International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Online Chat Website Application

Prof. Aditya Lavhale[1], Arjun Jaiswal[2], Ayush Yenprediwar [3]

[1]Asst.Professor, Department of Computer Science Engineering, Tulsiramji Gaikwad Patil College of Engineering & Technology, Nagpur Maharashtra, India

[234]Students, Department of Computer Science Engineering Tulsiramji Gaikwad Patil College of Engineering & Technology, Nagpur Maharashtra, India

Abstract**—** Real-time communication platforms have become a foundational requirement in today's digital world, powering social media, professional collaboration systems, customer support environments, gaming ecosystems, and enterprise communication tools. This project presents the end-to-end design, development, deployment, and analysis of a **Full-Stack Real-Time Chat Application built** using modern web technologies including **Node.js, Express, React, MongoDB, WebSockets (Socket.io), Docker, and Kubernetes.**

The goal of this research is to demonstrate how full-stack engineering, cloud technologies, real-time networking, and container-based deployment can be integrated to build a highly scalable, low-latency, secure communication system that mirrors industrial standards used by applications such as WhatsApp Web, Facebook Messenger, Slack, and Microsoft Teams.

This study focuses on solving real-world challenges such as authentication security, message reliability, concurrency handling, state synchronization, database consistency, and deployment scalability. A detailed architectural analysis is presented, including the frontend–backend interaction model, **WebSocket event lifecycle, JWT authentication,** database schema optimization, and Docker-based environment standardization.

I. Introduction

Communication has always been one of the most fundamental requirements of modern digital systems. With the rapid expansion of the internet and the increasing dependency on web-based applications, real-time communication has become an essential component of both personal and professional interactions. Over the years, the transformation of technology has redefined how individuals exchange information, collaborate across distances, and engage in online communities. From traditional email systems to advanced instant messaging platforms, there has been a continuous evolution in delivering communication that is faster, more reliable, and more interactive. This evolution has paved the way for real-time chat applications that can exchange data instantly between users, irrespective of geographical location or device platform.

In recent years, real-time chat systems have become deeply integrated into various sectors such as business, education, healthcare, customer support, gaming, entertainment, and social networking. The demand for uninterrupted, seamless, and instantaneous messaging has grown significantly, driving the development of modern systems that prioritize scalability, fault tolerance, responsiveness, and user-centric design. Applications such as WhatsApp, Messenger, Telegram, Slack, and Discord have set global standards for real-time communication, demonstrating how essential features such as instant text delivery, media sharing, authentication, encryption, and presence detection have become part of everyday digital experience.

Historical Context of Obstacle Detection: The concept of obstacle detection has evolved alongside the development of machines, robotics, and intelligent systems. Its origins can be traced back to early automation research in the mid-20th century, when scientists first began exploring ways for machines to perceive their surroundings. During the 1950s and 1960s, obstacle detection was primarily studied in the context of simple mobile robots that relied on basic infrared and ultrasonic sensors to avoid collisions. These early systems had limited range and accuracy, but they laid the foundation for machine perception.

By the 1980s and 1990s, advances in computing power and sensor technology significantly transformed the field. Researchers began integrating computer vision techniques, allowing systems to interpret visual data from cameras. During this period, algorithms such as edge detection, pattern recognition, and stereo vision emerged, enabling robots and autonomous systems to detect obstacles with greater precision.

The 2000s marked a major turning point with the rise of artificial intelligence and machine learning. These technologies allowed obstacle detection to become more adaptive, enabling systems to "learn" from their environment. LiDAR, radar, and high-resolution cameras became widely used in robotics, drones, and early autonomous vehicle prototypes.

Principle of Ultrasonic Obstacle Detection: The principle of ultrasonic obstacle detection is based on the behavior of high-frequency sound waves and their interaction with surrounding objects. Ultrasonic sensors operate by generating sound waves at frequencies higher than the audible range for humans, typically around **40 kHz**. These waves are emitted into the environment by the sensor's transmitter and propagate through the air at the speed of sound, which is approximately **343 meters per second** under normal atmospheric conditions.

When these ultrasonic waves encounter an object, they are reflected back toward the source in the form of an echo. The sensor's receiver then detects this returning echo and accurately measures the **time interval** between the emission of the wave and the reception of the echo. Using this measured "time of flight," the sensor calculates the distance to the obstacle using the formula: **Distance = (Speed of Sound × Time) / 2**. The division by two is necessary because the sound wave travels to the obstacle and then returns to the sensor.

This technique allows the system to determine an object's distance without requiring physical contact, making it ideal for applications where safety, automation, and precision are critical. Unlike infrared sensors, ultrasonic sensors are generally unaffected by lighting conditions, dust, or color variations of surfaces, making them highly reliable in diverse indoor and outdoor environments. The consistent performance of ultrasonic detection enables its use in robotics, automotive parking systems, autonomous navigation, industrial machinery, and smart assistance devices.

Overall, the ultrasonic obstacle detection principle provides a robust, cost-effective, and energy-efficient method for sensing objects and estimating distances, forming the foundation for many modern automation systems.

II. Problem Identification

The increasing use of autonomous systems, mobile robots, assistive devices, and intelligent machines has made obstacle detection a critical requirement for ensuring operational safety and efficient navigation. However, most existing obstacle detection methods face significant limitations that affect their reliability in real-world environments. Technologies like infrared sensors, although inexpensive, are highly sensitive to ambient light, surface color, and environmental reflections, which can lead to inaccurate distance measurement or complete detection failure. Vision-based systems, including cameras and image-processing techniques, offer sophisticated analytical capabilities but require high computational power, complex algorithms, and controlled lighting conditions. These systems also struggle in low-light, foggy, or high-glare environments, making them unsuitable for small, low-cost, embedded devices.

Furthermore, many commercially available detection systems are expensive and difficult to integrate with microcontroller-based projects such as Arduino or Raspberry Pi. Their complexity and high power consumption also limit their use in battery-powered applications. As a result, there is a need for a detection mechanism that is low-cost, energy-efficient, and capable of functioning reliably across varying environmental conditions. Ultrasonic sensors offer a promising alternative, but they too require precise calibration, proper mounting orientation, and effective filtering of noise to ensure stable readings. Inaccurate time measurements, temperature variations, and sound absorption by soft materials can also influence sensor output, presenting additional challenges.

Therefore, the core problem identified in this study is the development of a **robust, accurate, and affordable obstacle detection system** that is easy to implement, works consistently in dynamic conditions, and integrates seamlessly with embedded platforms. Solving this problem is essential for applications involving autonomous navigation, indoor/outdoor robotics, intelligent mobility aids, and safety monitoring systems. Addressing these issues forms the foundation for designing a reliable ultrasonic obstacle detection model capable of meeting modern requirements for safety, performance, and accessibility.

III. LITERATURE REVIEW

A) Literature Survey

Obstacle detection has been widely studied across robotics, autonomous systems, and embedded devices, with research efforts spanning low-cost proximity sensors to high-precision LiDAR and vision systems. Early work established basic proximity sensing using infrared and simple contact sensors, but these approaches were soon found limited by lighting conditions, surface reflectivity, and short detection ranges. Ultrasonic sensing emerged as a practical alternative for many embedded applications because it measures distance using sound-wave time-of-flight and is largely insensitive to ambient light and surface color. The project report on this Online Chat / AI-ML internship also documents the practical suitability of ultrasonic sensors for low-cost embedded projects and robotic navigation.

Techniques and Algorithms from Literature

Research on obstacle detection has introduced a wide range of sensing techniques and computational algorithms designed to improve accuracy, reduce noise, and enhance system reliability. Many early works employed **Time-of-Flight (ToF)** measurement as the primary technique for determining the distance of an obstacle using ultrasonic waves. This simple but effective approach forms the foundation of most ultrasonic detection systems, relying on precise timing and signal reflection analysis. To improve the stability of ultrasonic readings, several studies have implemented **digital filtering algorithms** such as the **Moving Average Filter** and the **Kalman Filter**, which reduce random noise and smooth out fluctuations caused by environmental interference. These filtering techniques have been widely adopted in embedded and robotic applications due to their low computational cost and real-time suitability.

B) Literature Summary

The existing body of research on obstacle detection highlights a steady evolution of sensing technologies and methodologies aimed at improving accuracy, reliability, and adaptability in real-world environments. Early studies established the limitations of traditional sensors such as infrared, which proved highly sensitive to ambient light and reflective surfaces. In contrast, ultrasonic sensing gained prominence for its robustness, low cost, and consistent performance under varying lighting conditions. Literature across robotics and embedded systems demonstrates that ultrasonic sensors provide dependable short-range detection using the time-of-flight principle, making them well-suited for small-scale and low-power applications.

B) Literature Summary

The existing body of research on obstacle detection highlights a steady evolution of sensing technologies and methodologies aimed at improving accuracy, reliability, and adaptability in real-world environments. Early studies established the limitations of traditional sensors such as infrared, which proved highly sensitive to ambient light and reflective surfaces. In contrast, ultrasonic sensing gained prominence for its robustness, low cost, and consistent performance under varying lighting conditions. Literature across robotics and embedded systems demonstrates that ultrasonic sensors provide dependable short-range detection using the time-of-flight principle, making them well-suited for small-scale and low-power applications.

IV. RESEARCH METHODOLOGY

A. Proposed System

The proposed system aims to develop a reliable and cost-effective ultrasonic obstacle detection model capable of accurately identifying nearby objects using the time-of-flight principle. Unlike existing high-cost or computationally heavy detection methods, this system is designed to operate efficiently on lightweight embedded platforms such as Arduino or similar microcontrollers. The system utilizes an HC-SR04 ultrasonic sensor to emit high-frequency sound waves and measure the echo return time in order to compute the distance of an obstacle. The microcontroller processes this data using optimized algorithms that include noise filtering, averaging techniques, and threshold-based decision logic to improve measurement stability.

1. System Objective

- To design a low-cost ultrasonic obstacle detection system.
- To ensure accuracy, stability, and real-time measurement.
- To integrate easily with embedded platforms such as Arduino.

2. Key Components

- HC-SR04 Ultrasonic Sensor
- Microcontroller board (Arduino / similar)
- Power supply and connectors
- Output interfaces: LEDs, buzzer, or display

3. Working Principle

- Sensor emits ultrasonic waves.
- Waves reflect back after hitting an obstacle.
- Echo time is measured and converted to distance.
- Filtering algorithms reduce noise and improve accuracy.

V. APPLICATIONS

1. Robotics Ultrasonic obstacle detection is widely used in mobile robots and autonomous vehicles to help them sense nearby objects and avoid collisions. Because ultrasonic sensors work in different lighting conditions, they support smooth navigation indoors and outdoors, making them ideal for beginner robotics, warehouse robots, and educational robots.

2. Automotive Systems In modern cars, ultrasonic sensors are commonly used in parking assistance and reverse detection systems. These sensors help drivers detect nearby objects at low speeds, improving safety during parking and tight maneuvers. Their short-range accuracy makes them essential for vehicle safety features.

3. Assistive Devices Ultrasonic detection is useful in devices designed for visually impaired individuals, such as smart walking sticks and wearable navigation aids. These systems convert distance information into vibrations or sound alerts, helping users detect obstacles and move safely in unfamiliar surroundings.

4. Industrial Automation Industries use ultrasonic sensors to monitor objects on conveyor belts, detect product presence, and prevent machine collisions. Their ability to function in dusty or noisy environments makes them reliable for factory automation and smart manufacturing applications.

5. IoT and Smart Home Applications Ultrasonic sensors are used in smart home systems such as automatic doors, touchless switches, and distance-based lighting control. Their responsiveness and low power consumption make them suitable for IoT devices that rely on motion or object detection.

VI. ADVANTAGES

1. Real-Time Communication The application provides instant message delivery using WebSockets and Socket.io, ensuring smooth, real-time interaction between users. This eliminates delays and makes the chat experience similar to modern messaging apps. Real-time updates like typing indicators and online/offline status enhance responsiveness.

2. Secure User Authentication The system uses JWT-based authentication and hashed passwords, making user login highly secure. By verifying tokens on every request, unauthorized access is prevented, giving users confidence that their private messages remain protected.

3. Message Persistence All chats are permanently stored in the MongoDB database with timestamps and message status. Users can access their complete chat history even after logging out, providing reliability and convenience for personal and professional communication.

4. Scalable Architecture The modular Node.js and React.js architecture, along with Docker containerization, allows the system to scale easily. More servers can be added to support thousands of users without affecting performance, making the application suitable for future expansion.

5. User-Friendly Interface The UI is designed to be clean, responsive, and easy to use on both mobile and desktop browsers. Features like chat rooms, sidebar navigation, and message search create an intuitive communication experience for all types of users.

6. Multi-Device Accessibility Since the application runs in a web browser, users do not need to install software. It can be accessed from laptops, mobiles, or tablets, making communication flexible and convenient from any location.

7. Advanced Chat Features Features like group chat, media sharing, typing indicators, and read receipts provide a modern and professional chatting experience. These functions enhance user engagement and make the platform suitable for teams, institutions, or organizations.

VII. LIMITATIONS

1. Limited Offline Support The application requires an active internet connection for all communication. Users cannot send or receive messages offline, and no local caching is available for offline access. This limits usability in areas with weak or unstable networks.

2. Scalability Challenges Under Very Heavy Load Although the system is scalable, extremely large numbers of concurrent users may cause delays in message delivery if the server is not upgraded or load-balanced. Without advanced clustering or distributed socket management, performance can degrade under heavy traffic.

3. No End-to-End Encryption (E2EE) While the system uses JWT authentication and secure hashing, messages are not encrypted end-to-end. This means that messages could theoretically be accessed on the server level, making it less secure compared to apps like WhatsApp or Signal.

4. Media Handling Limitations Large files, videos, or high-resolution images may take longer to upload or may fail if the server bandwidth and storage capacity are limited. Without a CDN or optimized storage system, media sharing can be slower.

5. Browser Dependency The chat system works primarily through a web interface. As a result, it depends on browser compatibility, and certain features may not work the same across all devices or browsers without additional optimization.

VIII. RESULTS

1. Successful Real-Time Messaging Performance The system demonstrated smooth and instant message delivery using Socket.io, with users able to exchange messages without noticeable delay. Real-time features such as typing indicators and online/offline status worked consistently during testing, confirming the reliability of the WebSocket implementation.

2. Secure and Stable User Authentication The JWT-based authentication system functioned correctly during login and message operations. Only authenticated users could access chat rooms, and token verification prevented unauthorized access. This validated the security and stability of the backend authentication module.

3. Efficient Message Storage and Retrieval MongoDB successfully stored all chat messages with timestamps, sender information, and message status. Users were able to retrieve complete chat history instantly, confirming that the database schema and indexing were optimized for fast read/write operations.

4. Smooth Multi-User Communication The system supported multiple users chatting simultaneously without performance breakdown. Group chats, private chats, and message synchronization were handled efficiently, showing that the application can scale to support larger user groups with minimal lag.

5. User Interface Performance and Responsiveness The frontend UI performed smoothly across devices, providing quick navigation between chat rooms and clean rendering of messages. Testing showed that the interface remained responsive even when handling large message lists, validating the efficiency of the React-based frontend. enabling better performance and a more native user experience.

IX. CONCLUSION

The development of the online chat application successfully demonstrates how modern web technologies can be integrated to create a fast, secure, and user-friendly communication platform. Through the use of Socket.io for real-time messaging, JWT for secure authentication, and MongoDB for efficient data storage, the system delivers a reliable chatting experience similar to widely used messaging platforms. The results confirm that the application performs smoothly under multi-user conditions, maintains stable backend operations, and provides a responsive interface across devices. Although the system includes certain limitations—such as lack of end-to-end encryption and limited media optimization—it achieves its primary objective of enabling real-time, browser-based communication with strong performance and scalability. Overall, the project proves to be a meaningful implementation of modern full-stack development concepts and serves as a strong foundation for future enhancement, expansion, and real-world deployment.

X. FUTURE WORKS

1. Implementation of End-to-End Encryption (E2EE) A major future enhancement is adding full end-to-end encryption to make messages readable only to the sender and receiver. This will significantly improve privacy and bring the application closer to security standards used in platforms like WhatsApp and Signal.

2. Support for Voice and Video Chat The system can be expanded to include voice calls, video calls, and group conferencing using WebRTC. This would transform the project from a simple chat application into a complete communication platform.

3. Cloud-Based Auto Scaling Future versions can integrate cloud services such as AWS, Azure, or Google Cloud to support auto-scaling. This allows the system to automatically handle thousands of users without performance issues.

4. Advanced Media Sharing and Storage Optimization Improving support for large files, videos, and images—possibly with dedicated cloud storage (S3 buckets or Firebase)—will make the application more media-friendly and efficient.

5. AI-Based Chat Moderation and Spam Detection Artificial intelligence can be used to detect spam messages, inappropriate content, and abusive behavior. This will ensure safe and clean communication in public or group chats.

6. Push Notifications and Offline Message Syncing By adding push notifications and offline syncing, users will receive message alerts even when the browser is closed. Messages typed while offline can sync automatically when the internet connection is restored.

7. UI Personalization and Themes Future updates can include customizable themes, chat backgrounds, dark mode, and adjustable fonts, giving users more control over their chatting experience.

8. Multi-Language Interface Support Providing multilingual UI options will make the application accessible to users from different regions, improving usability and adoption.

9. Group Management Features Advanced group features such as admin controls, mute settings, message deletion, and role-based permissions can enhance community and team-based communication.

10. Mobile App Development Finally, the project can be expanded into a dedicated Android/iOS mobile application using React Native or Flutter, enabling better performance and a more native user experience.

XI. REFERENCES

1. S. Grinberg, *Express.js Guide: The Comprehensive Book on Express.js*. Leanpub, 2014.

2. A. Banks and E. Porcello, *Learning React: Functional Web Development with React and Redux*. O'Reilly Media, 2017.

3. Node.js Documentation. Available: https://nodejs.org/docs

4. React.js Official Documentation. Available: https://react.dev

5. MongoDB Documentation. Available: https://www.mongodb.com/docs

6. Socket.io Documentation. Available: https://socket.io/docs

7. JSON Web Token (JWT) Documentation. Available: https://jwt.io/introduction

8. Docker Documentation. Available: https://docs.docker.com

9. A. Velu, "Real-time Messaging and WebSocket Communication in Modern Web Applications," *International Journal of Computer Applications*, vol. 182, no. 48, pp. 1–6, 2020.

10. M. Collina, "Building Real-Time Applications with WebSockets," *ACM Queue*, vol. 15, no. 5, pp. 1–10, 2017.

11. R. Fielding, "Representational State Transfer (REST)," *Doctoral Dissertation*, University of California, Irvine, 2000.

12. GitHub – MERN Stack Best Practices. Available: https://github.com/topics/mern-stack