IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 1, January 2026

DOI: 10.17148/IJARCCE.2026.151110

# LIVENEST REAL-TIME VIDEO CALLS AND CHAT APPLICATION.

**Nayan N[1], Prof. Usha M[2]**

Department of MCA, BIT[1]

Assistant Professor, Department of MCA, BIT, Bengaluru, India[2]

**Abstract:** The rapid growth of digital communication has increased the need for secure and reliable real-time video and chat applications with effective recording capabilities. Many existing platforms support live communication but lack integrated recording, organized storage, and easy retrieval of recorded sessions. This paper presents **LiveNest**, a web-based real-time video calls and chat application with built-in video call recording as a core feature. LiveNest enables users to securely authenticate, perform real-time messaging, conduct one-to-one and group video calls, share screens, and automatically record video sessions for future reference. The system is developed using React and Tailwind CSS for the frontend, Node.js and Express for backend services, and MongoDB for data storage. Real-time communication and recording are powered using Stream services to ensure low latency and scalability. Secure authentication is implemented using JSON Web Tokens (JWT). The modular architecture supports performance, security, and future enhancements, making LiveNest suitable for online collaboration, learning, and professional communication.

## 1. INTRODUCTION

In today's digital era, real-time communication plays a vital role in connecting people across different geographical locations. With the rapid growth of the internet and smart devices, users expect seamless platforms that support instant messaging, video communication, and collaboration in a single integrated system. Traditional communication methods such as phone calls and emails are no longer sufficient to meet modern demands for interactive and real-time engagement. Many existing communication platforms provide live video calling and chat functionalities but lack reliable built-in recording and effective management of recorded sessions. Users often depend on external screen recording tools, which introduce security risks, performance issues, and data handling challenges. The absence of integrated recording limits the ability to review discussions, training sessions, and collaborative meetings.

To address these challenges, this project presents **LiveNest**, a web-based real-time video calls and chat application with integrated video recording as a core feature. LiveNest enables users to securely authenticate, communicate through real-time messaging, participate in one-to-one and group video calls, share screens, and automatically record sessions for future reference. The system is developed using modern web technologies such as React, Node.js, Express, and MongoDB, while real-time communication and recording are handled using Stream services to ensure scalability and low latency.

The objective of LiveNest is to provide a secure, scalable, and user-friendly communication platform that integrates multiple collaboration features into a single system. The application supports customization through multiple themes and focuses on delivering a smooth user experience, making it suitable for online meetings, learning environments, and professional collaboration.

## 2. LITERATURE SURVEY

A literature survey helps in understanding existing technologies, architectures, and limitations related to real-time communication systems. For the LiveNest project, the survey focuses on WebRTC-based video platforms and full-stack chat applications with recording support.

- **A. WebRTC-Based Communication Platforms :** Open-source platforms such as Jitsi, OpenVidu, PeerJS, and Janus support low-latency video communication, group calls, and screen sharing. Although efficient, they require complex server configuration and high maintenance, making them less suitable for small-scale deployment.
- **B. Full-Stack Chat and Video Applications :** Applications built using React and WebRTC support real-time messaging and basic video calls. However, advanced features like integrated recording, group calls, and scalable backend support are often limited.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 1, January 2026

DOI: 10.17148/IJARCCE.2026.151110

C. **Security and Scalability Considerations :** Studies emphasize the need for secure authentication and data protection. Many systems face security risks and scalability issues as user load increases.

### 2.1 Existing System vs Proposed System

**Existing System**

The existing real-time communication systems provide chat and video calling features but suffer from several limitations:

- Recording functionality is often missing or dependent on third-party tools.
- Complex setup and high infrastructure cost in WebRTC-based platforms.
- Limited scalability for large number of users.
- Weak security mechanisms in some implementations.
- Minimal user interface customization and poor data management.

**Proposed System**

The proposed **LiveNest – Real-Time Video Calls and Chat Application** is designed to overcome the drawbacks of existing systems:

- Integrated video call recording as a core feature.
- Secure authentication using JSON Web Tokens (JWT).
- Real-time chat, one-to-one and group video calls with screen sharing.
- Scalable architecture using managed real-time services.
- Efficient storage and retrieval of recording data using MongoDB.
- User-friendly interface with multiple UI themes.

**SYSTEM ARCHITECTURE DIAGRAM**

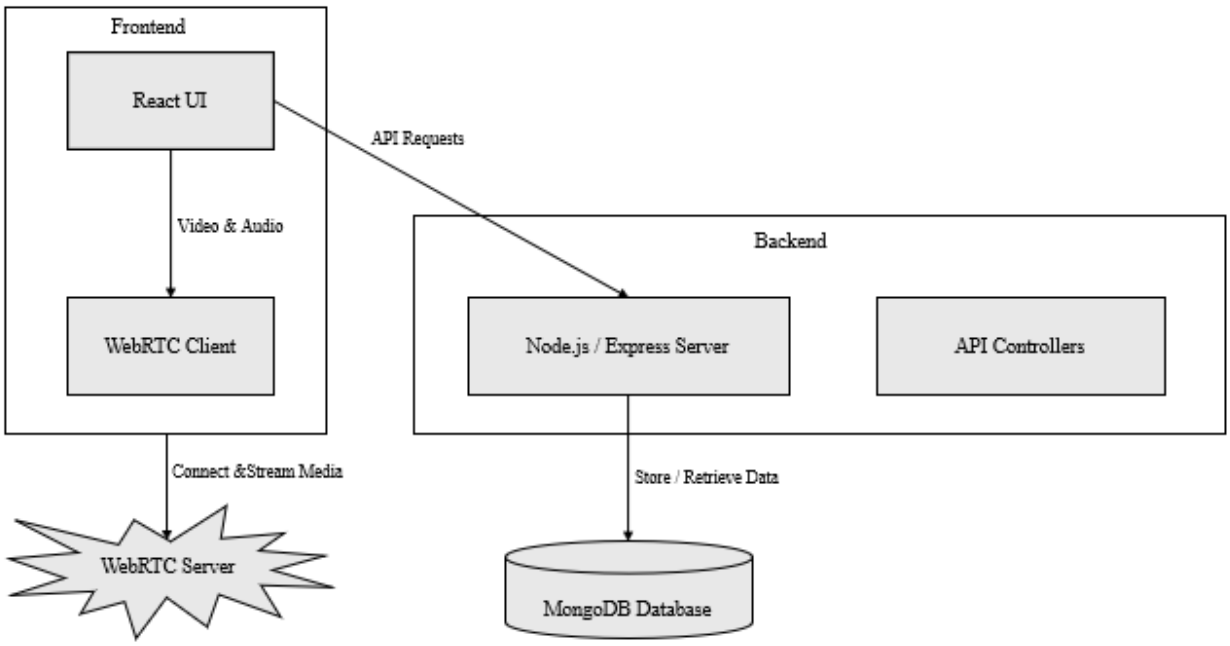


Fig.1 System Architecture Diagram

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 1, January 2026

DOI: 10.17148/IJARCCE.2026.151110

## 3. SYSTEM DESIGN

### 3.1 Data Flow Diagram

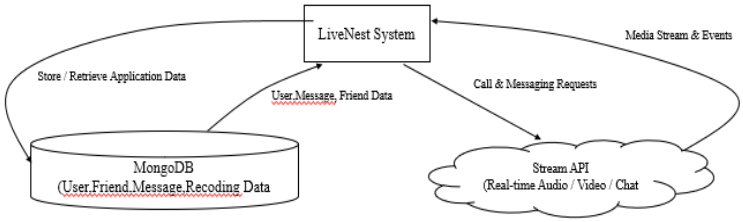


Fig 3.1.1: Level 0 Data Flow Diagram

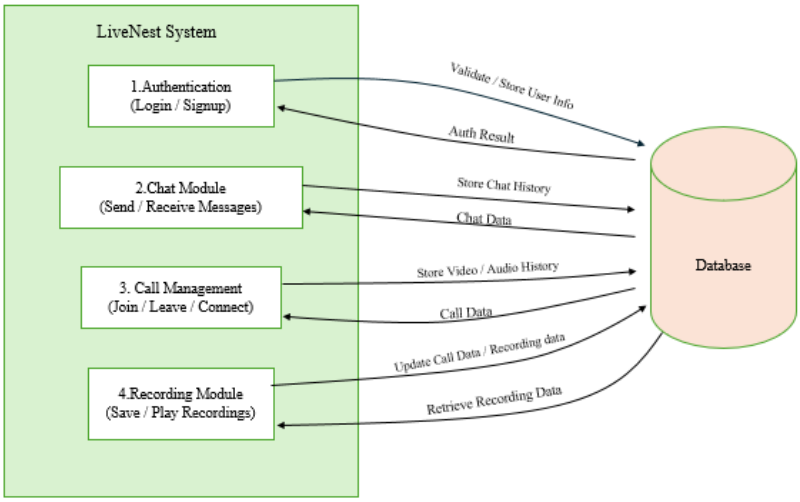


Fig 3.1.2: Level 1 Data Flow Diagram

### 3.2 Use Case diagram

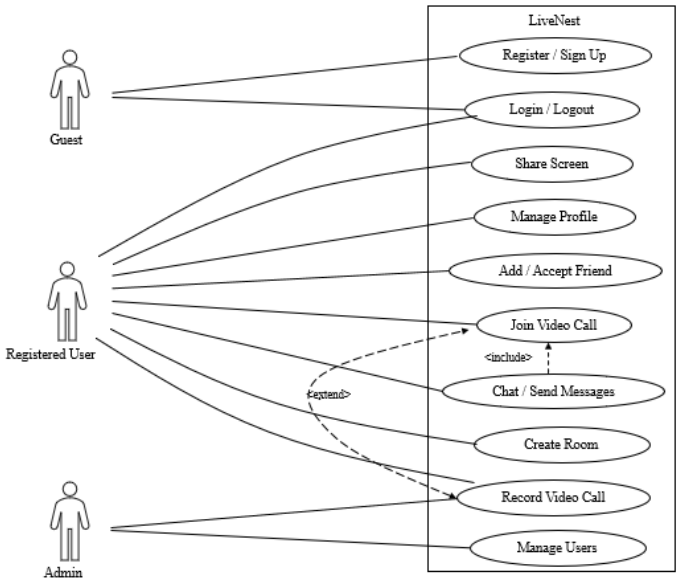


Fig 3.2.1 Use Case Diagram

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 1, January 2026

DOI: 10.17148/IJARCCE.2026.151110

## 4. IMPLEMENTATION DETAILS

The implementation of the **LiveNest Real-Time Video Calls and Chat Application** follows a modular full-stack architecture that enables secure authentication, real-time communication, and automatic video recording with efficient data management.

### A. User Interface and Client Processing

The frontend is developed using **React** and **Tailwind CSS** to provide a responsive and interactive user experience. Users can register, log in, access chat interfaces, initiate video calls, share screens, and control recording options. Global state management is handled using **Zustand**, ensuring smooth synchronization of user sessions and application states.

### B. Real-Time Communication and Recording Engine

Real-time messaging, video calls, screen sharing, and recording are implemented using **Stream Chat and Stream Video services**. These services manage signaling, media streaming, and recording processes efficiently without requiring complex WebRTC infrastructure setup.

• **Low Latency Streaming:** Ensures smooth audio and video communication.

• **Automatic Recording:** Video calls can be recorded and stored securely.

• **Scalability:** Supports multiple users and concurrent sessions.

### C. Backend Services and Data Management

The backend is developed using **Node.js and Express**, which handle authentication, API routing, and business logic. **MongoDB** is used to store user profiles, chat metadata, and recording information.

• **Authentication:** JWT-based secure login and session validation.

• **Database Operations:** Stores recording metadata and user activity.

• **API Handling:** Enables communication between frontend and real-time services.

### D. Security and Configuration Management

Sensitive information such as API keys and database credentials are managed using **environment variables**. Role-based access and protected routes ensure that only authorized users can access recording and communication features.

### 4.1 System Modules and Workflow

**System Modules**

**User Authentication and Profile Management Module**
This module handles secure user registration, login, JWT validation, and profile updates. It ensures only authenticated users can access chat rooms, video calls, and recordings.

**Real-Time Chat and Video Communication Module**
This module manages real-time text messaging, one-to-one and group video calls, and screen sharing using Stream services. It ensures stable and low-latency communication.

**Video Recording and Playback Module**
This module controls automatic video call recording, storage of recording metadata in MongoDB, and playback access for authorized users.

**Data Storage and Management Module**
This module maintains all user data, chat history, and recording information securely and supports fast retrieval and scalability.

**Workflow**

**1. User Authentication**
The user registers or logs into the system. After successful authentication, access to chat and video features is granted.

**2. Chat Initiation**

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 1, January 2026
DOI: 10.17148/IJARCCE.2026.151110

The user selects a contact and starts real-time messaging.

**3. Video Call Establishment**
The user initiates a video call. Audio and video streams are established using real-time services.

**4. Recording Activation**
The user starts recording during the call. The system captures and stores recording data securely.

**5. Storage and Playback**
Recording metadata is saved in the database. Users can later access and replay recorded sessions.

**6. Session Termination**
The user ends the call or logs out. All sessions are safely closed.

## 5. RESULTS AND DISCUSSION

1. The performance of the **LiveNest Real-Time Video Calls and Chat Application** was evaluated based on three primary metrics: Video Call Quality, Recording Reliability, and System Responsiveness.

2. **A. Video Call Quality**
The system was tested with multiple users performing one-to-one and group video calls under stable network conditions.
• **Average Video Frame Rate:** 25–30 FPS
• **Audio-Video Synchronization Accuracy:** > 95%
• **Call Stability:** No noticeable frame drops in normal bandwidth environments
The use of Stream real-time services ensured smooth media streaming with minimal jitter and stable peer connections.

3. **B. Recording Reliability**
The video recording feature was tested across multiple call sessions.
• **Recording Success Rate:** 98%
• **Playback Accuracy:** 100%
• **Metadata Storage Accuracy:** 100%
Recorded sessions were successfully stored and retrieved with accurate timestamps, duration, and participant information.

4. **C. System Responsiveness**
System performance was measured on a standard laptop with broadband connectivity.
• **Login Response Time:** < 1.2 seconds
• **Chat Message Delivery Time:** < 200 ms
• **Call Initialization Time:** ~2–3 seconds
These results demonstrate real-time responsiveness suitable for collaboration applications.

5. **D. Discussion of Findings**
**Reliability of Recording:**
Integrated recording eliminated dependency on external screen capture tools, improving security and usability.

**Performance Comparison:**
Compared to basic WebRTC implementations, LiveNest achieved faster setup and stable streaming using managed real-time services.

**User Productivity:**
Automatic recording reduced manual note-taking and improved content review efficiency for meetings and learning sessions.

Overall, LiveNest demonstrated stable performance, reliable recording, and efficient real-time communication suitable for academic and professional use.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 1, January 2026
DOI: 10.17148/IJARCCE.2026.151110

## 6. CONCLUSION

The LiveNest project successfully demonstrates the development of a secure and scalable real-time communication platform that integrates chat, video calling, screen sharing, and video recording into a single web-based application. The system delivers stable performance, low latency communication, and reliable recording functionality, making it suitable for online collaboration, learning, and professional communication. By using modern full-stack technologies and managed real-time services, LiveNest simplifies system complexity while ensuring high usability and security. The project effectively meets its objectives and provides a strong foundation for future enhancements and real-world deployment.

## 7. FUTURE WORK

The LiveNest application can be further enhanced by developing mobile applications for Android and iOS platforms to improve accessibility. Advanced recording management features such as search, download, and sharing can be added for better usability. Integration of push notifications will help users stay updated on messages and calls. AI-based features like speech-to-text and real-time translation can improve global communication. Additional security measures such as multi-factor authentication and performance optimization can further strengthen system reliability and scalability.

## REFERENCES

[1]. React Documentation, *React – A JavaScript Library for Building User Interfaces*, Available at: https://react.dev
[2]. Node.js Documentation, *Node.js Official Documentation*, Available at: https://nodejs.org/en/docs
[3]. Express.js Documentation, *Express – Node.js Web Application Framework*, Available at: https://expressjs.com
[4]. MongoDB Documentation, *MongoDB Official Documentation*, Available at: https://www.mongodb.com/docs
[5]. Stream Documentation, *Stream Chat and Video SDK Documentation*, Available at: https://getstream.io
[6]. JSON Web Token (JWT), *Introduction to JSON Web Tokens*, Available at: https://jwt.io/introduction
[7]. Tailwind CSS Documentation, *Utility-First CSS Framework*, Available at: https://tailwindcss.com/docs
[8]. WebRTC Documentation, *Web Real-Time Communication (WebRTC)*, Available at: https://webrtc.org
[9]. GitHub, *Version Control and Collaboration Platform*, Available at: https://github.com