RESEARCH ARTICLE OPEN ACCESS

Simulation-Based Real-Time Polling System Using WebSockets: A Study on EventCue

Ruchika Jain*, Kalpana Swami**, Kritika Kumari**

*Associate Professor, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India
**B.Tech Student, Department of CSE, Global Institute of Technology, Jaipur, Rajasthan, India

ABSTRACT

Real-time audience interaction has become an important component in classrooms, conferences and live events. Traditional HTTP/HTTPS protocols require continuous response to fetch the updated data, this results in real-time data transmission issues, such as the inability of servers to actively push messages and resource wastage. The performance of live polling applications suffers from these system restrictions which impact their ability to respond quickly. The research introduces EventCue as a web-based polling system which operates in real-time to collect responses quickly and show results immediately. The system implements WebSocket technology to establish a bi directional data transmission medium between clients and server for real-time data transmission. The system uses React to create the user interface and Node.js for backend operations such as establishing websocket connection, poll management and response collection. The system shows better performance in response time and system reaction when multiple users join at the same time according to experimental results.

Keywords **—** Real-time polling, WebSockets, low-latency systems, Node.js, React.

1. INTRODUCTION

Live polling systems operate throughout educational facilities and corporate gatherings and public gatherings to obtain immediate feedback from audience members. The systems allow presenters to maintain audience participation through real-time data analysis for making immediate decisions. The current web-based polling systems that use HTTP for communication create excessive network traffic which results in longer response times and delayed result displays when numerous users participate at once. Real-time web communication methods have become more popular because they solve the current system restrictions. The WebSocket handshaking process is based on the HTTP protocol, where the client first sends an HTTP request to the server with specific fields in the request header indicating that it is a WebSocket connection request.[5]

The research introduces EventCue as a web-based polling system which uses WebSockets to establish bi-directional, full-duplex communication medium to achieve fast response collection and quick result updates. The system uses a web structure with React to create the user interface and the Node.js for managing polls and real-time data processing. The system provides instant result updates to all the connected clients. The present research interest is covered for:

- The paper describes a WebSocket-based polling system which optimizes data exchange between servers and clients.
- The system includes a fast response collection system which handles responses from multiple users at the same time.
- The research shows through different studies that WebSockets give faster responses than standard HTTP-based polling systems.

2. LITERATURE REVIEW

Online polling and audience response systems receive extensive research because they function in educational environments and live events and team-based workspaces. Multiple existing tools allow presenters to collect participant feedback which they can display to the audience through combined results. The current systems operate through request–response communication which forces clients to maintain continuous server requests for new data updates. The server load increases while user response times become longer when the system handles more users who access the system at the same time.

Guan Shiqi et al. [7] implemented WebSocket technology in the Network Control System Lab (NCS Lab) to solve HTTP/HTTPS protocol limitations for real-time data transfer by enabling servers to send messages actively and reducing resource consumption which resulted in better real-time performance and

system reliability.

WebSockets enable the servers to maintain continuous two-way communication with the clients through persistent data exchange. The WebSocket protocol stands out from HTTP because it enables servers to send messages to clients at any time after connection establishment without requiring new connections [6].

The implementation of real-time communication methods remains underdeveloped for web-based polling systems which operate as lightweight solutions for live events. The current solutions operate on commercial platforms but they lack detailed architectural explanations and performance assessment results. The research fills this knowledge gap through its development of a basic real-time polling system which assesses its operational speed and system reaction time.

3. PROPOSED SYSTEM ARCHITECTURE

The EventCue system operates as a web-based real-time polling system which allows users to collect responses instantly while showing updated results in real time. The system uses a client-server design that helps reduce response delays and manage large number of users at the same time.

A. System Overview

EventCue operates through three essential system elements which include:

- Frontend client
- Backend server
- Real-time communication layer

The system interface of the frontend enables users to create polls and submit responses while the backend system handles poll events and combines user feedback. The frontend and backend maintain continuous data exchange through WebSockets which eliminates the need for repeated requests.

The proposed real-time polling system architecture appears in Fig. 1.

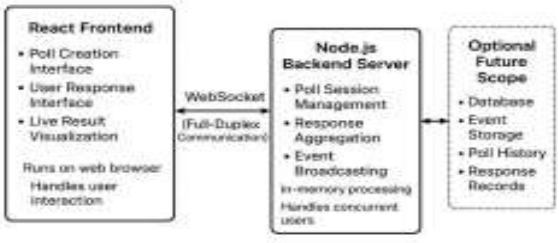

Fig. 1. The EventCue real-time polling platform operates through its system architecture design.

B. Frontend Module

The frontend is a simple web application that lets users to access polls directly in their web browsers. Users who participate in events can access current polling questions and respond to them in real-time through the system. The system updates poll results instantly after users submit their responses which creates an uninterrupted interactive experience.

The frontend system shows polling results in real time so users and presenters can track the response that are collected and present the next question on the screen. The system then requires immediate data collection because live sessions need quick audience responses to the make decisions or start discussions.

C. Backend Module

The backend server is very much responsible for the managing poll sessions, receiving responses, and then broadcasting updated results to all the connected clients. The system maintains each poll event as an independent session which provides synchronized updates to all users who access the same event.

Currently, the system operates without any persistent data storage and focusing on the live interaction during an event. The poll data always exists during the active session only. This design simplifies the system and then reduces processing overhead. In future versions, a database can be then integrated to store the event details, questions, and responses for post-event analysis.

D. Real-Time Communication Using

WebSockets then form the core of the proposed system. Once a client connects to the server then a persistent communication channel is established. Whenever a user submits a response then the backend processes the input and then immediately pushes the updated results to all the connected clients. This event-driven communication model then eliminates the delays associated with the traditional polling techniques.

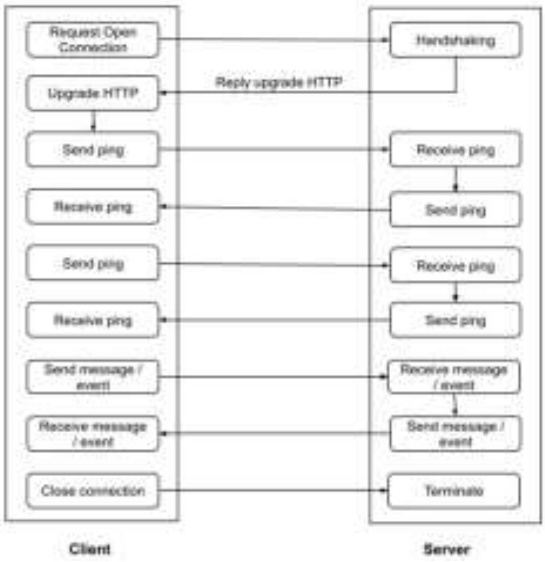

Fig. 2. Websocket Communication process

EventCue uses Websocket technology to quickly update the results and use network resources more efficiently when users interact with the system.

4. IMPLEMENTATION AND PERFORMANCE EVALUATION

The following section explains how to build EventCue polling system and shows its performance results when users participate in live polling activites. The evaluation process checks how well the system answers the users and how long it takes to respond to all the users.

A. Implementation Details

The EventCue frontend operates through a component based structure. It divides the user into two categories: creator of the event and user of the event. Creator can create an event and share the unique identifier of that event with the users. Users can join the event using the unique event identifier, they can view the active polls and submit their answers through the web interface. The client starts a WebSocket connection with the backend after the user sends the response to the system.

The backend system performs three essential functions which include creating polls, handling responses and generating final results. The system keeps active polls in the memory so that it can access and update the data quickly without using a database. When the server gets answers from clients, it updates the total results and then sends the new information to all the clients present in the event.

WebSockets enable clients to maintain the persistent communication with the server. The system operates differently from HTTP-based systems because it sends updates to clients without requiring them to send repeated requests. The system uses poll data changes to create server messages which get sent to all active clients.

B. Performance Evaluation Setup

The system performance assessment involved creating a scenario where multiple users accessed the same polling event while sending responses at the same time. The system performance assessment measured response update latency which represented the time span between user response submission and when all connected clients displayed the updated results.

The study compared two ways of sending updates: real time WebSocket updates and HTTP polling. In the HTTP polling method, users had to keep sending requests at fixed time intervals to check if there were new poll results. In WebSockets the system sends updates instantly to the user, without sending requests over and over to the server.

5. RESULTS AND DISCUSSION

The experimental results indicate that the WebSocket based approach significantly improves system responsiveness. In the HTTP polling model, noticeable delays were observed between response submission and result updates due to fixed request intervals and increased server workload. As the number of concurrent users increased, this delay became more significant.

In contrast, EventCue demonstrated consistently low response latency using WebSockets. Updated polling results were delivered almost instantly to all participants, even during simultaneous submissions. Additionally, the server experienced reduced network overhead since persistent connections eliminated redundant requests.

Creator View

Participant View-

Table 1: Comparison Between HTTP Polling and WebSocket-Based System

	HTTP Polling	**WebSocket**
Communication Model	Request-Response	Full Duplex
Update Latency	Higher	Low
Network Overhead	High	Reduced

Real-Time Updates	Limited	Immediate
Scalability	Moderate	Improved

These results demonstrate that WebSockets are well-suited for real-time polling applications requiring fast and synchronized data updates.

6. CONCLUSION AND FUTURE WORK

Performance evaluation results indicate that the proposed system achieves reduced response latency and improved user experience, particularly under concurrent participation. The simplicity of the architecture also makes it suitable for lightweight deployments without complex infrastructure requirements.

Future enhancements to EventCue include the integration of a database for persistent storage of events and responses, support for large-scale deployments, and the incorporation of analytics to provide deeper insights into audience behavior. Security features such as authentication and access control can also be added to enhance system reliability.

REFERENCES

[1] P. Lubbers and F. Grecco. HTML5 Web Sockets: A Quantum Leap in Scalability for the Web [Online]. Available: http://www.websocket.org/quantum.html

[2] W. Łasocha and M. Badurowicz, “Comparison of WebSocket and HTTP protocol performance”, J. Comput. Sci. Inst., vol. 19, pp. 67–74, Jun. 2021.

[3] J. Smith, “Real-time Web Communications: A Comparative Study of WebSockets and HTTP Polling,” Journal of Web Development, vol. 12, no. 3, pp. 45–60, 2019.

[4] J. Lu, P. Cao, C. Mo, and G. Li, "Design and Implementation of a Real-Time Collaborative Multi-End

[5] S. Guan, W. Hu and H. Zhou, "Real-Time Data Transmission Method Based on Websocket Protocol for Networked Control System Laboratory," 2019 Chinese Control Conference (CCC), Guangzhou, China, 2019, pp. 5339-5344.

[6] Wu Jie, Sun Zhenyong, Han Songtao, Application of Web Socket between server and client [J]. Hoisting and Conveying Machinery,2024,(13):100 104.

[7] R. Ajmera and D. Dharamdasani, “Comparative study of existing food delivery app,” Global Research Journal, pp. 454–463, 2022.

[8] A. Chauhan, R. Misra, "Outline of Web Development Life cycle in Software Engineering", International National Conference on Recent Trends in Engineering & Technology, 2023.

[9] A. Kalwar and R. Ajmera, “ARQI: Model for developing web application,” Int. J. on Technical and Physical Problems of Engineering (IJTPE), vol. 13, no. 47, pp. 7–13, Jun. 2021.

[10] A. Kalwar, R. Ajmera, and C. S. Lamba, “An empirical study in small firms for web application development and proposed new parameters,” Int. J. of Innovative Technology and Exploring Engineering, vol. 8, no. 4, Feb. 2019.

[11] R. Ajmera, A. Kalwar, and C. S. Lamba, “A modern study on progressions & issues of web applications development in small firms,” Int. J. of Scientific Research in Science and Technology, vol. 3, no. 8, Nov.–Dec. 2017.

[12] H. Fulwani, Harsh, H. Dhakad, A. Bohra, Dr. M. Mathur, "Web Development: Technologies, Trends and Future Scope", International Journal of Global Research in Science and Technology (IJGRST), Vol. 10, pp. 161-165, 2025.

[13] Pradeep Jha, Manju Mathur, Abhay Purohit, Apoorva Joshi, Anantika Johari, "LibUno: A React-Based Digital Platform for Smart Library Management", International Journal of Global Research in Science and Technology (IJGRST), Vol. 9, pp. 38-51, 2024.

[14] Tanush Kataria, Tanushka Saxena, Apoorva Joshi, Dr. Sangeeta Soni, Dr. Manju Mathur, "Web-Based Automation of Campus Examination and Placement Processes Using MERN Stack", International Journal of Global Research in Science and Technology (IJGRST), Vol. 10, pp. 236-241, 2025.

[15] Amit Bohra, Kritika Paliwal, Dr. Sangeeta Soni, "Online Code Editor: A Cloud-Based Platform for Real-Time Web Development", International Journal of Global Research in Science and Technology (IJGRST), Vol. 9, pp. 52-76, 2024.

[16] Harshit Sharma, Abhishek Kumar Agrawal, Kavita Bishnoi, Pankaj Jain, "From HTML

Pages to AI-Driven Platforms: A Review of Web Development Evolution", International Journal of Global Research in Science and Technology (IJGRST), Vol. 9, pp. 120-124, 2024.