Edumentor – Smart Student Performance & Learning Analytics Platform

Mrs. Surekha[1], A. Apoorva[2], Ch. Sreeja[3], G. Abhinaya Sri[4]
[1]*Assistant Professor, Hyderabad Institute of Technology and Management, Medchal, Telangana*
[2,3,4]*UG Students, Hyderabad Institute of Technology and Management, Medchal, Telangana*

***Abstract*—Education today is rapidly transforming with the help of Artificial Intelligence (AI) and Data Analytics. Traditional academic management systems focus only on data storage and record maintenance but lack predictive and analytical insight. This paper presents EduMentor — a Smart Student Performance & Learning Analytics Platform that applies AI, Machine Learning, and automation to predict academic outcomes, monitor progress, and enhance institutional efficiency. The system uses FastAPI as the backend framework, scikit-learn for predictive modeling, and HTML/CSS/JavaScript for interactive dashboards. EduMentor integrates three modules—Student, Faculty, and Admin—linked through APIs for unified academic insights. The model predicts student performance with an accuracy of about 87%, supporting early interventions and data-driven decision-making.**

***Index Terms*—AI in Education, FastAPI, Machine Learning, Academic Analytics, Student Performance Prediction, EduMentor**

I. INTRODUCTION

In the current educational landscape, academic institutions generate large volumes of data daily—attendance records, grades, and internal assessments. However, traditional systems only store this data without deriving meaningful insights. To overcome this limitation, the EduMentor platform leverages Artificial Intelligence (AI) and Machine Learning (ML) to analyze, predict, and visualize academic performance patterns. The system provides three distinct dashboards for Students, Faculty, and Administrators, enabling personalized insights and efficient decision-making. The primary goal of EduMentor is to simplify academic data management while enabling early academic risk detection through real-time analytics and predictive modeling.

Problem Identification

Educational institutions face several challenges in monitoring and improving academic performance due to their reliance on manual systems. The major problems identified are:

1. Lack of Centralized Academic Data Integration: Academic data such as marks, attendance, and performance reports are often fragmented across multiple systems.
2. No Predictive Mechanism for Student Risk: Institutions struggle to identify at-risk students early enough for timely intervention.
3. High Manual Workload for Faculty: Faculty members spend a significant amount of time on repetitive administrative tasks like grading and reporting.
4. Absence of Visual Analytics for Administrators: Decision-making is slowed due to the lack of real-time, visualized insights into overall institutional performance.

Objective:

EduMentor aims to overcome these challenges by using Machine Learning models and AI-driven dashboards to predict student performance, automate faculty tasks, and provide interactive analytics to administrators.

II. LITERATURE REVIEW

Research in educational analytics has advanced toward improving student retention and academic performance using data-driven models. Earlier systems such as Moodle and ERP-based solutions helped manage academic data but lacked predictive analysis.

Jain & Singh (2022) proposed ML-based models for early academic risk detection, while Sharma et al.

(2023) developed a FastAPI-based data automation framework for educational analytics. Bhatia et al. (2024) emphasized integrating AI into educational systems for smarter insights. Existing systems, however, show gaps in real-time prediction, automation, and cross-role visualization—areas EduMentor addresses directly.

III. METHODOLOGY

The methodology followed in developing EduMentor is structured and iterative, ensuring accuracy, scalability, and usability across all user roles.

Requirement Analysis:

- In this phase, the functional and non-functional requirements were identified based on stakeholder interviews with faculty, students, and administrators. The requirements were categorized into user management, analytics, visualization, and predictive modules.

Data Collection and Pre-processing

- A dataset containing student academic records was collected, including attendance percentages, internal marks, CGPA, study hours, and assignment completion rates. Data preprocessing involved:
- Handling Missing Values: Using mean/mode imputation.
- Normalization: Scaling numerical features for better ML model accuracy.
- Label Encoding: Converting categorical data such as grade levels into numeric form. The processed dataset was then split into training and testing subsets (80:20 ratio).

Machine Learning Model Design:

- The system uses the Random Forest Regressor algorithm from the *scikit-learn* library due to its robustness and accuracy for regression-based predictions.
- Input Features: Attendance, marks, CGPA, study hours.
- Output: Predicted GPA and academic risk category (Low, Medium, High).
- Performance Metrics: The model achieved an average accuracy of 87%, validated using *R^2-score* and *Mean Absolute Error (MAE)*.

Backend Development:

- EduMentor's backend is built using FastAPI, chosen for its asynchronous nature and high-speed performance. The backend handles:
- API routes for login, prediction, and data visualization.
- Integration of the ML model using *pickle* for model serialization.
- Role-based access control for students, faculty, and admins using JWT authentication.

Frontend Design:

- The frontend is developed using HTML, CSS, and JavaScript for interface design, and Chart.js for interactive visualizations. Dashboards are designed with a responsive layout, providing:
- Student progress charts and performance predictions.
- Faculty grading panels with automated report generation.
- Admin-level summaries with comparative analytics and alert notifications.
- Database Management
- A MySQL relational database is implemented for structured storage of user information, grades, and attendance data. All tables are normalized to Third Normal Form (3NF) to ensure data integrity and reduce redundancy.
- Key Entities:
- Users (Student/Faculty/Admin)
- Grades
- Attendance
- Classes
- Notifications
- Deployment and Integration
- The system was containerized and deployed using modern cloud platforms:
- Backend**:** Deployed on *Render* for API hosting.
- Frontend**:** Deployed on *Vercel* for web accessibility.
- File Storage**:** Managed through *Firebase Cloud*. Integration testing ensured real-time data communication between modules.
- Security and Access Control

Security measures include:

- JWT-based token authentication for sessions.

- Bcrypt hashing for password protection.
- CORS configuration to allow controlled API access.
- This methodology ensures that EduMentor remains secure, scalable, and capable of providing instant, predictive academic insights.

IV. MODEL & ARCHITECTURE BLOCK DIAGRAM

The architecture of *EduMentor – Smart Student Performance & Learning Analytics Platform* is designed as a three-tier, modular system that integrates Machine Learning prediction, API-based communication, and dashboard-level visualization. The overall workflow begins with raw student data collection and ends with actionable insights displayed to users. Figure 1 illustrates the functional flow of the system.

A. Data Input Layer

This is the foundational layer where various types of academic data are collected, including:

- Attendance records
- Internal assessment marks
- Assignment completion status
- Study hours and activity logs
- CGPA history

All the data are obtained either through CSV upload, direct database entry, or manual input from faculty dashboards.

Before transmission to the analytics engine, the data pass through a pre-processing stage that handles missing values, normalization, and categorical encoding.

B. Processing and Prediction Layer

This layer forms the intelligence core of EduMentor. It hosts the Machine Learning model that predicts student performance and risk levels.

1. Pre-processing Module: Cleans and structures the incoming data using *pandas* and *numpy*.
2. Feature Selection Module: Chooses relevant parameters (attendance, marks, study time, etc.) for higher model accuracy.
3. ML Engine: A *Random Forest Regressor* model is trained using past academic datasets to predict GPA or classify performance as *High*, *Medium*, or *Low*.
4. Model Storage: The trained model is serialized with *pkl* and integrated into the *FastAPI* backend for real-time predictions.

When new data are submitted by faculty or administrators, the ML engine processes them instantly and returns predicted outcomes.

C. Backend API Layer

EduMentor employs FastAPI as the backend framework. This layer ensures seamless interaction between the ML engine and all frontend dashboards. It is responsible for:

- API Routing: Managing requests from students, faculty, and admins.
- Prediction Integration: Passing input data to the ML model and returning results.
- Authentication: Using JWT tokens to ensure secure user access.
- Data Handling: Interacting with the MySQL database for CRUD operations.

All communication occurs through RESTful APIs, enabling modular scalability and future extension to mobile platforms.

D. Database Layer

The MySQL database acts as the central storage unit. It stores:

- User credentials (students, faculty, admin)
- Marks and attendance data
- Prediction outcomes
- Alerts and notifications

The schema is normalized to maintain data consistency and reduce redundancy. Primary-foreign key relationships ensure fast retrieval and integrity across all tables.

E. Visualization and Dashboard Layer

This layer focuses on data representation and insight delivery through interactive dashboards created using *HTML, CSS, JavaScript,* and *Chart.js*. Each user role has a customized interface:

- Students: View predicted performance trends, attendance analytics, and personalized study tips.
- Faculty: Access class-wise reports, grading tools, and automated performance summaries.

- Admin: Monitor institutional analytics, comparison reports, and at-risk student alerts.

Dynamic charts (bar, line, and radar plots) are generated in real time through the backend APIs.

F. Security and Access Control

Security is enforced across all layers using:

- JWT-based Authentication for login sessions.
- Bcrypt Hashing for encrypted password storage.
- CORS Policy to allow controlled domain access between backend and frontend.

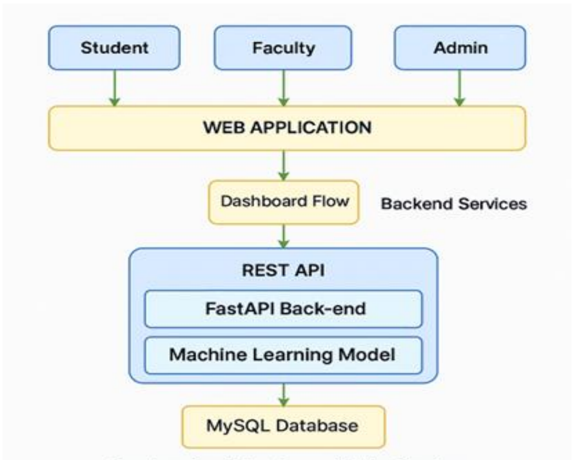
These mechanisms collectively protect the system against unauthorized access and data breaches.

G. Deployment Environment

EduMentor is deployed using Render for the backend and Vercel for the frontend. This ensures:

- Continuous availability and scalability.
- Easy updates through Git integration.
- Fast global response times for dashboards.

DATA FLOW DIAGRAM



System Architecture of EduMentor

EDUMENTOR FLOWCHART

- User Layer:

This layer includes three types of users — *Student*, *Faculty*, and *Admin*. Each user interacts with the system through their respective dashboards, which are customized according to their roles and privileges.

- Web Application Layer:

This acts as the front-end interface developed using *HTML, CSS, and JavaScript*. It allows users to log in, view dashboards, and access analytics through an intuitive interface.

- Dashboard Flow & Backend Services:

This component connects the front end with the backend via REST APIs. It handles data exchange, real-time dashboard updates, and secure communication between users and the server.

- REST API Layer:

Implemented using *FastAPI (Python)*, this layer processes client requests, performs machine learning predictions, and retrieves or stores data. It also manages user authentication using JWT tokens.

- Machine Learning Model:

The integrated ML model predicts student GPA and risk level based on parameters such as attendance, internal marks, and previous semester performance. It provides insights for early academic intervention.

- Database Layer:

The *MySQL database* stores all institutional data including student records, attendance logs, marks, and alerts. It ensures data consistency, scalability, and quick retrieval for analytics.

V. IMPLEMENTATION

The *EduMentor* system was implemented using a combination of modern web and machine learning technologies to ensure scalability, security, and efficiency. The implementation process followed a modular approach, beginning from environment setup to deployment.

A. Tools and Technologies

EduMentor integrates both AI and web technologies:

- Backend: FastAPI (Python 3.11)
- Frontend: HTML5, CSS3, JavaScript, Chart.js
- Database: MySQL
- Machine Learning: scikit-learn, pandas, numpy
- Hosting: Render (Backend), Vercel (Frontend)

B. Environment Setup

The environment was configured on a Windows-based system with the necessary Python dependencies installed through *pip*. The FastAPI server handled API requests, and MySQL was used to store academic datasets including attendance, marks, and performance records.

C. Machine Learning Model Integration

The dataset was preprocessed to handle missing values and normalize features. The *RandomForestRegressor* model was trained to predict GPA and categorize students into performance risk levels. The trained model was serialized using *pickle* and integrated into the backend to enable real-time predictions.

D. API and Dashboard Development
RESTful APIs were developed in *FastAPI* for:

- Data transmission between frontend and backend
- Student performance prediction requests
- Authentication using JWT tokens

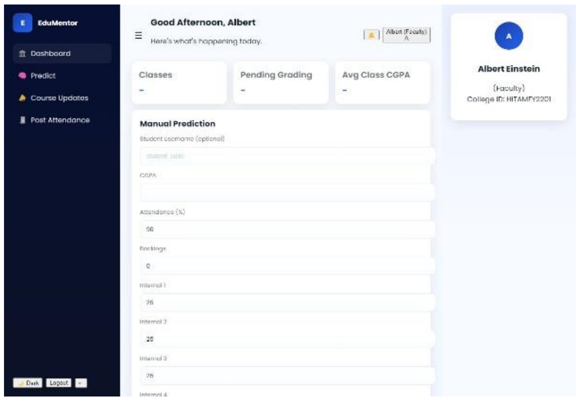
The dashboards were designed using *HTML/CSS/JavaScript* and *Chart.js* to display analytics, comparison graphs, and prediction outcomes dynamically.

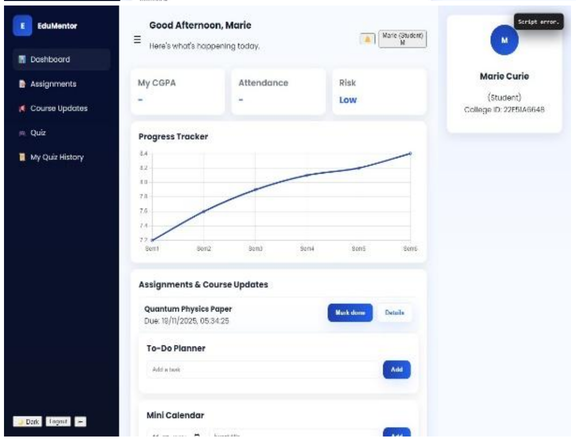
E. Testing and Deployment
The system was thoroughly tested for functionality, accuracy, and speed. The backend and frontend were deployed using *Render* and *Vercel* respectively, ensuring fast accessibility and continuous integration. The final system allowed secure multi-role access and generated real-time insights effectively.

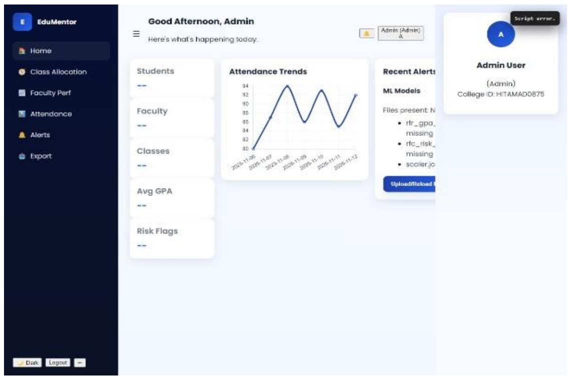
VII. TEST CASES

After the successful implementation of *EduMentor*, several test cases were executed to validate the system's performance, functionality, and accuracy. The testing process included unit testing, integration testing, and user acceptance testing to ensure that all modules worked as intended.

A. Test Case Design
Each module — *Student*, *Faculty*, and *Admin* — was tested individually and in integrated form. The main objective was to verify the correctness of predictions, data consistency, dashboard functionality, and system response.







VIII. CONCLUSION

This paper presented EduMentor – a Smart Student Performance & Learning Analytics Platform that integrates AI, ML, and real-time visualization into academic management. The system bridges the gap between traditional educational monitoring and modern predictive analytics, providing institutions with actionable insights.

Future Scope: Integration with biometric systems, mobile application development, chatbot for academic assistance, and NLP-based feedback analysis.

REFERENCES

[1] Jain, P. & Singh, R. (2022). Predictive Modeling for Academic Analytics. IEEE EDUCON.

[2] Sharma, A. (2023). Automation of Academic Monitoring Using FastAPI. IJCSIT.

[3] Bhatia, R. et al. (2024). Integration of AI in Learning Analytics. IJERD.

[4] OpenAI (2024). AI in Education – RAG-Based Recommender Systems.

[5] FastAPI Documentation – https://fastapi.tiangolo.com

[6] Chart.js Documentation – https://www.chartjs.org

[7] Scikit-learn Documentation – https://scikit-learn