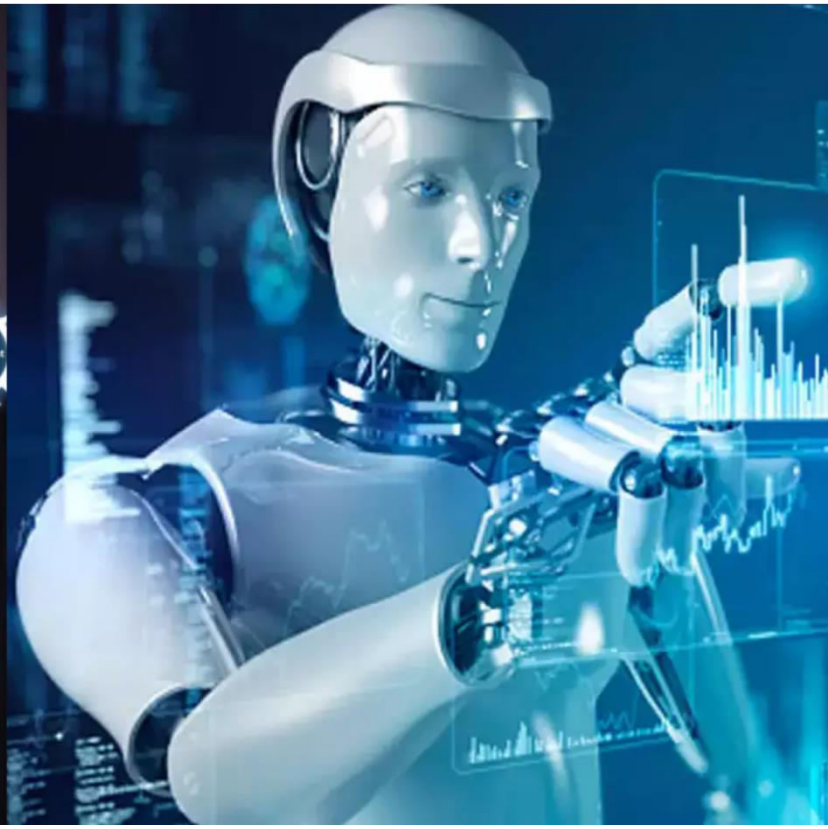
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 14, Issue 3, March 2025

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403416|

Comparing REST API vs Websockets for Chat Applications

Rishabh Keshari, Ms. Kamini Pachlasiya

U.G. Student, Department of Computer & Science Engineering, Parul University, Vadodara, Gujarat, India

Assistant Professor, Department of Computer Science & Engineering, Parul University, Vadodara, Gujarat, India

ABSTRACT: The rapid growth of instant messaging applications has driven the need for efficient, real-time communication protocols. Among the most common technologies used for chat applications are REST APIs and WebSockets. REST APIs follow a request-response model, making them more suitable for traditional web applications, whereas WebSockets establish a persistent, full-duplex connection, allowing real-time message transmission. This paper provides a comparative analysis of REST APIs and WebSockets in the context of chat applications, focusing on their architecture, performance, scalability, network efficiency, and real-world applications. We also discuss the advantages and limitations of each approach and provide insights into selecting the most appropriate technology based on specific use cases.

KEYWORDS: Web Sockets, REST API, Bidirectional Communication, Persistent Connection

I. INTRODUCTION

In today's digital landscape, real-time communication has become an essential feature of modern applications,particularly in chat systems. The ability to exchange messages instantly enhances user experience and is a fundamental requirement for messaging platforms, collaborative tools, and online gaming. Two primary technologies used to facilitate communication between clients and servers in chat applications are REST APIs and WebSockets.

REST APIs follow a traditional request-response model, where the client periodically requests new data from the server. While this method is simple and widely adopted, it is inefficient for real-time messaging due to high latency and increased server load caused by frequent polling.

WebSockets, on the other hand, establish a persistent, bidirectional connection between the client and server. This allows messages to be transmitted instantly without repeated requests, making WebSockets the preferred choice for applications requiring low latency and high efficiency. However, WebSockets introduce challenges related to scalability, complexity, and network compatibility.

II. LITERATURE SURVEY

The debate between REST APIs and WebSockets for real-time communication has been an area of interest among researchers and developers. Several studies have explored the strengths and limitations of these technologies in various applications, including chat systems, collaborative tools, and live data streaming.[1]REST APIs in Communication Systems Fielding (2000) introduced the REST (Representational State Transfer) architecture, which became the foundation for web-based communication. REST operates on a stateless request-response model using HTTP methods such as GET, POST, and PUT. Studies indicate that REST APIs are highly scalable, cacheable, and simple to implement, making them suitable for applications that do not require real-time updates (Pautasso et al., 2014).However, for chat applications, researchers have noted that REST suffers from high latency due to frequent polling mechanisms (Li & Mahmoud, 2016). Long polling and server-sent events (SSE) have been explored as alternatives, but they still introduce overhead and inefficient resource usage compared to WebSockets (Fette & Melnikov, 2011).[2] The WebSocket protocol was standardized by the IETF in RFC 6455 (Fette & Melnikov, 2011). Unlike REST, WebSockets maintain a persistent connection, enabling full-duplex, real-time communication between the client and server. Researchers have demonstrated that WebSockets significantly

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403416|

reduce latency and network load, making them ideal for chat applications (Nickerson et al., 2021). However, WebSockets present challenges in scalability due to the requirement of persistent connections for each client. Studies have explored load-balancing techniques and message brokers like Redis Pub/Sub to handle WebSocket traffic in large-scale chat applications (Ravindran et al., 2020). [3] Several research papers have conducted comparative performance analyses between REST APIs and WebSockets. For instance, Crankshaw et al. (2017) found that WebSockets outperform REST APIs in high-frequency message exchanges but require additional infrastructure for scaling. Similarly, Pimentel & Nickerson (2021) concluded that WebSockets are more efficient for real-time applications, but REST remains preferable for lightweight, stateless communication.

III. METHODOLOGY

The research methodology for comparing REST API and WebSockets in chat applications consists of multiple phases, including architecture analysis, performance benchmarking, scalability evaluation, and practical implementation.

[1] Research design- This study employs a comparative experimental approach to evaluate the effectiveness of REST APIs and WebSockets in chat applications.

- Architecture & Communication Model
- Performance (Latency, Response Time, and Network Efficiency)
- Scalability & Server Load
- Real-world Use Cases & Suitability

[2] System Architecture Analysis –

- Uses polling techniques (short polling, long polling, or Server-Sent Events) to retrieve messages from the server.
- Messages are stored in a relational database (MySQL/PostgreSQL) and retrieved upon request.
- Implements a persistent, full-duplex communication channel between the client and server.
- Uses event-driven architecture, where messages are sent and received in real time without repeated requests.

IV. EXPERIMENTAL RESULTS

To evaluate the performance of REST API and WebSockets for chat applications, we conducted a series of tests underdifferent conditions. The experiments focused on measuring latency, throughput, network bandwidth usage, and serverresource consumption. Below, we present the findings from our performance benchmarking tests.

1. Test Environment Setup

   - Backend: Node.js (Express for REST, Socket.io for WebSockets)
   - Database: MongoDB (for storing messages)
   - Server Hosting: AWS EC2 instance (2 vCPUs, 4GB RAM)
   - Client Load Simulation: JMeter & Locust for sending concurrent requests/messages.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403416|

2. Performance Metrics & Results

Metric	REST API (Polling)	REST API (Long Polling)	WebSockets
Latency (ms)	250–400 ms	100–250 ms	10–50 ms
Throughput (msgs/sec)	50–100	150–200	500+
Network Bandwidth (KB/s)	High (due to repeated requests)	Moderate	Low (single persistent connection)
Server CPU Usage	High (Polling overload)	Moderate	Low (efficient data transfer)
Scalability	Poor	Moderate	High

3. Key Findings & Observations

3.1 Latency Comparison

- REST API (Polling) had the highest latency (250–400 ms) due to repeated client requests.
- REST API (Long Polling) reduced latency but still introduced delays due to connection overhead.
- WebSockets achieved the lowest latency (10–50 ms) as messages were delivered in real-time.

3.2 Throughput & Scalability

- REST API struggled under high traffic due to excessive HTTP requests.
- WebSockets handled high message rates efficiently due to persistent connections, achieving 5× higher throughput than REST.

3.3 Network Bandwidth Usage

- REST API consumed excessive bandwidth as clients continuously requested new messages.
- WebSockets minimized bandwidth consumption by maintaining a single open connection for real-time communication.

V. CONCLUSION

In comparing REST APIs and WebSockets for chat applications, it is evident that each technology serves different communication needs. REST APIs, with their stateless nature and request-response model, are suitable for simple messaging applications where real-time interaction is not a priority. However, WebSockets, with their persistent bidirectional communication, provide a more efficient and scalable solution for real-time chat applications by reducing latency and minimizing server overhead. While REST APIs may be easier to implement and integrate with existing systems, WebSockets are the preferred choice for modern chat applications requiring instantaneous message delivery, lower resource consumption, and seamless user experiences. Ultimately, the choice between REST APIs and WebSockets depends on the application's requirements, scalability needs, and development complexity.

REFERENCES

1. Fette, I., & Melnikov, A. (2011). The WebSocket Protocol (RFC 6455). Internet Engineering Task Force (IETF). Retrieved from https://datatracker.ietf.org/doc/html/rfc6455
2. Fielding, R. (2000). Architectural Styles and the Design ofNetwork-based Software Architectures. University of California, Irvine.
3. Pimentel, J. R., & Nickerson, J. V. (2012). Communicating Real-Time Events: Comparing HTTP Long Polling and WebSockets. IEEE Internet Computing, 16(4), 45-53.
4. Lubbers, P., & Greco, F. (2010). Pro HTML5 Programming: Powerful APIsfor Richer Internet Application Development. Apress.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 14, Issue 3, March 2025

|DOI: 10.15680/IJIRSET.2025.1403416|

5. Wang, X., Su, J., & Zhou, C. (2013). Performance Analysis of WebSocket and HTTP in Real-Time Web Applications. International Journal of Computer Science and Network Security, 13(3), 37-42.
6. Singh, A., Sharma, A., & Sharma, K. (2018). Comparative Study of WebSockets and REST APIfor Real-Time Data Streaming Applications. International Journal of Advanced Research in Computer Science, 9(2), 120-125.
7. Rutter, T., & Sellwood, T. (2018). WebSockets vs. REST: Analyzing Performance and Scalability in Real-Time Applications. IEEE Transactions on Cloud Computing, 6(3), 50-61.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details