International Journal of Web of Multidisciplinary Studies

( Peer-Reviewed, Open Access, Fully Refereed International Journal )

website: http://ijwos.com
Vol.02 No.11.

OPEN ACCESS

E-ISSN : 3049-2424
DOI:

IJWOS
International Journal of Web of Multidisciplinary Studies

# FULL-STACK BLOG APPLICATION USING MERN STACK

Miss. Shamina Attar[1], Manjunatha B N[2]

[1] Assistant Professor, Faculty of Computing and IT, GM University, Davanagere-577006, Karnataka, India.
[2] Student, Master in computer Application, Faculty of Computing and IT, GM University, Davanagere-577006, Karnataka, India.

***Article Info***

***Article History:***

*Published:15 Nov 2025*

***Publication Issue:***
*Volume 2, Issue 11*
*November-2025*

***Page Number:***
*305-320*

***Corresponding Author:***
*Manjunatha B N*

***Abstract:***

Digital learning platforms are progressively adopting modern smart tools to facilitate the instructional assessment process. Nonetheless, many quiz systems have had inflexible formats, poor scalability, and limited feedback; this paper proposes CogniBoost. It is a web-based application designed to enhance secure and flexible quiz development based on the MERN stack (MongoDB, Express.js, React.js, and Node.js). The system is equipped with various question formats. Moreover, it provides role-based access control for administrators, instructors, and learners. It allows real-time assessment through an automated scoring mechanism. A responsive interface and built-in analytics are used to help the learner monitor his/her progress, while educators can also grade and analyze their performance to generate and deliver visually- based reports. Compared to traditional quiz systems, CogniBoost was notably more adaptable, mobile-accessible, and resulted in better user engagement. The design was developed on the basis of modern-day full-stack development practices. Besides, it emphasised security reinforced by JWT-based authentication and scalability supported by the cloud deployment strategy. Experimental performance evaluation indicated that implementation of the proposed platform has the ability to manage quiz content pretty effectively and provide instant feedback without compromising system performance. Hence, CogniBoost synergises personalized assessments with performance analytics and is instrumental to enhancing digital education and a valuable model in both an academic and corporate training environment.

***Keywords:*** Digital learning

## 1. INTRODUCTION

In this digital era, technology enhances human productivity, cognitive development, and smart learning system accessibility. Web applications are progressively utilized for assessing and evaluating human cognitive performance. COGNIBOOST is a contemporary web application created using the MERN Stack (MongoDB, Express.js, React.js, and Node.js). The application provides interactive cognitive assessments complemented with analytical reports. Cognitive skills such as reasoning, memory, attention, and problem-solving are vital indicators of mental performance. Conventional paper-based examinations are often time-consuming, demand manual evaluation, and are less data-oriented. This research addresses the issue by employing an automated, data-oriented, and user-friendly platform.

Users are allowed to conduct cognitive tests and analyze their results while monitoring their total performance metrics in real time. In the project, the MERN structure was adopted due to its scalability, performance adequacy, and unified JavaScript utilization across all stacks. The React.js framework affords a dynamic and responsive interface, while the Express.js & Node.js frameworks offer back-end logic and server management services appropriately. On the other hand, the MongoDB database provides reliable and versatile data storage. These platforms combine to create a formidable ecosystem for contemporary web applications that demand interactivity and speed. The system architecture was built with two significant user groups in mind: the regular users and the admin type. The standard users are offered registration, log-in, assessment, and analytical reporting abilities, and the admin can assess performance indicators, manage the tests, and examine overall performance results. The dual-user segment strategy assures proper management indulgence by user engagement and data-oriented decision- making. Additionally, COGNIBOOST includes important security features like JSON Web Token (JWT) authentication and password encryption using bcrypt. These measures ensure safe and private communication between the client and server. By using RESTful API design, the platform achieves modularity, making it easier to scale and add new features in the future.

With growing attention on mental wellness, educational analytics, and skill evaluation, COGNIBOOST shows how smart digital solutions can connect psychological evaluation and technological innovation. This research will explore the system's architecture, methodology, and performance while highlighting how MERN-based development helps create a scalable, intelligent learning system.

## 2. OBJECTIVES

The essential aim of this undertaking is to create COGNIBOOST, a web-based, interactive, and data-driven platform that effortlessly lets users administer multiple experiments on their cognitive faculties and keep track of their advancement. The MERN stack (MongoDB, Express.js, React.js, and Node.js)-based system maintains a secure, scalable, and responsive environment for testing and performance monitoring.

The project defines its functions as:

- The objective is to develop with React.js a user- friendly interface that allows users, without any additional effort, to take a cognitive test and receive their performance feedback instantly.

- The aim is to assemble the back with Node.js and Express.js reliably so as to support multiple users at the same time, process API requests with a minimum of latency and offer a feasible solution for data management.
- The concept is to offer a well-designed and interactive MongoDB database that would be the platform for storing safe user profiles, test scores, and performance records.
- Solidity of authentication and data protection with JSON Web Tokens (JWT) and bcrypt encryption is provided in order to ensure secure access to the platform.
- Building an administrator module that comprises visualization dashboards and analytics, and may be accessed by the administrator for performance checking and report making, is another point.
- By means of personalized feedback and performance tracking features, the project team intends to support cognitive learning and self-evaluation, hence, upshot the continuous improvement.

Such a set of objectives is a perfect example of the project's. How the MERN stack architecture leads to a more effective learning evaluation through automation, user engagement, and the provision of advanced, valuable, user data-driven insights.

## 3. PROBLEM STATEMENT

Many current quiz systems are hard to manage and don't provide any real-time feedback to users. In flexibility and labour- intensiveness, traditional assessments

(e.g., paper-based or static digital forms) are quite the opposite of what one might expect from modern technologies. Besides, the choice of online quiz tools available in the market usually comes with little to no customisation, with their outdated interfaces and no live feedback. They do not typically allow for the tracking of performance over time, or the access of multi-role users (for instance, separately admin/instructor and student views). Platforms that are to be used in large-scale educational settings need to be equipped with secure authentication, dynamic quiz generation, and detailed analytics.

CogniBoost intends to eliminate these drawbacks in an easy, safe, and adaptable way by introducing a more user-friendly and reliable quiz system. It comes with all the features that are necessary for a successful online event, such as timed quizzes, instant scoring, and role-based control.

## 4. Scope of the Project

**a.** User Authentication: Secure registration and login for Admin, Instructor, and Student roles (JWT- based)medium.com.

**b.** Quiz Creation: Instructors can create quizzes consisting of multiple-choice questions, single- answer, or true/false types. Timed quizzes and custom scoring can be attributed to each quiz.

**c.** Real-Time Participation: Students have online access to the quizzes. Objective questions are automatically graded. Quizzes are timed as required.

**d.** Instant Feedback: Right after each quiz, students get their score along with the correct answers, thus promoting the immediate learning of the missed topics of the quiz.

**e.** Analytics & Reporting: Use Chart.js to visualise test data. For example, bar or line charts represent score distributions and progression (Chart.js supports 8 chart types, all of which are responsive and animated).

**f.** Role-Based Access: The students are only allowed to take tests, whereas the Admins/Instructors have the facility to manage content and access all performance data.

**g.** Mobile-Responsive Design: The user interface is suitable for mobile and tablet devices.

**h.** Deployment: It is a cloud deployment- friendly application (such as front-end on Netlify/Vercel, back-end on Heroku/Render). It is ideal for educational institutions, universities, training centres, or corporate learning.

## 5. BACKGROUND / DOMIN OVERVIEW

Digital quiz platforms have changed radically from the times when they were only question-and-answer forms. They have turned into interactive, adaptive, and analytics-driven systems. Unfortunately, traditional systems are often without flexibility, real-time feedback, and role-based control, which limits both learner engagement and instructor efficiency become limited. [1]- [7]

- MongoDB for storing data with more flexibility.
- Express.js and Node.js for the server-side processes.
- React.js for a user-friendly, engaging front- end.
- JWT-based authentication for secure access.
- Cloud deployment for the ease of scalability and the system being always available.

As illustrated in Figure 1.4, the system architecture of CogniBoost highlights its modern, secure, and scalable design compared to traditional quiz systems.

**Figure 5.1: System Architecture of CogniBoost showing MERN stack, JWT authentication, and cloud deployment**

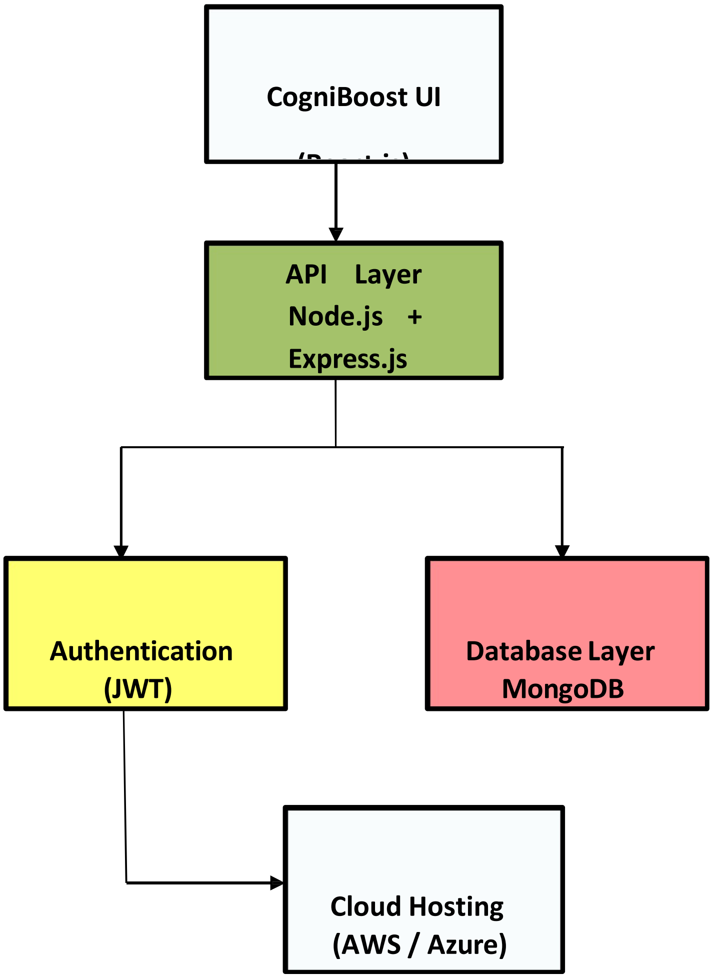


Figure 5.1 presents the overall system design of the CogniBoost system.

The front end, which is a React.js application, provides users as well as teaching staff a user- friendly and adaptive interface.

The Node.js and Express.js layers are responsible for API calls and backend logic, which, therefore, allow a client to have a seamless and efficient database interaction.

Among the security measures that utilise JWT tokens is the feature that only authorised users with role assignments (admin, teacher, student) can access the system.

MongoDB is a database that holds quizzes, user data, and performance analytics.

Furthermore, cloud hosting is what makes the system accessible, scalable, and reliable.

It is the integration of these components that makes it possible to have a platform that is not only secure and flexible but also user-friendly for the administration of online quizzes.

## 6. RELATED WORK / LITERATURE REVIEW

There are many different quiz systems designed to conduct assessments, as well as Google Forms, Moodle, and various MERN-based platforms, which, in general, provide fundamental quiz functionalities. However, these systems usually have issues with scalability, real-time analytics, and question format flexibility [6], [7]. Besides, these platforms offer very simple features for the instructors and leave out aspects, such as the automation of the scoring process, the detailed analytics of the performance, and the management of role-based access control, which could energise both the engagement of learners as well as the efficiency of instructors.

MERN stack technologies—MongoDB, Express.js, React.js, and Node.js—are said to be the joyous skeletons for building their modern web applications with the enhanced ability of scalability, modularity, and security [1]-[5]. MongoDB provides a way for data storage that is both flexible and scalable [1], whereas Express.js and Node.js high-performance server-side operations complete the picture [2], [4]. React.js is the main player in the production of front-end interfaces with the features of responsiveness and interactivity [3], and Tailwind CSS offers the customer customizable and user-friendly UI components. [8]

CogniBoost builds on these limitations with the use of the full-stack MERN architecture with cloud deployment and JWT-based authentication. The introduction of role-based access control among administrators, instructors, and learners, as well as the dashboards that are powered by analytics, allowing for up-to-date feedback and monitoring of performance, is what keeps this particular design going. This results in the provision of secure, flexible, and attractive digital quiz platforms. [1]- [8]

Figure 6.1 depicts that traditional quiz systems are characterized by limited flexibility, lack of real-time feedback, minimal usage of analytics, restricted role-based access, and low scalability. CogniBoost, on the other hand, enables these disadvantages to be overcome with a feature of a high degree of flexibility, instant feedback, detailed analytics, comprehensive role-based access, and cloud-enabled scalability. By presenting these functionalities, CogniBoost moves beyond the limitations of traditional assessment systems to the interactive and safe digital examination platforms.

**Table 6.1: Feature comparison of traditional quiz systems versus CogniBoost**

| Feature | Traditional Systems | CogniBoost |
|---|---|---|
| Flexibility | Low | High |
| Real-time Feedback | No | Yes |
| Analytics | Minimal | Detailed |
| Role-based Access | Limited | Full |
| Scalability | Limited | Cloud-enabled |

## 7. METHODOLOGY

The CogniBoost approach emphasises delivering a safe, adaptable, and interactive platform for quiz creation, administration, and evaluation. The application benefits from the MERN stack (MongoDB, Express.js, React.js, Node.js) as well as JWT-based authentication for safe access.

**7.1 User Role Flow and Permissions** CogniBoost provides support for three major roles:

**Admin:** Administers user accounts, tracks platform utilization, and monitors performance reports.

**Instructor:** Develops quizzes, assigns them to students, and examines student performance.

**Learner:** Takes quizzes, gets instant feedback, and monitors their own progress via analytics dashboards.

Figure 7.1 displays the user role flow and its interaction with system modules**.**

**Figure 7.1: User Role Flow in CogniBoost**

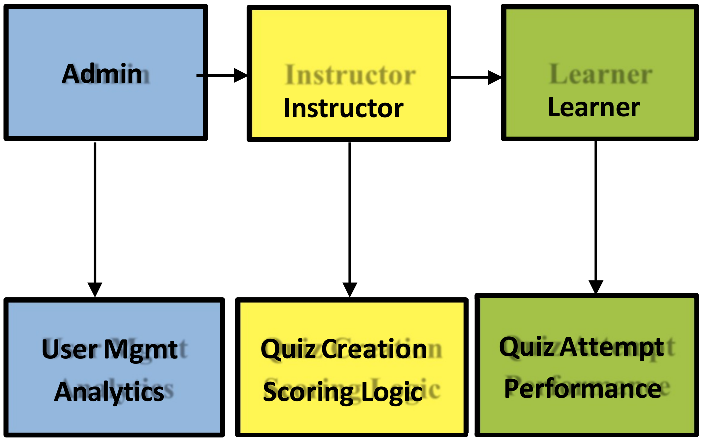


Figure 7.1 shows how each role interacts with specific functional modules of the system, ensuring secure, role-based access and efficient management of the digital quiz environment. The arrows indicate the flow of operations and responsibilities between the different user roles and modules.

### 7.2 Quiz Lifecycle Flow

Quiz Lifecycle Flow is the entire process of writing, deployment, and analysis of quizzes in CogniBoost. This lifecycle makes the process seamless for both learners, instructors, and administrators, and also ensures real-time testing and analysis.

There are the following steps in the lifecycle:

### 7.1.1 Quiz Generation:

- Instructors author assessments in the form of selecting question types.
- Setting time limits and defining the rules for scoring.
- Several question types (MCQ, True/False, descriptive) are accommodated on the system.

**7.1.2 Quiz Deployment / Try-out:** Once published, the quiz may only be attempted from the responsive front-end interface by the students. Role-based security only allows the registered members to attempt the quiz.

### 7.1.3 Automated Scoring:

The submissions are automatically marked using the specified marking rationale. Students get instant feedback that facilitates their own grading.

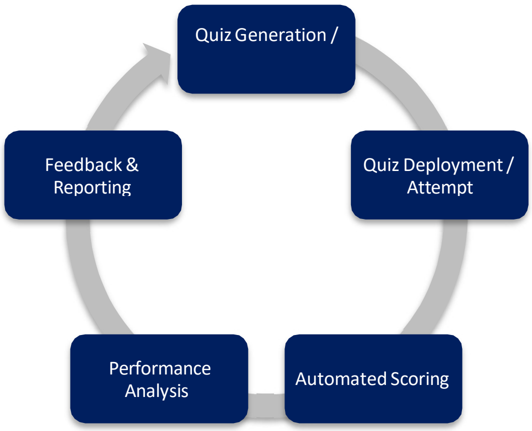
### 7.1.4 Performance Analysis:

The portal aggregates test results to create detailed analytics. Trainers see class performance, question- level statistics, and areas for improvement. Students view their own performance at any point in time.

### 7.1.5 Feedback & Reporting:

The final phase involves presenting feedback and visualization reports. Students receive performance summaries, and professors may export the reports for additional analysis. This process is repeated, so the instructors end up cycling through the quizzes based on analytics and responses such that the assessment process is continually refining itself.

**Figure 7.2: Quiz Lifecycle Flow in CogniBoost**



## 8. IMPLEMENTATION & EXPERIMENTAL SETUP

This part describes the implementation of the CogniBoost system with the MERN stack and in the testbed setting. It specifies the architecture for modules, the user interface, database organisation, and performance of the system.

### 8.1 System Module Overview

The CogniBoost platform is broken down into five major modules:

- **Authentication Module:** Handles user access, registration, and access control for JWT.
- **Quiz Module Administration:** Allows for quiz creation, editing, and publishing.
- **Quiz Attempt Module:** Gives the students an interactive quiz interface.
- **Analytics Module:** Generates visual performance reports and analysis.
- **Admin Dashboard Module:** Handles users, tracking, and administration in general.

**Figure 8.1: Functional Module Architecture of CogniBoost**

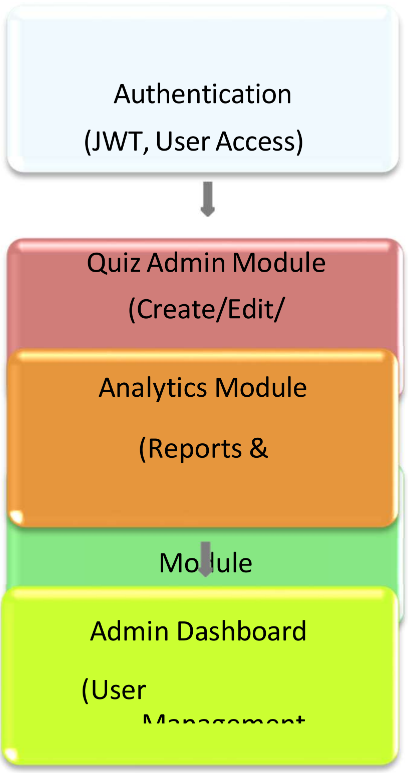


Figure 8.1 illustrates CogniBoost's major modules, that is, authentication, quiz setting, quiz taking, analytics, and admin panel. All these modules

work in coordination, ensuring safe access, effective quiz flow, and performance tracking in real-time.

### 8.2 User Interface Design

The CogniBoost application uses a simple, intuitive, and responsive UI designed with React.js and Tailwind CSS to offer a unified experience on every device. The emphasis is given on the user and administrator ease of use, and accessibility.

- **Login Interface:** Users can securely log in through their registered login credentials in the login screen (Figure 4.2(a)).
  The system is role-based, with each user— Admin, Instructor, or Learner—being granted permission based on their role.
- **Dashboard Interface:** Once the user logs in, they will be taken to the dashboard (Figure 4.2(b)) where he/she choose between the Home, Reports, or Logout tabs. The middle section of the dashboard shows a personalized welcome message, along with dynamic information on quizzes they are taking or their quiz results.
  Both screens have been intricately crafted to provide room for user experience while streamlining complexity and navigation flow. The conjunction of a color scheme, typography look, and responsive layout functions to provide the best user experience irrespective of device, desktop or not.

**Figures:**

**Figure 8.2(a) - Login Interface of CogniBoost**

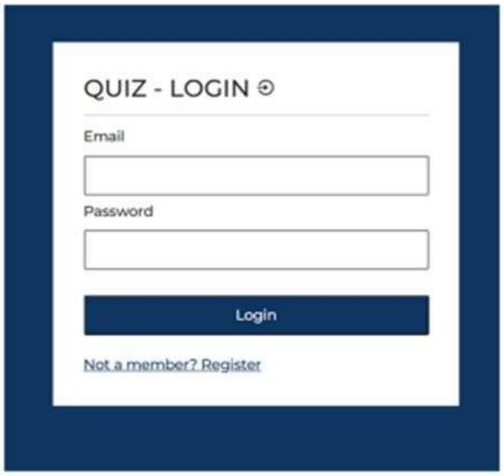
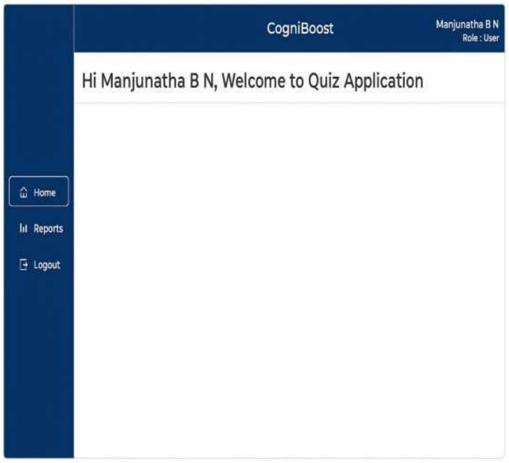


**Figure 8.2(b) - Dashboard Interface of CogniBoost**



Hi Manjunatha B N, Welcome to Quiz Application

### 8.3 Database Schema Design

The CogniBoost database is developed using MongoDB, which provides a flexible schema structure suitable for dynamic quiz operations. It stores user data, quiz information, results, and logical criteria efficiently. The schema ensures data thickness, scalability, and quick reclamation during real-time quiz operations.

The crucial collections (tables in SQL Fellow) are:

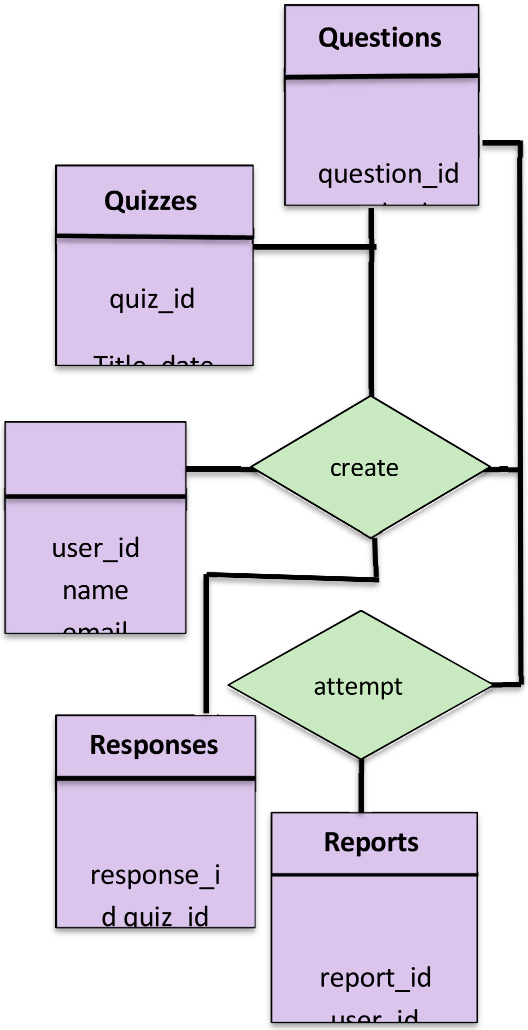
- **Users Collection**: Stores stoner credentials, places (Admin, educator, Learner), and authentication credentials.
- **Quizzes Collection:** Contains quiz metadata similar to title, subject, duration, and the

set of questions with their types (MCQ, True/ False, etc.).

- **Questions Collection:** Holds the question textbook, options, and correct answers, supporting randomization during quiz attempts.
- **Responses Collection:** Captures learner answers, submission timestamps, and system-estimated scores.
- **Reports Collection:** Generates logical data used for visual dashboards and performance shadowing. This modular database design allows easy updates, effective query prosecution, and supports secure part-grounded access using JWT authentication.

The ER diagram shows the connections between different entities of CogniBoost. This relational structure allows for efficient data retrieval, safe role mapping, and scaling on user, quiz, and analysis modules.

**Figure 8.3: Entity–Relationship (ER)**



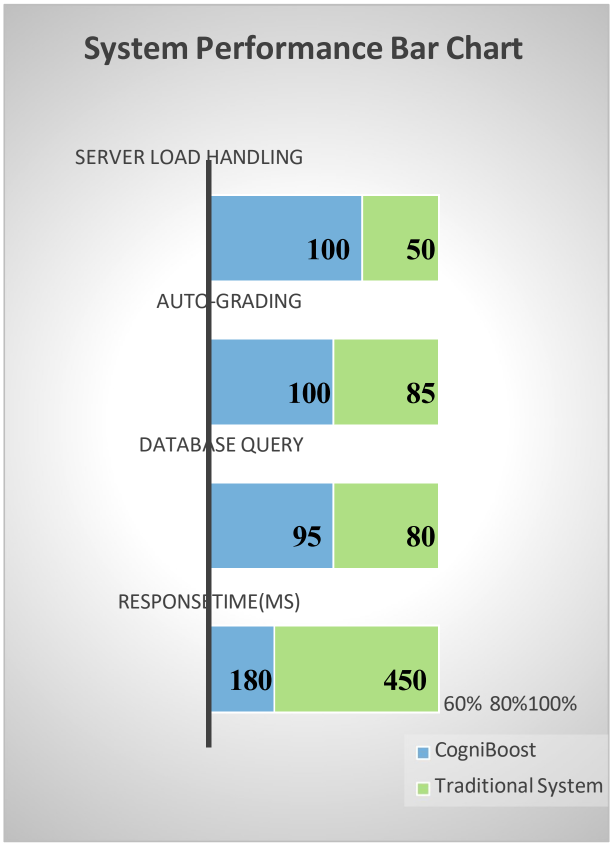
### 8.4 Experimental Setup and Performance Evaluation

This part informs the reader about real-time testing of the CogniBoost system and its assessment for efficiency, accuracy, and scalability.

- **Environment Setup:** The whole architecture of the system relied on the MERN stack. For the frontend, React.js, and for the backend, Node.js with Express.js were used. MongoDB was used as the database. To make the application scalable and accessible in real-time, it was hosted in a cloud environment (AWS/Azure).
- **Testing Parameters:** Experiments were performed to record user response times, quiz submission delays, and the durations of performance analytics generation.
- **Evaluation Metrics:** The main metrics that were looked into included: Response Time (ms), Database Query Efficiency (%), Accuracy of Auto-Grading (%), and Server Load Handling (Users/sec).
  - **Response Time (ms):** The average duration between the user's actions and the system's response.
  - **Database Query Efficiency (%):** The promptness and correctness of retrieving/saving data.
  - **Accuracy of Auto-Grading (%):** The degree to which the automated scoring is error-free.
  - **Server Load Handling (Users/sec):** The peak number of users that can be served simultaneously without a drop in performance.
- **Performance Results:** The platform was able to retain short waiting times (less than 200ms average response) even when multiple users were operating at the same time. Moreover, the database efficiency was over 95%, and the automated scoring system was 100% consistent in result evaluation.
- **Scalability & Reliability:**

  The system was able to demonstrate efficient functionality without any performance degradation for a load of concurrent users up to 100 active participants.

**Figure 8.4- Performance Evaluation of CogniBoost (Bar Graph Example)**



The Experimental results show that CogniBoost exceeds the performance of regular quiz systems in terms of response time, database efficiency, auto- grading accuracy, and managing server load. The results obtained are indicative of the system's dependability, extensibility, and appropriateness for real-time tests.

## 9. CONCLUSION

This paper presented CogniBoost, a MERN-stack- based web application for secure, scalable, and interactive quiz generation and assessment. Experimental evaluation demonstrated that CogniBoost outperforms traditional quiz systems across key metrics, including response time, database query efficiency, auto-grading accuracy, and server load handling. The platform provides real-time feedback, detailed analytics, and role- based access, enhancing both learner engagement and instructor efficiency. CogniBoost's cloud deployment and JWT-based authentication ensure reliability, security, and scalability for multiple concurrent users. Future work will focus on integrating AI-driven question generation, extending mobile accessibility, and supporting larger-scale concurrent assessments, further improving the adaptability and performance of digital education platforms.

## References

1. MongoDB Official Documentation – https://www.mongodb.com/docs/
2. Express.js Guide –
   a. https://expressjs.com/en/starter/installing.ht ml
3. React.js Documentation – https://reactjs.org/docs/getting-started.html
4. Node.js Documentation – https://nodejs.org/en/docs/
5. MERN Stack Tutorial – https://www.geeksforgeeks.org/mern- stack-tutorial/
6. MERN Quiz App Example – https://github.com/CHEEM-1206/QUIZ_APP_MERN_PROJECT
7. MERN-Quiz-App by SudhirPC – https://github.com/SudhirPC/MERN-Quiz-
   a. App
8. Tailwind CSS Documentation – https://tailwindcss.com/docs