

MERN Stack Comparison with previous technologies

Ms. Deepika Khurana, Dhruv Kumar Jha, Chaaahat Chowla, Ishita Arya
Delhi Technical Campus, Greater Noida, India,
dhruvkkumar0603@gmail.com, Chaaahatchowla@gmail.com

Abstract

The MERN stack, comprising MongoDB, Express.js, React.js, and Node.js, has emerged as a powerful and cohesive framework for developing modern web applications. This full-stack JavaScript solution enables seamless development across both front-end and back-end, offering significant advantages in terms of performance, scalability, and development speed. MongoDB provides a flexible NoSQL database, Express.js handles server-side logic, React.js offers a robust front-end framework for building dynamic user interfaces, and Node.js serves as the runtime environment for JavaScript on the server. This paper explores the architecture, core components, and advantages of the MERN stack, highlighting its role in the development of scalable, responsive, and maintainable web applications. Furthermore, we examine the challenges associated with adopting the MERN stack in large-scale production environments and provide insights into best practices for developers.

Keywords: MERN Stack, Full-stack JavaScript, MongoDB Express.js, React.js, Node.js,

1. Introduction

demand for high-performance, scalable, and maintainable web applications has led to the rise of modern full-stack frameworks. Among the most popular and widely adopted solutions is the MERN stack, which consists of MongoDB, Express.js, React.js, and Node.js. This stack offers a seamless, unified development experience using JavaScript for both client-side and server-side code, creating a cohesive and efficient environment for building dynamic web applications.

The MERN stack has gained significant attention due to its simplicity, flexibility, and performance. It allows developers framework, React.js for the front-end, and Node.js as the runtime environment, MERN provides a robust foundation for building modern web applications.

However, before the advent of the MERN stack, web development relied on a variety of other frameworks and technologies, each with its own strengths and limitations. Technologies such as LAMP (Linux, Apache, MySQL, PHP) and MEAN (MongoDB, Express.js, Angular, Node.js) were widely used for building web applications, but they often lacked the flexibility, speed, and rich user experiences offered by newer tools like React.js in the MERN stack. Other alternatives, such as the traditional Ruby on Rails or

Django, while powerful, required more complex or less integrated solutions for handling JavaScript on both the client and server sides.

This paper aims to compare the MERN stack with previous technologies commonly used in full-stack web development, such as LAMP, MEAN, and other widely used frameworks. By evaluating factors like ease of use, scalability, performance, community support, and ecosystem, we will highlight the advantages and challenges of adopting MERN for modern web application development. Additionally, this comparison will provide insights into how MERN stacks up against traditional technologies in terms of development speed, user experience, and long-term maintainability, helping developers make informed decisions about the best tools for their projects.

Through this research, we aim to provide a clear understanding of how the MERN stack has emerged as a dominant choice for full-stack development, and where it

2. RELATEDWORK

A web application in Linux Apache MySQL PHP(LAMP) responsive and closer to native mobile applications in terms of performance and interactivity [5]. Today, web applications continue to evolve with the integration of cloud computing, microservices architecture, and progressive web apps (PWAs), making them more scalable, reliable, and capable of providing native-like experiences across various devices [6].

4. METHODOLOGY

Web application development using the MERN stack (MongoDB, Express.js, React, Node.js) has become a popular approach for building modern, scalable, and dynamic web applications. The MERN stack is based on JavaScript, offering a unified language for both the front-end and back-end, which simplifies the development process. MongoDB, a NoSQL database, allows for flexible data storage, making it suitable for applications with dynamic data structures. Express.js, a web framework for Node.js, provides a robust and minimal set of features for building server-side applications. Node.js itself allows for efficient handling of asynchronous requests, making it ideal for applications that require high concurrency, such as social media platforms or real-time web forums. React, a front-end JavaScript library, enables the creation of dynamic and interactive user interfaces with a component-based architecture, making it easy to manage and update the user interface efficiently.

The use of MERN in web development promotes rapid development due to its rich ecosystem of tools and libraries. For instance, React facilitates the creation of single-page applications (SPAs) that offer seamless user experiences, while Node.js supports real-time communication with its event-driven architecture. Additionally, the integration of MongoDB with Express.js and Node.js simplifies handling JSON-like data structures across the entire stack, improving scalability. Together, these technologies allow developers to build full-stack applications with streamlined development processes, improving maintainability and performance. reload, enhancing user experience [4]. More recently, the shift towards Single Page Applications (SPAs) powered by frameworks like Angular, React, and Vue.js has made web apps more the following section will discuss MERN Stack components and their implementation.

4.1 MONGODB

MongoDB is a NoSQL database developed with a focus on high performance, scalability, and flexibility for handling large volumes of unstructured or semi-structured data. It was initially created by 10gen (now MongoDB, Inc.) and released in 2009 as an open-source document-oriented database. The database is designed to store data in a format called BSON (Binary JSON), which allows it to store complex and nested data structures, making it more flexible than traditional relational databases.

The primary programming language used in the development of MongoDB is C++. C++ was chosen for its performance advantages, especially when it comes to memory management and high-speed data operations. MongoDB's core components, such as the query engine and storage system, are written in C++ to optimize the database's efficiency and scalability. Over time, MongoDB has incorporated other programming languages and technologies for supporting various features and client interfaces, including JavaScript, Python, Go, and Java, among others, for its various client libraries, drivers, and tools.

MongoDB is schema-less, meaning documents within a collection do not need to follow the same structure. This allows applications to handle semi-structured and unstructured data more easily. This is especially useful when dealing with evolving data models where fields may change over time.

MongoDB supports horizontal scaling through a feature called sharding, which distributes data across multiple servers. This allows it to handle massive amounts of data by scaling out across multiple nodes instead of relying on a single server.

Replica Sets ensure high availability by maintaining multiple copies of data across different servers, providing redundancy and failover capabilities. MongoDB is designed for high-speed read and write operations. It supports in-memory processing for faster queries and indexing.

MongoDB indexes data automatically for faster access and offers advanced indexing options, such as While MongoDB is a NoSQL database, it has added support for multi-document ACID transactions starting with version 4.0, making it suitable for applications that require transaction-based operations and data consistency across multiple documents. This brings MongoDB closer to the reliability and data integrity features typically found in traditional relational databases.

MongoDB is highly useful for big data management due to its flexibility, scalability, and performance characteristics that align well with the demands of handling large volumes of complex data.

4.2 EXPRESSJS

Express.js is built on top of Node.js, which is a runtime environment for executing JavaScript on the server side. While Node.js allows you to build server-side applications using JavaScript, it doesn't include built-in utilities for handling web server functionalities (like routing, middleware, or HTTP request handling). Express.js fills this

gap by providing a simple, unopinionated framework that as geospatial and text indexes, to optimize query performance.

PUT, DELETE) and map them to specific functions or handlers. This is especially helpful in modern RESTful API development, which is a common requirement in modern web applications.

One of Express's defining features is its middleware system. Middleware functions sit between the request and response cycle, allowing you to process requests, perform operations like authentication, data parsing, logging, etc., before sending a response back to the client. Express allows you to add middleware globally or for specific routes. This modularity makes it very useful in modern applications, where you may need to process requests at different stages of the application lifecycle.

Express.js has become one of the most popular frameworks for building modern web applications, particularly in server-side JavaScript (Node.js) environments. Its flexibility, minimalist approach, middleware support, and ease of use make it an ideal choice for developing both traditional web applications and modern APIs. Whether you're building a RESTful API, a real-time web application, or integrating with front-end frameworks, Express.js provides a robust foundation to support these functionalities.

4.3 REACTJS

React.js is an open-source JavaScript library developed by Facebook (now Meta) for building user interfaces, especially for single-page applications (SPAs) where a dynamic, interactive user experience is needed. React allows developers to build reusable UI components that can manage their own state and render efficiently based on data changes. Since its release in 2013, React has become one of the most widely adopted front-end libraries, powering large-scale web applications like Facebook, Instagram, and Airbnb.

React's core concept is based on components, which are reusable, self-contained units that manage their own state and lifecycle. This component-based structure makes it easy to break down complex UIs into smaller, more manageable parts, improving code organization and maintainability.

Developers can create custom components for various parts of the application (e.g., buttons, forms, modals) and reuse them across different parts of the application, reducing the facilitates building web servers.

Express.js follows a minimalist design philosophy, providing just enough tools to build powerful web applications, without enforcing a specific structure or architecture. This gives developers the flexibility to structure their application according to their own preferences or specific project needs. Unlike more opinionated frameworks, Express does not impose specific patterns or dependencies, allowing developers to use third-party libraries, tools, and modules freely.

Express simplifies handling HTTP requests. You can easily define routes for different Hyper Text Transfer Protocol (HTTP) methods GET, POST, updates the Virtual DOM first and then compares it to the real DOM using a process called reconciliation. Only the

differences between the Virtual DOM and the real DOM are applied, which results in fewer and more efficient updates to the actual DOM. This makes React applications highly performant, especially when handling large and dynamic UIs, by reducing unnecessary re-renders and providing a smoother user experience.

React can also be used for mobile app development through React Native. React Native enables developers to build native mobile applications for iOS and Android using the same concepts and components as React for web development. With React Native, you can reuse the same codebase (or most of it) for both web and mobile platforms, reducing development time and cost.

4.4 NODEJS

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to execute need to write redundant code.

One of the key innovations in React is the Virtual DOM, which is a lightweight in-memory representation of the real Document Object Model (DOM). When the state of an application changes, React Node.js is built on several key components that work together to provide a powerful, scalable, and efficient runtime environment for building server-side applications. At its core, Node.js is a single-threaded event-driven engine powered by the V8 JavaScript engine (developed by Google for Chrome). It employs a highly efficient non-blocking, asynchronous I/O model that enables it to handle many simultaneous connections with high throughput.

Node JS Platform does not follow the Request/Response Multi-Threaded Stateless Model. It follows Single Threaded with Event Loop Model. Node JS Processing model mainly based on JavaScript Event based model with JavaScript callback mechanism. You should have some good knowledge about how JavaScript events and callback mechanisms work. If you don't know, please go through those posts or tutorials first and get some ideas before moving to the next step in this post. As Node JS follows this architecture, it can handle more and more concurrent client requests very easily. Before discussing this model internals, first go through the diagram below. I tried to design this diagram to explain each and every point of Node JS Internals. The main heart of Node JS Processing model is "Event Loop" [7].

The MERN stack is an excellent choice for developing modern web forum applications due to its flexibility, speed, and scalability. It allows for the creation of feature-rich forums with real-time communication, efficient data JavaScript code outside of a web browser. Built on the V8 JavaScript engine (developed by Google for Chrome), Node.js is designed to build scalable, high-performance web applications, especially for server-side scripting. Since its release in 2009 by Ryan Dahl, Node.js has become a dominant technology in the web development world, particularly for microservices, real-time applications, APIs, and Node.js enables full-stack JavaScript development, allowing developers to write both server-side and client-side code using the same language, JavaScript. This consistency in the language reduces complexity and learning curves, streamlining development processes. real-time user interactions are more efficient. Features like real-time notifications, live chat, and instant updates to posts or threads can be implemented seamlessly. For forums that rely on live discussions or instant feedback, this real-time capability is crucial.

5. CONCLUSION AND FUTURE WORK

The MERN stack offers a strong foundation for building modern web forums that are scalable, interactive, and efficient. However, continued improvements in performance optimization, security, AI integration, and real-time features will ensure that MERN remains a powerful tool for building future-proof web forum applications. As technology advances, leveraging these enhancements will allow developers to provide even richer, more engaging experiences for users, making MERN an even more compelling choice for web forum development in the years to come.

6. ACKNOWLEDGMENT

We would like to acknowledge the creators and maintainers of the MERN technologies—MongoDB, Express.js, React.js, and Node.js—for providing the open-source tools that made this research and development possible. Their commitment to the growth and innovation of web technologies continues to inspire developers around the world.

7. REFERENCES

7.1 Webpage:

- [1]. TutorialsPoint. (2021). PHP MySQL Forum. https://www.tutorialspoint.com/php/php_mysql_forum.htm
- [2]. W3Schools. (2023). JavaScript Tutorial. <https://www.w3schools.com/js/>
- [3]. Tutorials point. (2021). Web Application Development. https://www.tutorialspoint.com/web_development/
- [4]. MDNWebDocs. (2023). AJAX- Asynchronous JavaScript and XML. <https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX>
- [5]. Google Developers. (2023). Single Page Application (SPA) Overview. <https://developers.google.com/web/updates/2018/05/inside-browser-part1>
- [6]. Microsoft. (2023). Progressive Web Apps. <https://learn.microsoft.com/en-us/microsoft-edge/progressive-web-app>
- [7]. <https://www.digitalocean.com/community/tutorials/node-js-architecture-single-threaded-event-loop>
- [8]. <https://www.mongodb.com/resources/languages/mern-stack>
- [9]. <https://www.bocasay.com/how-does-the-mern-stack-work/>
- [10]. <https://expressjs.com/en/advanced/best-practice-performance.html>
- [11]. <https://www.icoderzsolutions.com/blog/most-popular-websites-built-with-reactjs/>