As per UGC guidelines an electronic bar code is provided to secure your paper

International Journal for Modern Trends in Science and Technology
Volume 12, Issue 04, pages 1351-1356
ISSN: 2455-3778 online
Available online at: http://www.ijmtst.com/vol12issue04.html
DOI: https://doi.org/10.5281/zenodo.19702130

Infinite Canvas A Real-Time Collaborative Digital Whiteboard Using Web Sockets and MERN Stack

Jalli Priyanka, K. Ram Sai, T. Venu Sai Durgarao, Bhanu Prakash, A. Bhuvana Chandrika

Department of Computer Science and Engineering, D.N.R. College of Engineering & Technology, Balusumudi, Bhimavaram, Andhra Pradesh, India

To Cite this Article
Jalli Priyanka, K. Ram Sai, T. Venu Sai Durgarao, Bhanu Prakash & A. Bhuvana Chandrika (2026). Infinite Canvas A Real-Time Collaborative Digital Whiteboard Using Web Sockets and MERN Stack. International Journal for Modern Trends in Science and Technology, 12(04), 1351-1356. https://doi.org/10.5281/zenodo.19702130

Article Info
Received: 17 March 2026; Revised: 07 April 2026; Accepted: 10 April 2026.

KEYWORDS

Real-time collaboration, Digital whiteboard, Infinite Canvas, Web Sockets, MERN stack, Creative design workflows

ABSTRACT

The rapid growth of distributed teams, online learning, and digital collaboration has highlighted the demand for intuitive and real-time interactive tools. This paper presents InfiniteCanvas, a real-time collaborative digital whiteboard system designed to facilitate seamless teamwork and brainstorming in both professional and educational environments. The system enables multiple users to simultaneously draw, write, and annotate on a shared digital canvas with synchronized updates powered by WebSockets. A diverse set of features, including customizable drawing tools, color palettes, and text annotations, enhances user creativity and interaction. To ensure reliability and scalability, the backend leverages the MERN stack (MongoDB, Express.js, React.js, Node.js) for session management, persistent storage, and efficient client–server communication. Additionally, the HTML5 Canvas API is employed to render high-performance graphics, while session persistence allows users to save, retrieve, and resume collaborative whiteboard activities. The system emphasizes collaborative editing, low-latency synchronization, and user-friendly design, making it suitable for online classrooms, virtual meetings, and creative design workflows. By bridging the gap between static whiteboard applications and modern collaborative platforms, InfiniteCanvas provides an effective solution for enhancing remote teamwork, communication, and knowledge sharing in real time.

1. INTRODUCTION

InfiniteCanvas is a real-time collaborative digital whiteboard designed to enhance communication and creativity among users working from different locations. It allows multiple users to interact simultaneously on a shared canvas where they can draw, write, sketch, and visualize ideas freely [1][5]. Unlike traditional communication tools, this platform focuses on visual

interaction, which plays a crucial role in understanding complex concepts [3][4].

The system is built using modern web technologies to ensure smooth performance and responsiveness. It uses WebSockets for instant updates, ensuring that all users see changes in real time without delay [9][11]. The HTML5 Canvas API enables high-quality rendering of graphics directly in the browser [8][10]. The application is developed using the MERN stack, which provides scalability and efficient data management [6][7].

Users can save their work and continue later, making it suitable for long-term projects [2]. The platform is accessible across devices such as mobiles, tablets, and desktops, ensuring flexibility and ease of use [10]. Overall, InfiniteCanvas provides an interactive and user-friendly environment for collaborative work, supporting real-time synchronization and efficient communication among users [1][5][9].

Purpose

The main purpose of Infinite Canvas is to provide an effective platform for real-time visual collaboration, enabling users to interact and share ideas seamlessly in a digital environment [1][5]. It aims to bridge the gap between physical and digital teamwork by offering a shared workspace where multiple users can express their ideas freely through drawing, writing, and sketching [3].

The system is designed to improve communication by allowing users to represent thoughts visually rather than relying solely on text-based interaction, which enhances understanding and creativity [4]. It supports various collaborative activities such as brainstorming, teaching, planning, and designing, making it useful across different domains [8][10].

Another key objective is to make collaboration simple, user-friendly, and accessible to all users, including beginners, by providing an intuitive interface and real-time interaction capabilities powered by modern web technologies [7][9].

Motivation

The motivation behind Infinite Canvas arises from the growing need for effective online collaboration tools in today's digital environment. With the increasing adoption of remote work and online education, users require platforms that support interactive and creative communication [1][7]. Traditional tools such as emails, chat applications, and document-based systems are limited in their ability to represent visual ideas, making it difficult to convey complex concepts clearly [3][5].

These limitations often reduce user engagement and creativity, especially in activities such as brainstorming, teaching, and collaborative design. Observing these challenges highlights the necessity for a more dynamic and visually interactive solution that enables real-time collaboration [8][10].

Additionally, advancements in web technologies such as WebSockets and real-time synchronization systems have made it possible to build responsive and scalable collaborative platforms [9][11]. Leveraging these technologies motivates the development of InfiniteCanvas as an efficient and user-friendly system for enhancing communication and creativity in digital collaboration.

Problem Statement

In today's digital environment, many collaboration tools fail to provide effective real-time visual interaction. Most existing platforms primarily focus on text-based communication, which limits creativity and makes it difficult to clearly explain complex ideas [3][5]. This creates a gap in communication, especially in activities that require visual representation such as brainstorming, teaching, and design.

Existing digital whiteboards often face challenges such as high cost, limited functionality, and poor performance when multiple users interact simultaneously [1][6]. Some systems also suffer from synchronization issues, resulting in delays and inconsistencies that can confuse users during real-time collaboration [9][11]. Furthermore, many tools lack features for saving and revisiting previous work, which affects long-term usability and productivity [2].

Accessibility is another major concern, as several platforms are not fully compatible across different devices such as mobiles, tablets, and desktops [10]. Additionally, the lack of an intuitive and user-friendly interface makes it difficult for beginners to effectively use these tools [7].

These limitations highlight the need for a scalable, efficient, and easy-to-use real-time collaborative system that ensures smooth synchronization, cross-device accessibility, and enhanced user experience.

2 LITERATURE SURVEY

[1] WebSocket Protocol for Real-Time Communication

Fette and A. Melnikov (2011) introduced the WebSocket protocol, which enables full-duplex communication between client and server over a single TCP connection. Unlike traditional HTTP, WebSockets maintain a persistent connection, allowing real-time data exchange with low latency. This makes it highly suitable for applications such as chat systems and collaborative platforms. The protocol reduces communication overhead and improves efficiency by avoiding repeated request-response cycles. It serves as a fundamental technology for enabling instant synchronization in real-time collaborative systems.

[2] M. Fowler (2015) – Event-Driven Architecture.

Martin Fowler explains the concept of event-driven architecture, where system components communicate through events. This approach is widely used in real-time applications to handle multiple user interactions simultaneously. In collaborative systems, actions like drawing or editing are treated as events and broadcasted to all connected users. This ensures that every participant sees updates instantly. The architecture improves scalability and responsiveness, making it ideal for systems with many concurrent users. This concept is important for designing the backend of real-time collaboration tools using Web Sockets.

[3] N. Lawson (2019) – Performance Analysis of Web Socket-Based Applications.

This study focuses on evaluating the performance of Web Socket-based systems in real-time environments. It highlights how Web Sockets reduce latency and improve synchronization compared to traditional polling methods. The research demonstrates that Web Sockets provide faster data transmission and better user experience in applications like live chat and collaborative editing. It also discusses challenges such as network delays and handling large numbers of users. These findings are useful for optimizing performance in real-time whiteboard systems. The study supports the use of Web Sockets for efficient and scalable collaboration.

[4] S. Tilkov and S. Vinoski (2010) – Node.js: Using JavaScript for Server-Side Programming. This paper explains how Node.js enables scalable server-side development using non-blocking, event-driven architecture. Node.js is particularly suitable for real-time applications because it can handle multiple connections efficiently. It works well with WebSockets to manage continuous communication between users. In collaborative platforms, Node.js ensures smooth handling of multiple simultaneous users without performance degradation. The paper highlights the advantages of using JavaScript across both client and server sides. This makes development faster and more consistent for real-time systems.

[5] J. Resig (2009) – JavaScript and AJAX Applications.

This work discusses the importance of JavaScript in building dynamic and interactive web applications. It explains how asynchronous communication improves user experience by updating content without reloading the page. While AJAX was widely used earlier, the study also shows its limitations in real-time systems. This led to the adoption of Web Sockets for better performance.

3 PROPOSED METHODOLOGY

The proposed system is a real-time collaborative whiteboard application. It allows multiple users to interact on a shared canvas simultaneously. Web Sockets provide fast and continuous communication. The MERN stack ensures scalability and storage. The system improves collaboration efficiency and user experience.

3.1 System Architecture

The above diagram illustrates the overall architecture of the Smart Recruitment System, designed to automate resume processing and job matching. The system is structured into multiple layers, including the User Layer, Application Layer, and Database Layer, with interactions between job seekers, employers, and administrators. The InfiniteCanvas system follows a **Client–Server Architecture with Real-Time Communication**, integrating the MERN stack and WebSockets for efficient synchronization.

1 Presentation Layer (Client Layer)

This is the **frontend layer** where users interact with the system. Built using **React.js** provides drawing tools, shapes, text, and UI controls.Uses **HTML5 Canvas API** for rendering drawings \Captures user actions (mouse/touch events) Output: User actions (draw, erase, text)

2 Real-Time Communication Layer (WebSocket Layer)

This is the **core real-time engine** of the system. Uses **WebSockets (Socket.IO)**. Maintains persistent

connection between client & server. Sends and receives updates instantly. Eliminates HTTP request delays
Output: Instant data transmission across users

3 Application Layer (Server Layer)

This layer handles **business logic and collaboration control**. Built using **Node.js + Express.js**. Manages: User sessions, Whiteboard rooms, Event broadcasting. Receives drawing data and distributes it to all clients
Output: Processed and broadcasted updates

4. Data Management Layer (Database Layer)

Handles data persistence and storage. Uses MongoDB
Stores: User details, Whiteboard states Saved sessions, enables users to resume previous work
Output: Stored and retrieved data

5. Synchronization Layer (Collaboration Engine)

Ensures **consistency among multiple users**. Synchronizes all client views. Handles concurrent updates. Maintains same canvas state for all users.
Output: Unified real-time view

System Architecture diagram

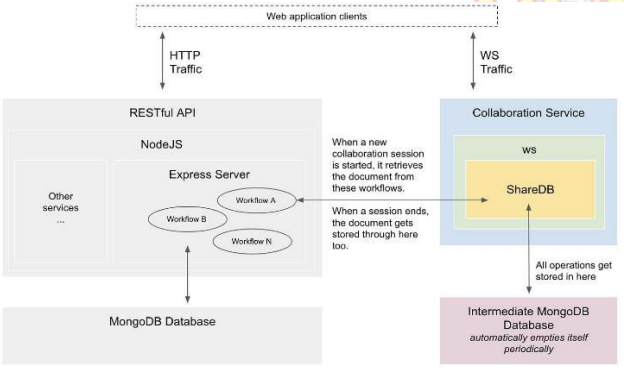


Fig:3.1 System Architecture

3.2 Use Case diagram

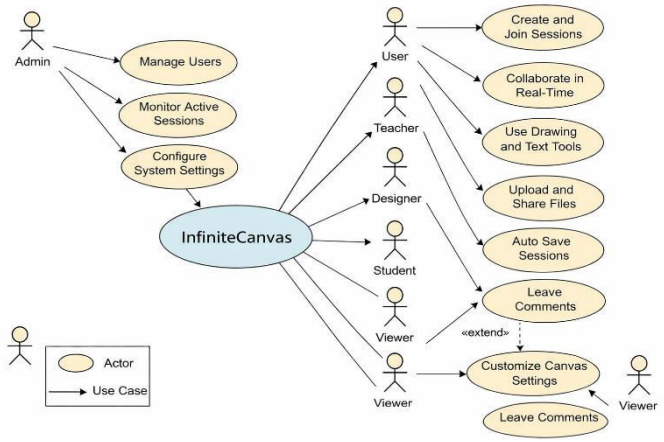


Fig:3.2 Use Case Diagram

The UML Use Case Diagram of InfiniteCanvas represents the interaction between different types of users (actors) and the system functionalities. It provides a high-level overview of how users engage with the platform and what operations they can perform. The system includes multiple actors such as Admin, User, Teacher, Designer, Student, and Viewer, each having specific roles and access levels. The Admin is responsible for managing users, monitoring active sessions, and configuring system settings, ensuring smooth operation of the platform.

3.3 Class diagram

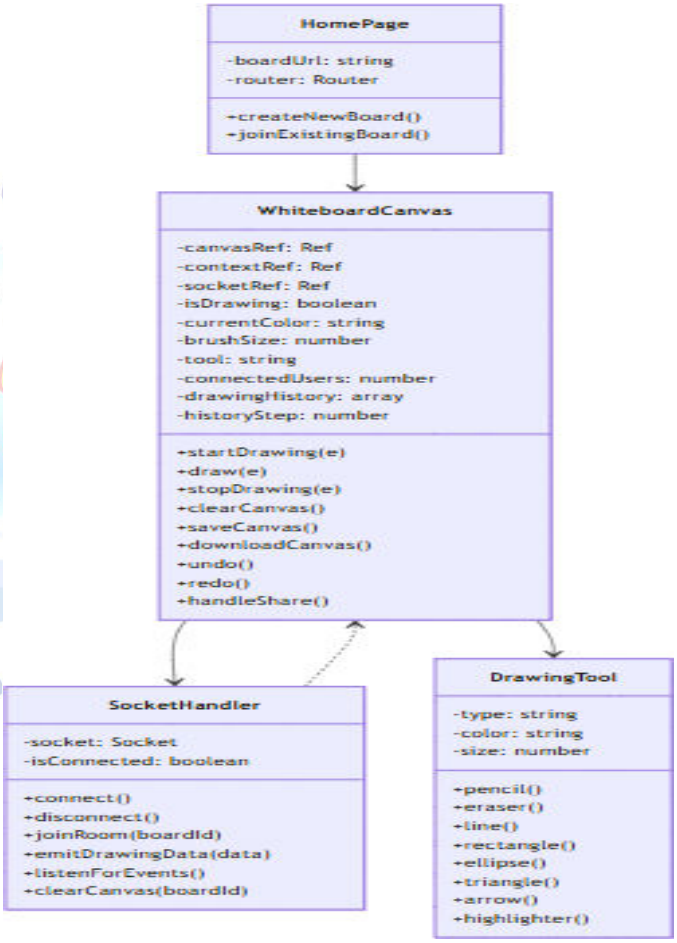


Fig: 3.3 Class Diagram

The image illustrates a class diagram for a collaborative whiteboard application, detailing the main components and their relationships. At the top of the hierarchy, the HomePage class handles board navigation using two primary attributes: boardUrl and router. The HomePage provides two essential methods: createNewBoard(), for initiating a new drawing session, and joinExistingBoard(), allowing users to access an existing collaborative board.

4 RESULTS

The system successfully enables real-time collaboration, allowing multiple users to draw, write, and annotate simultaneously with minimal delay. Integration of WebSockets ensures low-latency communication and instant synchronization of user actions across all connected clients. The use of HTML5 Canvas API provides smooth and high-performance rendering of graphical elements in the browser. The MERN stack architecture demonstrates good scalability and efficient handling of client–server interactions. The platform supports session persistence, allowing users to save, retrieve, and resume their work effectively.

Canvas Whiteboard

The Canvas Whiteboard homepage presents a user-friendly interface designed for seamless real-time collaboration on an infinite digital canvas. It highlights three core features: real-time collaboration allowing multiple users to draw together instantly synchronized on the same canvas; lightning-fast performance driven by WebSocket technology that ensures smooth and responsive drawing experiences; and rich drawing tools that provide a full suite of options.

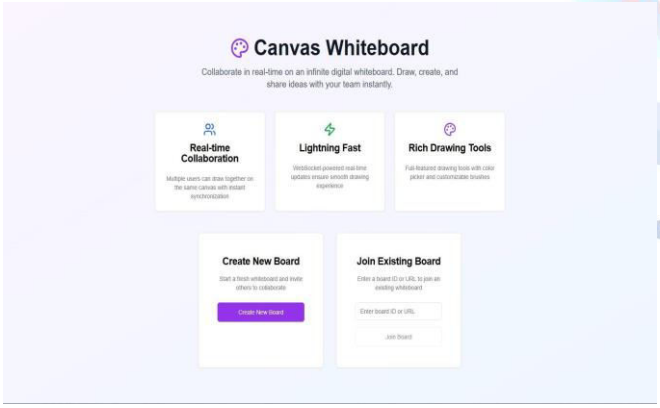


Fig: 4.1 Canvas Whiteboard

Canvas Whiteboard

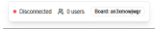
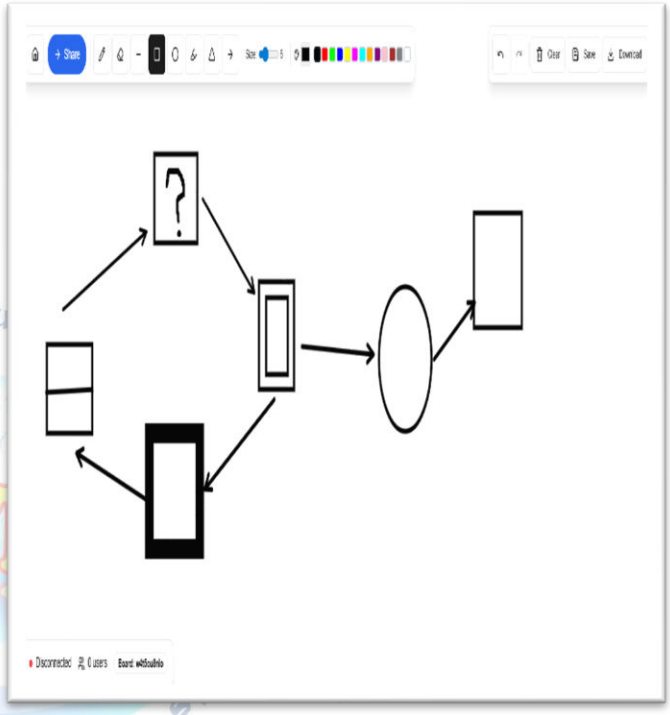


Fig:4.2 Canvas Whiteboard

This image displays the main workspace of the Canvas Whiteboard application, offering users a clean, minimalist interface centered around a large digital canvas for drawing and collaboration. At the top, there is a toolbar with various drawing and editing options, including tools for freehand drawing, erasing, creating lines, rectangles, circles, triangles, and arrows.

Canvas Whiteboard



The image shows an active Canvas Whiteboard session featuring a variety of geometric shapes and arrows drawn using the toolbar options. The workspace contains a rectangle, a circle, and a triangle, each outlined with precise, straight lines. Several arrows of different colors—black, red, green, and purple—are also present, illustrating directionality and interaction among the shapes. The arrows intersect through the geometric figures, and multiple straight lines are drawn in various orientations, adding complexity to the canvas.

5 CONCLUSION

In conclusion, the development of a real-time collaborative whiteboard application encompasses a complex interplay of frontend interactivity, backend synchronization, and robust data management to deliver seamless user experiences. Through careful implementation leveraging technologies like React for the client interface and Socket.IO for real-time

communication, the system enables multiple users to draw, edit, and share content synchronously. Rigorous testing—spanning unit, integration, stress, and fault tolerance— ensures reliability, scalability, and responsiveness under diverse conditions. Incorporating features such as undo/redo, session replay, and persistent storage, along with considerations for security, performance optimization, and accessibility, further enhances the platform's robustness and usability. Overall, this collaborative whiteboard solution stands as a scalable, resilient, and user-centric tool, empowering effective remote collaboration in educational, professional, and creative environments.

6 FUTURE SCOPE

The future scope of collaborative whiteboard applications is vast and promising, driven by evolving technologies and user demands for more immersive and versatile remote collaboration tools. One significant direction is the integration of advanced artificial intelligence (AI) and machine learning to enhance user experience, such as intelligent shape recognition, handwriting-to-text conversion, and context-aware suggestions, enabling faster and more accurate content creation. Augmented reality (AR) and virtual reality (VR) technologies hold the potential to transform whiteboards into 3D collaborative spaces, allowing users to interact naturally with virtual objects and participants regardless of physical location. Further expansion includes deeper integration with popular productivity and communication platforms, enabling seamless workflows across video conferencing, project management, and document editing tools. Real-time voice transcription and multilingual support can break language barriers, fostering more inclusive collaboration. Scalability improvements will support larger user groups and complex projects, while enhanced permission controls will allow granular management of roles and access rights for better security and organization.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] Anonymous. Real-Time Collaboration Whiteboard. IJERT 2025.

[2] Khan, M.; Patel, H.; Roy, S. Hybrid WebRTC-WebSocket Communication and Adaptive State Synchronization for Scalable Real-Time 3D Collaborative Whiteboard. arXiv 2025, arXiv:394065151.

[3] Singh, R.; Mehta, A. Sketch Sync: A Real-Time Collaborative Whiteboard Web App Built with Next.js. Scribd 2025 (accessed on 20 August 2025).

[4] Yu, J.; Lee, K.; Choi, M. COLiER: Collaborative Editing of Raster Images. arXiv 2021, arXiv:2107.05962.

[5] Coudo AI. Design a Real-Time Collaborative Whiteboard System. Coudo.ai Blog 2025.

[6] Rayan, C.R. Realtime Collaborative Whiteboard Using NodeJS, MongoDB, Nginx LB and Redis. Medium 2024.

[7] Rao, R. Elevating User Experience: Building Real-Time Collaboration with the MERN Stack and WebSockets. Medium 2025.

[8] MoldStud. Real-Time Drawing Applications with SocketIO: Building Interactive Whiteboards. MoldStud 2024.

[9] Anayat, S.M. Realtime Collaboration App with WebSockets & Node.js. Dev.to 2025.

[10] GeeksforGeeks. Real-Time Collaborative Editing App Using React & WebSockets. GeeksforGeeks 2025.

[11] Ably. The Complete Guide to WebSockets with React. Ably Blog 2025.

[12] Reddit. Real-Time Whiteboard App Made Using React and Socket.IO. Reddit 2022.

[13] YouTube. Develop Collaborative Whiteboard: WebSocket, Node.js. YouTube 2024. [14] YouTube. Real-Time Collaborative Whiteboard—Digiboard Project. YouTube 2023.