International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 15 Issue 3, March-2026

Design and Implementation of a Full MERN Stack–Based Job Portal for Efficient Recruitment Management

Prof. Rucha R. Bhuvad[1], Shivkesh Sachin Shinde[2], Aditya Ramesh Jadhav[3], Mugdha Prabhakar Mule[4], Aishwarya Shivnath Darade[5]

[1,2,3,4,5]Department of (BE) Computer Engineering, Faculty of Navsahyadri Education Society's Group Of Institutions, (SPPU) Pune University, Pune, Maharashtra, India

ABSTRACT

Online recruitment systems have reshaped modern hiring practices by allowing organizations to publish vacancies and receive applications through digital platforms. Despite this progress, many current job portals provide only basic functionalities and often lack comprehensive administrative management, organized resume handling, and integrated real-time communication features.This paper introduces the development of a comprehensive job portal built using the MERN stack to enhance and simplify recruitment operations. The system incorporates secure dual-role authentication for candidates and administrators, real-time job listing management, resume uploading and editing tools, structured application monitoring, and an embedded video interview feature accessible from any location.Developed with MongoDB, Express.js, React.js, and Node.js, the platform ensures scalable architecture, secure data processing, and responsive user interaction. By combining secure authentication with integrated remote interview capabilities, the system reduces reliance on third-party tools and improves workflow efficiency. The proposed solution delivers streamlined recruitment management, enhanced communication, and a user-friendly digital hiring environment.

Keywords - MERN Stack, Job Portal, Recruitment Management, Role-Based Authentication, Resume Management, Video Interview.

INTRODUCTION

The recruitment industry has experienced a significant transformation with the adoption of digital technologies, replacing traditional paper-based hiring practices. Earlier methods relied heavily on manual resume evaluation, face-to-face interviews, and fragmented communication processes, often leading to delays and inefficient coordination between recruiters and candidates. While existing job portals provide basic functionalities such as job posting and application submission, many lack a comprehensive management framework that integrates communication, tracking, and administrative control within a unified system.To address these limitations, this project proposes and develops a comprehensive job portal using the MERN stack architecture. The primary objective of the system is to create a centralized platform that enhances recruitment efficiency through automation and integrated functionality. The application supports secure registration and login for both administrators and candidates, implementing role-based authentication to ensure controlled system access.The platform offers real-time job listings, streamlined application submission, resume uploading capabilities, and editable CV management features. Additionally, it incorporates an integrated video interview module, enabling recruiters and applicants to conduct remote interviews directly within the system. This eliminates the need for third-party communication tools and supports modern remote hiring practices.The technological foundation of the system is built on MongoDB, Express.js, React.js, and Node.js. React.js is utilized to create a responsive and dynamic user interface, ensuring smooth user interaction. Node.js and Express.js manage backend operations, RESTful APIs, and server-side logic. MongoDB serves as the database solution, securely storing user credentials, job information, resumes, and application data. Advanced authentication and security mechanisms are implemented to safeguard sensitive information and maintain data integrity.By integrating all essential recruitment functions into a single web-based environment, the proposed solution improves workflow coordination, reduces operational complexity, and provides a scalable, secure, and user-centric digital recruitment platform.

LITERATURE REVIEW

The rapid advancement of digital technologies has significantly influenced the evolution of recruitment systems. Modern organizations increasingly rely on web-based platforms to streamline hiring processes and improvecandidate engagement. Initial online job portals were primarily designed to display employment opportunities and enable

applicants to submit resumes electronically. Although these platforms enhanced convenience and broadened access, many lacked comprehensive administrative features and structured workflow coordination for recruiters.With the growth of full-stack web development, scalable and interactive recruitment solutions have become more achievable. Technologies such as Node.js and React.js are widely utilized for developing responsive and high-performance web applications. Their flexibility and dynamic rendering capabilities support real-time user interaction and efficient front-end performance. Studies suggest that adopting a unified JavaScript-based technology stack enhances development consistency, simplifies maintenance, and improves overall system reliability.Database management is another critical component in recruitment platforms. NoSQL databases like MongoDB are particularly suitable for managing diverse and large-scale recruitment data, including user profiles, resumes, job postings, and application records. Their flexible schema design allows efficient handling of structured and semi-structured information.Security remains a fundamental requirement in multi-user systems. Modern web applications commonly implement token-based authentication methods, including JSON Web Tokens (JWT), to manage user sessions securely and enforce role-based authorization. Additionally, secure password encryption and hashing mechanisms are employed to protect sensitive user credentials and prevent unauthorized access.The growing trend of remote hiring has further increased the need for integrated communication tools within recruitment systems. While video conferencing solutions are frequently used for interviews, many job portals depend on third-party applications instead of embedding communication features directly into their platforms.Despite technological progress, there is limited research focused on a fully integrated recruitment system that combines secure authentication, resume handling, job monitoring, and built-in video interviewing within a single MERN-based framework. This project addresses that gap by proposing a centralized, scalable, and efficient digital hiring solution.

Problem Statement

Many organizations continue to face challenges in their recruitment operations due to disconnected digital tools, manual processing, and dependence on multiple third-party platforms. Although existing job portals offer fundamental features such as publishing job vacancies and accepting applications, they often fail to provide comprehensive administrative management, organized resume handling, and secure role-based access control. As a result, recruiters are required to navigate between separate systems for job postings, candidate evaluation, communication, and interview scheduling, which increases workload and reduces overall efficiency.The lack of a centralized application monitoring system further complicates the hiring process. When managing a large pool of applicants, tracking candidate status, shortlisting progress, and interview outcomes becomes time-consuming and prone to errors. In addition, inadequate authentication and security measures can put confidential user information at risk, threatening data privacy and system credibility.The rise of remote hiring practices has highlighted additional limitations of traditional job portals. Most platforms rely on external video conferencing tools rather than embedding interview functionality directly into the recruitment system. This fragmented approach disrupts workflow continuity and negatively impacts user experience for both recruiters and applicants.Therefore, there is a strong demand for an integrated, secure, and scalable web-based recruitment solution. Such a system should combine administrator and user authentication, real-time job management, structured resume uploading and editing, organized application tracking, and built-in video interview capabilities within a unified platform. Implementing this approach can significantly improve operational efficiency, strengthen security, and enhance the overall effectiveness of digital recruitment management.

Iv. Objectives

The main aim of this project is to develop a reliable, scalable, and user-centric web-based job portal utilizing the MERN stack framework to enhance the overall efficiency of recruitment management. The system is designed to incorporate distinct login modules for administrators and candidates, supported by role-based authorization to maintain secure and controlled system access.One of the key objectives is to enable recruiters to publish, update, and manage active job openings effectively, while allowing applicants to browse available opportunities and submit applications through a well-structured and transparent workflow. The platform also intends to provide integrated resume management features, including resume uploading, viewing, and in-platform CV editing, thereby removing dependence on external document handling tools.Ensuring data protection is another critical objective. The system implements secure password hashing techniques and token-based authentication mechanisms to safeguard confidential user information and maintain session integrity. Furthermore, the project aims to integrate a real-time video interview module within the application, enabling remote interviews without the need for third-party communication platforms.Overall, the objective is to build a unified recruitment solution that streamlines hiring processes, improves interaction between recruiters and candidates, strengthens data security, and supports modern digital hiring practices through efficient full-stack web development technologies.

V. Proposed system

The proposed recruitment platform is structured using a three-tier architectural model to promote modular design, scalability, and secure interaction among system components. By dividing the system into separate layers for presentation, business logic, and data storage, the architecture simplifies development, testing, and future system upgrades while maintaining organized communication between modules.The first tier, referred to as the Presentation Layer, is implemented using React.js to create a dynamic and responsive user interface. This layer delivers separate dashboards for candidates and administrators, ensuring an intuitive user experience. It includes features such as user

and admin registration and login pages, an administrative control panel, real-time job listing displays, structured job application forms, resume uploading functionality, CV modification tools, and an integrated video interview interface. Communication between the frontend and backend occurs through secure RESTful API endpoints. Role-based access control is applied at the interface level to ensure that each user type can only view and access permitted features.The second tier, known as the Application Layer, is developed using Node.js and Express.js. This layer is responsible for processing client requests, executing core business logic, and managing system workflows. It authenticates users through JSON Web Token (JWT) mechanisms to maintain secure session handling. Password security is ensured through hashing algorithms before credentials are stored in the database. The backend manages operations such as job posting, candidate application processing, resume data handling, and interview coordination. Additionally, it oversees the signaling mechanisms necessary to establish real-time communication for video interviews between recruiters and applicants.The third tier, the Data Layer, utilizes MongoDB as the primary database solution. It organizes data into structured collections, including user profiles, administrator credentials, job postings, resume records, and application tracking details. MongoDB's document-oriented schema provides flexibility in handling evolving and diverse recruitment data.For remote interviews, browser-based video communication technology is embedded directly within the system. The backend supports connection setup, while peer-to-peer data exchange enables smooth audio and video transmission. Overall, this layered architecture ensures secure authentication, systematic data flow, scalability, and efficient recruitment management within a cohesive MERN stack framework.

A. Overall System Architecture

The system architecture of the proposed MERN stack–based job portal is structured to facilitate efficient communication between candidates, administrators, server-side components, and database services while ensuring security and optimized performance. The platform primarily serves two types of users: applicants and administrators. Both interact with the system through a web-based interface that requires authentication before granting access to role-specific features and permissions.After successful login, user actions such as browsing available jobs, submitting applications, uploading resumes, or scheduling interviews are transmitted securely to the backend using RESTful API requests. The server functions as the core processing component of the application. It evaluates incoming requests, applies predefined business logic, enforces authorization rules, and maintains secure user sessions through token-based authentication mechanisms. Administrative operations—including job creation, candidate evaluation, and account management—are also handled within this layer.All essential data, such as user profiles, job postings, resume documents, and application tracking information, is stored in a NoSQL database system. This data layer ensures flexible document storage and efficient retrieval of dynamic recruitment records.To support remote hiring, the architecture incorporates an embedded real-time communication module for conducting video interviews. When an interview session is initiated, the backend coordinates the connection process, enabling direct browser-to-browser interaction without relying on external conferencing platforms.Overall, the architecture provides centralized management, secure data transmission, scalability, and streamlined recruitment operations within a unified and integrated digital framework.

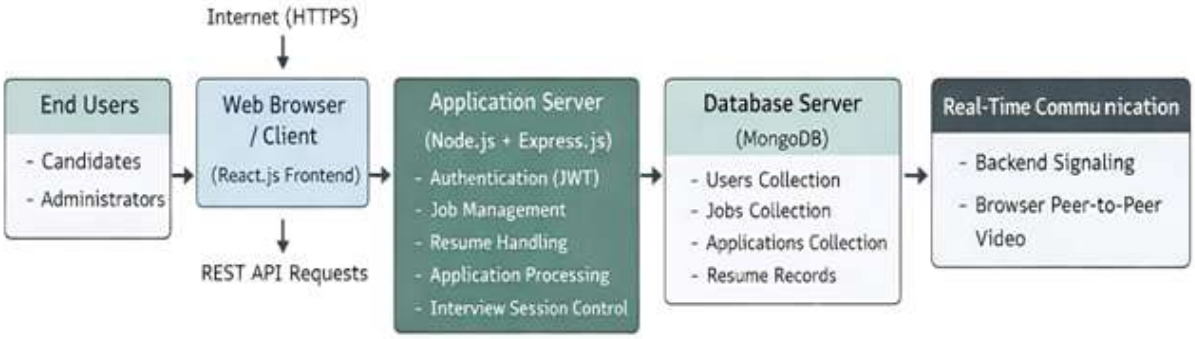

Fig. 1. Overall System Architecture

B. Software Architecture

The proposed MERN stack–based job portal is developed using a modular client–server architecture to ensure scalability, maintainability, and secure recruitment management. The system separates the frontend, backend, and

database layers to promote organized development and efficient performance.The frontend is built with React.js and is responsible for handling user interaction, rendering dashboards, validating forms, and managing client-side sessions. It provides separate interfaces for candidates and administrators. Communication between the client and server occurs through secure RESTful API requests, enabling structured data exchange.The backend is implemented using Node.js and Express.js, which manage core application logic, process user requests, and enforce authentication and authorization policies. Role-based access control ensures that users can only access features relevant to their responsibilities. Middleware components are used for token verification, request validation, and error handling to maintain system stability and security.MongoDB serves as the database layer, storing user profiles, job postings, resume data, and application records in a flexible document-based structure. Additionally, the system integrates a real-time video interview module supported by secure backend signaling.This architecture ensures reliable performance, secure data handling, and efficient recruitment workflow management within a unified digital platform.

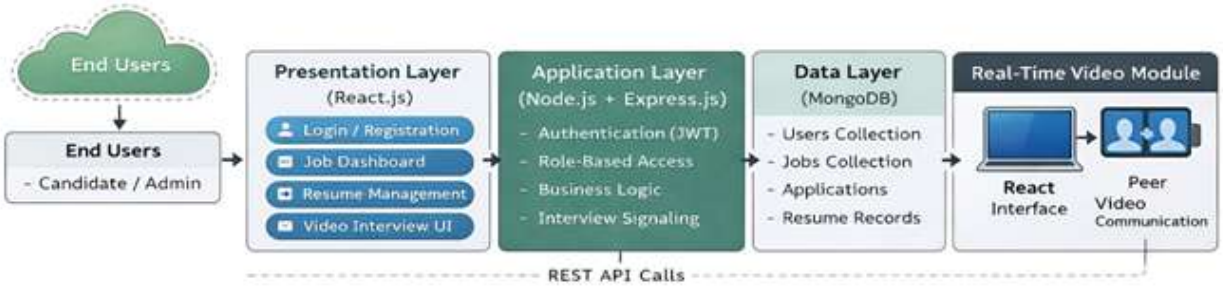

Fig. 2. Software Architecture

VI. SYSTEM METHODOLOGY

The development of the proposed MERN stack–based job portal followed a systematic approach to ensure security, scalability, and efficient recruitment management. The process began with requirement analysis, where the needs of candidates and administrators were clearly identified, including secure authentication, job posting, resume management, application tracking, and video interview functionality. After defining these requirements, a structured database schema was designed to manage collections for users, jobs, resumes, and applications. The backend was then developed using Node.js and Express.js to create RESTful APIs for handling authentication, job operations, and application processing. Token-based authentication and encrypted password storage were implemented to maintain secure session management and protect sensitive data. The frontend was developed using React.js to build responsive dashboards, interactive forms, and dynamic content rendering for both user roles. Resume upload and editing features were integrated to allow candidates to manage their information effectively. The video interview module was implemented using real-time communication coordinated through the backend. Finally, system testing was conducted to verify secure login, accurate data handling, smooth workflow, and stable communication performance.

VII. IMPLEMENTATION DETAILS

The implementation of the proposed job portal was carried out using the MERN stack to ensure seamless integration between frontend, backend, and database components. The backend was developed using Node.js and Express.js, where RESTful APIs were created to manage user authentication, job posting, application processing, and resume handling. Secure login functionality was implemented using JSON Web Tokens (JWT) to manage session authentication, while password encryption was achieved using hashing techniques to protect sensitive user data. Role-based access control was integrated to differentiate permissions between candidates and administrators. The frontend was developed using React.js to create an interactive and responsive user interface. Separate dashboards were designed for users and admins, enabling job browsing, application submission, resume upload, and management operations. Form validation and error handling were implemented at both client and server levels to improve reliability. MongoDB was used as the database to store user credentials, job listings, resume documents, and application records in structured collections. For the resume management feature, file upload functionality was implemented to allow candidates to store and update their documents efficiently. The real-time video interview feature was integrated using browser-based communication mechanisms, where the backend handled session coordination and connection establishment between participants. The entire system was tested to ensure proper API response, secure data storage, smooth user interaction, and stable interview sessions. This implementation ensures scalability, security, and efficient recruitment workflow management.

VIII. WORKING PRINCIPLE

The working principle of the proposed MERN stack–based job portal focuses on secure authentication, structured data handling, and integrated communication within a centralized recruitment platform. The system allows candidates and administrators to register and log in through a secure authentication process. When credentials are submitted, the backend verifies them using encrypted password comparison and generates a secure token to maintain the user session. Based on role-based authorization, users are redirected to their respective dashboards. Administrators can create and manage job postings, review applications, and monitor recruitment activities. These job listings are stored in the database and displayed dynamically to candidates as active opportunities. Candidates can search for jobs, review descriptions, upload resumes, edit profile details, and apply directly through the portal. All application data and resume information are securely stored in organized database collections. Communication between the frontend and backend is handled through structured API requests, ensuring proper validation and controlled data flow. When an interview is scheduled, the system initiates a real-time session that connects the recruiter and candidate through browser-based audio and video communication. Access control policies ensure that users can perform only authorized actions. Continuous data storage and retrieval maintain consistency, reliability, and smooth operation throughout the entire recruitment lifecycle.

IX. ROLE-BASED ACCESS CONTROL FRAMEWORK

The Role-Based Access Control (RBAC) framework implemented in the proposed MERN stack–based job portal ensures structured authorization and secure system operation by assigning permissions according to user roles. The platform defines two primary roles: candidate and administrator, each with clearly differentiated access privileges. During authentication, the backend verifies user credentials and generates a secure token containing role information. This role identifier is used throughout the session to regulate access to protected routes and functionalities. Candidates are permitted to browse live job listings, upload and edit resumes, apply for jobs, and participate in scheduled video interviews. Administrators, on the other hand, are authorized to create and manage job postings, review applications, monitor user activities, and control interview sessions. Middleware components within the backend validate role permissions before processing each request, preventing unauthorized operations and protecting sensitive data. This structured access control model minimizes security risks, reduces accidental data manipulation, and ensures accountability within the system. By separating responsibilities and enforcing controlled access at both frontend and backend levels, the RBAC framework strengthens overall platform security while maintaining operational clarity and system integrity.

X. DATA MANAGEMENT AND STORAGE

Effective data management and storage form the foundation of the proposed MERN stack–based job portal, directly influencing system performance, security, and long-term reliability. The platform manages diverse categories of information, including candidate credentials, administrator accounts, job postings, resume documents, application records, and interview-related metadata. To efficiently handle this structured and dynamic data, MongoDB is utilized as the primary database due to its flexible, document-oriented architecture. Unlike traditional relational databases that rely on rigid table schemas, MongoDB stores data in JSON-like documents, allowing adaptive schema design and easier modification as system requirements evolve. Separate collections are maintained for users, administrators, job listings, applications, and resumes to ensure organized storage and optimized querying. Sensitive authentication details such as passwords are securely protected using hashing algorithms before being stored, preventing unauthorized access and enhancing data confidentiality. Job postings include structured attributes such as title, description, qualifications, location, and status, enabling efficient filtering and retrieval operations. Resume information, including uploaded files and profile details, is stored systematically to maintain consistency and eliminate redundancy. Application records create a structured linkage between candidates and job postings, enabling administrators to monitor hiring progress effectively. Backend validation mechanisms and controlled API access ensure data accuracy and integrity, while indexing techniques improve retrieval speed for job searches and candidate filtering. Backup strategies and role-based access permissions further strengthen reliability. Overall, the data management framework ensures secure information handling, efficient data retrieval, scalability, and sustainable maintenance of the recruitment platform.

RESULTS AND ANALYSIS

The developed MERN stack–based job portal was evaluated to assess its functionality, performance, security, and overall recruitment efficiency. Functional testing was carried out by creating multiple candidate and administrator accounts to verify secure role-based authentication and controlled system access. The login and registration modules successfully generated token-based sessions, restricting protected routes to authorized users only. Job posting operations performed through the administrator dashboard were instantly reflected on the candidate interface, confirming accurate real-time data retrieval from the database. The job application workflow was tested by submitting multiple applications, and the system correctly stored and linked candidate information with respective job listings. Resume upload and editing features were validated by updating profile details and confirming consistent database updates without redundancy or data loss. The video interview feature was tested under varying network conditions, where stable browser-based audio and video communication was established with minimal delay under normal

connectivity. Performance analysis showed that backend APIs handled concurrent requests efficiently, and MongoDB provided acceptable response times due to organized collections and indexing. Security testing confirmed that encrypted passwords and token verification mechanisms prevented unauthorized access. Error-handling processes effectively managed invalid inputs and maintained system stability. Overall, the system demonstrated reliable performance, secure data handling, and improved recruitment workflow efficiency within a unified platform.

CONCLUSION

The proposed MERN stack–based job portal successfully demonstrates how modern web technologies can be utilized to build a secure, scalable, and efficient recruitment management platform. By integrating user and administrator authentication, live job posting, structured application tracking, resume upload and editing functionality, and real-time video interview capability within a single system, the platform simplifies the complete hiring lifecycle. The use of MongoDB, Express.js, React.js, and Node.js ensures seamless communication between frontend and backend components while maintaining flexibility and performance. Secure authentication mechanisms and role-based access control enhance data protection and system reliability. The inclusion of a built-in video interview module further strengthens the platform by supporting remote hiring without relying on third-party applications. Overall, the system provides a centralized and user-friendly solution that improves recruitment workflow efficiency, reduces operational complexity, and aligns with modern digital hiring practices. The project highlights the effectiveness of full-stack development in solving real-world recruitment challenges.

FUTURE SCOPE

The proposed job portal system can be further enhanced by integrating advanced intelligent and scalable features to improve recruitment efficiency. In the future, artificial intelligence–based resume screening can be implemented to automatically shortlist candidates based on skills, experience, and job requirements. Machine learning algorithms may assist in recommending suitable jobs to candidates and identifying the best applicants for recruiters. The system can also include an interview recording and playback feature for evaluation and documentation purposes. Automated email and notification services can be added to update users about application status and interview schedules. Cloud deployment and microservices architecture can improve scalability and performance for handling large numbers of users. Additionally, a mobile application version can be developed to increase accessibility. Data analytics dashboards for administrators may provide insights into hiring trends and platform usage, further enhancing decision-making and recruitment strategy optimization.

REFERENCES

1. R. T. Fielding and R. N. Taylor, “Principled design of the modern web architecture,” ACM Trans. Internet Technol., vol. 2, no. 2, pp. 115–150, 2002.
2. M. Tilkov and S. Vinoski, “Node.js: Using JavaScript to build high-performance network programs,” IEEE Internet Comput., vol. 14, no. 6, pp. 80–83, 2010.
3. S. Tilkov and S. Vinoski, “Server-side JavaScript with Node.js,” IEEE Comput., vol. 43, no. 10, pp. 95–98, 2010.
4. D. Flanagan, JavaScript: The Definitive Guide, 6th ed. Sebastopol, CA, USA: O’Reilly Media, 2011.
5. MongoDB Inc., “MongoDB: The application data platform,” MongoDB Documentation, 2023.
6. ReactJS Documentation, “A JavaScript library for building user interfaces,” Meta Platforms Inc., 2023.
7. Barth, “HTTP state management mechanism,” IETF RFC 6265, 2011.
8. M. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT),” IETF RFC 7519, 2015.
9. Jennings et al., “WebRTC 1.0: Real-time communication between browsers,” W3C Recommendation, 2021.
10. R. Buyya et al., “Cloud computing and emerging IT platforms: Vision and reality,” Future Gener. Comput. Syst., vol. 25, no. 6, pp. 599–616, 2009.
11. L. Richardson and S. Ruby, RESTful Web Services. Sebastopol, CA, USA: O’Reilly Media, 2007.
12. E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Boston, MA, USA: Addison-Wesley, 1994.
13. Sommerville, Software Engineering, 10th ed. Boston, MA, USA: Pearson, 2016.
14. K. Hwang, J. Dongarra, and G. C. Fox, Distributed and Cloud Computing: From Parallel Processing to the Internet of Things. Burlington, MA, USA: Morgan Kaufmann, 2012.
15. P. Mell and T. Grance, “The NIST definition of cloud computing,” NIST Special Publication 800-145, 2011.
16. Silberschatz, H. F. Korth, and S. Sudarshan, Database System Concepts, 6th ed. New York, NY, USA: McGraw-Hill, 2011.
17. M. Fowler, Patterns of Enterprise Application Architecture. Boston, MA, USA: Addison-Wesley, 2003.
18. Crockford, JavaScript: The Good Parts. Sebastopol, CA, USA: O’Reilly Media, 2008.
19. OWASP Foundation, “OWASP Top 10: The Ten Most Critical Web Application Security Risks,” 2021.
20. R. Sandhu and P. Samarati, “Access control: Principle and practice,” IEEE Commun. Mag., vol. 32, no. 9, pp. 40–48, 1994.

21. G. Coulouris, J. Dollimore, and T. Kindberg, Distributed Systems: Concepts and Design, 5th ed. Boston, MA, USA: Addison-Wesley, 2011.
22. T. Berners-Lee, R. Fielding, and H. Frystyk, "Hypertext Transfer Protocol – HTTP/1.1," IETF RFC 2616, 1999.
23. S. Newman, Building Microservices. Sebastopol, CA, USA: O'Reilly Media, 2015.
24. Banks and R. Gupta, "MQTT version 3.1.1," OASIS Standard, 2014.
25. W3C, "Web Applications Architecture," World Wide Web Consortium (W3C), 2020.
26. M. Fowler, Refactoring: Improving the Design of Existing Code, 2nd ed. Boston, MA, USA: Addison-Wesley, 2018.
27. K. Beck, Test-Driven Development: By Example. Boston, MA, USA: Addison-Wesley, 2003.
28. Grigorik, High Performance Browser Networking. Sebastopol, CA, USA: O'Reilly Media, 2013.
29. Evans, Domain-Driven Design. Boston, MA, USA: Addison-Wesley, 2004.
30. N. Zakas, Professional JavaScript for Web Developers, 4th ed. Hoboken, NJ, USA: Wiley, 2020.
31. S. Chacon and B. Straub, Pro Git, 2nd ed. New York, NY, USA: Apress, 2014.
32. P. Deitel and H. Deitel, Internet & World Wide Web: How to Program, 5th ed. Boston, MA, USA: Pearson, 2012.
33. R. Pressman and B. Maxim, Software Engineering: A Practitioner's Approach, 8th ed. New York, NY, USA: McGraw-Hill, 2014.
34. T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design. Upper Saddle River, NJ, USA: Prentice Hall, 2005.
35. B. Schneier, Applied Cryptography, 2nd ed. New York, NY, USA: Wiley, 1996.
36. C. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, 2006.
37. Gourley and B. Totty, HTTP: The Definitive Guide. Sebastopol, CA, USA: O'Reilly Media, 2002.
38. M. Kleppmann, Designing Data-Intensive Applications. Sebastopol, CA, USA: O'Reilly Media, 2017.
39. Hunt and D. Thomas, The Pragmatic Programmer, 2nd ed. Boston, MA, USA: Addison-Wesley, 2019.
40. Reese, Database Programming with JDBC and Java. Sebastopol, CA, USA: O'Reilly Media, 2000.