iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** II **Month of publication:** February 2025

DOI: https://doi.org/10.22214/ijraset.2025.66969

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue II Feb 2025- Available at www.ijraset.com

Boosting Scalability and Performance: Transitioning from PHP to the Dynamic MERN Stack

Mr. Gaurav Omprakash Prajapati, Mr. Omkar Ganesh Mahangare, Mr. Ahemad Rafik Shaikh, Mr. Rohit Shahaji Thite, Prof. Manisha Patil, Dr. Geetika Narang
Computer Engineering, Trinity College Of Engineering and Research , Pune, (SPPU), Maharastra ,India

Abstract: ***The existing PHP-based application suffers from significant scalability and performance issues, limiting its ability to handle increasing user demand and leading to slow response times. This situation necessitates a migration to a more efficient architecture that can provide enhanced performance and seamless scalability, ensuring a better user experience and support for future growth PHP applications often face challenges when trying to scale efficiently, especially in handling multiple concurrent requests under heavy load. This can lead to slow performance and bottlenecks. In contrast, the MERN stack, particularly with Node.js, is designed to handle a large number of simultaneous connections thanks to its non-blocking, event-driven architecture, making it inherently more scalable for modern web applications.***

I. INTRODUCTION

In today's rapidly evolving technological landscape, organizations are increasingly seeking to modernize their web applications to stay competitive and meet growing user demands. One effective approach to achieving this is migrating from a traditional PHP-based codebase to a more contemporary and robust technology stack, such as the MERN stack, which includes MongoDB, Express.js, React.js, and Node.js.

PHP has been a reliable and widely-used server-side scripting language for many years, powering numerous web applications with its ease of use and extensive ecosystem. However, as web applications become more complex and performance expectations rise, the limitations of PHP, such as its scalability challenges and less flexible architecture, become more apparent. To address these challenges and leverage advancements in web development, many organizations are transitioning to the MERN stack.

The MERN stack offers a modern, full-stack JavaScript solution that streamlines development through a unified language across both client and server sides. Node.js, with its non-blocking I/O and event-driven architecture, provides a high-performance runtime environment for server-side applications. Express.js, a minimal and flexible Node.js web application framework, simplifies server-side logic and routing. React.js, a powerful front-end library developed by Facebook, enables the creation of dynamic and responsive user interfaces.

Migrating to the MERN stack presents several compelling benefits, including improved application performance, enhanced scalability, and more straightforward code maintenance. This transition will also involve careful planning to ensure that existing features are preserved and that data integrity is maintained throughout the migration process. By adopting the MERN stack, the application can better meet current and future demands, providing users with a more seamless and efficient experience while positioning the organization for future growth and innovation.

II. METHODOLOGY

A. Migrate the Database

1) *Analysis of SQL Database Structure*: Begin by thoroughly examining the existing SQL database to understand its schema and relationships between tables.
2) *ER Diagram Creation*: Create an Entity-Relationship (ER) diagram to visually represent the database structure and relationships, which will guide the migration process.
3) *MongoDB Database Design*: Based on the ER diagram, design a MongoDB database schema that aligns with the NoSQL paradigm, ensuring proper data representation and efficiency.

©IJRASET: All Rights are Reserved

B. Analyze the Architecture and Code

1) *Frontend and Backend Code Review*: Conduct a comprehensive review of both frontend and backend codebases to identify existing functionalities and areas for improvement.
2) *Architectural Analysis*: Evaluate the current architecture, focusing on the MVC (Model-View-Controller) structure. This analysis will facilitate the division of the code into manageable components aligned with the new architecture.

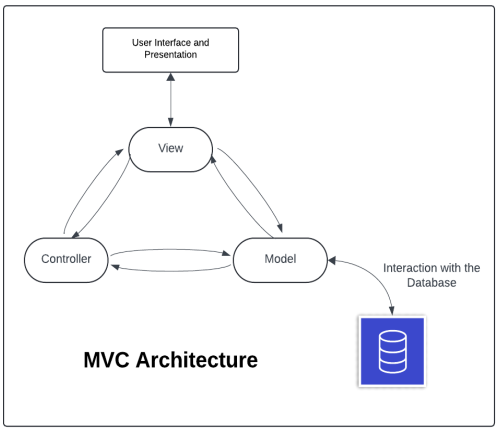


Fig. 2.1: MVC Architecture

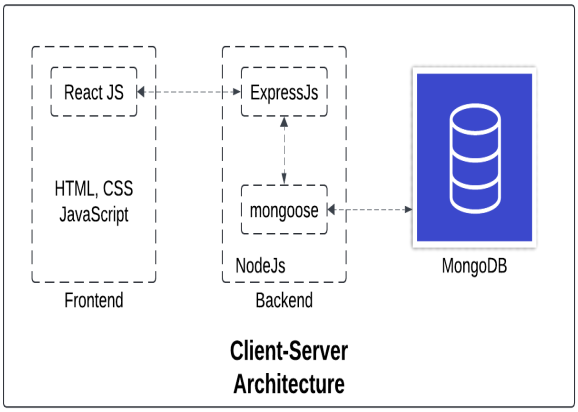


Fig. 2.2: Client-Server Architecture

C. API Controller Development

1) *Separation of Concerns*: After establishing the models, organize the code by separating concerns to enhance maintainability.
2) *Request and Response Analysis*: Analyze the API requests received from the frontend and the corresponding responses required.
3) *Code Migration*: Begin migrating the codebase by converting existing functionalities into a RESTful API format suitable for the MERN stack.

D. Frontend Development

1) *Building the Frontend with React*: Concurrently with backend development, initiate the creation of the frontend using React. This step includes setting up components, state management, and connecting to the backend APIs.

E. Testing

1) *Comprehensive Testing*: Once development is completed, conduct thorough testing of the entire application. This should include unit tests, integration tests, and end-to-end testing to ensure functionality, performance, and reliability.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

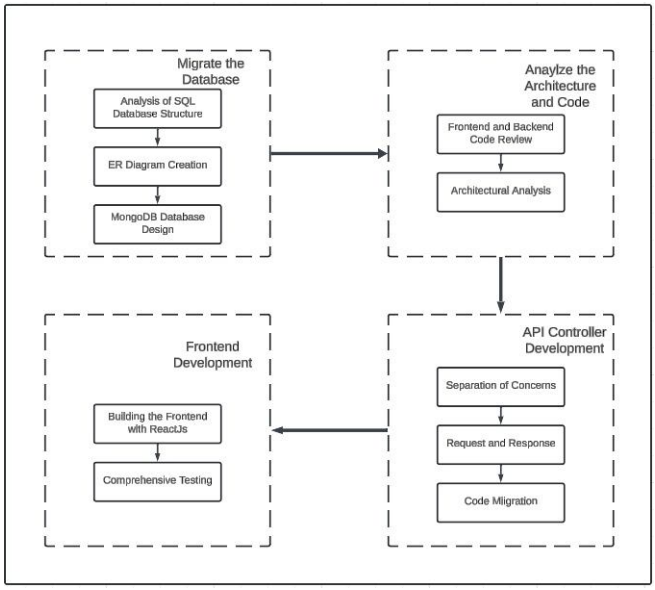


Fig. 2.3 Development Process

III. LITERATURE REVIEW

In the realm of web development and data migration, numerous studies have emerged that highlight both the functionalities and challenges associated with modern technologies. This section reviews relevant literature, focusing on the MERN stack and data migration processes.

A. MERN Stack Overview

Bafna et al. [1] provide a comprehensive review of the MERN stack, emphasizing the functionalities and benefits of its components—MongoDB, Express.js, React.js, and Node.js. The authors articulate how these technologies integrate to form a cohesive web development framework, suitable for various web applications. However, the study identifies notable limitations in the MERN stack, particularly its complexity arising from reliance on third-party libraries. This complexity may hinder developer productivity and complicate large-scale projects. Additionally, the authors note potential security vulnerabilities inherent to the stack, which necessitate extra precautions.

To address these limitations, the authors propose streamlining dependencies through well-maintained libraries and adopting a modular architecture for large projects. They also recommend implementing regular security audits and utilizing security-focused libraries to enhance data protection and authentication processes.

B. Data Migration Process

Sarmah [2] explores the data migration process, focusing on key phases such as data analysis, cleansing, transformation, and reconciliation. The study outlines strategies for effective migration, including the identification of hidden costs, risk management, and rollback strategies essential for successful data transfer. Sarmah employs a structured approach based on the Software Development Life Cycle (SDLC), utilizing techniques such as ETL (Extract, Transform, Load) and data profiling.

However, the paper acknowledges a limitation in its scope, as it may not encompass emerging technologies and modern practices that have evolved since its publication. Additionally, it does not fully address industry-specific requirements or complex migration scenarios involving big data and real-time processing. The author suggests that future research should integrate contemporary tools and methodologies, particularly leveraging AI and machine learning for advanced data cleansing and validation.

C. Challenges in Data Migration

Goyal et al. [3] investigate the intricacies of data migration from legacy systems, positing that data migration should be recognized as a distinct field of study. They analyze key challenges and practices related to successful migration processes. Their descriptive analysis differentiates between data migration and simple data movement, underscoring the importance of integrating data migration within broader application management frameworks.

Despite its contributions, this study lacks coverage of modern data migration technologies that have emerged after its publication. Moreover, it does not address complex, real-time migration scenarios or industry-specific challenges that may have evolved since 2012. The authors recommend updating research to include recent advancements in data migration technologies and methodologies, supported by case studies on complex and real-time migrations to enhance relevance to current industry practices.

IV. FUTURE SCOPE

Converting a PHP project to a MERN (MongoDB, Express.js, React, Node.js) stack can offer several advantages and opportunities for future development. Here are some key points regarding the future scope of such a transition.

A. Scalability

Node.js is designed for high scalability, making it easier to handle increased loads as your application grows. The non-blocking architecture can improve performance in concurrent user scenarios.

B. Single Page Applications (SPAs)

React enables the development of SPAs, which provide a smoother user experience by reducing page reloads. This can lead to better user engagement and retention.

C. Full JavaScript Stack

Using a single language (JavaScript) across the stack (client and server) can simplify development, reduce context switching for developers, and improve productivity.

D. Improved Performance

By leveraging asynchronous processing and efficient database interactions with MongoDB, you can optimize application performance and responsiveness.

E. API-First Approach

Transitioning to MERN encourages an API-first approach, allowing you to decouple the front end and back end. This can facilitate easier integration with other services and microservices.

F. Cloud Compatibility

Node.js applications are well-suited for cloud environments, making it easier to deploy and scale your application using services like AWS, Azure, or Heroku.

G. Enhanced Security

Modern tools and frameworks often come with built-in security features or libraries, making it easier to implement best practices for securing your application.

V. CONCLUSION

This project highlights the critical necessity for web applications to adapt to the dynamic digital landscape characterized by increasing user demands and evolving technological standards.

Our evaluation of the existing PHP-based application has revealed significant challenges, including scalability limitations and performance bottlenecks, which result in slow response times during peak usage. These issues not only impair user experience but also restrict the application's potential for growth. The proposed migration to a MERN stack (MongoDB, Express.js, React.js, Node.js) is a strategic response to these challenges. By leveraging the strengths of the MERN stack, particularly its non-blocking, event-driven architecture via Node.js, we anticipate substantial improvements in both performance and scalability.

This transition aims to create a more efficient, robust, and user-friendly application capable of seamlessly

handling multiple concurrent requests, thereby enhancing overall user satisfaction. This preliminary work establishes a solid foundation for the modernization of the application, setting the stage for future enhancements and ensuring that it can effectively meet the demands of both current and future users in an ever-evolving technological landscape.

The insights gained throughout this project will guide subsequent development efforts, ultimately leading to a more resilient and adaptable application.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

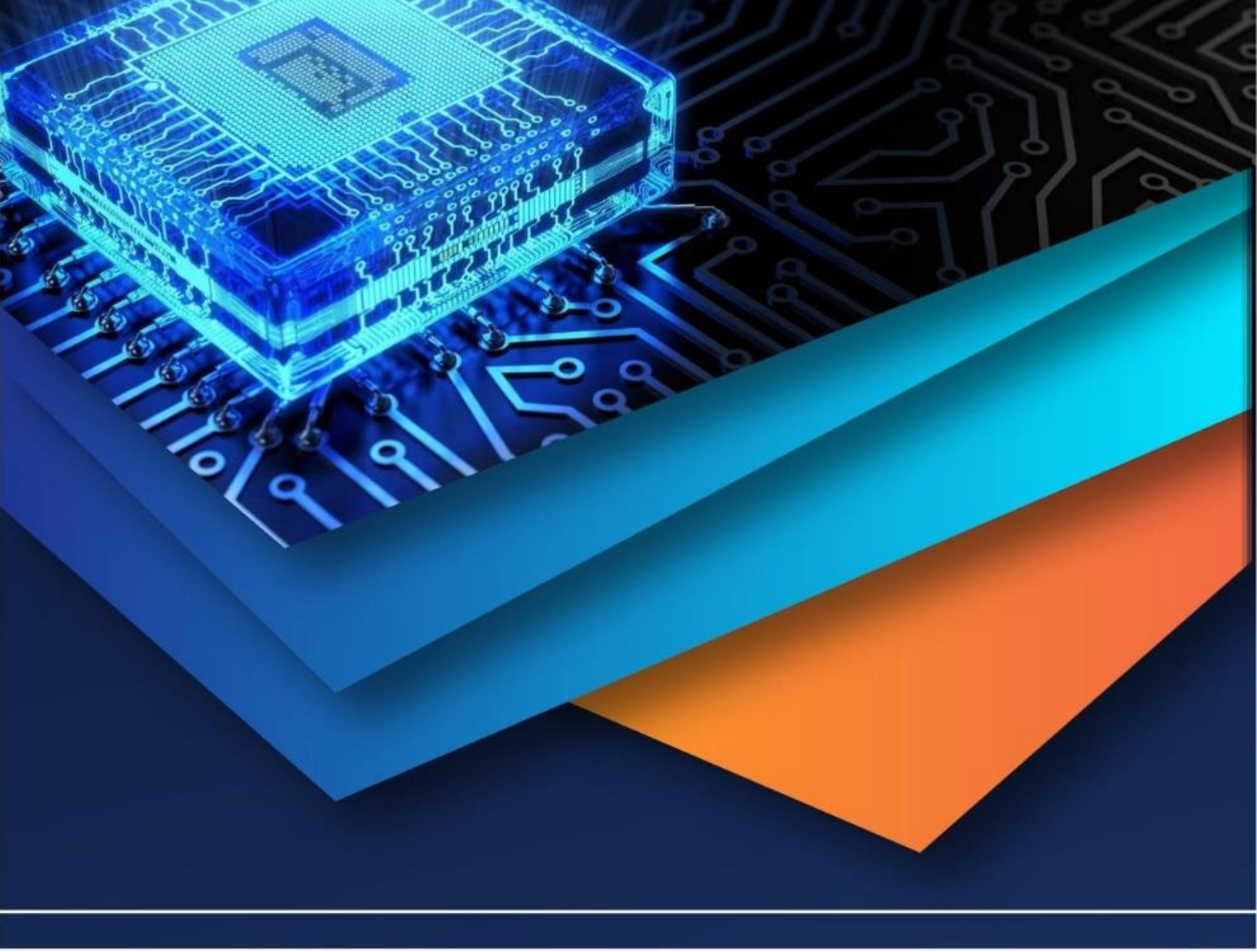
VI. ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

REFERENCES

[1] Bafna, S. A., Dutonde, P., Mamidwar, S., Korvate, M. S., Shirbhare, D. (Year). Study and Usage of MERN Stack for Web Development.
[2] Goyal, V., Jain, A., Gupta, V. K. Data Migration & its Issues.
[3] Jadhav, M. B., Badre, R. R. GUI for Data Migration and Query Conversion.
[4] Kadam, Y., Goplani, A., Mattoo, S., Gupta, S. K., Amrutkar, D., Dhanke, J. Introduction to MERN Stack & Comparison with Previous Technologies.
[5] Goyal, V., Mishra, A. K., Singh, D. Implementation and Comparison of MERN Stack Technology with HTML/CSS, SQL, PHP & MEAN in Web Development.
[6] Singh, A. Data Migration from Relational Database to MongoDB.
[7] Simanta Shekhar Sarmah. Data Migration.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)