IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

Customer Relationship Management (CRM)

Prof. Snehal Rewatkar, Mr. Yash Mendhe, Mr. Pranay Ambule, Mr. Karan Jaiswal
Dept. Computer Science & Engineering
Tulsiramji Gaikwad Patil College of Engineering and Technology, Nagpur, Maharashtra, India.
snehal.cse@gmail.com mendheyash455@gmail.com
Pranayambule20@gmail.com, Ksjaiswalngp@gmail.com

Abstract: *Customer Relationship Management (CRM) systems are essential for organizations to handle customer data, track interactions, and deliver better service through a unified and organized platform. In many companies, outdated manual processes and scattered ways of keeping records cause problems like inconsistent data, limited access, slow responses, and lower efficiency. To fix these issues, this paper introduces the design and development of a web-based CRM system built using the MERN stack—MongoDB, Express.js, React, and Node.js. The system is made to be scalable, responsive, secure, and easy to maintain, allowing businesses to manage customer activities in real time. It features important tools like user login, managing customer details, tracking potential customers, recording interactions, and using dashboards for monitoring, which help organizations manage and retrieve data more effectively. By using a full-stack JavaScript approach, the system makes development easier, improves communication between the front and back ends, and allows flexible data management through a document-based database. The system was tested for its usability, performance, functionality, and reliability, and the results show it works well as a real-world CRM solution. Overall, the study shows that a MERN-based CRM can boost business efficiency, improve how customer relationships are managed, and support smarter decision-making in today's web-based environments..*

Keywords: Customer Relationship Management, CRM, MERN Stack, MongoDB, Express.js, React, Node.js, Web Development, Customer Data Management, Lead Tracking, Full-Stack JavaScript

I. INTRODUCTION

Customer Relationship Management (CRM) is a strategic approach that uses technologies and business practices to manage interactions with current and potential customers in an organized and effective way.

It plays a key role in modern business operations because it helps organizations gather, store, analyze, and use customer data to improve service quality and build long-term relationships. A CRM system goes beyond just keeping contact information; it also helps track communication, monitor sales, provide customer support, and support decision-making with accurate and up-to-date data.

In today's competitive market, businesses need to respond to customer needs quickly and maintain consistent communication throughout the customer journey.

Using old methods like spreadsheets, paper records, or separate software tools often causes problems such as repeated data, missed follow-ups, and difficulty finding information when needed. These issues can lower customer satisfaction and reduce productivity. For these reasons, more organizations are turning to digital CRM platforms that offer a central place to manage all customer-related activities.

A well-designed CRM system lets users manage customer profiles, track interactions like calls, emails, and meetings, and follow leads or opportunities from the first contact to final conversion.

This improves teamwork between sales, marketing, and support teams and gives a full picture of customer behavior and needs. Having all customer data in one place helps organizations make better decisions, personalize communication, and respond to service requests more effectively.

The development of web technologies has made CRM systems more accessible, flexible, and scalable.

DOI: 10.48175/IJARSCT-33675

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

Web-based solutions let users access the system from any device and location, making them suitable for businesses of all sizes. In this project, the CRM system is built using the MERN stack, which offers a modern full-stack JavaScript environment for creating interactive user interfaces, efficient backend services, and flexible databases. This approach supports real-time data handling, modular development, and easier maintenance.

Overall, the goal of this CRM project is to create a central platform that makes customer data management easier and improves business operations.

By replacing manual and scattered processes with a structured web-based solution, the system helps organizations increase efficiency, reduce mistakes, and build stronger customer relationships.

II. LITERATURE SURVEY

Customer Relationship Management (CRM) systems have changed a lot over the years. They started as simple tools for storing contact information, but now they are powerful platforms that help with sales, customer support, marketing, and analyzing data. In the beginning, these systems were often used on desktop computers or were built around strict databases, which worked well for keeping records organized but weren't very flexible or easy to use. They also didn't support real-time teamwork very well.

At first, the main focus of CRM was on managing customer contacts and tracking sales.

By the late 1990s and early 2000s, more businesses started using these systems because they helped keep customer information in one place and improved how work was done. However, a lot of these systems used old technology and were hard to update or connect with newer software. During this time, people talked a lot about having a clear view of customers, but the tools available weren't very mobile-friendly, didn't automate things very well, and had a rough user experience.

As the 2000s and 2010s progressed, web technologies made things better.

CRM systems started moving to web-based platforms built with tools like HTML, CSS, JavaScript, PHP, Java, .NET, MySQL, and PostgreSQL. These allowed users to access CRM from anywhere and automate a lot of basic tasks. Researchers at the time focused on how web access improved workflow and how data could be reported more effectively. Still, many web-based systems struggled with how well they could scale, how easy they were to modify, and how well they could handle data that wasn't strictly organized.

The rise of modern JavaScript frameworks and NoSQL databases brought a new direction for CRM development.

One popular choice is the MERN stack, which uses JavaScript for both the front and back ends of applications. React helps create reusable UI parts and responsive designs that are easier to manage. Node.js and Express.js allow for quick server-side processing and API creation, which makes it easier for different parts of the app to talk to each other. MongoDB, with its flexible structure, works well for CRM because it can store a variety of data like customer profiles, leads, interactions, and roles without needing a strict database layout. In recent years, developers and researchers have favored MERN-based systems because they offer scalability, flexibility, and real-time features, which are all important for today's CRM needs.

III. METHODOLOGY OF THE SYSTEM –

The CRM system we're building is made using a full-stack web approach based on the MERN stack.

This combines MongoDB, Express.js, React, and Node.js into one unified framework. We chose this method because it helps create a web application that's easy to maintain, scalable, and interactive, which is perfect for managing customer activities efficiently. The development process is broken down into several phases, starting with understanding what the system needs to do and then planning the design, setting up the database, building the system, testing it, and finally preparing it for launch.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

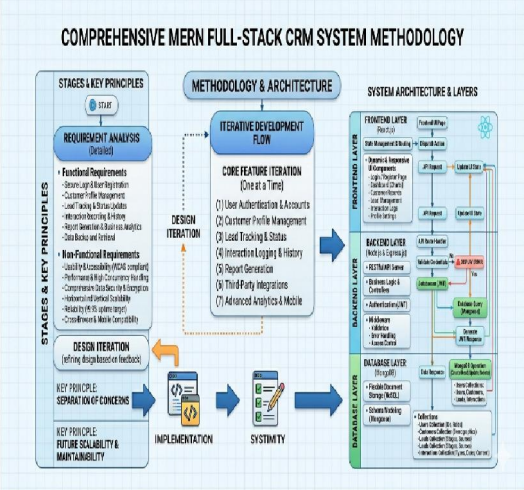

Fig-1 : CRM System Methodology

The first step is requirement analysis.

Here, we figure out exactly what the CRM system needs to do, both in terms of features and how well it should perform. The features include things like user registration, secure login, managing customer profiles, tracking leads, recording interactions, and generating reports. These features are what the system needs to handle daily CRM tasks. On the other hand, non-functional requirements are about how well the system works overall. These include usability, speed, security, scalability, dependability, and compatibility across different browsers. These make sure the app is not only functional but also reliable and practical for everyday use.

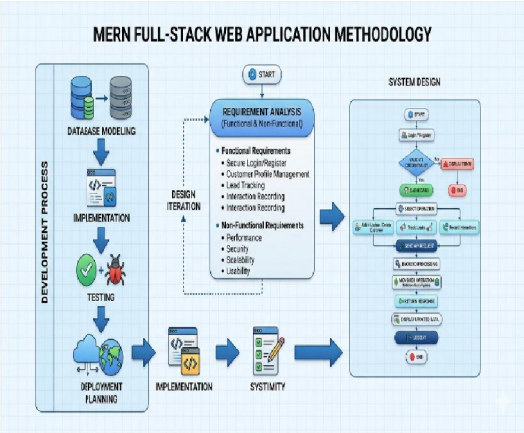

Fig-2 : proposed CRM System

Once the requirements are clear, the system is divided into three main parts: frontend, backend, and database.

The frontend is built with React, which is great for creating a dynamic and responsive user interface. React allows developers to reuse components, which makes the interface faster to build, update, and maintain. Pages like the login screen, dashboard, customer records, lead management, and interaction logs are built as separate components, making it easier for users to move around the system. Using state management and routing also helps the app respond quickly to user actions, improving the overall experience.

The backend is developed using Node.js and Express.js.

This part of the system deals with the logic of the application, handles user authentication, validates data, and connects the frontend with the database. Express.js is used to create RESTful APIs that let the frontend perform actions like

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 11, April 2026 Impact Factor: 8.2

creating, reading, updating, and deleting data. Node.js provides a fast runtime environment that helps manage multiple requests at once. Middleware is used for tasks like checking input, managing errors, and controlling access, which makes the system more secure and reliable.

The database is built with MongoDB, a type of NoSQL database that's very suitable for CRM systems.

MongoDB stores data in JSON-like documents, which is flexible enough to handle various kinds of customer information, lead details, and interaction records. Collections like users, customers, leads, and interactions help organize the data efficiently. This structure allows easy access, changes, and expansion as the system grows. The flexibility of MongoDB also helps the system scale better than traditional relational databases.

One of the main principles in this approach is separating the system into distinct parts so each layer has a specific job.

The React frontend sends user requests to the backend, the backend processes the request and interacts with MongoDB, and the database returns the needed data back to the backend, which then sends the response to the frontend. This modular structure makes the system easier to develop, debug, maintain, and improve. It also means changes in one part of the system won't affect other parts unnecessarily.

The methodology also uses an iterative approach, where the system is built in stages, and each feature is developed, tested, and added step by step.

For example, authentication is built before customer management, and customer management is completed before lead tracking and interaction logging. This method reduces errors, improves code quality, and makes testing more effective. It also ensures the final product meets both technical goals and user expectations.

A. Workflow

The user starts by opening the CRM application and either creates a new account or signs in using their correct login information.

The authentication system checks the details provided and allows access only to users who are authorized.

Once the login is successful, the system takes the user to the main dashboard.

On the dashboard, the user sees important information like customer details, the status of leads, and recent activities.

From the dashboard, the user can perform various CRM tasks, such as adding, modifying, removing, or viewing customer information.

They can also keep track of customer interactions and update the status of leads through the same screen.

When the user fills out a form, the React-based frontend sends an HTTP request to the Express.js backend.

The backend system checks the request and carries out the necessary steps for the business logic.

It then connects with MongoDB to either save, update, retrieve, or remove the relevant data.

The database sends the processed information back to the backend.

The backend then shares this data with the frontend.

The frontend refreshes the screen automatically without completely reloading the entire page.

This process ensures that data flows smoothly, responses are quick, and the user experience is real-time.

B. Algorithm (Pseudocode)

Algorithm:

The CRM system uses a step-by-step process that starts with checking if a user is logged in and then moves on to managing customers and potential leads. This system is built to keep things secure, organize data properly, and make sure the different parts of the app—like the user interface, the server, and the database—work well together.

Start.

Show the login or sign-up page.

Get the user's login details.

Check if the details are correct.

If they are correct, let the user in.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

If not, show an error and stop.
Once the user is logged in, show the main dashboard.
Let the user do different things in the CRM, like:
Add a new customer.
Update customer information.
Delete a customer.
View customer details.
Add or keep track of leads.
Record any interactions.
Send the request from the React front end to the Node.js/Express back end.
Check the request on the back end.
Connect to MongoDB.
Do what the request asks the database to do.
Send the result back to the front end.
Show the user the updated information.
Keep doing this until the user logs out.
End.

C. Flowchart

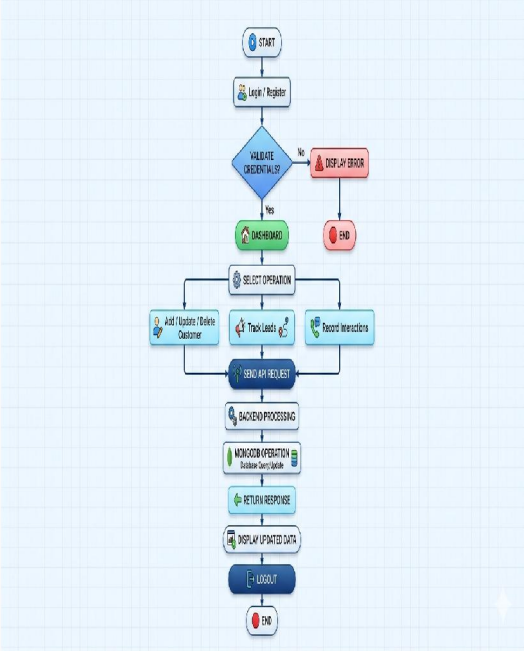

Fig-3 : Flowchart

IV. IMPLEMENTATION

The Customer Relationship Management (CRM) application was built as a modular, full-stack web system using the MERN stack.

Each part of the system had a specific job, which made the code clearer, easier to manage, and more efficient. The whole setup was divided into three main sections: the frontend for users to interact with the system, the backend for

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 11, April 2026 Impact Factor: 8.2

handling all the business rules and connecting to the database, and the database for keeping all the data safe and long-lasting. This way of organizing the system made it easier to keep things in order, grow the application, and add new features later on.

The frontend was made using React, which allowed for a user-friendly and responsive interface.

Components like login and registration forms, customer tables, dashboards, navigation menus, and input dialogs were created so that the user experience was consistent throughout the app. React Router was used to move between different pages such as login, registration, dashboard, customer management, lead tracking, and interaction history. The system used state management to keep everything up to date, so changes in customer or lead details appeared instantly without needing to reload the whole page. This made the interface quick, clean, and easy to use.

The backend was built with Node.js and Express.js, which helped manage the app's logic, handle requests from the frontend, and communicate with the database.

RESTful APIs were set up for key CRM actions like user login, creating customer records, updating details, removing entries, getting dashboard info, and tracking interactions. Each API route was designed for one task and returned the right response to the frontend. Extra tools such as middleware were added to check if inputs were correct, ensure security, and handle any errors before the data ever reached the database. This made the system more dependable and secure.

MongoDB was chosen as the database because of its flexible document-based structure, which fits well with CRM data that can vary in format.

Separate groups were used to store users, customers, leads, and interaction records. User data included login info and roles, customer records had contact and profile details, lead records tracked possible business chances, and interaction records kept a history of communication like calls, emails, or meetings. The JSON-like format of MongoDB let the application store related data in a more natural and adaptable way compared to strict relational databases. This made it easier to change the system later and support growth as data volumes increased.

Keeping users safe was a priority, so the system only allowed authorized users to access important features.

During sign-up, user details were checked before being saved. When logging in, the system matched entered credentials with stored user data, and only valid users were given access to the dashboard. This kept sensitive customer data protected and stopped unwanted use. More access control could be added to limit certain actions based on user roles if needed.

The communication between the frontend, backend, and database was done through APIs.

When a user filled out a form in the React frontend, the system sent an HTTP request to the backend. The backend checked the request, processed any needed actions, and used MongoDB to save or get the data. After completing the database task, the backend sent the response back to the frontend, which then updated the page without needing a full reload. This setup made the system feel like a real-time web app and improved the user experience.

Overall, this implementation turned the system design into a working CRM platform.

Using React, Node.js, Express.js, and MongoDB gave a good balance of performance, flexibility, and ease of maintenance. The modular approach also made it simpler to add features like analytics, notifications, reports, and integrations with other tools in the future.

V. RESULTS AND ANALYSIS

The CRM system that was put in place works well as a central place to manage all customer-related information in a clear and efficient way. By bringing together customer data, lead details, and communication history into one web-based tool, the system cuts down on the need for handwritten records and data stored in different places. Having everything in one place makes it easier to access, keeps the information consistent, and ensures it's reliable, which is key for managing customer relationships effectively.

The screens in the system, like the login, dashboard, customer management, lead management, and interaction tracking pages, work as planned and give a clear picture of what the CRM can do.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

The login screen keeps things secure so only authorized users can access it. The dashboard shows important data in an easy-to-read format. The customer management page lets users add, edit, delete, and view customer details. The lead management tool helps track potential business leads, and the interaction tracking feature keeps a record of all communications, making it easier to follow up on customer activities over time.

From the user's point of view, the system has an easy-to-use layout that makes it simple to navigate and complete tasks quickly.

The frontend, built using React, makes the pages load smoothly and update in real time, which improves the overall experience. Users can do things efficiently without long waits, and the system's modular design makes it easy to understand even for those who are new to it. This is really important because a CRM only works well if people can use it easily and consistently.

When it comes to performance, the MERN stack—made up of React, Node.js, Express.js, and MongoDB—supports fast data handling and reliable performance.

React helps the user interface respond quickly, while Node.js and Express.js handle requests on the server side efficiently. MongoDB helps store and retrieve customer data quickly. Together, these technologies make the system run smoothly, even as more data is added. This makes the system suitable for real business environments where speed and reliability are important.

The system underwent several types of testing, including unit, integration, system, and validation testing, to ensure every part works correctly both on its own and as part of the whole system.

Unit testing confirmed that individual components work as they should, while integration testing made sure the frontend, backend, and database work together properly. System testing checked the whole application, and validation testing confirmed that it meets the required features. The results show that the system is stable, works correctly, and is ready to be used in real situations.

The discussion also shows that the system meets its main goals.

It makes customer data easier to organize, simplifies tracking leads, improves communication records, and reduces the need for manual work. These improvements help businesses work more efficiently and make better decisions using accurate and up-to-date information. Plus, using the MERN stack provides a flexible base for future growth, like adding analytics, reports, mobile support, or connecting with other services.

Overall, the results show that the CRM system is effective, can grow with business needs, and is practical for modern web-based customer management.

The implementation shows that a full-stack JavaScript approach can support a responsive and easy-to-maintain CRM solution. The discussion also suggests that the project could be developed into a more advanced enterprise system with additional features in the future.

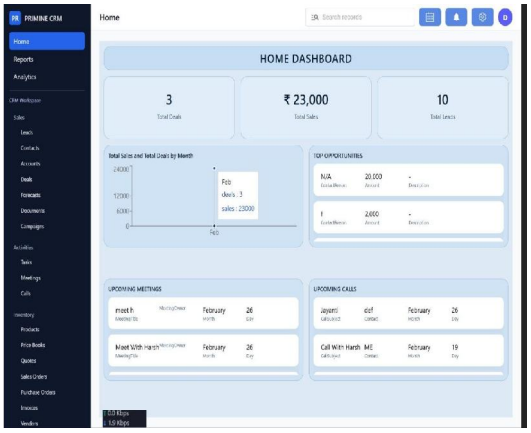

Fig-4 : Home Dashboard

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

VI. FUTURE SCOPE

The current CRM system is built with the key features needed for managing customers effectively. These features include secure login, keeping track of customer information, following up on leads, and recording customer interactions. While these basics help with everyday business tasks, the system can be improved in several ways to make it smarter, easier to use, and better for big companies. Future work should focus on making it more automated, helpful for decision-making, easier to use on mobile devices, better connected with other systems, and more secure.

One major improvement is using AI for predictive analytics.

By looking at how customers behave, what they buy, how they communicate, and how leads move through the sales process, the CRM can predict what customers might need, which leads are most likely to become sales, and when the best time is to follow up. This helps businesses shift from reacting to customer needs to being proactive. Predictive models can also help with grouping customers, predicting who might leave, and forecasting sales, giving companies a better edge in the market.

Another important change is adding support for mobile apps.

In many workplaces, people need to access CRM information while traveling, in meetings, or working away from the office. A mobile version or a mobile-friendly app would let sales teams and managers update customer details, check reports, and reply to customer activity using their phones or tablets. This would make the system easier to access, speed up responses, and boost productivity.

The system can also be expanded by connecting with communication tools like email and SMS.

These connections would help send automated messages, remind people about appointments, update leads, and notify customers directly from the CRM. For example, when a new lead comes in, the CRM could automatically send a welcome email or SMS. This would cut down on the need for manual communication and keep customers in touch more consistently.

Adding a payment gateway is another useful step, especially for businesses that handle billing, subscriptions, or transactions.

This would make the CRM more helpful for service providers, online stores, and organizations managing regular payments. Linking customer records with payment history gives a fuller picture of how customers interact with the company over time.

A better analytics dashboard is another strong area to improve.

The system can be expanded to show charts, trends, sales reports, summaries of customer activities, and performance metrics. These dashboards would help managers understand how the business is doing quickly and spot areas that need attention. Real-time reports can also help with planning and quicker decision-making.

Security is another area that can be improved.

Future versions might include stronger login methods, better password protection, secure data storage, and more detailed access controls. These steps would lower the chance of unauthorized access and help keep customer data safe. For bigger companies, adding logs to track who accesses the system and what they do can help with accountability and security monitoring.

The CRM can also become a more collaborative tool by connecting with other business apps like calendars, cloud storage, marketing platforms, and support systems.

This would let companies link CRM tasks with other tools they use, creating a smoother and more connected way to manage customer relationships. These integrations are especially helpful for big teams that use multiple digital tools.

Overall, the future of this CRM project has a lot of potential.

By making it smarter, more mobile-friendly, better connected, more analytical, and more secure, the system can grow from a simple customer management tool into a full-scale enterprise solution. These changes would make the CRM more powerful, flexible, and better suited to today's business needs.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

VII. CONCLUSIONS

This paper introduces the design and creation of a web-based Customer Relationship Management (CRM) system using the MERN stack. It aims to solve the problems of outdated and disorganized customer management methods. The system offers a single place where customer data, lead details, and communication history can be stored securely and neatly. By replacing old, scattered records with a well-organized digital system, the application helps ensure data is consistent, easily accessible, and well-managed.

The system shows how modern web technologies can be used to create a powerful and scalable business solution.

The React frontend gives users a clean and easy-to-use interface. Node.js and Express.js handle the server-side logic and communication with other parts of the app. MongoDB stores data in a flexible and dependable way, making it suitable for changing CRM needs. These technologies work together to build a full-stack solution that is both functional and simple to maintain.

The project successfully achieved its main goals by offering secure login, customer management, lead tracking, and record-keeping features.

These tools help make workflows more efficient, reduce the need for manual tasks, and give better insight into customer activities. As a result, the system supports more effective customer relationship management and helps businesses make smarter decisions.

This project also shows that the MERN stack is a great choice for building modern web applications for businesses.

Its use of a single programming language simplifies development and upkeep, while its structure makes it easy to expand in the future. The system is therefore a good fit for companies that want a flexible, fast, and scalable CRM solution.

In short, the developed CRM system is a useful and effective tool for managing customer relationships through the web.

It has the potential to grow with features like AI-driven analytics, detailed reports, mobile access, and connections with other platforms. With these upgrades, the system can become a smarter and more complete platform for businesses in the real world.

VIII. ACKNOWLEDGEMENT

I want to sincerely thank my project guide for their valuable guidance, constant encouragement, and helpful suggestions throughout the development of this CRM project. Their support helped me steer the project in the right direction and enhanced both the technical aspects and the overall presentation of the work.

I am also deeply grateful to the faculty members and the institution for providing the academic environment, resources, and encouragement needed to complete this project successfully.

Their support and collaboration were crucial in helping me understand the practical sides of web development and research writing.

I would like to recognize the developers and open-source communities behind React, Node.js, Express.js, and MongoDB.

Their powerful technologies made this project possible. These tools provided a strong foundation for creating a modern, scalable, and efficient web-based CRM system.

I also want to thank everyone who gave feedback, motivation, and technical help during the preparation of this research paper.

Their suggestions and support were very helpful in improving the quality of the project and finishing it with confidence.

Finally, I am thankful to my family, friends, and well-wishers for their continuous support and patience throughout this journey.

Their encouragement has been a great source of motivation in completing this work successfully.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 11, April 2026
Impact Factor: 8.2

REFERENCES

[1]. Anonymous. "BUILDING CRM USING MERN STACK." IJSREM, 2024. Link: https://ijsrem.com/download/building-crm-using-mern-stack/
[2]. Hemant Singh Dev. "CRM Web App (MERN Stack + Figma Design)." GitHub, 2025. Link: https://github.com/hemantsinghdev/crm-system
[3]. Anonymous. "Customer Relationship Management (CRM) Using Python Full Stack (Django and MySQL)." IJRASET. Link: https://www.ijraset.com/best-journal/customer-relationship-management-crm-using-python-full-stack-django-and-mysql
[4]. Anonymous. "Building CRM Using Mern Stack." Scribd, 2025. Link: https://www.scribd.com/document/788646385/BUILDING-CRM-USING-MERN-STACK
[5]. Anonymous. "Build MERN Stack CRM Ticket System #0 - Series Intro." YouTube, 2020. Link: https://www.youtube.com/watch?v=7UjdDjgAD2E
[6]. Anonymous. "Usefulness of business simulations using CRM in the opinion of students of an economic and technical university." ScienceDirect, 2025. Link: https://www.sciencedirect.com/science/article/pii/S187705092503039X
[7]. Harish Vinayagamoorthy. "CRM Project with MERN Stack." GitHub, 2023. Link: https://github.com/HarishVinayagamoorthy/CRM-MERN
[8]. MA Fadilah. "Development Web-Based Customer Relationship Management (CRM) System with Dynamic Workflow for Enhancing the Sales Process." IJCS, 2025. Link: https://www.ijcs.net/ijcs/index.php/ijcs/article/view/4840/1153
[9]. Anonymous. "Customer Relationship Management (CRM) in Small Organizations." DiVA Portal, 2025. Link: https://www.diva-portal.org/smash/get/diva2:1972737/FULLTEXT01.pdf
[10]. React Team. "React Documentation." Link: https://react.dev/
[11]. Node.js Contributors. "Node.js Documentation." Link: https://nodejs.org/en/docs
[12]. Express.js Team. "Express.js Documentation." Link: https://expressjs.com/
[13]. MongoDB, Inc. "MongoDB Documentation." Link: https://www.mongodb.com/docs/
[14]. ECMA International. "ECMAScript Language Specification." Link: https://tc39.es/ecma262/