A Project Report on

# STUDENT PERFORMANCE IN TRAININGS

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

**Bachelor of Technology**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

Submitted by

D.SAI KIRAN
(20H51A05G7)

B. NIKHIL
(20H51A05B7)

A. HARI PRIYA
(20H51A0582)

Under the esteemed guidance of

Dr.Siva Skandha Sanagala
(Associate Professor and HOD)

**Department of CSE**

**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

(UGC Autonomous)

*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A$^+$ Grade

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

**2023- 2024**

**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

# CERTIFICATE

This is to certify that the Major Project report entitled **"Student Performance in Trainings"** being submitted by **D. Saikiran (20H51A05G7), B .Nikhil (20H51A05B7), A. Hari Priya (20H51A0582)** in partial fulfillment for the award of **Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING** is a record of bonafide work carried out under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

| | | |
|---|---|---|
| **Dr. Siva Skandha Sanagala** | **Dr.Siva Skandha Sanagala** | **EXTERNAL EXAMINER** |
| **Associate Professor and HOD** | **Associate Professor and HOD** | |
| **Dept. of CSE** | **Dept. of CSE** | |

# ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project a grand success.

We are grateful to **Dr.Siva Skandha Sanagala (Associate Professor and HOD)**, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank**, Dr.Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete our project work successfully.

We are very grateful to **Dr. Ghanta Devadasu**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Dept Name for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary& Correspondent, CMR Group of Institutions, and Shri Ch Abhinav Reddy, CEO, CMR Group of Institutions for their continuous care and support.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly or indirectly in completion of this project work.

| | |
|---|---|
| D. Sai Kiran | 20H51A05G7 |
| B. Nikhil | 20H51A05B7 |
| A. Hari Priya | 20H51A0582 |

# TABLE OF CONTENTS

| CHAPTER NO. | TITLE | PAGE NO. |
|---|---|---|
| | LIST OF FIGURES | ii |
| | ABSTRACT | iii |
| **1** | **INTRODUCTION** | 1 |
| | 1.1. Problem Statement | 2 |
| | 1.2. Research Objective | 2-3 |
| | 1.3. Project Scope and Limitation | 4 |
| **2** | **BACKGROUND WORK** | 5 |
| | 2.1. Manual Systems | 6-7 |
| | 2.1.1. Introduction | 7 |
| | 2.1.2. Merits,Demerits and Challenges | 7 |
| | 2.1.3. Implementation of Manual System | 7-8 |
| | 2.2. Online Coding Platforms | 8 |
| | 2.2.1. Introduction | 8 |
| | 2.2.2. Merits,Demerits and Challenges | 9 |
| | 2.2.3. Implementation of online Coding Platform | 9-10 |
| **3** | **PROPOSED SYSTEM** | **11** |
| | 3.1. Objective of Proposed Model | 12 |
| | 3.2. Implementation of Proposed Model | 12-13 |
| | 3.3. Designing | 13-14 |
| | 3.3.1.UML Diagram | 15 |
| | 3.4. Stepwise Implementation and Code | 16-30 |
| | 3.4.1 Frontend and Backend Development | |
| **4** | **RESULTS AND DISCUSSION** | **31** |
| | 4.1. Performance metrics | 32-34 |
| **5** | **CONCLUSION** | **35** |
| | 5.1 Conclusion and Future Enhancement | 36 |
| 6 | **REFERENCES AND CODE** | **37-48** |

## List of Figures

| FIGURE NO. | TITLE | PAGE NO. |
|---|---|---|
| 3.3.1 | Architecture | 15 |
| 3.3.2 | UML Diagram of Department and Admin | 15 |
| 3.3.3 | UML Diagram of Student | 15 |
| 3.3.4 | UML Diagram of TPO | 16 |
| 3.4.1 | Stepwise Implementation and Code | 16 |
| 3.4.2 | Dashboard | 27 |
| 3.4.3 | Admin login page | 27 |
| 3.4.4 | Admin Dashboard | 28 |
| 3.4.5 | Overall Student Performance Analysis | 28 |
| 3.4.6 | Faculty login | 29 |
| 3.4.7 | create test | 29 |
| 3.4.8 | upload marks | 30 |
| 3.4.9 | upload Attendance | 30 |
| 3.4.10 | faculty profile | 30 |
| 3.4.11 | Student login | 31 |
| 3.4.12 | Student performance analysis | 31 |
| 3.4.13 | Student check his marks | 32 |
| 3.4.14 | Student check his attendance | 32 |
| 4.1.1 | Student Graph Analysis | 34 |
| 4.1.2 | TPO Graph Analysis | 35 |

## ABSTRACT

Student performance in training programs is a critical area of concern in education and professional development. This study investigates the multifaceted factors influencing student performance and explores effective strategies to enhance learning outcomes in various training contexts. A comprehensive literature review reveals the interplay of individual and environmental factors, including prior knowledge, motivation, teaching methods, resources, and support systems. The findings suggest that motivation, relevance of training, and the quality of instructional methods for student performance. The study highlights the importance of creating a supportive learning environment and optimize the effectiveness of training programs.

The project aims to analyses the performance of the student in training that include the no of trainings he/she attended, filtering and separation of students based on their performance. We can able to understand training performance of students or particular his/her training and the particular data is analyzed represented in the form graphs. In this project the particular student can also gave feedback to the trainings that he/she attended and can able to know the type of trainings, when it starts and the training and placement department can share the materials through the website. It is web application constitute on MERN stack these technologies include Mongo dB, Express, Reacts, Nodejs.

# CHAPTER 1
# INTRODUCTION

## 1.1. Problem Statement

The primary objective of this study is to delve into the root causes of the challenges and issues faced by students during their assessment process and training sessions. By identifying and understanding these issues, the research aims to assess their impact on students' overall performance within the training program. This problem statement serves as the foundation for conducting a comprehensive investigation, which will ultimately lead to the development of effective strategies to enhance the effectiveness of the training program.

In order to achieve these goals, the research adopts a mixed-methods approach that combines both quantitative and qualitative analyses. The quantitative analysis focuses on objective data and performance metrics, such as aptitude scores, to provide a numerical assessment of student capabilities. Additionally, it examines verbal learning skills, enabling the research to gauge linguistic and language-related aptitude that might influence performance.

By integrating both quantitative and qualitative data, the research aims to provide a holistic view of the challenges faced by students and how these challenges impact their performance. This comprehensive assessment will enable the development of targeted strategies and interventions to improve the training program's overall effectiveness. The ultimate goal is to create a learning environment that supports student success and addresses their unique needs, resulting in enhanced performance and outcomes.

## 1.2. Research Objective

By using the student performance in trainings, we can detect the number of students who can participate in the trainings, attendance of the students, number of students attended for the assessments etc. Adequate resources and support systems provided during training will enhance student performance. Analyzing student performance in trainings allows educators and administrators to gain valuable insights into various aspects of student engagement and achievement. The following will give clear insights about the performance of students in trainings

### 1.2.1 Individual Student Performance in Training Programs:

Individual student performance data is a fundamental component of the research. This data includes records of how each student is faring in the training programs. It may encompass various performance metrics, such as test scores, project completion, attendance, and participation.

### 1.2.2 Grades, Scores, or Assessments Related to Different Training Modules:

This category of data pertains to the specific evaluation of student performance in various training modules or courses within the program. It includes grades, scores on assignments and exams, and assessments that measure a student's understanding and mastery of the subject matter. By analyzing these scores, you can assess how students are progressing in different areas and pinpoint the modules where they may be struggling.

### 1.2.3 Information on Resources Provided During Training:

The data related to the resources provided during training is essential for understanding the learning environment. This includes details about the materials, equipment, and technology made available to students. It's vital to evaluate whether students have access to the necessary tools and resources to support their learning. Any limitations or disparities in resource access can be identified and addressed to enhance the overall training experience.

### 1.2.4 Reports:

Reports from course evaluation, feedback from instructors, and program assessments provide a qualitative perspective on the training program. These reports Collectively, these data types offer a comprehensive view of student performance, their ultimately leading to the development of strategies to enhance the quality and outcomes of the training program.

## 1.3. Project Scope and Limitations

- MERN stack allows for easy scalability. As your user base grows, you can seamlessly expand your system to handle more traffic and users.
- Students can able to give feedback to lectures of trainers in this way the management can get know if the trainer is efficient.
- React Native, which is based on React.js can be used to develop mobile applications for both iOS and Android platforms.
- MongoDB's flexibility allows you to store and analyze vast amounts of user data. By analyzing user behavior, helps in graphical analysis.

# CHAPTER 2
# BACKGROUND WORK

# CHAPTER 2
# BACKGROUND WORK

We started conducting research on various fields, such as academic databases, journals, and books for related studies, and analysed key theories and frameworks that were used for analysing the topic. We identified the key factors that are involved in student performance in training, such as teaching methods, learning environment, finding students who are not good at particular topics, and pushing or motivating them. We have researched the historical development of training and education systems. Understand how training programs have evolved over time and the impact of changing educational philosophies and technologies on student performance.

Some of the topics we have gone through

A Critical Analysis of MERN Stack Application in Educational Environments: Implications for Student Engagement and Learning Outcomes [1], it says the role of MERN application in education settings and the combination of React, Express, Node, and MongoDB influences student engagement programs [2].

A Comprehensive Factor Affecting Student Performance in Training Programs: It says how various factors influence student performance in training, examining approaches, and technological approaches [3].

## 2.1 Factors Affecting Student Performance:

Investigate the various factors that can affect student performance in training programs, such as: Individual factors (e.g., cognitive abilities, learning styles, prior knowledge, motivation). Environmental factors (e.g., teaching methods, resources, support systems). Curriculum and program design[4] (e.g., content relevance, assessment methods). Technological advancements and their impact on training[5]. Look for examples of best practices and effective strategies for improving student performance in training.

## 2.2 Manual Systems

### 2.2.1 Introduction

The introduction section provides an overview of the manual records system used for analyzing student performance. In educational institutions, manual records have traditionally been the primary method for storing and organizing student data, including marks and other relevant information [5-6]. Faculty members are typically responsible for entering and managing this data, organizing it by department for ease of access and analysis by management. However, manual records have limitations, including the potential for human error in data entry and the lack of robust security measures to protect sensitive information [6]. As technology advances, educational institutions are exploring digital solutions to overcome these drawbacks and improve the efficiency and security of student performance analysis.

### 2.2.2 Demerits, and Challenges

The demerits, and challenges section delves into the strengths and weaknesses of the manual records system the drawbacks, including the risk of human error and the lack of security measures[7]. Additionally, this section explores the challenges faced by educational institutions in maintaining and managing manual records effectively, particularly in the context of increasing data volumes and security concerns[8-9]. It highlights the need for modernization and explores potential solutions to address the limitations of the manual records system[10].

### 2.2.3 Implementation of Manual System

The implementation section outlines the process of implementing the manual records system within educational institutions. It covers various aspects, including data collection, entry, organization, and analysis. Once the data requirements are defined, faculty members or designated staff members are responsible for entering the data into the manual records system. Organizing the data in a logical and systematic manner is essential for easy retrieval and analysis

Additionally, this section discusses the roles and responsibilities of faculty members involved in managing manual records, as well as any tools or resources utilized to streamline the process[10-11]. Furthermore, it may explore best practices and recommendations for effectively implementing and maintaining manual records to ensure accurate and reliable data for student performance. In this context,

## 2.3 Online coding platform (Hacker rank)

### 2.3.1 Introduction

It Introduces the MERN Stack and its relevance to contemporary web development. It may describe the MERN Stack components (MongoDB, Express.js, React.js, Node.js) and their individual roles in building modern web applications[12]. The described application is a comprehensive platform for coding skills and increasing their problem-solving abilities. It offers users a wealth of coding challenges, categorized by data structures like linked lists, trees, and graphs, enabling them to refine their coding skills. These problems span various difficulty levels, making it an ideal space for both novices and seasoned programmers to practice and expand their coding horizons[12-13]. Additionally, the platform hosts global coding contests where users can compete against participants from around the world.

### 2.2.1 Merits

The MERN Stack offers several significant advantages that make it a popular choice among developers for building modern web applications

### 1. Problem Solving and Practice:

The application offers a wide range of coding problems, categorized by data structures and algorithms[13]. Users can choose problems of varying difficulty levels to challenge themselves Users can practice problem-solving, sharpen their coding skills, and gain confidence in tackling real-world coding challenge

### 2. Learning and Improvement

By providing problems related to different data structures, the application helps users learn and understand how to apply these structures effectively[13-14]. Users can experiment with different algorithms and approaches to solve problems, gaining a deeper understanding of computer science concepts.

### 3 Skill Diversification & knowledge sharing:

Users have the opportunity to explore a wide range of coding problems, which helps them diversify their skill set They can move from problems related to basic data structures to more advanced topics like dynamic programming, graph algorithms, and more. The application often includes discussions, forums, or chat features where users can collaborate, ask questions, and share insights about solving problems

### 2.3.1 Demerits, and Challenges

1. It is not dedicated to particular organizations or community members; in this context, the performance is not evaluated by any other person or external individuals[15-16].
2. This platform sometimes susceptible to data breaches like external gain access to sensitive information present in our account
3. Some coding platform like exhibit student performance analysis weak. For instance, certain may lack comprehensive tool or features like graphs to effectively evaluate the student performance

### 2.3.2 Implementation of coding platform:

It develops into the practical application of the MERN Stack to address specific challenges or requirements[15]. It may provide case studies, code examples, or best practices for implementing the MERN Stack in real-world projects

React is frontend Technology it contains components and props and Hooks

**1. Components:**

In React, the user interface is broken down into smaller, reusable pieces called components [16]. Each component is responsible for rendering a part of the user interface. For example, in the Hacker Rank platform, you might have components for user authentication, problem statements, code editors.

**2. Props (Properties):**

Props are a mechanism for passing data from a parent component to its child components. They are used to customize and configure how a component behaves or what it renders[16-17]. In the context of the Hacker Rank platform, props might be used to pass information like problem descriptions, user profiles, or contest details to various UI component[17]

**3. State:**

React components can have state, which is used to store and manage data that can change over time. State is typically used to handle user interactions and dynamic content For example, in coding practice sections [18], React might use state to manage code input, track user progress, and update the user interface in real-time as the user solves problems. React introduced hooks, such as `use State` and `use Effect`, to manage component state and side effects in functional components[18].

- Backend Like Node.js is used to create a backend environment to run the server helpful in creating Http request and response from the server and also developing API
- The Existing System uses Mango DB for data access like Authentication[19], features like problem solving and express is a framework helps Nodejs to Work on API(Application Programming Interface)
- MongoDB is a NoSQL database used in the MERN Stack for data storage and access[20]. It differs from traditional relational databases by providing a flexible, schema-less structure, which is particularly advantageous for applications where data schemas evolve over time.

# CHAPTER 3
# PROPOSED SYSTEM

# CHAPTER 3
# PROPOSED SYSTEM

## 3.1. Objective of Proposed System

The objective of the proposed model is to develop a comprehensive system for analyzing and evaluating student performance within training programs. This system will leverage a range of metrics, including test scores, class attendance, and participation levels, to gain insights into the effectiveness of training initiatives and the factors influencing student outcomes. By collecting and analyzing these metrics, the model aims to provide stakeholders with valuable insights into student performance trends, enabling them to identify areas for improvement and make informed decisions to optimize the training experience. Through rigorous analysis of student performance data, management will be able to gain a deeper understanding of the effectiveness of training programs. They will be able to identify patterns and trends in student performance, pinpoint areas of strength and weakness, and identify factors that contribute to student success. Armed with this information, stakeholders can make data-driven decisions to enhance the overall effectiveness and impact of training programs.

Furthermore, the proposed model aims to facilitate continuous improvement efforts within training programs. By providing stakeholders with actionable insights derived from data analysis, the model empowers them to implement targeted interventions and strategies to address areas of concern and enhance student outcomes. This iterative process of analysis, intervention, and evaluation enables training programs to evolve and adapt over time, ensuring that they remain relevant and effective in meeting the needs of students.

## 3.2 Implementation of Proposed Model

The implementation of a student training portal that empowers students to track their progress has introduced significant benefits. This system enables students to gain a clear overview of their performance throughout the training program it allows to access review their grades, scores, and assessment results, providing a real-time understanding of their academic achievements and areas where they may need improvement. This not only enhances students' awareness of their own progress but also fosters a sense of accountability and self- directed learning. It encourages students to take ownership of their education and set personal goals for improvement. This feature is particularly useful for self-assessment and for students to tailor their study strategies to address their individual strengths and weaknesses .Additionally, MongoDB's flexibility for storing and analyzing user data has been instrumental in facilitating graphical analysis of user behavior within the training portal. The database's ability to handle unstructured and evolving data makes it well-suited for this purpose. With graphical analysis, the system can generate visual representations of how students interact with the platform. This includes tracking which training modules they engage with the most, the duration of their sessions, and patterns of access to learning materials. By leveraging these insights, the system can make data-driven recommendations in essence, this data-driven analysis helps tailor the training program to better suit the needs and preferences of the students.

## 3.3 Designing

The architecture of student performance in trainings contains various roles these helps easy of understanding the flow of process of our website they are

**Roles**

1.TPO (Training and Placement officer)

2.HOD (Head of Department)

3.Students

4.Admin

## TPO

In the context of a comprehensive training program, several key responsibilities come into play. The schedule for training planning, coordination, and execution is a fundamental element that ensures the program runs smoothly, guiding the sequence of activities and maintaining deadlines. Finally, gathering feedback from students is an indispensable process that helps evaluate the program's efficacy.

## HOD

Assessing individual performance allows educators to tailor their approach, recognizing high-achieving students and providing assistance to those who may be struggling. Meanwhile, the overall departmental graph offers a bird's-eye view of department-wide progress, enabling administrators to make strategic decisions, allocate resources efficiently, and drive improvements in the training program.

## Students

His feature allows students to monitor their progress, identify areas where they excel, and pinpoint those in need of improvement. It fosters a sense of ownership over their educational outcomes, encouraging them to set personal goals and track their achievements Furthermore, students can leverage the system for seeking improvements in their academic pursuits.

## Admin

The role of an administrator in the educational system is multifaceted, and it involves crucial responsibilities that ensure the smooth operation and management of various aspects. Administrators play a pivotal role in maintaining student data, overseeing the integrity and security of records, and managing the information that underpins the educational process. Overall, administrators are central figures in the educational ecosystem, wielding the power to facilitate student success and career advancement.

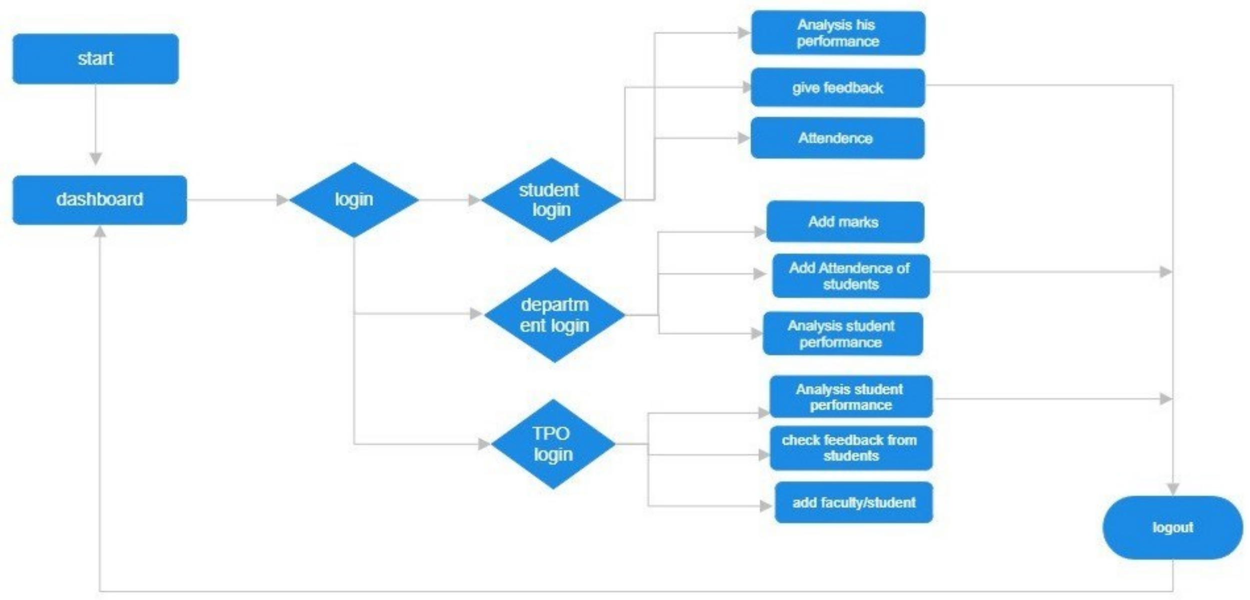


Fig 3.3.1. Architecture

## UML Diagrams

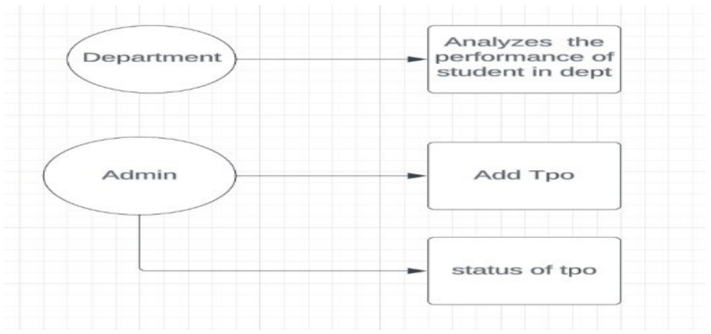


Fig 3.3.2 UML diagram of Department and ADMIN

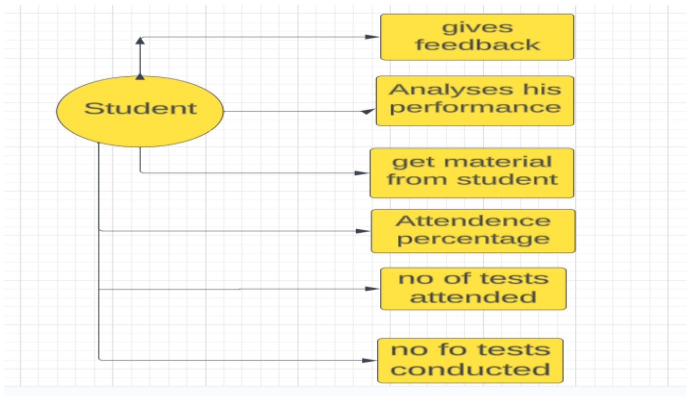


Fig 3.3.3 UML diagram of Student

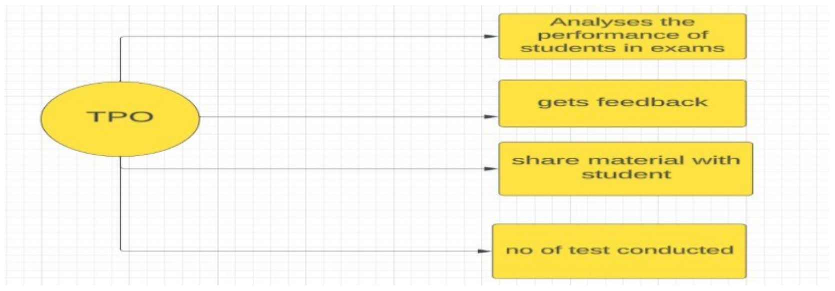

TPO
Analyses the performance of students in exams
gets feedback
share material with student
no of test conducted

Fig 3.3.4 UML diagram of TPO

## 3.4 Stepwise Implementation and Code

The frontend, built using React.js, is responsible for presenting the application's user interface and interacting with users. It comprises a collection of components that render HTML elements, handle user input, and manage state. Through Reacts component-based architecture and virtual DOM, developers can create dynamic and responsive UIs that update efficiently in response to user action and it uses react redux state management On the other hand, the backend, powered by Express.js and Node.js, serves as the foundation for data processing, storage, and business logic. It handles incoming HTTP requests from the frontend, processes data, interacts with the database, and sends responses back to the client

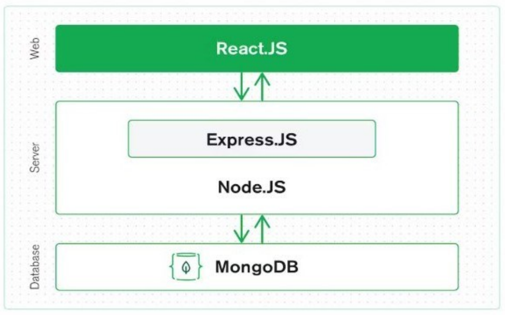


Fig 3.4.1 Working of MERN Project

The above figure shows the workings of MERN project. React interacts with the Express framework, moving the data from the frontend to the backend through APIs (application program interfaces).

Reacts component-based structure allows for the modular organization of these requests, facilitating a clear and organized approach to data retrieval and manipulation. Upon receiving these requests, the Express.js server processes the incoming data and executes the necessary actions. This can include querying the MongoDB database to retrieve specific data records, up-dating existing data entries, or performing other server-side operations based on the request parameters. Express.js' routing capabilities allow for the mapping of incoming requests to spe-cific controller functions, ensuring that each request is handled appropriately and efficiently.

The communication between Express.js and MongoDB forms the backbone of the backend data management process. Through the Mongoose ODM (Object Data Modeling) li-brary, Express.js interacts with the MongoDB database, allowing for seamless manipulation of data stored in the database. This interaction involves executing MongoDB queries, updating documents, and performing other database operations as required by the incoming re-quests.Overall, this exchange of data between the frontend and backend components facilitates a dynamic and responsive user experience, where user interactions trigger server-side actions, and the resulting data updates are reflected in real-time on the frontend interface. By leveraging the capabilities of both frontend and backend technologies, MERN stack applications are able to deliver efficient, scalable, and feature-rich solutions for a wide range of use cases.

### 3.4.1 Frontend and Backend Development:

#### 1. Frontend Development:

The Frontend part is developed by ReactJs and it react redux used for state management and integrating React Redux for state management enhances the efficiency and scalability of the application. React Redux is a library that provides a predictable state container for man aging the application's state in a centralized manner. Here's how React Redux is used for state management in the frontend:

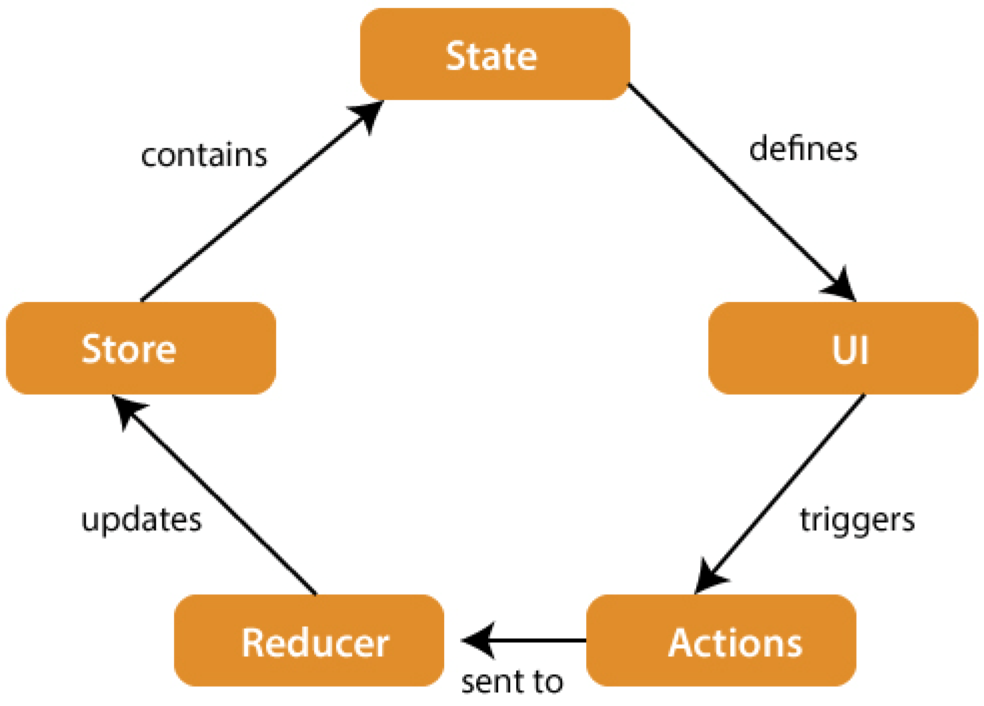
1. **Setup Redux Store**: The first step is Setting up the Redux store is the foundational step in integrating Redux for state management in a React.js application. The Redux store serves as a centralized container that holds the entire state tree of the application, allowing any component within the application to access and interact with the state. This centralized approach streamlines state management, as it eliminates the need to pass state down through multiple levels of component hierarchy using props. By initializing the Redux store, developers create a single source of truth for application state, enabling consistent and predictable data access and updates across the entire application. Additionally, the Redux store facilitates efficient data retrieval and manipulation, enhancing performance and scalability in complex React.js applications.
2. **Define Reducers**: Reducers play a crucial role in managing state changes within a Redux-based React.js application. These pure functions are responsible for processing actions dispatched by components, determining how the application's state should be updated, and producing a new state object accordingly. Each reducer takes the current state and an action object as input, and based on the type of action, it computes and returns a new state object. To manage multiple slices of the application's state, reducers are combined using the combine Reducers utility provided by Redux. This utility enables developers to organize and manage reducers efficiently, allowing them to handle specific slices of state in a modular and maintainable manner.
3. **Create Actions**: Actions Actions serve as messengers that carry information from components to the Redux store, triggering state updates. These plain JavaScript objects encapsulate data relevant to state modifications, with a mandatory type property defining the action's purpose. Action creators, functions responsible for generating actions, streamline this process by encapsulating the action creation logic. When invoked, action creators return action objects, providing a structured and consistent approach to dispatching actions within a Redux-based React.js application.

4. **Connect Components to Redux Store**: Connecting components to the Redux store is essential for accessing state and dispatching actions within a React.js application. This linkage is established using either the connect function or React Redux hooks like useSelector and use Dispatch. By connecting components to the Redux store, developers empower them to access relevant data from the store and dispatch actions to modify the state. This seamless integration facilitates efficient data flow and enables components to respond dynamically to changes in the application's state, fostering a robust and interactive user experience.

5. **Dispatch Actions**: Dispatching actions is a fundamental mechanism for initiating state changes within the Redux store in a React.js application. Components utilize the dispatch function, provided by the connection to the Redux store, to dispatch actions. Once dispatched, actions are transmitted to all reducers, which in turn process the action and update the application's state accordingly. This orchestration ensures that any changes to the state are handled uniformly and consistently across the application, maintaining the integrity of the data flow and facilitating a predictable state management process.

6. **Access State:** Accessing state in connected components is a fundamental aspect of Redux-based React.js applications, facilitated through either the mapStateToProps function or the useSelector hook. These mechanisms enable components to retrieve specific slices of state from the Redux store, providing access to relevant data for rendering the user interface. By accessing state in this manner, components can dynamically adapt their appearance and behavior based on changes in the application's state, ensuring a synchronized and responsive user experience. This seamless integration of state management empowers developers to create more interactive and data-driven UIs while maintaining a clear separation of concerns between state management and presentation logic.

7. **Update State**: Components can update the state stored in the Redux store by dispatching actions. When an action is dispatched, it triggers the corresponding reducer, which updates which then processes the action and updates the state accordingly. This process ensures that state modifications follow a predictable and controlled flow, as defined by the reducers. By dispatching actions to update the state, components can trigger specific state changes throughout the application, enabling a cohesive and synchronized user experience. This approach to state management fosters a clear separation of concerns, with components responsible for triggering state updates while reducers dictate how the state should be modified based on the dispatched actions



3.4.15 React -redux working

8. **7. Middleware** Redux extends the core functionality of the Redux store, allowing developers to incorporate additional features such as handling asynchronous actions and logging. Redux Thunk is a popular middleware used for managing asynchronous operations within Redux, enabling action creators to return functions that can dispatch actions asynchronously. This facilitates tasks like fetching data from APIs or performing asynchronous operations before dispatching relevant actions. On the other hand, Redux Logger provides a convenient way to log actions and state changes, aiding in debugging and monitoring application behavior. By integrating middleware like Redux Thunk and Redux Logger, developers can enhance the capabilities of Redux-based applications, streamline development workflows, and ensure effective management of complex scenarios.

## Backend Development

Backend development in a MERN (MongoDB, Express.js, React.js, Node.js) stack application revolves around constructing the server-side components responsible for handling data logic, routing, and database interactions. Node.js serves as the runtime environment for executing JavaScript code on the server, while Express.js, a minimalist web application framework for Node.js, facilitates the creation of robust and scalable web servers and APIs. Interactions with the MongoDB database, a NoSQL database, are managed through Express.js using libraries like Mongoose, which provides a schema-based solution for modeling application data and simplifies the process of querying and manipulating data in MongoDB.

1. **Setup Node.js and Express.js:** Begin by Setting up a Node.js environment and installing Express.js is the foundational step in backend development for a MERN stack application. Node.js acts as the runtime environment, allowing JavaScript code to run server-side. Express.js, being a minimalist and flexible web application framework built on top of Node.js, simplifies the process of creating web servers and APIs by providing a rich set of features and middleware.

3 **Node.js and Express.js**: Setting up a Node.js environment and incorporating Express.js marks the initial stride in laying the groundwork for backend development in a MERN stack application. Node.js serves as the foundation, enabling the execution of JavaScript code on the server-side. Upon installing Express.js, developers gain access to a powerful toolkit tailored for web application development. Express.js streamlines server-side development by furnishing a comprehensive array of features, including routing, middleware management, and HTTP request handling.

4 **Create Server Structure**: structuring the server-side application involves organizing routes, middleware, and other components to ensure a coherent and maintainable codebase. Routes serve as the entry points for handling incoming HTTP requests and play a pivotal role in defining how the server responds to different endpoints. Each route is responsible for mapping a specific URL path to a corresponding handler function, which processes the request and generates an appropriate response. Middleware functions intercept incoming requests before they reach the route handlers, enabling developers to execute pre-processing tasks such as authentication, request validation, and error handling. By strategically organizing middleware functions, developers can modularize and reuse common request-processing logic across multiple routes, enhancing code readability and maintainability.

5 **Connect to MongoDB**: Use a MongoDB Utilizing a MongoDB client library like Mongoose is fundamental for establishing connectivity between the Express.js application and the MongoDB database in a MERN stack environment. Mongoose simplifies database interaction by providing an object data modeling (ODM) layer that abstracts away the complexity of raw MongoDB queries. Through Mongoose, developers define database models that represent data entities, allowing for structured and schema-based data management. These models encapsulate the schema definition, data validation rules, and methods for performing CRUD operations. With Mongoose, CRUD operations become intuitive and straightforward, as developers can utilize familiar JavaScript syntax and Mongoose's built-in methods to create, read, update, and delete documents in the MongoDB database.

6 **Implement CRUD Operations**: Developing route handlers for CRUD operations forms a critical component of backend development in a MERN stack application, facilitating seamless communication between the client-side application and the MongoDB database. These route handlers serve as the intermediary layer responsible for processing incoming HTTP requests, each representing a particular CRUD operation (Create, Read, Update, Delete). When a client-side application sends a request to the server, the corresponding route handler intercepts the request and executes the corresponding database operation using the MongoDB database. For instance, a POST request triggers the creation of a new document in the database, while a GET request retrieves data from the database based on specified criteria. Similarly, PUT and DELETE requests are responsible for updating and deleting existing documents, respectively..

7 **Authentication and Authorization**: Implement Implementing robust authentication and authorization mechanisms is paramount for securing backend API endpoints in a MERN stack application. Middleware solutions such as Passport.js offer a versatile approach, enabling developers to integrate various authentication strategies like JWT or OAuth seamlessly. By defining authorization logic based on user roles or permissions, developers can restrict access to specific routes or resources, ensuring that only authenticated and authorized users can interact with sensitive data. Leveraging JWT tokens for authentication enhances security by enabling stateless authentication while facilitating the exchange of secure user information between the client and server. Overall, the implementation of these mechanisms enhances the overall security posture of the application, safeguarding against unauthorized access and ensuring data integrity in the MERN stack ecosystem.

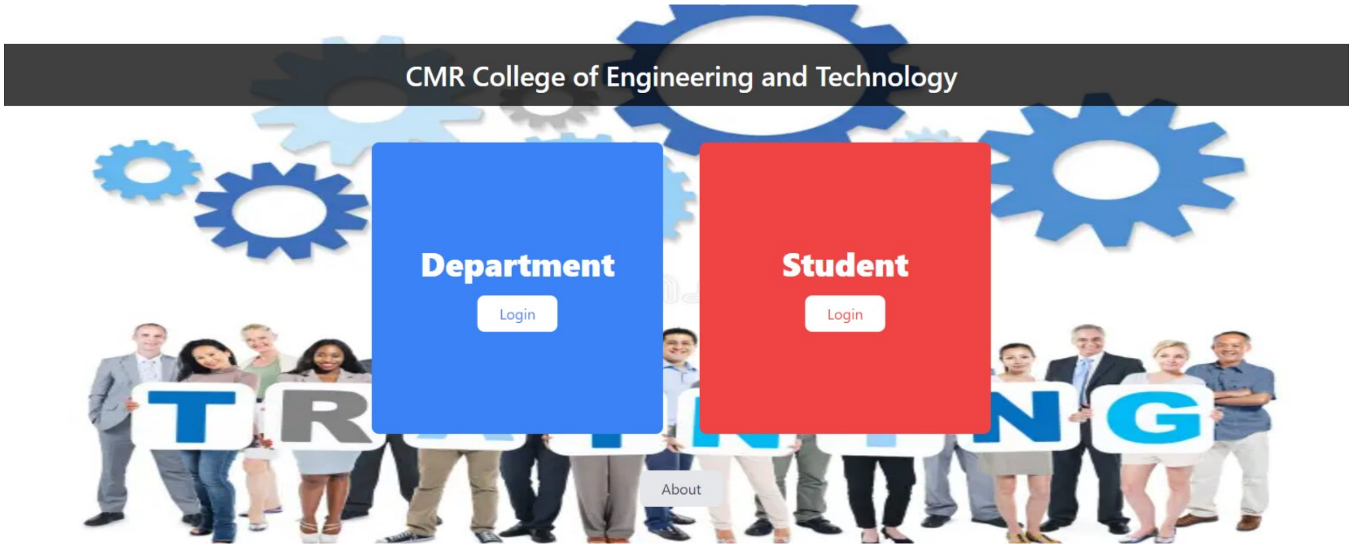


Fig 3.4.2. Dashboard

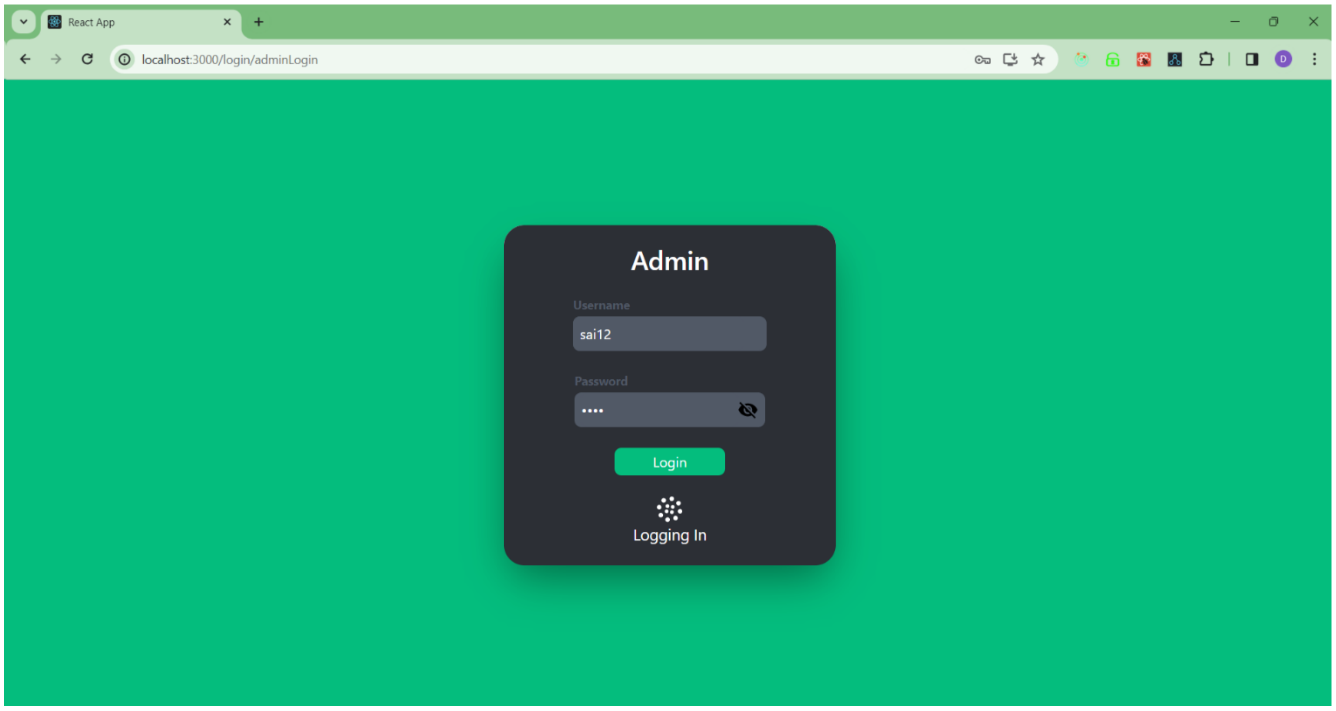


Fig 3.4.3. Admin login page

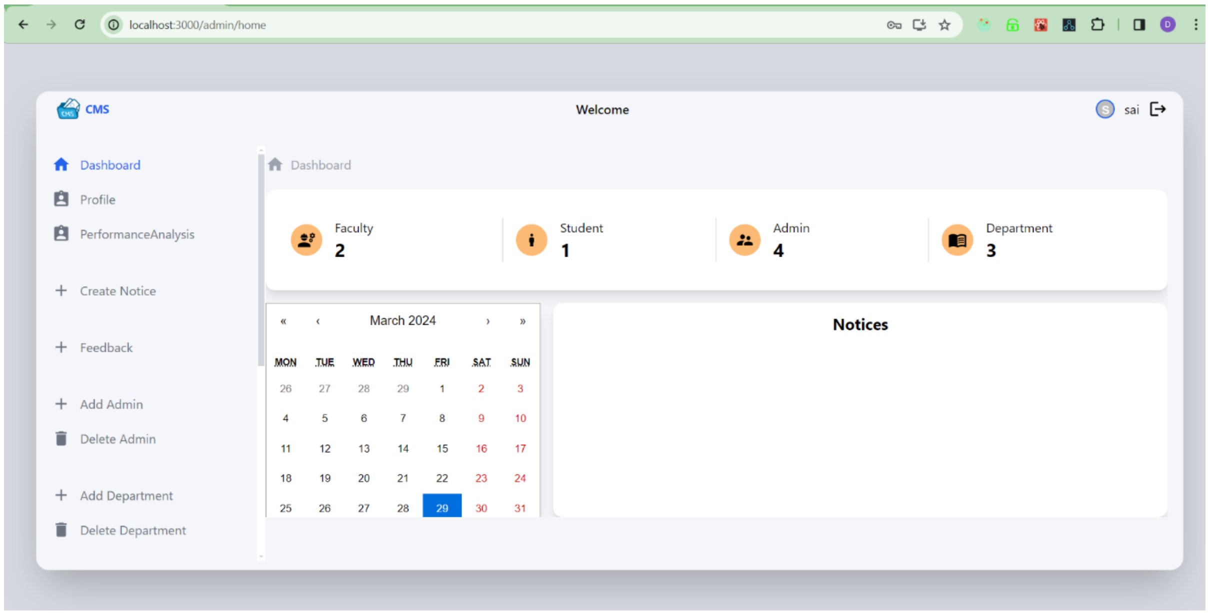


Fig 3.4.4. Admin Dashboard

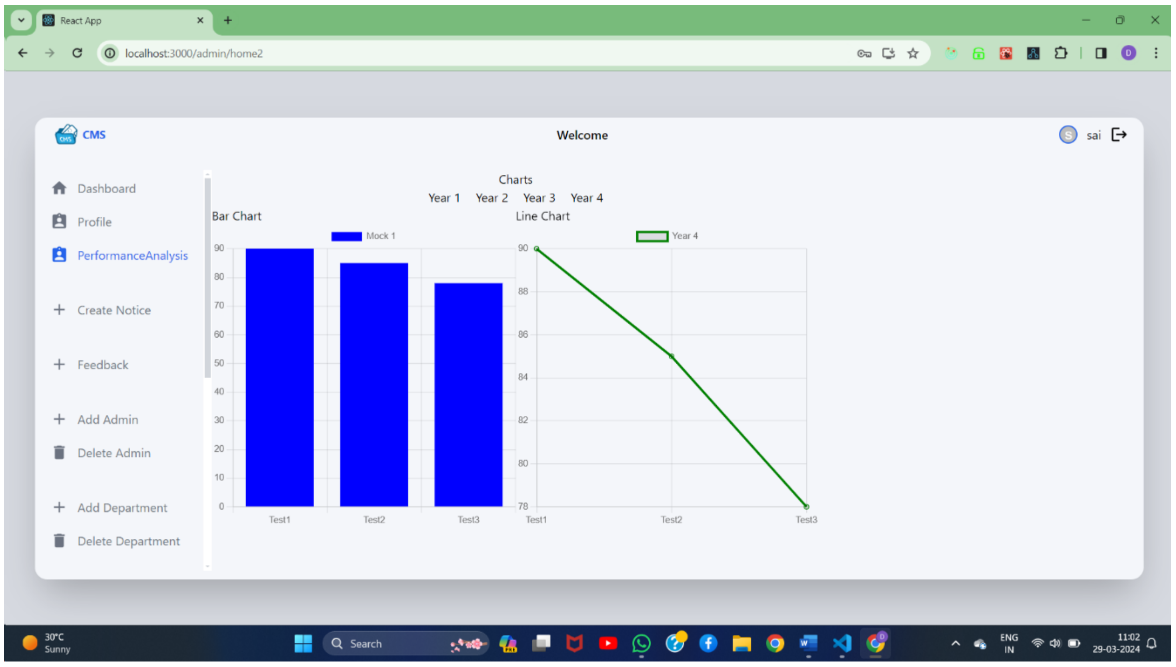


Fig 3.4.5. Overall Student Performance Analysis

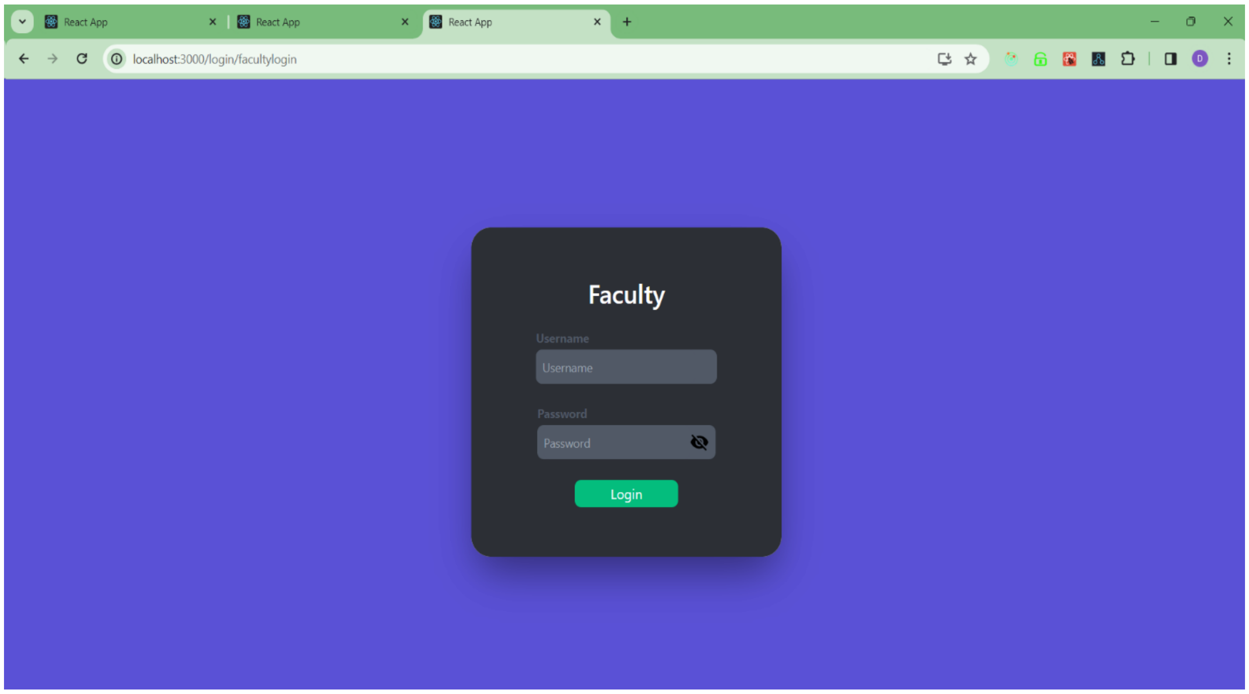
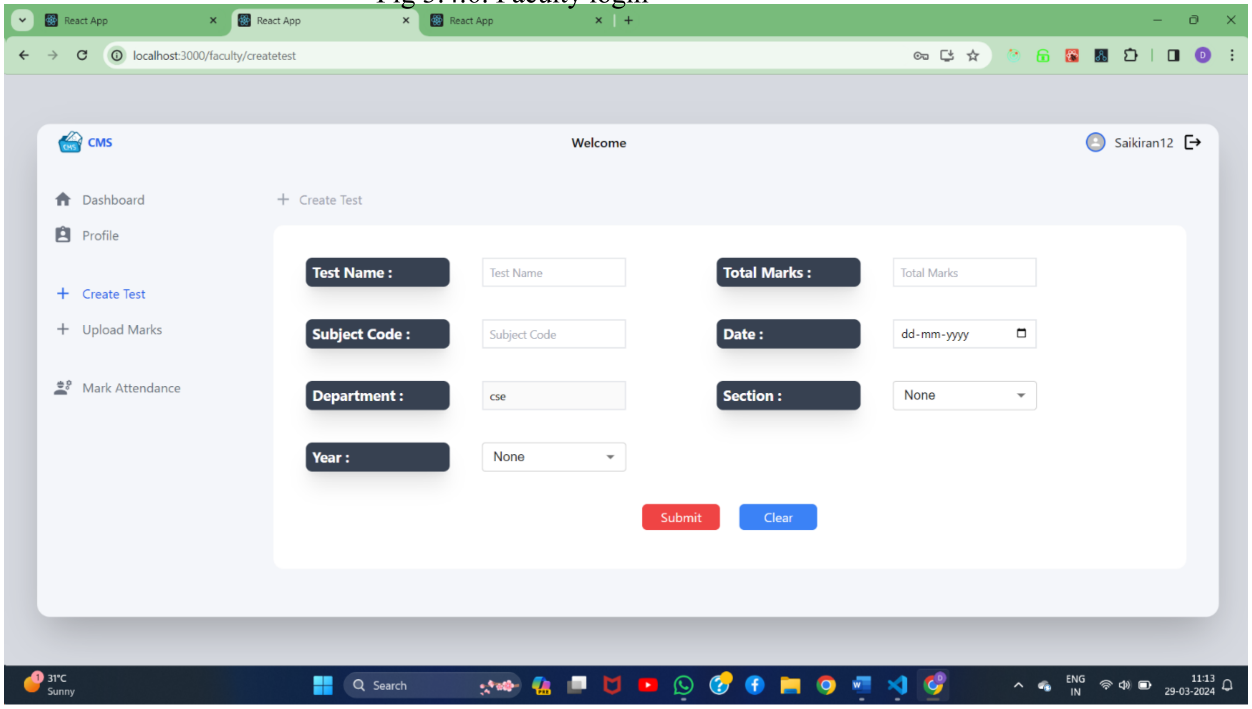
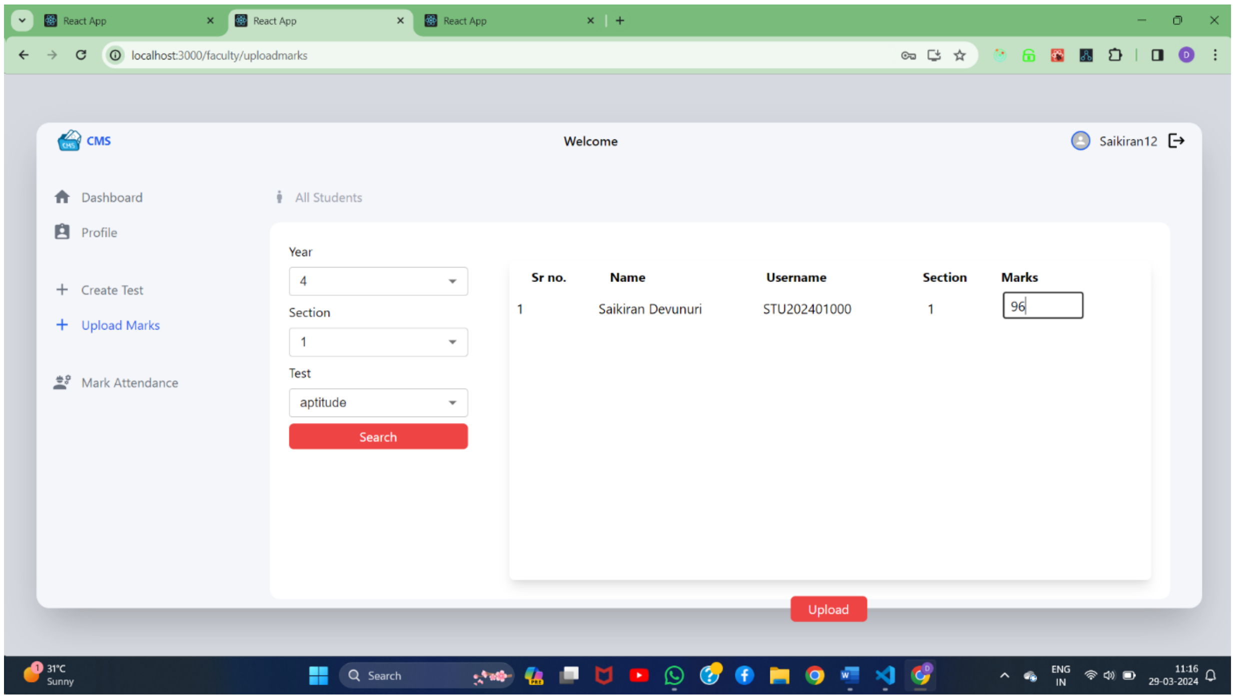


Fig 3.4.6. Faculty login



| | | | |
|---|---|---|---|
| Test Name : | Test Name | Total Marks : | Total Marks |
| Subject Code : | Subject Code | Date : | dd-mm-yyyy |
| Department : | cse | Section : | None |
| Year : | None | | |

Fig 3.4.7. create test



CMS  Welcome  Saikiran12

Dashboard
Profile
Create Test
Upload Marks
Mark Attendance

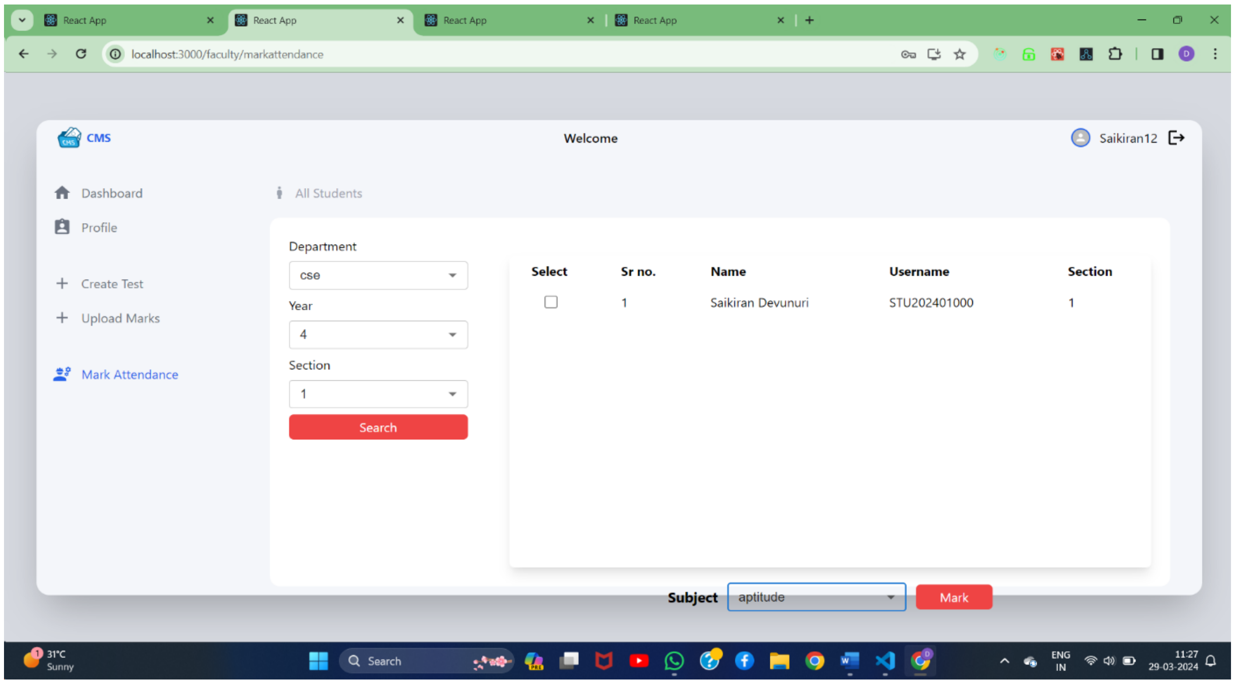
All Students

Year: 4
Section: 1
Test: aptitude
Search

| Sr no. | Name | Username | Section | Marks |
|---|---|---|---|---|
| 1 | Saikiran Devunuri | STU202401000 | 1 | 96 |

Upload

Fig 3.4.8. upload marks



CMS  Welcome  Saikiran12

Dashboard
Profile
Create Test
Upload Marks
Mark Attendance

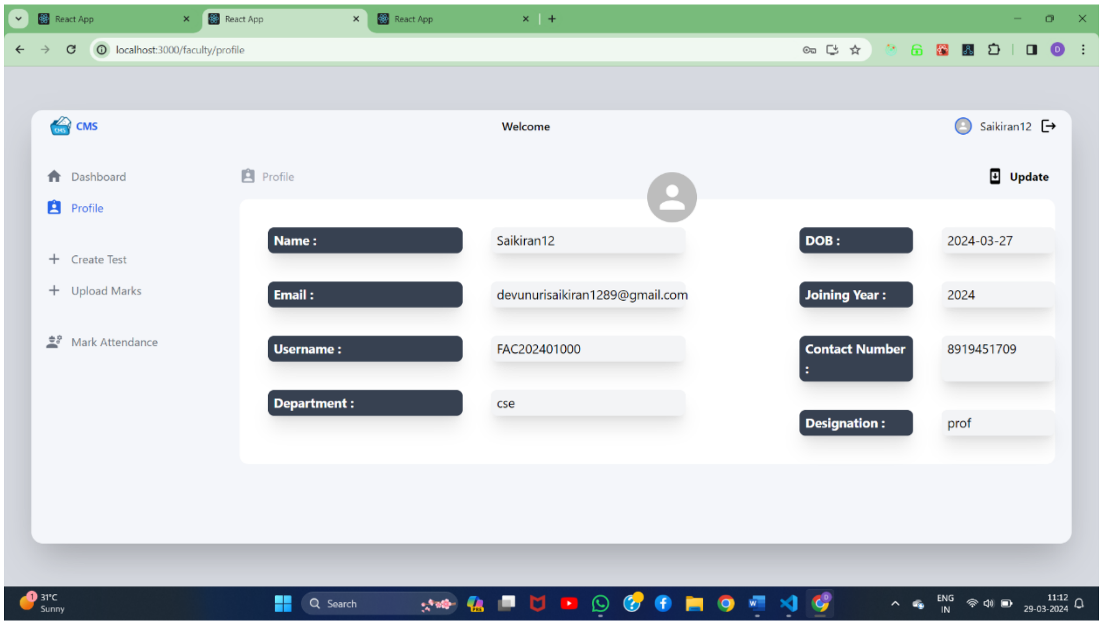
All Students

Department: cse
Year: 4
Section: 1
Search

| Select | Sr no. | Name | Username | Section |
|---|---|---|---|---|
| ☐ | 1 | Saikiran Devunuri | STU202401000 | 1 |

Subject: aptitude  Mark

Fig 3.4.9. upload Attendance



| | | | |
|---|---|---|---|
| Name : | Saikiran12 | DOB : | 2024-03-27 |
| Email : | devunurisaikiran1289@gmail.com | Joining Year : | 2024 |
| Username : | FAC202401000 | Contact Number : | 8919451709 |
| Department : | cse | Designation : | prof |

Fig 3.4.10. faculty profile

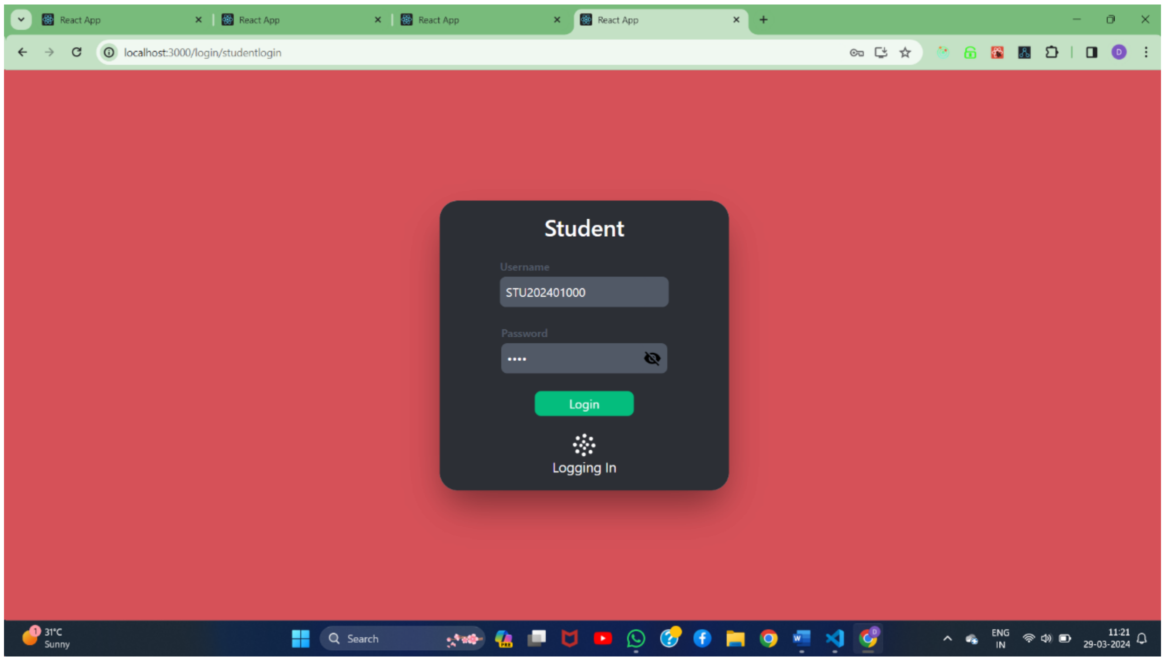


Fig 3.4.11. Student login

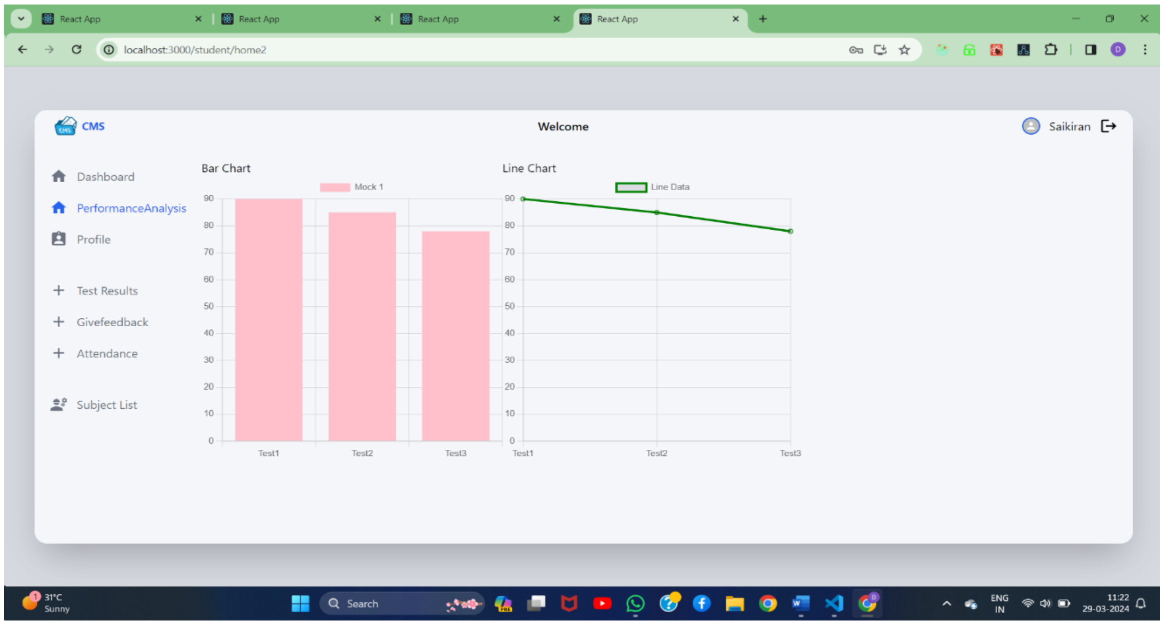
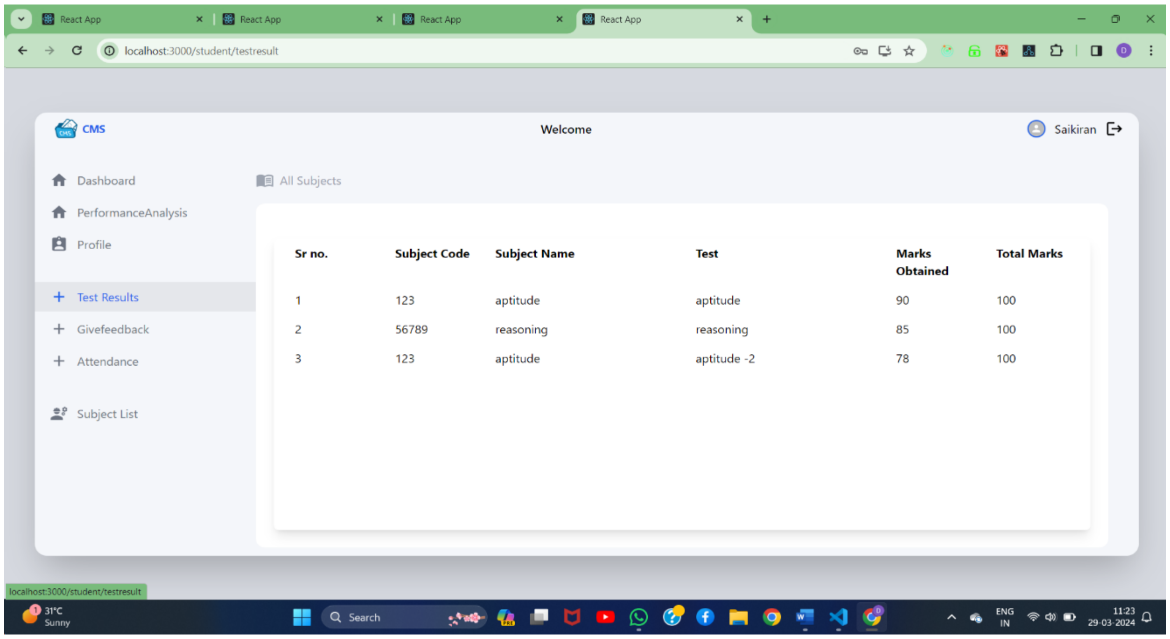
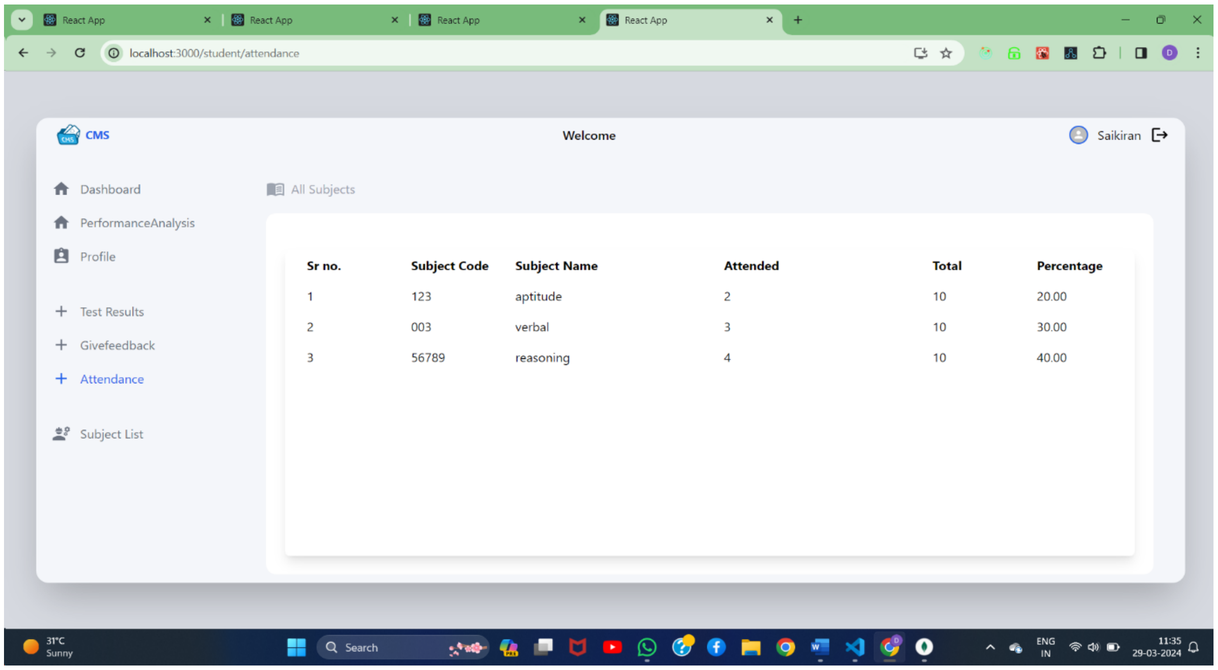


Fig 3.4.11. Student performance analysis



| Sr no. | Subject Code | Subject Name | Test | Marks Obtained | Total Marks |
|---|---|---|---|---|---|
| 1 | 123 | aptitude | aptitude | 90 | 100 |
| 2 | 56789 | reasoning | reasoning | 85 | 100 |
| 3 | 123 | aptitude | aptitude -2 | 78 | 100 |

Fig 3.4.12. student check his marks



| Sr no. | Subject Code | Subject Name | Attended | Total | Percentage |
|---|---|---|---|---|---|
| 1 | 123 | aptitude | 2 | 10 | 20.00 |
| 2 | 003 | verbal | 3 | 10 | 30.00 |
| 3 | 56789 | reasoning | 4 | 10 | 40.00 |

Fig 3.4.13. student check his attendance

# CHAPTER 4
# RESULTS AND DISCUSSION

# CHAPTER 4
# RESULTS AND DISCUSSION

The MERN stack, with MongoDB as its database component, offers inherent scalability, making it well-suited for accommodating growing user bases and increasing traffic. MongoDB's NoSQL database architecture provides flexibility in handling large volumes of user data efficiently. As the user base expands, the system can seamlessly scale to meet demand without sacrificing performance. This scalability ensures that the platform remains responsive and reliable, facilitating a smooth user experience even during periods of high activity. Additionally, MongoDB's scalability complements the feedback loop within the system, allowing for continuous improvement based on user interactions and preferences. By leveraging the scalability of the MERN stack and MongoDB, the platform can evolve to meet the evolving needs of users, ultimately contributing to an enhanced learning experience.

## 4.1 Performance Metrics

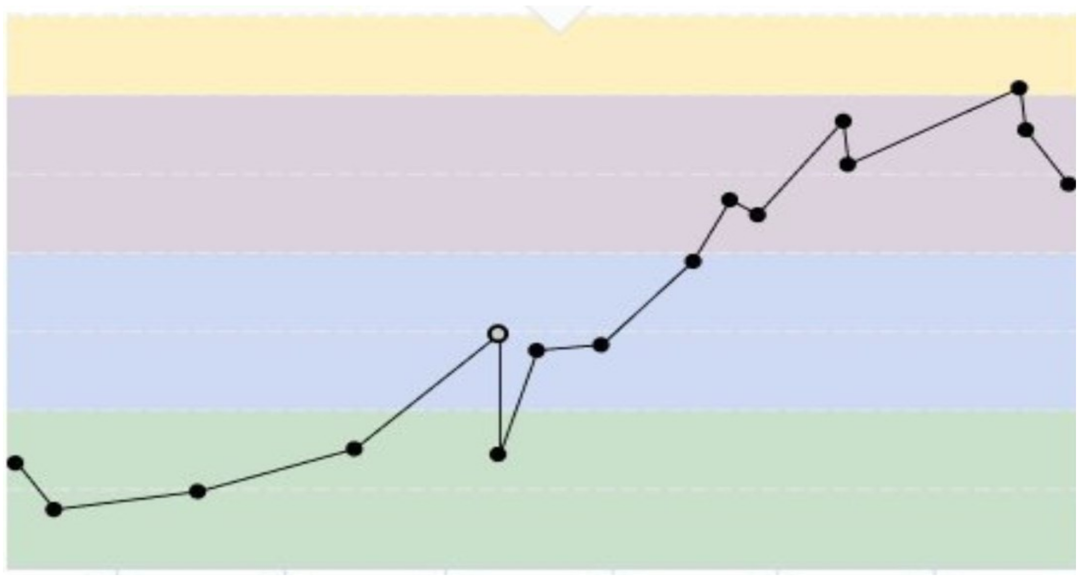


**Fig 4.1.1 Student graph analysis**

This graph shows a detailed view of how students ‘progress through their educational journey. It provides insights into their development, highlights areas where they good, and pinpoints those that need additional attention. By tracking student growth at specific stages and conducting a thorough analysis of their performance in tests and assignments, educators can tailor their teaching strategies and support to meet the unique needs of each student, ultimately fostering a culture of continuous improvement

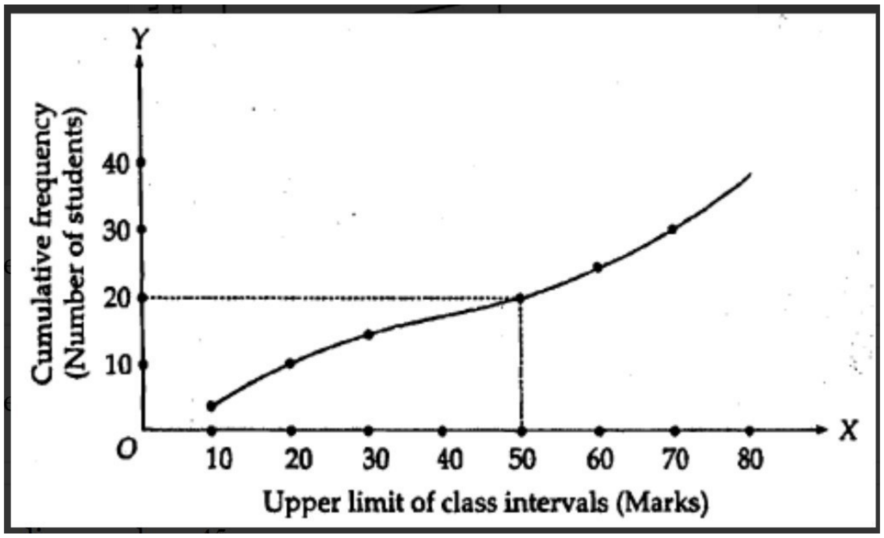


**Fig 4.1.2 TPO graph analysis**

In the second graph, the system enables the presentation of students' overall performance in training. This data is accessible to higher authorities, including Heads of Departments (HODs), Deans, and Principals of respective institutes. This information empowers top-level decision- makers to gain a comprehensive overview of the training program's effectiveness and the achievements of the students

Access to comprehensive data empowers higher authorities within educational institutions to make informed decisions, strategically allocate resources, and shape the institution's future direction. By leveraging insights derived from student performance metrics, attendance records, and engagement levels, decision-makers can identify areas of strength and areas for improvement within the institution. Strategic allocation of resources becomes more effective when guided by data-driven insights. By analyzing trends and patterns in student performance and behavior, administrators can allocate resources such as faculty, facilities, and funding to areas where they are most needed. For example, if data analysis reveals a correlation between student success and smaller class sizes, resources can be directed towards reducing class sizes to improve student outcomes.

Furthermore, access to data promotes transparency and accountability in educational management. When stakeholders have access to relevant data, they can understand the rationale behind decisions and evaluate the effectiveness of policies and initiatives. Transparent data-driven decision-making processes build trust among stakeholders and foster a culture of accountability within the institution.

.

# CHAPTER 5
# CONCLUSION

# CHAPTER 5
# CONCLUSION

In this project, we delve into a comprehensive analysis of student performance, focusing on their engagement with tests and attendance in classes. By scrutinizing these factors, we aim to uncover the intricate dynamics that influence student outcomes. The research not only explores the correlation between test participation, class attendance, and academic achievement but also delves into the underlying mechanisms driving these relationships.
Our approach incorporates scalability and flexibility in data storage, making it an ideal choice for handling large volumes of student records. This robust data management solution ensures that educators and administrators have seamless access to the information necessary for informed decision-making. By leveraging scalable storage options, we ensure that the system can accommodate the diverse needs of educational institutions, regardless of size or complexity. Furthermore, the implementation of the MERN stack – MongoDB, Express, React, and Node.js – underscores our commitment to developing a responsive, user-friendly, and adaptable platform Mobile Application

## 5. 1 Future Scope

In future we can Develop a mechanism of separation of students bases on their performance and Incorporate gamification elements such as badges, achievements, and leaderboards to enhance student motivation and engagement. By gamifying the learning experience, the platform can foster a sense of competition, collaboration, and achievement among students, driving active participation and progress. Integration with Learning Management Systems (LMS): Enable seamless integration with existing learning management systems to streamline data exchange and enhance interoperability

# REFERENCES

## REFERENCES

[1] Angela L. Duckworth and Martin E.P. Seligman Self-Discipline Outdoes IQ in Predicting Academic Performance of Adolescents

[2] Alf Lizzio, Keithia Wilson &Roland Simons University Students' Perceptions of the Learning Environment and Academic Outcomes: Implications for theory and practice

[3] Barry J. Zimmerman Self-efficacy & educational development Monique Lortie Development of a Website: An Experience of interface with Manual Workers and Web Specialists

[4] Zinovieva1, Artem Chuk Anna V Iatsyshyn The use of online coding platforms as additional distance tools in programming education

[5] Rasha Albashaireh; Hua Ming A Survey on Online Learning Platforms with Initial Investigation of SituaAwwal, Olarotimi Badru

[6] Vani Vasudevan,M.G.M. Khan MERN Stack Web-Based Education Management Information Systems for Pacific Island Countries

[7] Daniel McFarland &Diane Hamilton Factors Affecting Student Performance and Satisfaction: Online versus Traditional Course Delivery

[8] Akshitha Kool: Manual Accounting System And Computerized Accounting System

[9] Owan, Valentine Joseph & Bassey, Bassey A. Comparative Study of Manual and Computerized Software Techniques of Data Management and Analysis in Educational Research.

[10] Gilbert, M. (2019), "Student performance is linked to connecting effectively with teachers"

[11] Proff Yogesh Introduction to MERN Stack & Comparison with Previous Technologies

[12] R.; Snijders, C.; Kleingeld, A.; Matzat, U. Predicting Student Performance from LMS Data: A Comparison of 17 Blended Courses Using Moodle LMS. IEEE Trans. Learn. Technol. 2017, 10, 17–29. [Google Scholar]

[13] Yogesh Baiskar1 , Priyas Paulzagade Full Stack Development using MERN

[14] Full-Stack Development10. Dr. Poornima Mehta, Harsh Kumar, Amit Sharma, "study point website using MERN stack", International Research Journal of Modernization in Engineering Technology and Science.

[15] Predicting Student Academic Performance by Means of Associative Classification by Luca Cagliero *ORCID,Lorenzo CanaleORCID,Laura Farinetti ORCID,Elena BaralisORCID andEnrico VenutoORCID

[16] P. Kanchanamala, B. H. Sai, B. Balaji, A. Panigrahi, K. Srinivas and A. V. Vardhan, "Automated Programming Evaluation using MERN," 2023

[17] Ritesh Patil, Vaishali Gentyal, Vaishnavi Mudaliar, Gauri Kanpurne and Devyani Ambi, "College Website Using MERN Stack",

[18] Sanchit Aggarwal, Jyoti Verma et al., "Comparative analysis of MEAN stack and MERN stack"

[19] Proff Suvarn Patil "Web application for fitness tracking System"

[20] https://www.w3schools.com/whatis/whatis_fullstack_js.asp

## Frontend Code:

### Index.js:

```
import axios from "axios";

// const API = axios.create({ baseURL: pro-
cess.env.REACT_APP_SERVER_URL });

const API = axios.create({ baseURL: "http://localhost:5000/" });

API.interceptors.request.use((req) => {

  if (localStorage.getItem("user")) { req.headers.Au-

    thorization = `Bearer ${ JSON.parse(localStor-

    age.getItem("user")).token

    }`;

  }

  return req;

});

// Admin

export const adminSignIn = (formData) => API.post("/api/admin/login",
formData);

export const adminUpdatePassword = (updatedPassword) =>

  API.post("/api/admin/updatepassword", updatedPassword);

export const getAllStudent = () => API.get("/api/admin/getallstudent");

export const getAllFaculty = () => API.get("/api/admin/getallfaculty");

export const getAllAdmin = () => API.get("/api/admin/getalladmin");

export const getAllDepartment = () =>
API.get("/api/admin/getalldepartment");

export const getAllSubject = () => API.get("/api/admin/getallsubject");
export const updateAdmin = (updatedAdmin) =>
  API.post("/api/admin/updateprofile", updatedAdmin);
export const addAdmin = (admin) => API.post("/api/admin/addadmin",
admin);


export const createNotice = (notice) =>
  API.post("/api/admin/createnotice", notice);
export const deleteAdmin = (data) => API.post("/api/admin/deleteadmin",
data);
export const deleteFaculty = (data) =>
API.post("/api/admin/deletefaculty", data); ex-
port const deleteStudent = (data) =>
API.post("/api/admin/deletestudent", data); ex-
port const deleteSubject = (data) =>
API.post("/api/admin/deletesubject", data); ex-
port const deleteDepartment = (data) =>
API.post("/api/admin/deletedepartment", data);
export const getAdmin = (admin) => API.post("/api/admin/getadmin", admin);
export const addDepartment = (department) => API.post("/api/admin/adddepart-
ment", department);
export const addFaculty = (faculty) =>
API.post("/api/admin/addfaculty", faculty);
export const getFaculty = (department) =>
API.post("/api/admin/getfaculty", department);
export const addSubject = (subject) =>
API.post("/api/admin/addsubject", subject);
```

```
export const getSubject = (subject) =>
API.post("/api/admin/getsubject", subject);
export const addStudent = (student) =>
API.post("/api/admin/addstudent", student);
export const getStudent = (student) =>
API.post("/api/admin/getstudent", student);
export const getNotice = (notice) => API.post("/api/admin/getnotice", notice);
// Faculty
export const facultySignIn = (formData) => API.post("/api/fac-
  ulty/login", formData);
export const facultyUpdatePassword = (updatedPassword) =>
API.post("/api/faculty/updatepassword", updatedPassword); ex-
port const updateFaculty = (updatedFaculty) =>
API.post("/api/faculty/updateprofile", updatedFaculty);
export const createTest = (test) => API.post("/api/faculty/createtest", test);
export const getTest = (test) => API.post("/api/faculty/gettest", test); ex-
port const getMarksStudent = (student) => API.post("/api/faculty/getstu-
dent", student);
export const uploadMarks = (data) => API.post("/api/faculty/uploadmarks",
data);
export const markAttendance = (data) =>
  API.post("/api/faculty/markattendance", data);
// Student
export const studentSignIn = (formData) => API.post("/api/stu-
  dent/login", formData);
export const studentUpdatePassword = (updatedPassword) =>
  API.post("/api/student/updatepassword", updatedPassword);
```

```
export const update Student = (updated Student) =>
API.post("/api/student/updateprofile", updatedStudent); ex-
port const getTestResult = (testResult) => API.post("/api/stu-
dent/testresult", testResult);
export const getAttendance = (attendance) =>
API.post("/api/student/attendance", attendance);
```

**Actiontype.js:**

```
export const ADMIN_LOGIN = "ADMIN_AUTH";
export const LOGOUT = "LOGOUT";
export const UPDATE_PASSWORD = "UPDATE_PASSWORD";
export const UPDATE_ADMIN = "UPDATE_ADMIN"; ex-
port const ADD_ADMIN = "ADD_ADMIN";
export const ADD_DEPARTMENT = "ADD_DEPARTMENT"; ex-
port const ADD_FACULTY = "ADD_FACULTY";
export const GET_ALL_FACULTY = "GET_ALL_FACULTY"; ex-
port const ADD_SUBJECT = "ADD_SUBJECT";
export const GET_ALL_SUBJECT = "GET_ALL_SUBJECT";
export const ADD_STUDENT = "ADD_STUDENT";
export const GET_ALL_STUDENT = "GET_ALL_STUDENT"; ex-
port const GET_STUDENT = "GET_STUDENT";
export const GET_FACULTY = "GET_FACULTY";
export const GET_SUBJECT = "GET_SUBJECT";
export const GET_ALL_ADMIN = "GET_ALL_ADMIN";
export const GET_ALL_DEPARTMENT = "GET_ALL_DEPARTMENT";
export const SET_ERRORS = "SET_ERRORS";
export const FACULTY_LOGIN = "FACULTY_LOGIN";
export const UPDATE_FACULTY = "UPDATE_FACULTY";
export const ADD_TEST = "ADD_TEST";
export const GET_TEST = "GET_TEST";
export const MARKS_UPLOADED = "MARKS_UPLOADED";
export const ATTENDANCE_MARKED = "ATTENDANCE_MARKED";
export const STUDENT_LOGIN = "STUDENT_LOGIN"; ex-
port const UPDATE_STUDENT = "UPDATE_STUDENT";
export const TEST_RESULT = "TEST_RESULT";
export const ATTENDANCE = "ATTENDANCE";
export const GET_ADMIN = "GET_ADMIN";
export const DELETE_ADMIN = "DELETE_ADMIN";
export const DELETE_DEPARTMENT = "DELETE_DEPARTMENT";
export const DELETE_FACULTY = "DELETE_FACULTY";
export const DELETE_STUDENT = "DELETE_STUDENT";
export const DELETE_SUBJECT = "DELETE_SUBJECT";
export const CREATE_NOTICE = "CREATE_NOTICE"; ex-
port const GET_NOTICE = "GET_NOTICE";
```

## App.js:

```
const App = () => { re-
  turn (
    <Routes>
      <Route exact path="/" element={<Login />} />
      {/* Admin  */}
      <Route path="/login/adminlogin" element={<AdminLogin />} />
      <Route path="/admin/home" element={<AdminHome />} />
      <Route path="/admin/home2" element={<AdminHome2 />} />
      <Route path="/admin/profile" element={<AdminProfile />} />
      <Route path="/admin/update" element={<AdminUpdate />} />
      <Route path="/admin/update/password" element={<AdminPassword />}
/>
      <Route path="/admin/updatepass-
        word"
        element={<AdminFirstTimePassword />}
      />
      <Route path="/admin/feedback" element={<Adminfeed />} />
      <Route path="/admin/createnotice" element={<CreateNotice />} />
      <Route path="/admin/addadmin" element={<AddAdmin />} />
      <Route path="/admin/deleteadmin" element={<DeleteAdmin />} />
      <Route path="/admin/adddepartment" element={<AddDepartment />} />
      <Route path="/admin/deletedepartment" element={<DeleteDepartment />}
/>
      <Route path="/admin/addfaculty" element={<AddFaculty />} />
      <Route path="/admin/deletefaculty" element={<DeleteFaculty />} />
      <Route path="/admin/deletestudent" element={<DeleteStudent />} />
      <Route path="/admin/deletesubject" element={<DeleteSubject />} />
      <Route path="/admin/allfaculty" element={<GetFaculty />} />
      <Route path="/admin/addstudent" element={<AddStudent />} />
      <Route path="/admin/addsubject" element={<AddSubject />} />
      <Route path="/admin/allsubject" element={<GetSubject />} />
      <Route path="/admin/allstudent" element={<GetStudent />} />
      {/* Faculty  */}
      <Route path="/login/facultylogin" element={<FacultyLogin />} />
      <Route path="/faculty/home" element={<FacultyHome />} />
      <Route path="/faculty/password" element={<FacultyFirstTimePassword
/>} />
      <Route path="/faculty/profile" element={<FacultyProfile />} />
      <Route path="/faculty/update" element={<FacultyUpdate />} />
      <Route path="/faculty/update/password" element={<FacultyPassword />}
/>
      <Route path="/faculty/createtest" element={<CreateTest />} />
      <Route path="/faculty/uploadmarks" element={<UploadMarks />} />
      <Route path="/faculty/markattendance" element={<MarkAttendance />}
/>
      {/* Student  */}
      <Route path="/login/studentlogin" element={<StudentLogin />} />
      <Route path="/student/home" element={<StudentHome />} />
      <Route path="/student/home2" element={<StudentHome2 />} />
      <Route path="/student/feedback1" element={<Feedback1/>} />
      <Route path="/student/password" element={<StudentFirstTimePassword
/>} />
      <Route path="/student/profile" element={<StudentProfile />} />
      <Route path="/student/update" element={<StudentUpdate />} />
      <Route path="/student/update/password" element={<StudentPassword />}
/>
      <Route path="/student/subjectlist" element={<SubjectList />} />
      <Route path="/student/testresult" element={<TestResult />} />
      <Route path="/student/attendance" element={<Attendance />} />
     </Routes>
    );};
```

**Backend Code:**

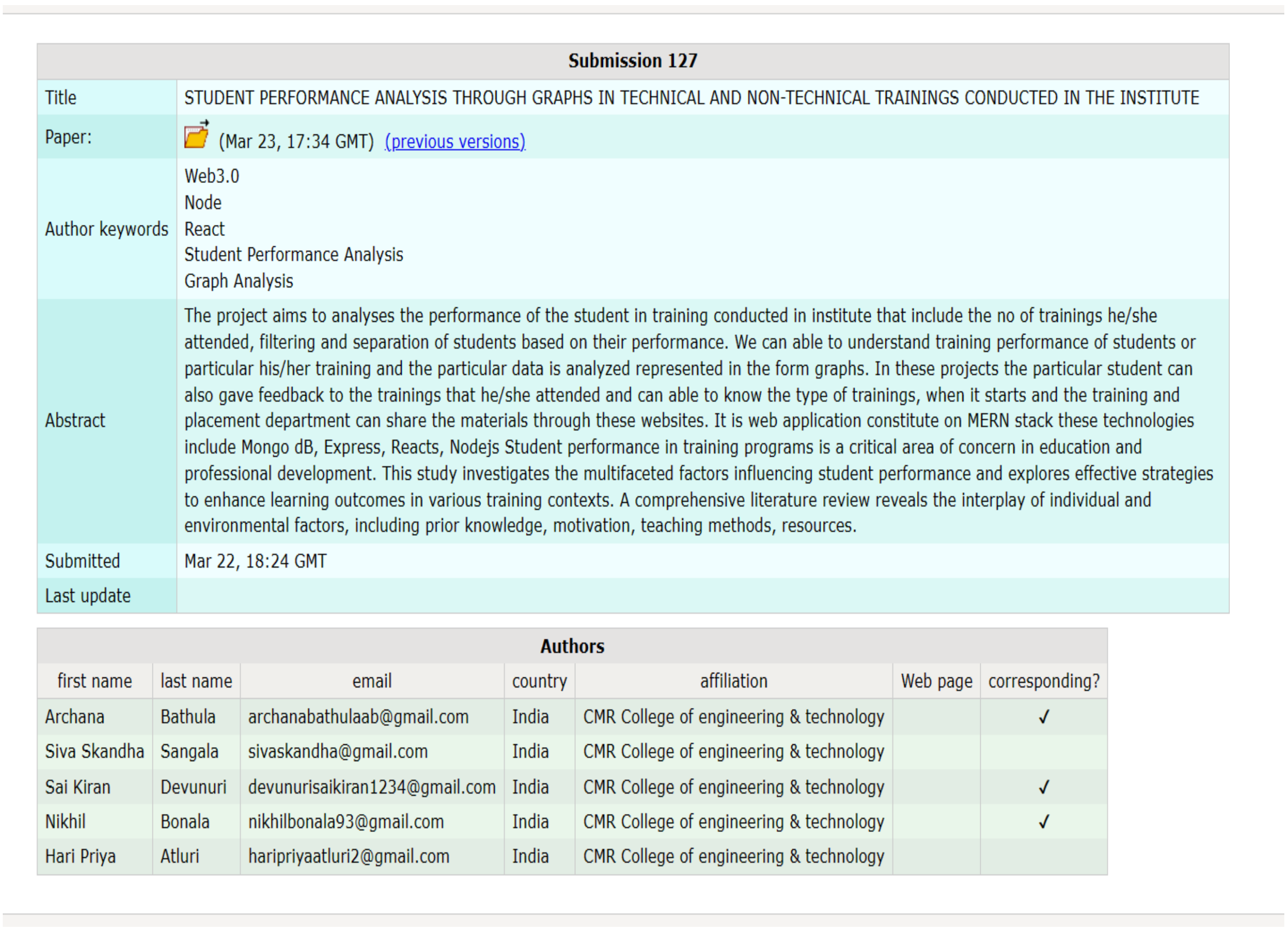
**Index.js:**

```
import express from "express";
import bodyParser from "body-parser";
import mongoose from "mongoose"; im-
port cors from "cors";
import dotenv from "dotenv";
import adminRoutes from "./routes/adminRoutes.js";
import studentRoutes from "./routes/studentRoutes.js";
import facultyRoutes from "./routes/facultyRoutes.js";
const app = express();
dotenv.config();
app.use(bodyParser.json({ limit: "30mb", extended: true }));
app.use(bodyParser.urlencoded({ limit: "30mb", extended: true }));
app.use(cors());
app.use("/api/admin", adminRoutes);
app.use("/api/faculty", facultyRoutes);
app.use("/api/student", studentRoutes);
const PORT = process.env.PORT || 5000;
app.get("/", (_req, res) => {
res.send("Hello to college erp API");
});
mongoose
  .connect('mongodb://127.0.0.1/placementcmr2', {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  })
  .then(() =>
    app.listen(PORT, () => console.log(`Server running on port ${PORT}`))
  )
  .catch((error) => console.log("Mongo Error", error.message));
```

**Auth.js:**

```
import jwt from "jsonwebtoken";
const auth = async (req, res, next) => {
try {
    let token;
    let authHeader=req.headers.Authorization || req.headers.authorization;
    if(authHeader)
        token = authHeader.split(" ")[1];
    let decodedData;
    if (token) {
      decodedData = jwt.verify(token, process.env.sEcReT);
      req.userId = decodedData?.id;
    }
    next();
  } catch (error) { con-
    sole.log(error);
  }
};
export default auth;
```

## Submission of Paper on ICIACS:



**Submission 127**

| | |
|---|---|
| Title | STUDENT PERFORMANCE ANALYSIS THROUGH GRAPHS IN TECHNICAL AND NON-TECHNICAL TRAININGS CONDUCTED IN THE INSTITUTE |
| Paper: | (Mar 23, 17:34 GMT) (previous versions) |
| Author keywords | Web3.0<br>Node<br>React<br>Student Performance Analysis<br>Graph Analysis |
| Abstract | The project aims to analyses the performance of the student in training conducted in institute that include the no of trainings he/she attended, filtering and separation of students based on their performance. We can able to understand training performance of students or particular his/her training and the particular data is analyzed represented in the form graphs. In these projects the particular student can also gave feedback to the trainings that he/she attended and can able to know the type of trainings, when it starts and the training and placement department can share the materials through these websites. It is web application constitute on MERN stack these technologies include Mongo dB, Express, Reacts, Nodejs Student performance in training programs is a critical area of concern in education and professional development. This study investigates the multifaceted factors influencing student performance and explores effective strategies to enhance learning outcomes in various training contexts. A comprehensive literature review reveals the interplay of individual and environmental factors, including prior knowledge, motivation, teaching methods, resources. |
| Submitted | Mar 22, 18:24 GMT |
| Last update | |

**Authors**

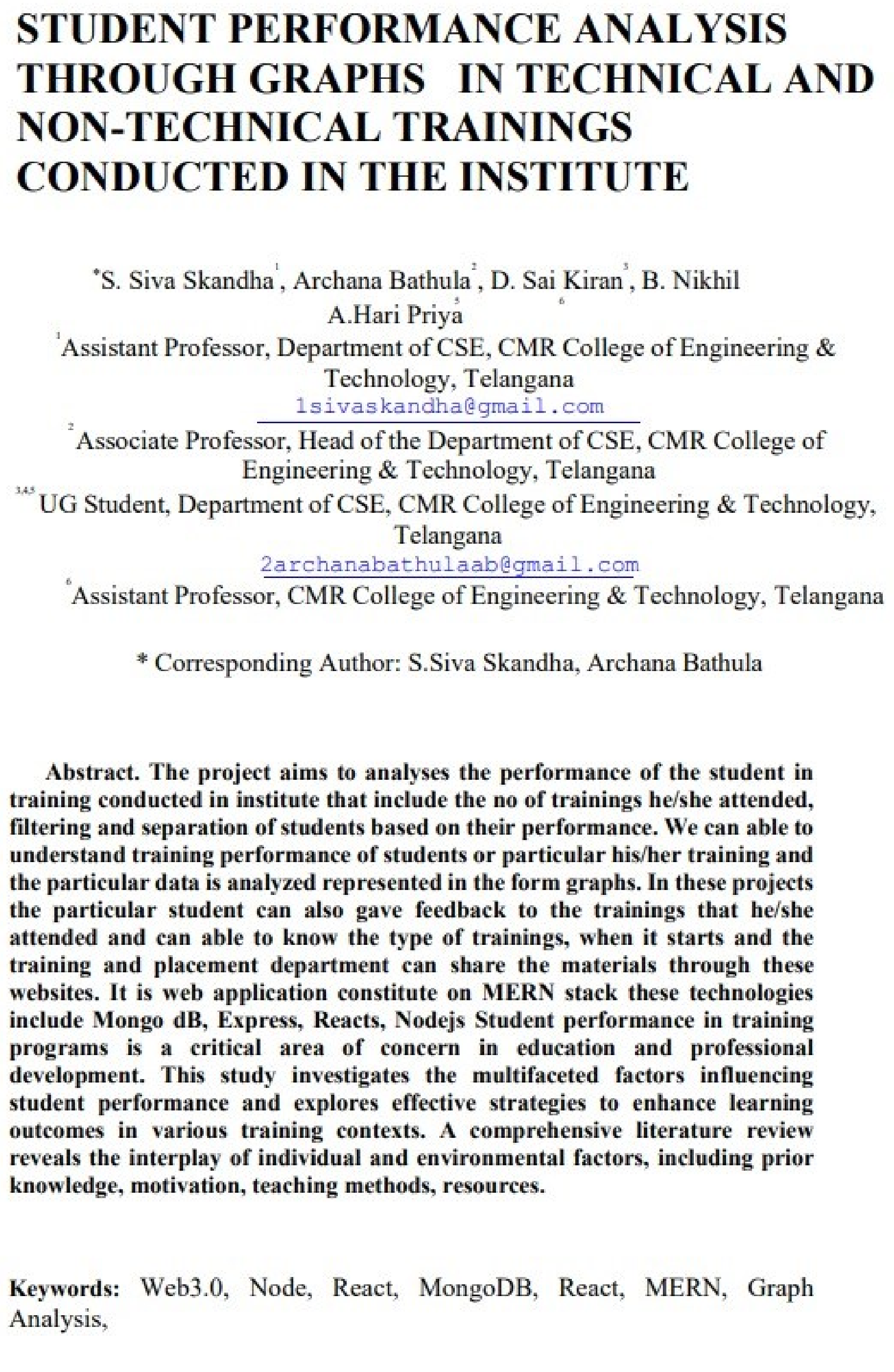
| first name | last name | email | country | affiliation | Web page | corresponding? |
|---|---|---|---|---|---|---|
| Archana | Bathula | archanabathulaab@gmail.com | India | CMR College of engineering & technology | | ✓ |
| Siva Skandha | Sangala | sivaskandha@gmail.com | India | CMR College of engineering & technology | | |
| Sai Kiran | Devunuri | devunurisaikiran1234@gmail.com | India | CMR College of engineering & technology | | ✓ |
| Nikhil | Bonala | nikhilbonala93@gmail.com | India | CMR College of engineering & technology | | ✓ |
| Hari Priya | Atluri | haripriyaatluri2@gmail.com | India | CMR College of engineering & technology | | |

**Fig No:5 Paper Submission**

## Link of the Submission of Springer Paper:

ICIACS 2024 Submission 127 (easychair.org)

## Submitted Paper:



# STUDENT PERFORMANCE ANALYSIS THROUGH GRAPHS IN TECHNICAL AND NON-TECHNICAL TRAININGS CONDUCTED IN THE INSTITUTE

*S. Siva Skandha[1], Archana Bathula[2], D. Sai Kiran[3], B. Nikhil
A.Hari Priya[5] [6]
[1]Assistant Professor, Department of CSE, CMR College of Engineering & Technology, Telangana
1sivaskandha@gmail.com
[2]Associate Professor, Head of the Department of CSE, CMR College of Engineering & Technology, Telangana
[3,4,5]UG Student, Department of CSE, CMR College of Engineering & Technology, Telangana
2archanabathulaab@gmail.com
[6]Assistant Professor, CMR College of Engineering & Technology, Telangana

* Corresponding Author: S.Siva Skandha, Archana Bathula

**Abstract. The project aims to analyses the performance of the student in training conducted in institute that include the no of trainings he/she attended, filtering and separation of students based on their performance. We can able to understand training performance of students or particular his/her training and the particular data is analyzed represented in the form graphs. In these projects the particular student can also gave feedback to the trainings that he/she attended and can able to know the type of trainings, when it starts and the training and placement department can share the materials through these websites. It is web application constitute on MERN stack these technologies include Mongo dB, Express, Reacts, Nodejs Student performance in training programs is a critical area of concern in education and professional development. This study investigates the multifaceted factors influencing student performance and explores effective strategies to enhance learning outcomes in various training contexts. A comprehensive literature review reveals the interplay of individual and environmental factors, including prior knowledge, motivation, teaching methods, resources.**

**Keywords:** Web3.0, Node, React, MongoDB, React, MERN, Graph Analysis,

## 1 Introduction

The test scores or grades are important for the students to analysis his performance and to improve his performance [1] The environment for monitoring the students by management is also required for better analysis of their performance in trainings conducted by the management [1-2] This serves the foundation for conducting a good investigation, which will ultimately lead to the development of effective strategies to improve the effectiveness of the training programs [2] it adopts a different methods approach that combines both quantitative and qualitative analyses. The quantitative analysis focuses performance metrics, such as aptitude scores, to provide a numerical assessment of student capabilities. And also, it examines verbal learning skills, enabling the research to get linguistic and language-related aptitude that might influence performance of students [2-3] By combining both quantitative and qualitative data, to provide a clear view of the challenges faced by students and how these challenges impact their performance and finding or addresses their unique needs, results in good performance and outcomes of Students.

## Related Work

There are different existing methods for analyzing the performance of students that exists are.

### 1.1 Manual Record

The performance of the students can be analyzed by a manual system, in which the marks of the students are stored in manual records [3]. There, the management can analyze the performance of the students, and these records are organized by the faculties of respective departments.

### 1.2 Drawbacks

1. Here, calculations are performed by faculties. It may result in human error.
2. There is no proper security for the records.

### 1.3 Online Coding Platforms

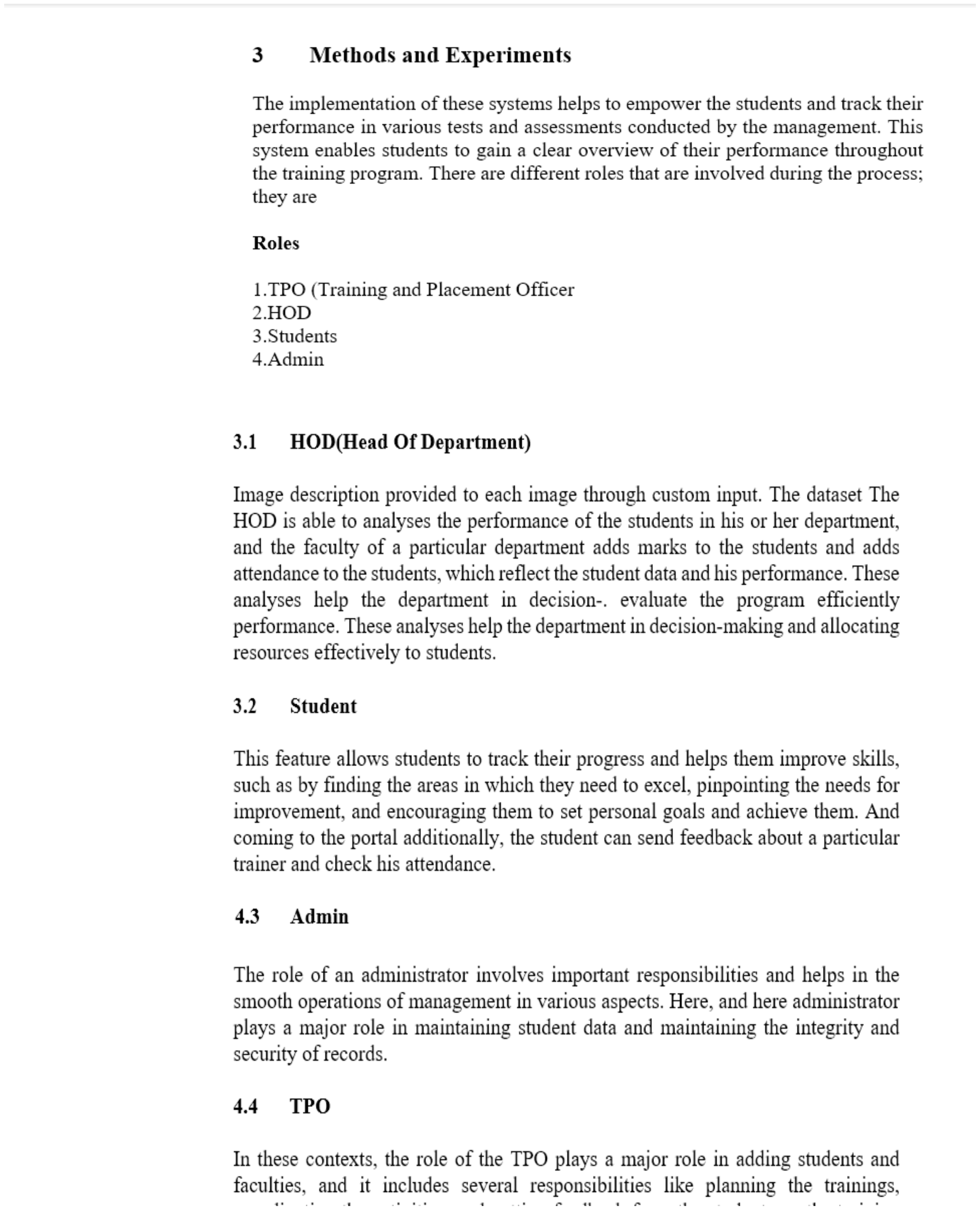
The coding platforms, which help to build problem-solving skills by solving various coding problems, offer a wealth of coding challenges, categorized by data structure, linked lists, trees, and graphs. [4], Using these platforms helps with learning and improvement. [5], By providing problems related to different data structures, the application helps users learn and understand how to apply these structures

## 1.4 Drawbacks

1. It is not dedicated to particular organizations or community members; in this context, the performance is not evaluated by any other person or external individuals.
2.This platform sometimes susceptible to data breaches like external gain access to sensitive information present in our account
3. Some coding platform like exhibit student performance analysis weak. For instance, certain may lack comprehensive tool or features like graphs to effectively evaluate the student performance.

# 2 Literature survey

We started conducting research on various fields, such as academic databases, journals, and books for related studies, and analyzed key theories and frameworks that were used for analyzing the topic.] We identified the key factors that are involved in student performance in training, such as teaching methods, learning environment [3-4], and some online platforms which are dedicated to particular platform which are not dedicated to particular organization [4-5]. Some of the topics we have gone through
A Critical Analysis of MERN Stack Application in Educational Environments [6]: Implications for Student Engagement and Learning Outcomes, and it says the role of MERN application in education settings and the combination of React, Express, Node, and MongoDB influences student engagement programs.
A Comprehensive Factor Affecting Student Performance in Training Programs[7]: It says how various factors influence student performance in training, examining approaches, and technological approaches.
We Understand how training programs have evolved over time and the impact of changing educational philosophies and technologies on student performance.



## 3 Methods and Experiments

The implementation of these systems helps to empower the students and track their performance in various tests and assessments conducted by the management. This system enables students to gain a clear overview of their performance throughout the training program. There are different roles that are involved during the process; they are

**Roles**

1.TPO (Training and Placement Officer
2.HOD
3.Students
4.Admin

### 3.1 HOD(Head Of Department)

Image description provided to each image through custom input. The dataset The HOD is able to analyses the performance of the students in his or her department, and the faculty of a particular department adds marks to the students and adds attendance to the students, which reflect the student data and his performance. These analyses help the department in decision-. evaluate the program efficiently performance. These analyses help the department in decision-making and allocating resources effectively to students.

### 3.2 Student

This feature allows students to track their progress and helps them improve skills, such as by finding the areas in which they need to excel, pinpointing the needs for improvement, and encouraging them to set personal goals and achieve them. And coming to the portal additionally, the student can send feedback about a particular trainer and check his attendance.

### 4.3 Admin

The role of an administrator involves important responsibilities and helps in the smooth operations of management in various aspects. Here, and here administrator plays a major role in maintaining student data and maintaining the integrity and security of records.

### 4.4 TPO

In these contexts, the role of the TPO plays a major role in adding students and faculties, and it includes several responsibilities like planning the trainings,

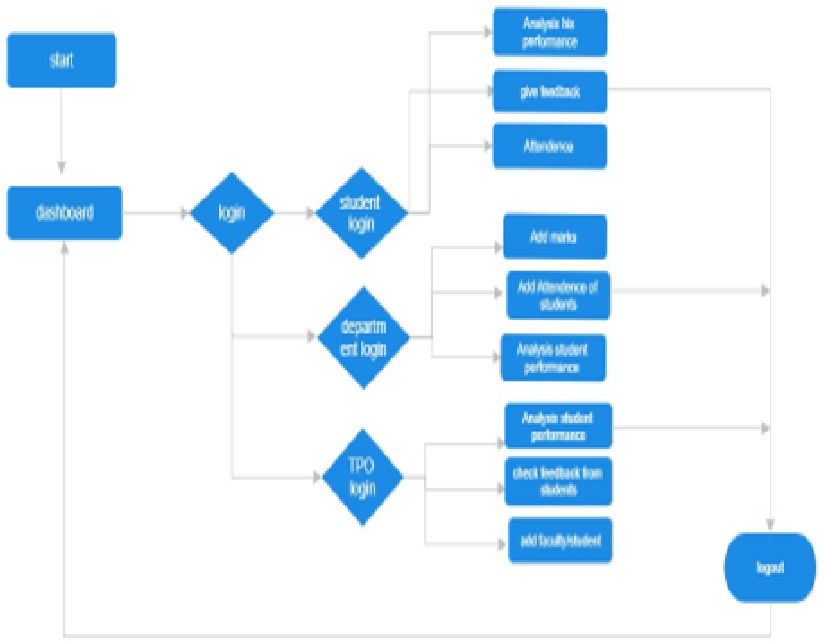


**Fig. 1.** Architecture diagram of proposed solution

## 5 Tech Stack Used

**5.1 Node:** it is open-source and allows to execute JavaScript code outside the browser and run applications on the server side. It is commonly used to build fast and scalable network applications, and it contains a package manager (NPM) that gives access to a vast ecosystem of libraries and modules.

**5.2 MongoDB**: it is a NOSQL database it uses a document-oriented data model it does not uses tables instead it stores the data in JSON-like documents and are Dynamic.

**5.3 React**: It is used to develop the frontend part of the application, and we used React Redux for state management, which helps connect the components and interact in a global state. It also contains the UI components, which are reusable, and the components update efficiently in response to changes in the data.

**5.4 Express**: it is a web application framework designed to create robust web applications. It is known for its simplicity, flexibility, and ease of use. allowing the developers to define the routers and handle the HTTP requests, and it facilitates rapid development of server-side components.

The above figure shows the workings of MERN project. React interacts with the Express framework, moving the data from the frontend to the backend through APIs (application program interfaces). These act as a bridge, facilitating the exchange of data between frontend and backend components. Here, react components make a specific request to specific end points defined by the express server, and triggering actions can fetch the data that is required. Upon receiving these requests to express, the express executes the necessary action and communicates with the MongoDB server to retrieve or manipulate the data.

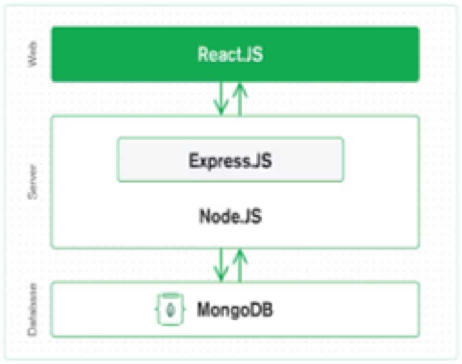


Fig 5.Working of MERN Project

## 6 Result and Discussion

The project results in showing the performance of the students in the form of form of graphs and the performance is analyzed in two ways the performance is analyzed by the student in different tests conducted by the management and the second analyzes is done by TPO (Training and placement department through the overall graph analysis of the students that which helps him to give clear view of understanding of students in different trainings that are conducted. It includes the faculty of particular department can assign the marks to students and these marks are converted in the form of graphs.

## 7 Conclusion

In this project, it analyzes the performance of students in various technical and non-technical trainings and assessments conducted at the institute. as it gives a comprehensive evaluation of students in various aspects of the training. It includes the attendance percentage and no tests attempted by the candidate, feedback given by students on classes that are conducted, and the separation of students on their performance with the above factors or assessments. It also helps the management take the necessary measures to improve their performance in training. Our project also focuses on the performance of the students through graphical analysis Which helps the students easily understand their performance in training and helps them improve their performance in the areas where they are back. The implementation of the project is done through MERN (MongoDB, Express, React, Node), which is), which is scalable, user-friendly, and adoptable

## 8 Reference

[1] Angela L. Duckworth and Martin E.P. Seligman Self-Discipline Outdoes the IQ in Predictin Academic Performance of Adolescents

[2] Alf Lizzio, Keithia Wilson &Roland Simons University Students' Perceptions of the Learning Environment and Academic Outcomes: Implications for theory and practice

[3] Barry J. Zimmerman  Self-efficacy & educational development Monique Lortie Development of a Website: An Experience of interface with Manual Workers and Web Specialists

[4] Zinovieva1, Artemchuk Anna V Iatsyshyn The use of online coding platforms as additional distance tools in programming education

[5] Rasha Albashaireh; Hua Ming A Survey on Online Learning Platforms with Initial Investigation of SituaAwwal, Olarotimi Badru

[6] Vani Vasudevan,M.G.M. Khan MERN Stack Web-Based Education Management Information Systems for Pacific Island Countries

[7]  Daniel McFarland &Diane Hamilton Factors Affecting Student Performance and Satisfaction: Online versus TraditionalCour

# GITHUB LINK

https://github.com/sai2382