M.S. RAMAIAH INSTITUTE OF TECHNOLOGY

MSR NAGAR, BENGALURU,560054

A Mini Project Report on

Student Performance Tracking System

Submitted in partial fullfillment of the OTHER COMPONENT requirements as a part of the

NOSQL Databases-ISAEC591 for the V Semester of degree of Bachelor of Engineering in

Information Science and Engineering

Submitted by

1MS24IS407	HARSHITA.M.V
1MS24IS412	SHREYA.K.G

Under the Guidance of

Faculty Incharge Dr. Kusuma S
Assistant Professor Dept. of ISE

Department of Information Science and Engineering

Ramaiah Institute of Technology 2025 – 2026

Contents

1. Project Overview

1.1 Project Description

The Student Performance Tracking System (SPTS) is a database-driven application designed to store, manage, and monitor student academic performance.

The system allows faculty and administrators to record student details, track subject-wise marks, attendance, and semester performance, and analyze academic progress efficiently.

SPTS uses MongoDB as its NoSQL backend, enabling flexible and scalable storage of student records, subject information, and performance data.

1.2 Requirements Collection

The following requirements were identified during project analysis:

- Centralized storage of student academic records.
- User authentication for staff and students.
- Recording subject-wise marks and attendance.
- Term-wise performance tracking for students.
- Ability to view and update student performance data.
- Secure access based on user roles (staff/student).
- CRUD operations for student and performance records.
- MongoDB-based backend for flexible document storage

1.3 NoSQL Database Used and Why

MongoDB was selected due to:

1. Flexible Schema:
Student performance data may vary across semesters, subjects, and departments. MongoDB supports a dynamic document structure without requiring schema migration.
2. JSON/BSON Support:
MongoDB documents integrate naturally with Node.js and Express applications used in the system.
3. High Scalability:
MongoDB supports horizontal scaling, making it suitable for handling a growing number of student records and performance data.
4. Performance:
Fast indexing, querying, and aggregation enable efficient retrieval of student details, marks, attendance, and performance records.

1.4 Application of the Project

The Student Performance Tracking System is applied in educational institutions to:

- Allow staff to record marks, attendance, and assessments.
- Track and manage student academic performance efficiently
- Enable students to view their performance details
- Reduce manual paperwork and calculation errors
- Provide quick access to student academic records
- Generate performance reports for academic evaluation

2. System Requirements

2.1 Software Requirements

Windows / Linux / macOS

• Node.js (v14 or above)

• Express.js Framework

• MongoDB

• npm package manager

• Web browser (Chrome / Firefox / Edge)

2.2 Hardware Requirements

- Minimum Intel i3 processor
- At least 4 GB RAM (8 GB recommended)
- Minimum 500 MB storage
- Stable internet connection (for MongoDB Atlas)3. Database Design

3.1 Data Model

The database follows a document-oriented data model.
Collections used:

• students — stores student personal and academic details

• performance — stores subject-wise marks, attendance, and term performance
• users — stores login and role-based authentication details

4. Implementation Steps

4.1 Installation & Steps

- Install Node.js
- Setup MongoDB
- Run server with npm start

6. Application Design

Programming Language & Stack:

• Language: JavaScript (Node.js, ES6+)
• Backend Framework: Seed.js
• Database: MongoDB
• Frontend Technologies: HTML, CSS
• Runtime Environment: Node.js
• Session Store: connect-mongo

Operations

1.Login Page

This is the authentication module of the system.
Operations performed

1. User enters username
2. User enters password
3. User selects role
    - Student
    - Staff
4. Click Login

System operation

- The system checks:
    - Username exists or not

    - Password matches
    - Selected role is correct
- If details are valid:
    - Redirects to Student Dashboard OR Staff Dashboard
- If invalid:
    - Shows error (wrong credentials)

Type of operation
Authentication
☑ Role-based access control

2.Student Dashboard – “My Performance” Page

This is the student view module where a student can only view their own performance.
Operations performed

1. Display student name and role
    - Example: *Welcome, Harshitha (Student)*
2. Fetch performance data from database
3. Display marks in tabular format

Data shown (Read-only)

- Subject Name
- Internal Marks
- Assignment Marks
- Attendance Percentage
- Term (Term 1)

Type of operation
☑ Read operation
☑ Data retrieval
☑ View-only access

3.Staff Login Page

Allows staff (teacher) to log into the system.
Operations performed

1. Staff enters:
    - Username
    - Password
2. Selects role as Staff
3. Clicks Login

System operation

- Validates staff credentials

- Redirects to Staff Dashboard

Type of operation

☑ Authentication

☑ Authorization (staff privileges)

4.Staff Dashboard – All Students Performance

This is the most important admin module.

A) View All Students Performance

Operations performed

1. System fetches all students list
2. Displays:
    - Student Name
    - Term
    - Action button (Edit Marks)

Staff capability

☑ Can view all students

☑ Can select any student

B) Edit / Update Student Marks

Operations performed

1. Staff clicks Edit Marks
2. System opens Update Student Marks form
3. Staff enters:
    - Term
    - Subject
    - Internal Marks
    - Assignment Marks
    - Attendance %
4. Clicks Save Marks

Backend operation

- Data is:
    - Validated
    - Updated in database
- Student record is modified

Type of operation

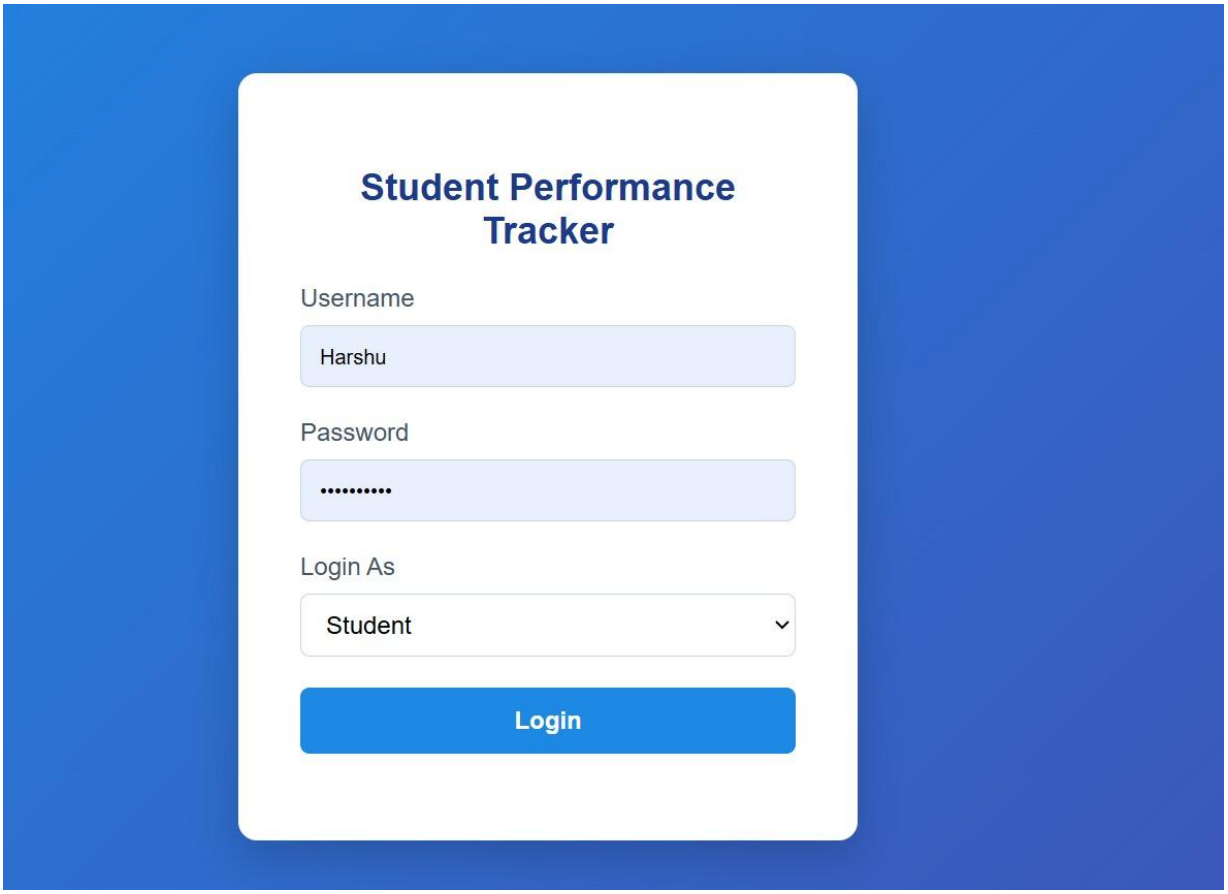
☑ Create operation (new record)

☑ Update operation (modify marks)

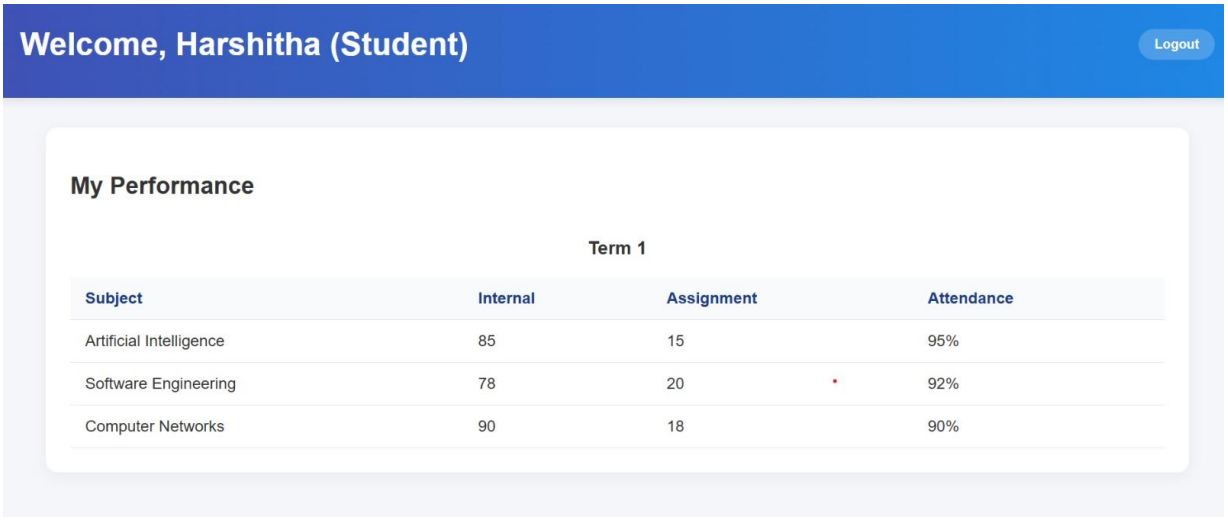
☑ Admin control

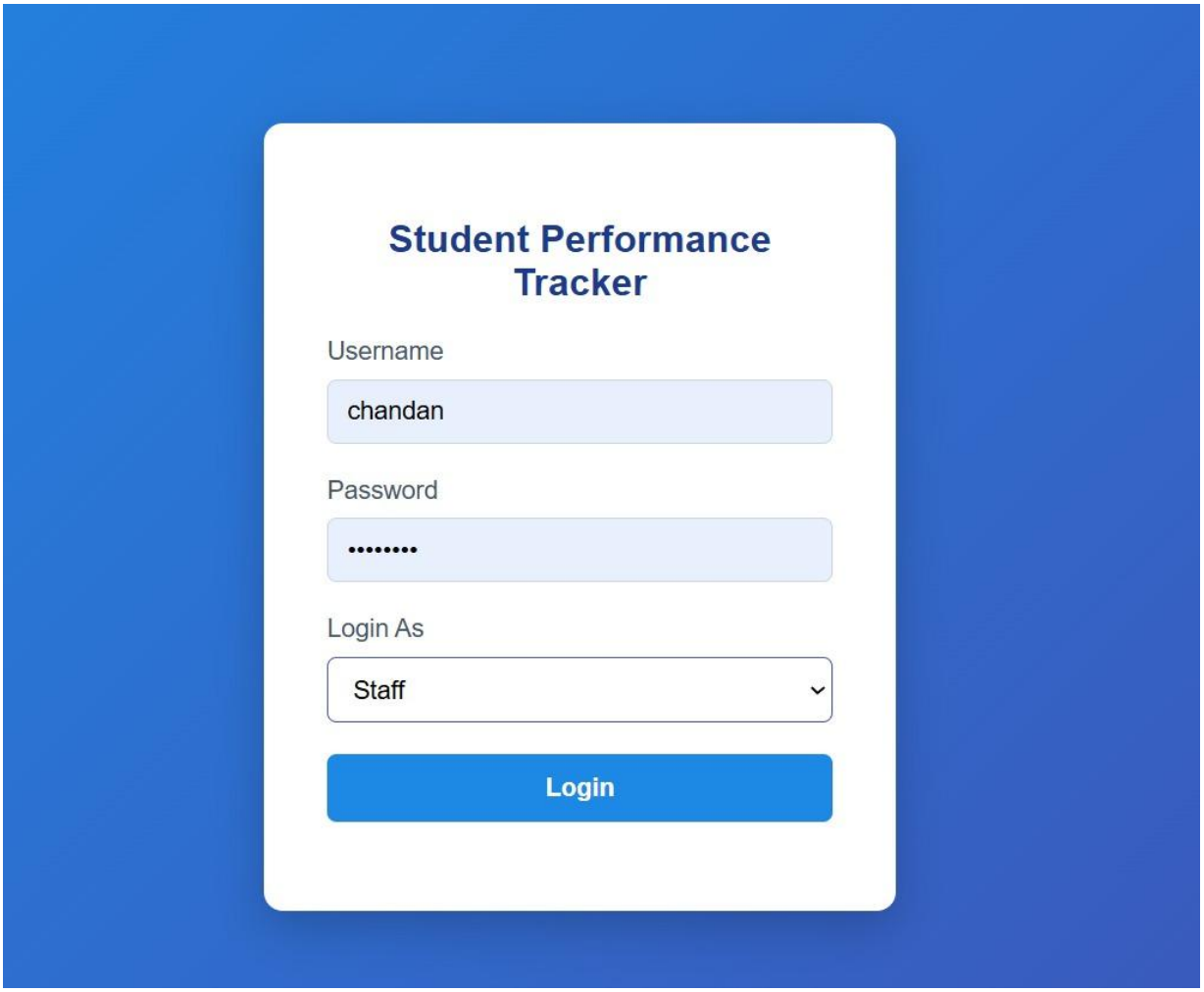
Screenshots

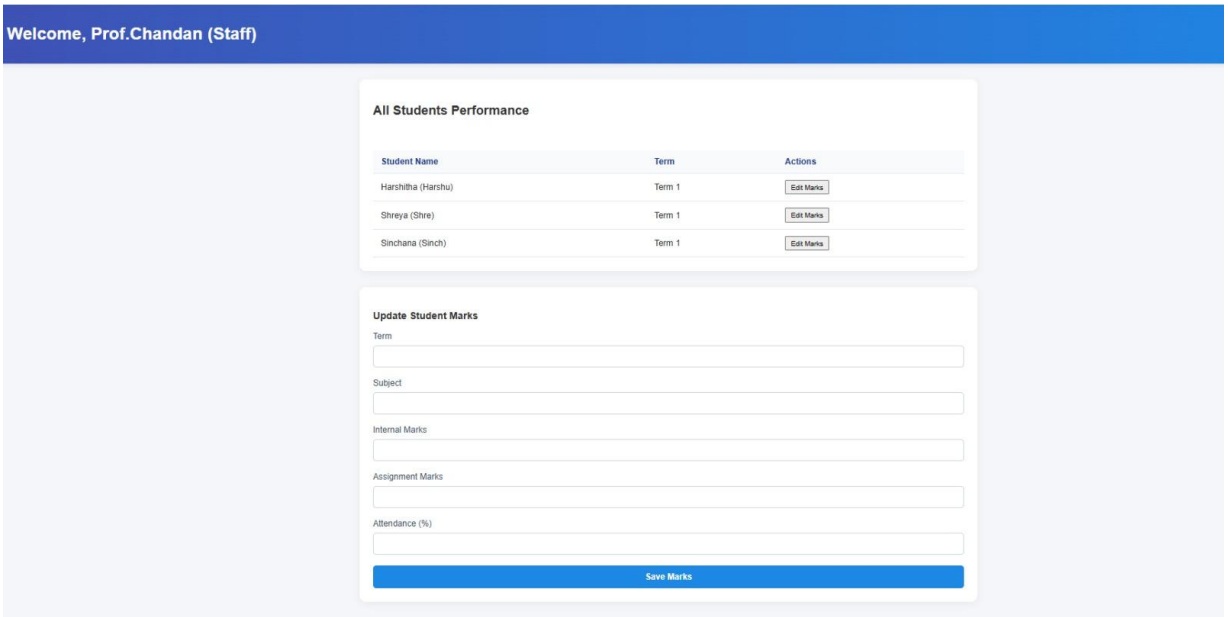
Dashboard:



Login page




Student Performance Tracker
Username
chandan
Password
••••••••
Login As
Staff
Login


Welcome, Prof.Chandan (Staff)
All Students Performance
Student Name
Term
Actions
Harshitha (Harshu)
Term 1
Edit Marks
Shreya (Shre)
Term 1
Edit Marks
Sinchana (Sinch)
Term 1
Edit Marks
Update Student Marks
Term
Subject
Internal Marks
Assignment Marks
Attendance (%)
Save Marks

. Challenges Faced

- Initial MongoDB connecting with frontend.
- Ensuring authentication works properly.
- Data validation for required fields
- Handling the data of every student.
- Updating the data
-

8. Conclusion

The **Student Performance Tracking System** provides an efficient and organized approach to managing student academic records using a NoSQL database. By utilizing MongoDB along with Node.js and Express.js, the system ensures flexible data storage, fast access, and easy scalability. The document-oriented data model allows efficient handling of student details, subject-wise marks, and attendance records. This system reduces manual effort, minimizes errors, and enables quick retrieval of performance data for staff and students. Overall, the project successfully demonstrates the practical use of MongoDB in building a scalable and efficient academic performance management system