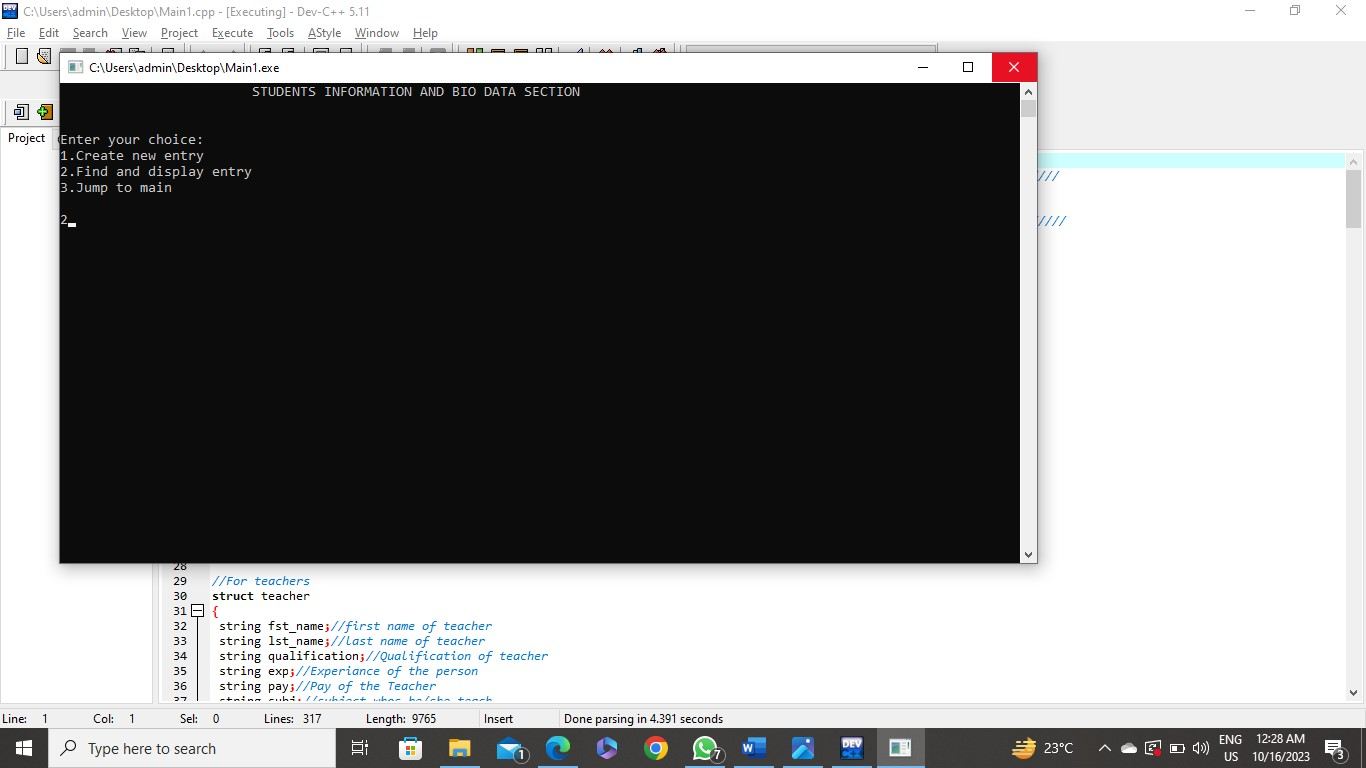
Management

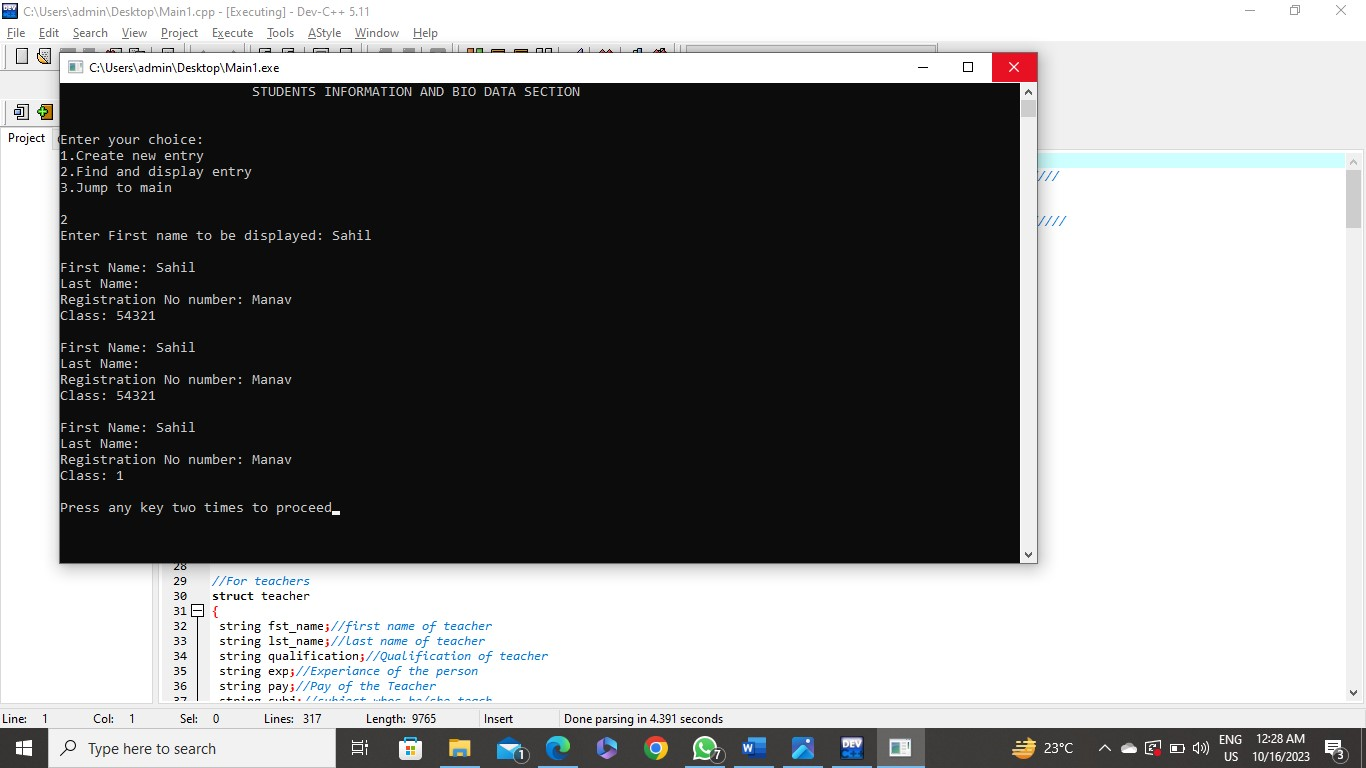
**Sample Inputs**





**Sample Outputs**





**Coding**

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//SCHOOL MANAGEMENT PROGRAM

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//Headers inclusion

#include <iostream>

#include <fstream>

#include <string>

#include <conio.h>

#include <windows.h>

using namespace std;

//Structure defining

//For students

struct student

{

string fname;//for student first name

string lname;//for student last name

string Registration;//for Registration No number

string classes;//for class info

}studentData;//Variable of student type

//For teachers

struct teacher

{

string fst\_name;//first name of teacher

string lst\_name;//last name of teacher

string qualification;//Qualification of teacher

string exp;//Experiance of the person

string pay;//Pay of the Teacher

string subj;//subject whos he/she teach

string lec;//Lecture per Week

string addrs;//Adders of teacher home

string cel\_no;//Phone number

string blod\_grp;//Bool Group

string serves;//Number of serves in School

}tech[50];//Variable of teacher type

///////////////////////////////////////////////////

//Main function

int main()

{

int i=0,j;//for processing usage

char choice;//for getting choice

string find;//for sorting

string srch;

while(1)//outer loop

{

system("cls");//Clear screen

//Level 1-Display process

cout<<"\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\";

cout<<"\n\n\t\t\tSCHOOL MANAGEMENT PROGRAM\n\n";

cout<<"\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\";

cout<<"\n\n\t\t\tMAIN SCREEN\n\n";

cout<<"Enter your choice: "<<endl;

cout<<"1.Students information"<<endl;

cout<<"2.Teacher information"<<endl;

cout<<"3.Exit program"<<endl;

cin>>choice;

system("cls");//Clear screen

switch(choice)//First switch

{

case '1': //Student

{

while(1)//inner loop-1

{

system("cls");//Clear screen

//Level-2 display

cout<<"\t\t\tSTUDENTS INFORMATION AND BIO DATA SECTION\n\n\n";

cout<<"Enter your choice: "<<endl;

cout<<"1.Create new entry\n";

cout<<"2.Find and display entry\n";

cout<<"3.Jump to main\n";

cin>>choice;

switch (choice)//Second switch

{

case '1'://Insert data

{ ofstream f1("student.txt",ios::app);

for( i=0;choice!='n';i++)

{

if((choice=='y')||(choice=='Y')||(choice=='1'))

{

cout<<"Enter First name: ";

cin>>studentData.fname;

cout<<"Enter Last name: ";

cin>>studentData.lname;

cout<<"Enter Registration number: ";

cin>>studentData.Registration;

cout<<"Enter class: ";

cin>>studentData.classes;

f1<<studentData.fname<<endl<<studentData.lname<<endl<<studentData.Registration<<endl<<studentData.classes<<endl;

cout<<"Do you want to enter data: ";

cout<<"Press Y for Continue and N to Finish: ";

cin>>choice;

}

}

f1.close();

}

continue;//control back to inner loop -1

case '2'://Display data

{ ifstream f2("student.txt");

cout<<"Enter First name to be displayed: ";

cin>>find;

cout<<endl;

int notFound = 0;

for( j=0;(j<i)||(!f2.eof());j++)

{

getline(f2,studentData.fname);

if(studentData.fname==find)

{

notFound = 1;

cout<<"First Name: "<<studentData.fname<<endl;

cout<<"Last Name: "<<studentData.lname<<endl;

getline(f2,studentData.Registration);

cout<<"Registration No number: "<<studentData.Registration<<endl;

getline(f2,studentData.classes);

cout<<"Class: "<<studentData.classes<<endl<<endl;

}

}

if(notFound == 0){

cout<<"No Record Found"<<endl;

}

f2.close();

cout<<"Press any key two times to proceed";

getch();//To hold data on screen

getch();//To hold data on screen

}

continue;//control back to inner loop -1

case '3'://Jump to main

{

break;//inner switch breaking

}

}

break;//inner loop-1 breaking

}

continue;//Control pass to 1st loop

}

case '2'://Teachers biodata

{

while(1)//inner loop-2

{

system("cls");//Clear screen

//Level-2 Display process

cout<<"\t\t\tTEACHERS INFORMATION AND BIODATA SECTION\n\n\n";

cout<<"Enter your choice: "<<endl;

cout<<"1.Create new entry\n";

cout<<"2.Find and display\n";

cout<<"3.Jump to main\n";

cin>>choice;

switch (choice)//Third switch

{

case '1'://Insert data

{

ofstream t1("teacher.txt",ios::app);

for(i=0;choice!='n'&& choice!='N';i++)

{

if((choice=='y')||(choice=='Y')||(choice=='1'))

{

cout<<"Enter First name: ";

cin>>tech[i].fst\_name;

cout<<"Enter Last name:: ";

cin>>tech[i].lst\_name;

cout<<"Enter qualification: ";

cin>>tech[i].qualification;

cout<<"Enter experiance(year): ";

cin>>tech[i].exp;

cout<<"Enter number of year in this School: ";

cin>>tech[i].serves;

cout<<"Enter Subject whos teach: ";

cin>>tech[i].subj;

cout<<"Enter Lecture(per Week): ";

cin>>tech[i].lec;

cout<<"Enter pay: ";

cin>>tech[i].pay;

cout<<"Enter Phone Number: ";

cin>>tech[i].cel\_no;

cout<<"Enter Blood Group: ";

cin>>tech[i].blod\_grp;

t1<<tech[i].fst\_name<<endl<<tech[i].lst\_name<<endl

<<tech[i].qualification<<endl<<tech[i].exp<<endl

<<tech[i].serves<<endl<<tech[i].subj<<endl<<tech[i].lec

<<endl<<tech[i].pay<<endl<<tech[i].cel\_no<<endl<<tech[i].blod\_grp<<endl;

cout<<"Do you want to enter data: ";

cin>>choice;

}//if

}//for loop

//for finding through name

system("cls");

t1.close();

}//case 1

continue;//Control pass to inner loop-2

case '2'://Display data

{

ifstream t2("teacher.txt");

cout<<"Enter name to be displayed: ";

cin>>find;

cout<<endl;

int notFound = 0;

for( j=0;((j<i)||(!t2.eof()));j++)

{

getline(t2,tech[j].fst\_name);

if(tech[j].fst\_name==find)

{

notFound = 1;

cout<<"First name: "<<tech[j].fst\_name<<endl;

getline(t2,tech[j].lst\_name);

cout<<"Last name: "<<tech[j].lst\_name<<endl;

getline(t2,tech[j].qualification);

cout<<"Qualification: "<<tech[j].qualification<<endl;

getline(t2,tech[j].exp);

cout<<"Experience: "<<tech[j].exp<<endl;

getline(t2,tech[j].serves);

cout<<" number of year in this School: "<<tech[j].serves<<endl;

getline(t2,tech[j].subj);

cout<<"Subject whos teach: "<<tech[j].subj<<endl;

getline(t2,tech[j].lec);

cout<<"Enter Lecture(per Week): "<<tech[j].lec<<endl;

getline(t2,tech[j].pay);

cout<<"pay: "<<tech[j].pay<<endl;

getline(t2,tech[j].addrs);

cout<<"Address: "<<tech[j].addrs<<endl;

getline(t2,tech[j].cel\_no);

cout<<"Phone Number: "<<tech[j].cel\_no<<endl;

getline(t2,tech[j].blod\_grp);

cout<<"Bool Group: "<<tech[j].blod\_grp<<endl;

}//if

}//for loop

t2.close();

if(notFound == 0){

cout<<"No Record Found"<<endl;

}

cout<<"Press any key two times to proceed";

getch();//To hold data on screen

getch();//To hold data on screen

}//case 2

continue;//Control pass to inner loop-2

case '3'://Jump to main

{

break;//inner switch

}//case 3

}//inner switch

break;//inner while

}//inner loop

continue;//control pass to 1st loop

}//outer case 2

case '3':

{

break;//outer case 3

}//outer case 3

}

break;//outer loop

}

}