STUDENT HUB

INTRODUCTION:

Every academic institute depends on data, but not all of them handles it well. In many schools, student records are scattered across paper files, spreadsheets, or outdated digital systems that lack structure, security, or scalability. This creates confusion, increases administrative workload, and miscommunication especially when it comes to grading. To address these challenges, educational institutions often turn to digital solutions that can streamline student data management.

A student management system is a software-based platform designed to store, process, and manage student information such as academic performance, personal details, and enrollment data. By digitizing this data, the system improves accuracy, accessibility, and organization. However most traditional systems fail to account for access control, they treat all users the same, granting unrestricted privileges regardless of roles or responsibility. This chapter seeks to represent information concerning background study, problem statement, objective, and the scope.

This project is a simple, locally-hosted Student Management System built using HTML, CSS, JavaScript, Node.js, and Express.js, with an SQLite backend database. It allows users to register as either **Students** or **Teachers**, offering tailored access to academic records **based on their role**. Students can **view their grades and report inaccuracies,** while teachers can **manage and update grades without modifying student personal information.**

BACKGROUND STUDY

Before the emergency of computerized systems, student's record-keeping was entirely manual. Academic institutions relied on paper files, handwritten grade books, and physical storage cabinets. This traditional method, though effective for small scale operations, became unsustainable as enrollment increased and data complexity grew.

The first steps toward digitizing student information began in the 1960s and 1970s, with universities in the United States and Europe experimenting with mainframe-based administrative software. These early systems were custom-built, often expensive, and used only for large institutions.

A major shift occurred in the 1980s and early 1990s with the widespread use of personal computers. Schools began using spreadsheet software such as lotus 1-2-3 and Microsoft excel to manage student data locally. This era marked the beginning of small-scale student information systems (SIS) that were more accessible to schools.

By the late 1990s and early 2000s, dedicated student management systems began to emerge commercially. Products like PowerSchool (launched in 1997), Infinite Campus (in 1993), and open-source tools like Fedena (launched in 2009) introduced web-based access and multi-role support for students, teachers, and administrators.

Problem Today:

In many institutes today, student data is still managed by overly complexed systems, but all these systems were expensive and mostly available for large institutes leaving the smaller institute behind the evolution of student management systems. Hence, the reason we are building a small scale, less expensive system to manage student data.

This project is inspired by the need for a lightweight, role-based academic record system that works offline without the need for external hosting, and maintains privacy and control for both students and educators. By separating permissions clearly between user types, this project simulates real-world systems while remaining beginner-friendly for developers.

LITERAURE REVIEW:

This chapter offers a critical examination of existing research and systems related to student management platforms. It aims to contextualize the academic information systems by evaluating methodologies, innovations, and limitations of the system.

ALIYU MUSA (2019) conducted research on the development of student grade management system. He designed a basic desktop application for entering and viewing grades.

In 2015 MOHAMMAD A. ALIA and HUSSEIN A. AL BAHADILI highlighted the importance of using role-based access to protect student data by create design and implementation of a student information system. The main focus was to introduce a centralized student information system for university and aim at secure access, data integrity, and multi-user functionality.

In 2017 E. OKONIGENE and E. EHIKIOYA designed and implement an offline-based student management system. The main focus was to create a student management system in areas with poor or no internet connection, emphasizing on offline performance and data security.

In 2013 OMOGBADEGUN Z.O highlights the importance of role-based access control in student management system, and the main focus was on how to separate user's permissions using role-based access control, preventing unauthorized access to sensitive student data, demonstrating secure login, data hiding, and limited privileges.

OBJECTIVES:

- ➢ To design and implement a web-based student management system with clearly defined user roles.
- ➢ To allow students to securely access and review their academic records.
- ➢ To enable teachers to manage student grades without access to sensitive personal data.
- ➢ To store all data locally using an SQLite database for simplicity and offline access.
- ➢ To reinforce secure and ethical data handling in software systems.

Scope:

This project will cover:

- Local user registration and login, requiring email and password credentials. The JavaScript on the login page will search the SQLite database for matching user credentials, directing authenticated users to their respective dashboards or alerting them to sign up if not found.
- Role-based dashboards for students and teachers, displayed on separate webpages. Users will be directed to their specific dashboard depending on the role they chose during login.
- Secure data handling and storage using Node.js, Express.js, and an SQLite database (stored as a .db file).
- CRUD (Create, Read, Update, Delete) options for grades (teachers only). The teacher dashboard will feature a table listing student names and their grades, and teachers will be able to edit grades by selecting a student and assigning one of the pre-defined grades from "A" to "F".

- Feedback systems for grade complaints (students only), with a dedicated notification dashboard for teachers to view and delete these complaints.
- Displaying only the courses taught by a specific teacher on their respective dashboard.

This project will not include:

- Online access hosting.
- Real-time updates or multi-user support.
- Advanced encryption or third-party integration.
- Admin role or automated grading.

Main Users of the System

- Students: Individuals enrolled in educational institutions who need to securely access and review their academic grades and report any inaccuracies or complaints.
- Teachers/Educators: Instructors responsible for managing, updating, and reviewing student grades for the courses they teach, without needing access to sensitive personal information.
- Small Schools/Educational Institutions: Organizations seeking a lightweight, locally-hosted, and cost-effective solution for basic student academic record management, prioritizing privacy and data control.

LIMITATION:

- Student cannot interact with each other
- Student cannot edit his/her grade
- Teacher cannot add or edit student personal details
- No SMS or email notification
- Basic password encryption
- No multi-user sessions
- All data must be managed locally

INOVATION:

Student Hub will introduce new unique features making our web app different from what others have built before. These features will include;

- Role-Based Access Control (RBAC):
  Separates permissions between students and teachers for better data security.

- Grade-Reporting Features:
  Students can report incorrect grades mimicking real academic systems
- Offline, Local-only Operation:
  No hosting or internet required, accessible to institutions with low resources

Functional and Non-Function Dependencies:

- **Functional Dependencies**
  These are the features the system must perform in other to call this project successful. They include;

    - User registration and login with role selection
    - Role-based access to dash-boards
    - Student access to grades and complaint system
    - Teacher access to grade editor, student list and student complaints.
    - Local data storage SQLite file (.db)
    - Display and validation of data in the browser

- **Non-Functional Dependencies:**

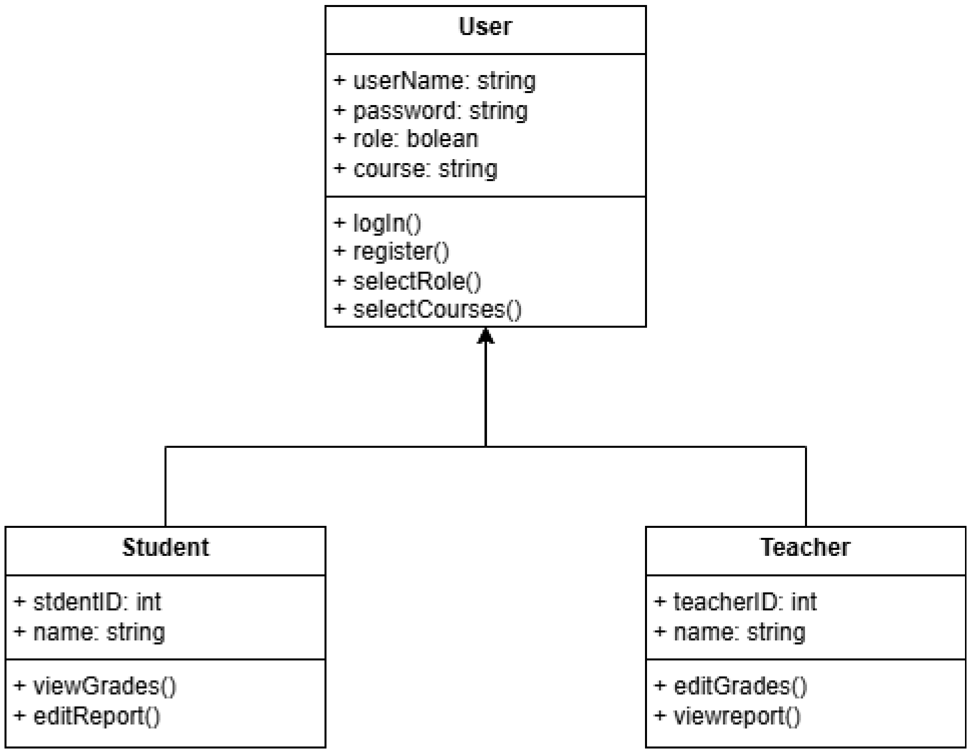
  These describe how the system should behave

    - Fast and responsive in modern browsers
    - Simple and intuitive interface
    - Reliable data access with minimal crashes
    - Consistent design across all screens
    - Fast communication between SQLite database and frontend.

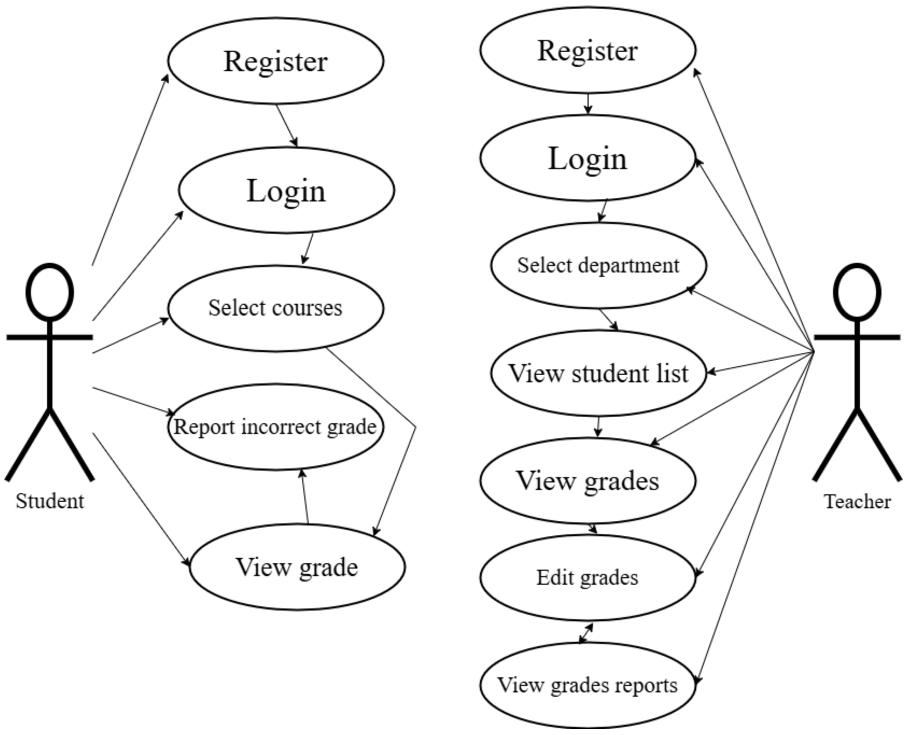
UML Diagrams:

The following diagrams were drawn using ***Draw.io***

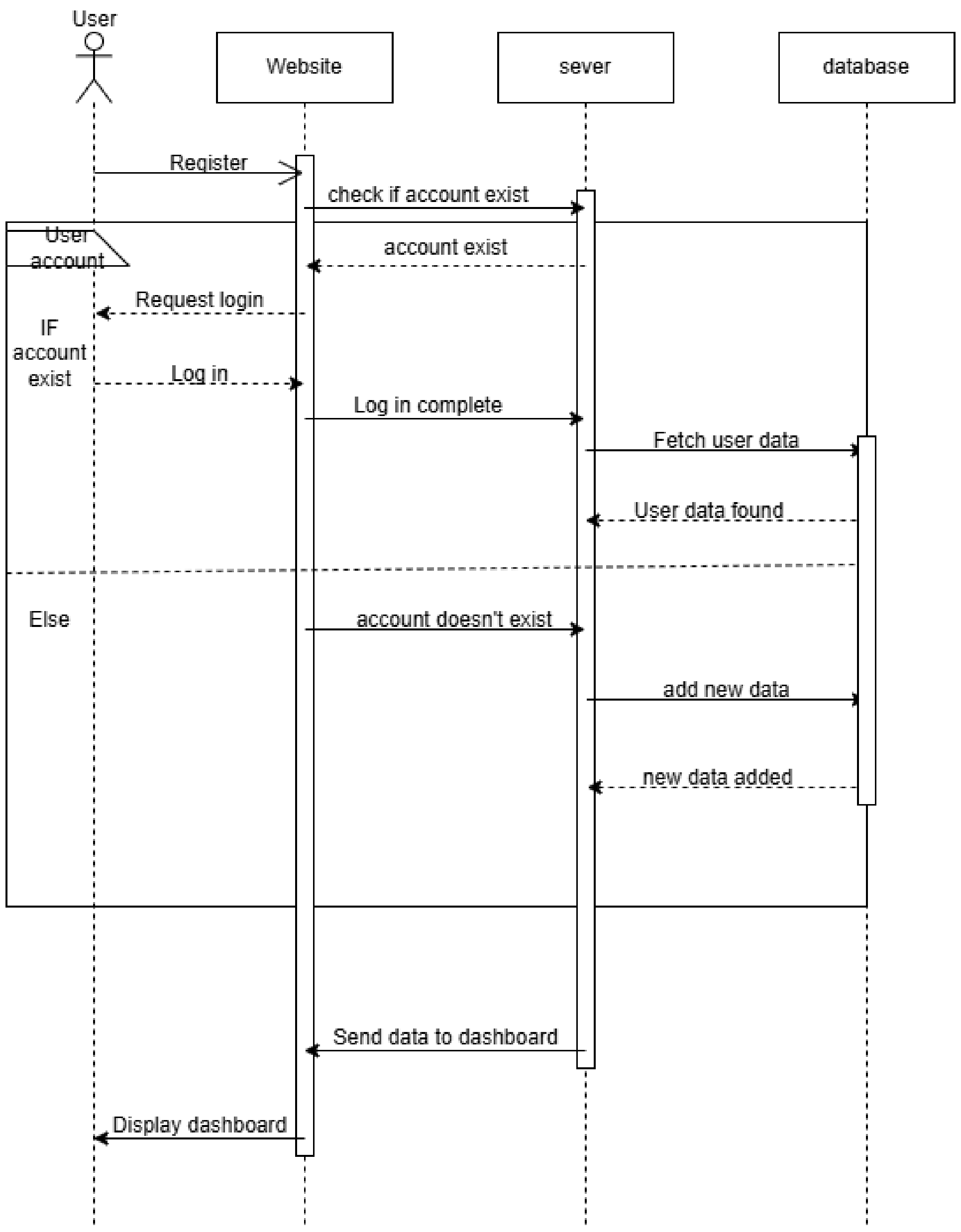
- Class Diagram;



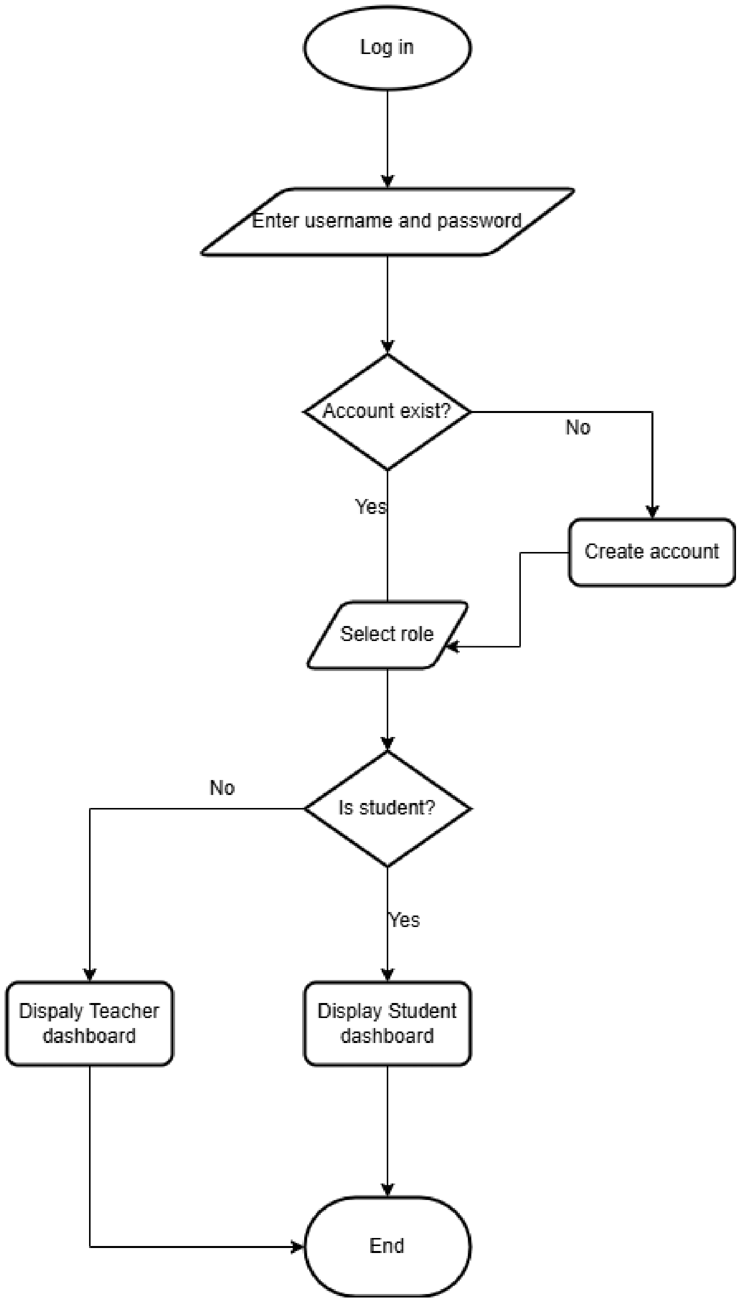
- Use-Case Diagram;



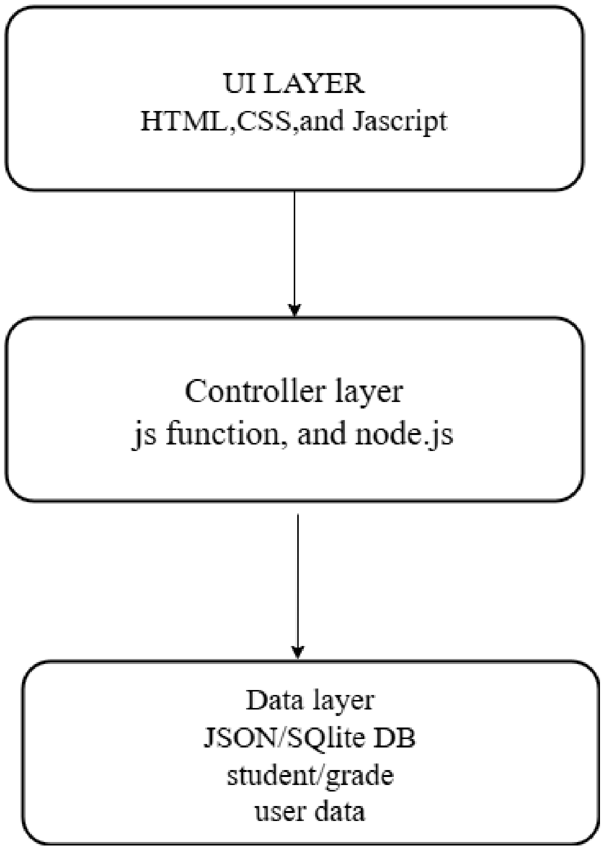
- Sequence Diagram;



- Flow Chart;

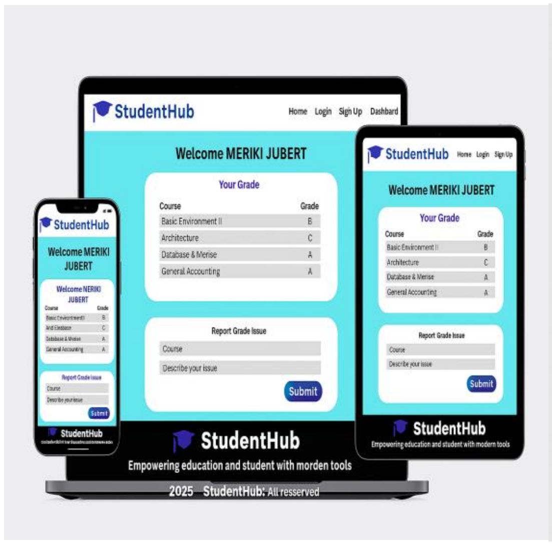

Log in
Enter username and password
Account exist?
No
Yes
Create account
Select role
Is student?
No
Yes
Dispaly Teacher dashboard
Display Student dashboard
End

- Architectural Diagram;

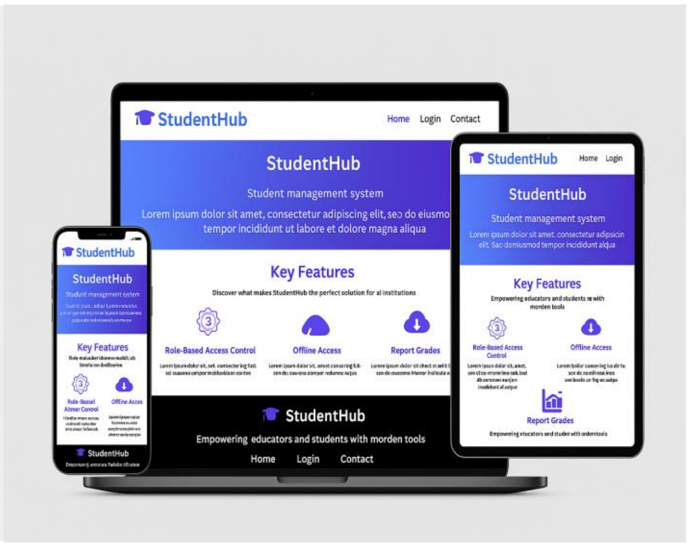

UI LAYER
HTML,CSS,and Jascript
Controller layer
js function, and node.js
Data layer
JSON/SQlite DB
student/grade
user data

Design UI/UX:

The following was designed using Canva. This design is responsive made for both mobile phones, Tablets and Laptops



Course	Grade
Basic Environment II	B
Architecture	C
Database & Merise	A
General Accounting	A


StudentHub
Home
Login
Contact
StudentHub
Student management system
Lorem ipsum dolor sit amet, consectetur adipiscing elit, seo do eiusmo tempor incididunt ut labore et dolore magna aliqua
Key Features
Discover what makes StudentHub the perfect solution for al institutions
Role-Based Access Control
Offline Access
Report Grades
StudentHub
Empowering educators and students with morden tools
Home
Login
Contact