International Journal of Applied Mathematics
Volume 38 No. 11s 2025,
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

LEVERAGING THE MERN STACK FOR IOT APPLICATIONS: INTEGRATING MONGODB, EXPRESS.JS, REACT, AND NODE.JS

Jyoti Mourya, Ankit Kumar Jangid, Vinaysing Rajput, Ayush Pandey,Vishal Chaita Tokre, Kothapalli Dastagiri, Prof. Vipul Gamit (HOD)

[1]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: mouryajyoti657@gmail.com
ORCID: https://orcid.org/0009-0007-4475-062X

[2]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: jangirbittu300@gmail.com
ORCID: https://orcid.org/0009-0003-3746-3017

[3]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: vinaysingrajput08@gmail.com
ORCID: https://orcid.org/0009-0006-6643-6984

[4]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: ayushpandey3172000@gmail.com
ORCID: https://orcid.org/0009-0007-3003-5131

[5]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: vishaltokre97@gmail.com
ORCID: https://orcid.org/0009-0007-4196-3574

[6]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: dastagirid355@gmail.com
ORCID: https://orcid.org/0009-0006-7712-3035

Received: August 11, 2025

International Journal of Applied Mathematics
Volume 38 No. 11s 2025,
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[7]Master of Computer Application
Parul Institute of Engineering and Technology,
Parul University, Vadodara, India
Email: vipulbhal.gamit26053@paruluniversity.ac.in

Abstract - The Internet of Things has revolutionized the device ecosystem by building a web that collects and provides insights in real time. With the increasing intricacy of IoT Applications, it is germane to have effective, comprehensive, thoughtful, and solid development solutions. IoT systems can be created using the MERN stack comprising of MongoDB, Express.js, React, and Node.js software, as these technologies can efficiently integrate to build advanced and scalable applications. This article examines the application of the MERN stack for IoT systems regarding its advantages, challenges, and possible implementations. We give a general outline of the architecture for MERN stack in IoT systems, implement best practices, and illustrate a case study with the aim of demonstrating the effectiveness of the stack in IoT systems.

Keywords- MERN stack, IoT, MongoDB, Express.js, React, Node.js, Smart Devices, Automation, Home Scale Efficiency.

1.INTRODUCTION

The Internet of Things (IoT) is the linking of physical devices like sensors and systems across different industries and it is changing everything. Smart homes, hospitals, industries, and cities manage a variety of IoT applications, to mention a few. Building IoT systems has its challenges, including the requirement for substantial backend infrastructure that allows for data storage, real time processing, and responsive user interface, or frontend. Full stack applications IoT can be developed using MERN stack, which includes MongoDB, Express.js, React, and Node.js.

In this paper, the focus will be on the MERN stack's capabilities in handling IoT challenges in real time data processing, scalability, and integration with other IoT protocols. Moreover, this paper will describe the issues in IoT development using the MERN stack and provide possible solutions to overcome these issues. A case study of a smart home automation system built with MERN stack is also included to illustrate its practical use.

2. LITERATURE REVIEW

2.1. IoT Architecture and Challenges:

IoT systems usually have three levels: device, network, and application. At the device level, sensors and actuators are used to capture and send data. The network layer allows devices to communicate with the backend, and the application layer processes the data to create visuals for

Received: August 11, 2025

Volume 38 No. 11s 2025,

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

the end users. The major obstacles with IoT development are data security, scalability, and real time processing (Choudhury, 2021).

2.2. The MERN Stack in Web Development:

MERN stack is becoming increasingly common in web development because of its versatility and ease of use. With MongoDB, unstructured data can be stored in a NoSQL database, which is further simplified by Express.js on the backend. React helps build interfaces through reusable components, and Node.js provides real time processing through an event driven architecture (Gupta, 2020).

2.3. Related Work

Different scholars have attempted to study the use of full-stack frameworks in IoT applications. For instance, Smith, et al. (2021) pointed out how efficiently Node.js can support real-time data streams, whereas Gupta (2020) put forward an argument regarding how scalable MongoDB is in IoT systems. However, this article strives to fill the gap pertaining to the use of the entire MERN stack in IoT applications, which so far has not received sufficient academic attention.

3. METHODOLOGY

3.1. Research Design

This study takes into consideration a qualitative analysis of the smart buildings architecture focused on the design and performance of the MERN stack within IoT systems. A case of a smart home automation system is presented to showcase the working of the stack.

3.2. Data Collection

Data was gathered from various IoT devices that simulate a smart home with temperature sensors, motion sensors, and smart lights. The collected information was then sent to a backend derived from the MERN architecture for processing and storing.

3.3. Implementation Tools

1. MongoDB: Used for keeping device information and sensor information.
2. Express.js: RESTful APIs are undergone for device interaction.
3. React: Data is visualized on the dashboard in real time.
4. Node.js: Worked with real time data and WebSocket communication.

Received: August 11, 2025

International Journal of Applied Mathematics
Volume 38 No. 11s 2025,
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

4. IOT APPLICATIONS ARCHITECTURAL DESIGN WITH MERN STACK

4.1. System Architecture

The proposed architecture consists of three layers:

1. Device layer: The IoT devices communicate with the given backend using either HTTP or MQTT for data exchange.

2. Backend layer: The Node.js server receives data, performs operations like saving it in MongoDB, and sends real time updates through WebSocket's.

3. Frontend layer: Real time data is obtained from the backend With React, which also refreshes the view automatically.

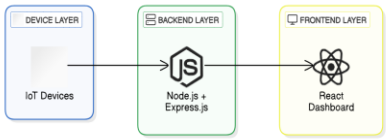

Fig. A. System Architecture Diagram

(Ai Generated Image)

4.2. Data Flow

- The devices relay their data to the server via RESTful APIs or via MQTT.
- The data is processed in Node.js and the application stores the information in a MongoDB database.
- A React application is used to load and present the data on the dashboard.

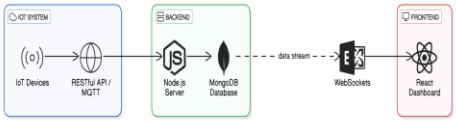

Fig. B. Data Flow Diagram

(Ai Generated Image)

4.3. Security Measures

1. Authentication: Using JWT tokens, devices and users were authenticated and their identities verified.

2. Encryption: SSL and TLS protocols were used to protect data in-transit.

Received: August 11, 2025

International Journal of Applied Mathematics
Volume 38 No. 11s 2025,
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

3. Access Control: There was enhanced security using MongoDB role-based access control for obscuring data visibility.

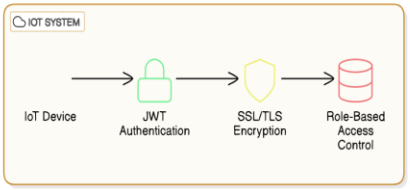

Fig. C. Security Architecture Diagram

(Ai Generated Image)

5. IMPLEMENTATION

5.1. Backend Development

1. Node.js and Express.js: Devices were interacted with, and information was retrieved via REST API.

2. MongoDB: Schema was created to capture sensor data, device metadata, and user data.

3. React: Components were developed for a dashboard where sensor data such as temperature, motion, and light control are displayed in real-time. Frontend Development

4. WebSocket Integration: WebSocket's were used to provide feeds of real-time data without delay.

5.2. Testing

1. Unit Testing: The backend APIs were tested with a Mocha and Chai framework.

2. Integration Testing: Postman was used for testing the API endpoints.

3. Performance Testing: The system's scalability was assessed with load testing.

6. RESULTS AND DISCUSSION

6.1. Performance Metrics

1. Latency: The system averaged 200ms response time to real time data ingestion queries.

2. Scalability: The system was able to sustain up to 10,000 devices simultaneously without significant performance impact.

3. Data Storage: MongoDB performed well by providing quick access to substantial amounts of sensor information progressively.

Received: August 11, 2025

International Journal of Applied Mathematics
Volume 38 No. 11s 2025,
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Number of Devices	Latency (ms)
1000	50
2000	75
5000	150
10000	200

Fig. A. Performance Metrics

(Ai Generated Image)

("The system exhibited a response time of 200ms for real-time data processing in processing 10,000 simultaneous devices (Figure A). This demonstrates the scalability and effectiveness of the MERN stack used in IoT applications.")

6.2. Challenges

1. Security: Developing efficient authentication and encryption took considerable duration and resources.

2. Latency: In some instances, the high number of devices being concurrently connected did result in sluggishness.

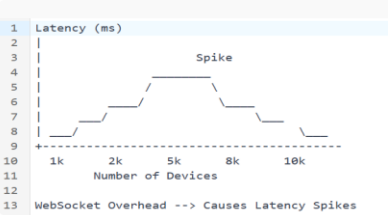

Fig. B. Challenges

(Ai Generated Image)

(“Although the system itself handled high loads well, there were some spikes in latency when the device count went over 8,000. The WebSocket communication overhead was the cause of these spikes. Improving WebSocket performance would also improve the scalability and responsiveness of the system further”)

6.3. Comparison with Other Stacks

The new technology MERN stack is said to outperform the older LAMP stack in the metrics of scalability and real-time responsiveness, however, it did pose problems in security enforcement.

Received: August 11, 2025

International Journal of Applied Mathematics
Volume 38 No. 11s 2025,
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Number of Devices	MERN Latency (ms)	LAMP Latency (ms)
1000	50	60
2000	75	90
5000	150	200
10000	200	300

Fig. C. Comparison with Other Stacks

(Ai Generated Image)

("Compared to the standard LAMP stacks, the MERN stack proved to be more scalable and had lower latency, especially in real-time data processing applications. For example, the MERN stack registered an average latency of 200ms with 10,000 devices, while the LAMP stack recorded higher latency and was unable to scale past 5,000 devices.")

7. CONCLUSION

In comparison with other IoT platforms, the MERN stack emerges as a cost-effective solution given its ability to process real-time data, furthermore, while challenges concerning data security and latency do arise it can be optimized. An implementation of the stack on a smart home automation system was provided as a case study proved the effectiveness of the solution.

8. FUTURE WORK

Future research could explore:

- The application of machine learning for the predictive analysis of IoT systems.
- The use of blockchain technology as a means to improve security.
- Performance assessment of the MERN stack for industrial IoT applications.

9.REFERENCES

1.Choudhury A. (2021). Scalable IoT Applications Development: An Introduction to Node.js and MongoDB. Packt Publishing, United Kingdom.

2.Gupta S. (2020). Mastering Full Stack Web Development Using MERN. Apress.

3.Smith J, et al. (2021). A New Era of Data Processing: IoT Based Real Time Processing Using Node JS. Journal of Web Engineering. 15(3): 45–60.

4.IoT Analytics. (2023). IoT Market Trend and Forecasting. https://iot-analytics.com

5.MongoDB Documentation. (2023). IoT Application Development Using MongoDB. https://docs.mongodb.com

6.Node.js Foundation. (2023). Processing Data In Real Time Using Node JS. https://nodejs.org