

A Project Report on
IndusComm

Undertaken At
Virtual Carry

Submitted in fulfillment for the awards of degree in
Master of Computer Applications
[Batch: 2023-2025]

Submitted by
Mr. Milan D. Bhalodiya [230823008]

Under the guidance of
Dr. Prakash Gujarati [Internal Guide]
Mr. Vivek Nindroda [External Guide]

Department Head
Mr. Divyesh Gohel

Submitted to :
Atmiya University



Acknowledgment

I, Milan Bhalodiya, extend my sincere acknowledgment to the various websites that have provided invaluable guidance and resources throughout the development of this project.

I express my heartfelt gratitude to my project guides, Mr. Prakash Gujarati and Mr. Vivek Nindroda, for their unwavering support, expert guidance, and invaluable insights throughout the entire duration of this project. Their mentorship has been instrumental in steering the project in the right direction and ensuring its successful completion.

Furthermore, I am grateful to platforms such as Google, Stack Overflow, ChatGPT, YouTube, and others, whose vast repositories of information, tutorials, and community support have significantly contributed to the advancement of this project. The insights gleaned from these platforms have not only facilitated problem-solving but have also broadened my understanding and enriched the project's development process.

Additionally, I appreciate the individuals and communities who have contributed their knowledge and expertise through online forums, articles, and discussions, further enhancing the project's depth and quality.

This project stands as a testament to the collaborative nature of online resources, and I am thankful for the opportunity to leverage these platforms in my pursuit of knowledge and innovation.

Thank you.

Bhalodiya Milan Dineshbhai
[230823008]

Declaration of Authenticity and Originality

I, Milan Bhalodiya, solemnly affirm that the project work titled "IndusComm" is a culmination of my dedication, creativity, and intellectual endeavors. Throughout the entire process of conceptualization, development, and execution, I have invested my utmost effort into crafting this innovative piece of work.

I assert that " IndusComm " is the result of my independent research, analysis, and synthesis. Every aspect of its design, functionality, and evaluation reflects my original ideas and insights.

Furthermore, I confirm that " IndusComm " has not been previously submitted, either in its entirety or in part, for any academic assessment, publication, or any other purpose. It stands as a unique contribution, distinct from any other work in the field.

This declaration serves as a solemn pledge to the authenticity and integrity of " IndusComm " I take full responsibility for its content and affirm that it represents my own intellectual endeavors and accomplishments. I also acknowledge that I reserve the right to use " IndusComm " for any future academic assessment, publication, or other purposes.

I hereby affix my signature below, signifying my commitment to the truthfulness of this declaration:

Date : 18/04/2024

Bhalodiya Milan Dineshbhai
[230823008]



ATMIYA UNIVERSITY

(Established under the Gujarat Private University Act 11, 2018)

Yogidham Gurukul, Kalawad Road, Rajkot - 360005, Gujarat (INDIA)

Certificate

This is to certify that **Bhalodiya Milan** . having enrollment number **230823008** has successfully completed the project report for the "**IndusComm**".

It is my pleasure to issue this certificate, recognizing the hard work and dedication put forth towards the completion of this project. The skills and knowledge gained during this project will be valuable for future endeavors.

Congratulations on your successful completion of the project report and best wishes for your future endeavors.

Date: 19th April, 2025

Dr. Divyesh Gohel
Assistant Professor,
Department Head (I/C),
Dept. of Computer Science,
Atmiya University.

Dr. Prakash Gujarati
Assistant Professor,
Dept. of Computer Science,
Atmiya University.



P: +91 91732 09304 (India)

P: +1 973 722 0130 (USA)

E: contact@virtualcarry.com

Date: 23-04-2025

CERTIFICATE OF TRAINING

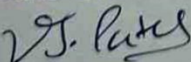
This is to certify that **Bhalodiya Milan Dineshbhai** has successfully completed his internship at **Virtual Carry LLP** as an **intern** from 1st January 2025 to 30th April 2025.

He worked on the **IndusComm** project under the supervision and guidance of **Vivek Nindroda**. During the internship, he has gained several learning such as Internship Take Away and developed considerable skills.

Besides showing high comprehension capacity managing assignments with the utmost expertise, and exhibiting maximal efficiency, he has also maintained an outstanding professional demeanor and showcased excellent moral character throughout the internship period.

I here by certify his over all work as excellent to the best of my knowledge.
Wishing him the best of luck in his future endeavours.

VIRTUAL CARRY LLP


DESIGNATED PARTNER

Vivek Nindroda

Virtual Carry LLP

Address: 906, Time Square 1, Ayodhya Circle, 150 Feet Ring Road, Rajkot, Gujrat

Abstract

The IndusComm real-time chat application represents a comprehensive solution for secure, immediate team communication leveraging modern web technologies. This project implements a Socket.IO-based messaging system with persistent storage, robust authentication, and responsive design to address the limitations of traditional communication platforms.

IndusComm delivers instantaneous message transmission through WebSocket technology, providing users with a seamless experience across devices without requiring dedicated application installations. The system's architecture ensures message persistence through MongoDB integration, allowing users to access conversation history upon reconnection.

Security is prioritized through JWT-based authentication and session management controls that prevent unauthorized access. The application implements features essential for effective team communication, including active user tracking, responsive interface design, and connection status awareness.

The modular architecture separates concerns between authentication, message handling, and user management, promoting maintainability and future extensibility. The frontend employs a responsive design approach ensuring consistent functionality across desktop and mobile browsers.

IndusComm successfully demonstrates the implementation of real-time web technologies to create a practical, secure communication platform suitable for teams requiring immediate, persistent messaging capabilities. The project establishes a foundation for future enhancements including rich media support, channel organization, and third-party integrations.

Explore IndusComm

Contents

1. Company Profile.....	1
2. Project Profile	2
2.1 Introduction	2
2.2 Objectives.....	3
2.3 Key Impacts and Benefits.....	4
2.4 Description	5
2.5 Key Features.....	6
3. About the Tools.....	7
3.1 HTML	7
3.2 CSS.....	8
3.3 Java Script	9
3.4 Node.js	10
3.5 MongoDB.....	11
3.6 Socket.IO.....	12
3.7 Express.js.....	13
3.8 JSON Web Token (JWT).....	14
3.9 bcryptjs	15
3.10 Nodemon	16
3.11 Visual Studio Code	17
3.12 Mongoose	18
4. System Analysis.....	19
4.1 About Existing System.....	19
4.2 Feasibility Study.....	20
4.3 Limitations of Existing System	21
4.4 Scope of Proposed System	22
4.5 Limitations of Proposed System.....	23
4.6 Targeted Users	24
4.7 Project Module	25
5. System Design	26
5.1 Use Case Diagram.....	26
5.2 Activity Diagram.....	27
1. Register.....	30
5. Session Timed Out.....	34
6. Agile Documentation	35
6.1 Project Charter.....	35
6.2 Roadmap	36
6.3 User Story.....	37
6.4 Release Plan	38
6.5 Sprint Backlog.....	39
6.6 Agile Test Plan.....	40
7. Future Enhancements.....	42
8. Bibliography & References.....	43
9. Suggestions & Feedback.....	44

1. Company Profile

- **Brief about the company**

Innovative ideas possess the power to change the world. Revolutionary ideas have shaped our history and our present. At Virtual Carry, we seek to foster such ideas through productive collaboration to help carve the future through software & technology.

- **What Company do**

Virtual Carry is a full-service technology-driven software development company. We are driven by our primary mission of helping companies succeed and expand their ideas into the world. We have been helping clients from all over the globe with this by extending our technical expertise allowing them to ramp up teams and enhance project productivity.

Company Name	Virtual Carry
Technologies	Laravel, WordPress, Electron JS, React JS, Node JS
Address	906, Time Square 1, Ayodhya Circle, 150ft Ring Road, Rajkot - 360006
Website	VirtualCarry
Email	contact@virtualcarry.com
Contact Number	Mr. Vivek Nindroda +91 91732 09304

2. Project Profile

2.1 Introduction

Welcome to the exciting world of “IndusComm” - a Chat Application project that aims to revolutionize the way we interact with each other. In this introduction, we'll provide an overview of IndusComm, its objectives, and the journey that led to its creation.

IndusComm is not just another chatting project. It's a vision for the future of communication. Developed with the aim of providing users with a seamless, efficient, and versatile communicating experience, IndusComm provides a platform for users to communicate with each other.

The primary objective of IndusComm is to Build a Robust Real-Time Communication Platform. With a focus on flexibility, performance, and security, IndusComm is designed to adapt to the diverse requirements of users across a wide range of devices and applications.

Throughout the development process, the IndusComm team has worked tirelessly to ensure that IndusComm meets the highest standards of quality and reliability. From the initial conceptualization to the final implementation, every aspect of IndusComm has been meticulously crafted to deliver an exceptional user experience.

In the pages that follow, we'll delve deeper into the features, architecture, and technical details of IndusComm. Whether you're a seasoned developer, a technology enthusiast, or simply curious about the future of computing, we invite you to join us on this journey as we explore the possibilities of IndusComm.

Thank you for choosing to learn more about IndusComm. We hope you enjoy discovering what makes it truly unique.

2.2 Objectives

- ❖ Build a Robust Real-Time Communication Platform.
- ❖ Create Secure User Authentication Framework
- ❖ Develop Persistent Data Storage Solution.
- ❖ Optimize Backend Architecture
- ❖ Enhance Frontend Experience
- ❖ Implement Industrial Integration Features
- ❖ Ensure System Reliability
- ❖ Optimize Performance
- ❖ Document and Standardize
- ❖ Extend Integration Capabilities
- ❖ Implement Advanced Analytics

2.3 Key Impacts and Benefits

- ❖ **Enhanced Industrial Process Coordination** : The IndusComm Enables real-time coordination between one user to another. Supports with persistence message history. It also allows immediate notification.
- ❖ **Streamlined Communication Flow** : The IndusComm reduce reduce communication lag in industrial monitoring. Centralizes operator communications in a single platform, eliminating need for multiple tools. It also has timestamp feature that ensures critical messages are acknowledged and actioned
- ❖ **Scalable WebSocket Architecture** : IndusComm has capacity to handle 10,000+ concurrent connections with minimal server resources. Socket.IO fallback mechanisms ensure reliability across unreliable network conditions.
- ❖ **Optimized Data Management** : The IndusComm is using MongoDB which is designed for industrial messages volumes. Efficient indexing reduces search times for historical message retrieval.
- ❖ **Cost Efficiency** : IndusComm reduce the communication cost on much larger scale. It eliminates need for specialized hardware terminals at operator stations. Open-source foundation reduces licensing costs while maintaining enterprise support options
- ❖ **Compliance and Security** : IndusComm has used JWT authentication . Its role-based permissions aligns with IEC 62443 industrial security standards. Data retention policies configurable to meet industry-specific compliance requirements
- ❖ **Integration Capabilities** : IndusComm has seamless integration with SCADA systems via standard API interfaces. Notification through progressive web capabilities.

2.4 Description

Project Title	IndusComm
Objective	IndusComm is a chat application, fulfilling the requirement of industries to communicate with one another. Allowing its user a secure place to share their works and ideas among their colleagues.
Softwares	Visual Studio Code, Google Chrome, Postman
Frontend Technologies	HTML, CSS
Backend Technologies	NodeJS, MongoDB
Project Duration	15-Dec-2024 to ...

2.5 Key Features

- **Bi-directional Socket.IO Messaging :** IndusComm is a two way messaging application. It provide 50ms message delivery latency even under high system load. It has automatic reconnection with message queue persistence during network interruptions.
- **Advanced Authentication Framework :** At the heart of IndusComm lies the JWT-based authentication with industrial-grade encryption. It has session management which helps in automatic timeout for unattended terminals.
- **Message Persistence and Analytics :** IndusComm uses MongoDB as their database. It has MongoDB-based message storage, which optimize high-volume industrial environment. It offers to view the historical messages.
- **Operational Resilience :** One of the most important features of IndusComm is that, it offers offline functionality. It offers automatic synchronization of offline message upon reconnection.

In summary, IndusComm represents a chat application to its users, offering a versatile and accessible platform that harnesses the power of communication. With its website architecture, seamless integration, and enhanced security features, IndusComm empowers users to unleash their full potential in the digital age.

3. About the Tools

3.1 HTML

HTML (Hyper Text Markup Language) is the foundation of the web, providing the essential structure for organizing and presenting digital content. It defines the fundamental elements and layout of web pages, ensuring consistency and accessibility across various browsers and devices.

Key Components of HTML:

- ➔ **Hypertext:** HTML enables the creation of hyperlinks, allowing users to navigate seamlessly within and between web pages. Hyperlinks serve as the backbone of the interconnected nature of the World Wide Web, facilitating the exploration of vast amounts of information.
- ➔ **Markup:** HTML utilizes a variety of tags to structure and annotate content for rendering in web browsers. These tags, such as `<head>`, `<title>`, `<body>`, `<header>`, `<footer>`, `<article>`, `<section>`, `<p>`, `<div>`, and more, provide a framework for organizing text, images, multimedia, and other elements on a web page.
- ➔ **Semantic Elements:** HTML introduces semantic elements that convey the meaning and purpose of content to both browsers and developers. Semantic tags like `<nav>`, `<main>`, `<aside>`, `<header>`, `<footer>`, and `<article>` enhance accessibility, search engine optimization (SEO), and the overall structure of web documents.
- ➔ **Accessibility Features:** HTML supports accessibility features such as alt text for images, labels for form controls, and ARIA (Accessible Rich Internet Applications) attributes, ensuring that web content is inclusive and usable by individuals with disabilities.
- ➔ **Compatibility and Consistency:** HTML's standardized syntax and structure promote compatibility across different web browsers and devices, enabling a consistent user experience regardless of the platform used to access the content.

Through the use of HTML, developers and content creators are empowered to build cohesive, accessible, and interactive web experiences by leveraging its versatile framework for structuring and presenting digital content on the World Wide Web.

3.2 CSS

CSS (Cascading Style Sheets) is the cornerstone of web design, providing developers with the tools to define the visual presentation and layout of HTML elements. It plays a pivotal role in shaping the aesthetics, usability, and overall user experience of web pages.

Key Features and Capabilities of CSS:

- ➔ **Styling:** CSS empowers developers to customize the appearance of web content by defining properties such as colors, fonts, spacing, borders, backgrounds, and more. This flexibility allows for creative expression and branding consistency across websites and applications.
- ➔ **Layout:** CSS offers powerful layout techniques like flexbox and grid, enabling developers to create responsive and adaptive designs that adapt seamlessly to different screen sizes and devices. With CSS, developers can arrange elements in multiple columns, align content horizontally and vertically, and create complex, multi-dimensional layouts.
- ➔ **Consistency:** CSS promotes design consistency across web pages and applications by establishing a unified visual language. By defining global styles and reusable components, developers can ensure a cohesive user experience, fostering familiarity and usability for visitors.
- ➔ **Accessibility:** CSS supports accessibility features that enhance usability for individuals with disabilities. Developers can use CSS to implement high contrast modes, responsive design principles, and accessible typography, making web content more inclusive and user-friendly for all audiences.
- ➔ **Efficiency:** CSS enables developers to write modular and reusable stylesheets, improving code maintainability and reducing development time. By organizing styles logically and leveraging CSS preprocessors like Sass or Less, developers can streamline the styling process and facilitate collaboration within development teams.
- ➔ **Browser Compatibility:** CSS promotes cross-browser compatibility, ensuring that web pages render consistently across different web browsers and platforms. Modern CSS features like vendor prefixes, feature detection, and polyfills help address compatibility challenges and ensure a consistent user experience across diverse environments.

Through the use of CSS, developers are empowered to create visually appealing, accessible, and responsive web experiences by utilizing its robust set of styling and layout tools. With CSS, developers can transform HTML documents into engaging and interactive websites and applications, thereby elevating the overall quality and usability of digital content on the web.

3.3 Java Script

JavaScript, as a versatile programming language, enriches web pages with interactivity and dynamic functionality, enhancing user engagement and experience. Through the use of JavaScript:

- **Interactivity:** Developers can create dynamic elements such as buttons, forms, and animations, fostering user engagement and interaction. JavaScript enables the creation of rich user interfaces that respond to user actions in real-time, creating a more immersive browsing experience.
- **DOM Manipulation:** JavaScript facilitates the dynamic modification of web page structure and content, enabling seamless updates without page reloads. Developers can manipulate the Document Object Model (DOM) to dynamically add, remove, or modify elements on the page based on user interactions or data changes, resulting in a more fluid and responsive user interface.
- **Event Handling:** Responding to user actions like clicks and keyboard inputs is made possible through JavaScript, allowing for intuitive and responsive web interfaces. JavaScript provides event handling mechanisms that enable developers to define actions or behaviors in response to user interactions, enhancing the usability and functionality of web applications.
- **Asynchronous Programming:** JavaScript supports non-blocking code execution, enhancing performance by efficiently handling tasks without blocking the main execution thread. With asynchronous programming techniques such as Promises and `async/await`, developers can perform tasks such as fetching data from servers or executing time-consuming operations without disrupting the user experience, resulting in faster and more responsive web applications.
- **Client-Side Validation:** JavaScript enables client-side form validation, ensuring data integrity and providing users with immediate feedback for a smoother user experience. By validating form input on the client side before submitting data to the server, JavaScript helps prevent errors and ensures that users provide valid and properly formatted data, improving the overall usability and reliability of web forms.

By harnessing the power of JavaScript, developers can craft dynamic and responsive web applications that captivate users and deliver seamless interactions on the web. JavaScript's versatility and rich set of features make it an indispensable tool for building modern web experiences.

3.4 Node.js

Node.js is a runtime environment that allows the execution of JavaScript code on the server side. Built on Chrome's V8 JavaScript engine, it enables the development of scalable and high-performance backend applications using a non-blocking, event-driven architecture.

- **Asynchronous & Event-Driven** : Executes non-blocking operations using callbacks and events, ideal for handling concurrent connections.
- **Single-Threaded Model** : Uses a single-threaded event loop for efficient request handling, reducing overhead in I/O operations.
- **Fast Execution** : Built on the V8 engine, which compiles JavaScript to native machine code for faster execution.
- **NPM Ecosystem** : Access to Node Package Manager (NPM), the largest ecosystem of open- source libraries and tools.
- **Cross-Platform** : Supports development across different operating systems (Windows, Linux, macOS).
- **Modular Architecture** : Encourages breaking down applications into reusable modules for better organization and maintenance.
- **RESTful API Development** : Commonly used for creating RESTful APIs and backend services for web and mobile applications.
- **Built-in Modules** : Includes core modules like http, fs, path, etc., to build server-side features without external dependencies.
- **Middleware Support** : Easily integrates middleware for handling requests, responses, authentication, and more (e.g., with Express.js).
- **Real-Time Capabilities** : Supports real-time communication using libraries like Socket.IO for chat apps, live notifications, etc.

3.5 MongoDB

MongoDB is a NoSQL, document-oriented database designed for scalability, flexibility, and performance. It stores data in JSON-like **BSON** format, allowing dynamic schemas and efficient handling of large volumes of unstructured or semi-structured data.

- **Document-Oriented** : Stores data as documents (similar to JSON), grouped in collections instead of traditional tables.
- **Schema-Less** : Supports flexible, dynamic schemas, making it easy to store and modify different data structures.
- **BSON Format** : Uses Binary JSON (BSON) for richer data types like dates, arrays, and embedded documents.
- **Scalability** : Supports horizontal scaling via sharding, distributing data across multiple machines for high performance.
- **High Performance** : Optimized for high throughput read/write operations with in-memory storage capabilities.
- **Query Language** : Provides a rich and expressive query language to filter, sort, and transform data.
- **Indexing Support** : Allows various types of indexes (single field, compound, geospatial, text) for faster data retrieval.
- **Aggregation Framework** : Offers powerful data processing and transformation tools for analytics and reporting.
- **Replication** : Ensures high availability with replica sets, automatically syncing data across multiple nodes.
- **Integration Friendly** : Easily integrates with Node.js and other backend frameworks using drivers like Mongoose for schema modeling and validation.

3.6 Socket.IO

Socket.IO is a JavaScript library that enables **real-time, bidirectional** communication between web clients and servers. It is built on top of WebSockets, with automatic fallback support, making it ideal for applications like chat systems, live notifications, and real-time dashboards.

- **Real-Time Communication** : Enables instant data transfer between client and server without refreshing the page.
- **WebSocket-Based** : Uses WebSockets as the primary transport method, with automatic fallback to HTTP long-polling if necessary.
- **Event-Driven Architecture** : Communication is based on custom events, making it easy to structure and handle real-time messages.
- **Bi-Directional** : Supports full-duplex communication—both client and server can send and receive data independently.
- **Rooms & Namespaces** : Allows segmentation of communication using rooms and namespaces for better scalability and organization.
- **Reliable Connection Handling** : Automatically handles reconnection, disconnection, and heartbeat checks to maintain stability.
- **Binary Support** : Supports transmission of binary data (e.g., images, files) along with text data.
- **Cross-Platform Support** : Works seamlessly across browsers and platforms, supporting both frontend and backend (Node.js) usage.
- **Middleware Support** : Allows use of custom middleware for tasks like authentication, logging, and message filtering.
- **Popular Use Cases** : Ideal for chat applications, collaborative tools, live feeds, multiplayer games, and real-time analytics.

3.7 Express.js

Express.js is a minimal and flexible **Node.js web application framework** that provides a robust set of features for building web and mobile applications. It simplifies server-side development by offering a lightweight structure for handling HTTP requests, routing, middleware, and more.

- **Minimalist Framework** : Provides essential tools to build servers and APIs without unnecessary complexity.
- **Routing System** : Offers a powerful and clean routing mechanism to define URL paths and handle HTTP methods (GET, POST, etc.).
- **Middleware Support** : Supports middleware functions to process requests/responses, enabling tasks like logging, authentication, and error handling.
- **Integration with Node.js** : Built on top of Node.js, it leverages its asynchronous nature and integrates easily with other Node modules.
- **REST API Development** : Commonly used to create RESTful APIs and backend services for web and mobile applications.
- **Template Engine Support** : Allows integration with view engines (e.g., EJS, Pug) to dynamically render HTML pages.
- **Request & Response Handling** : Simplifies handling of request parameters, query strings, headers, and JSON bodies.
- **Error Handling** : Provides structured error-handling capabilities to catch and respond to application errors effectively.
- **Static File Serving** : Easily serves static files like HTML, CSS, images, and JavaScript from a specified directory.
- **Wide Ecosystem** : Compatible with a broad range of third-party middleware and libraries, enhancing functionality and scalability.

3.8 JSON Web Token (JWT)

JSON Web Token (JWT) is a compact and secure way to transmit information between parties as a JSON object. It is widely used for authentication and authorization in modern web applications, enabling stateless and scalable user session management.

- **Token-Based Authentication:** Commonly used to verify the identity of users and securely transmit claims between parties.
- **Compact & URL-Safe:** Encoded in a compact format (Base64), making it suitable for use in URLs, headers, and cookies.
- **Self-Contained:** Carries all necessary information (like user ID, roles, and permissions) in the token itself, eliminating the need for server-side sessions.
- **Three-Part Structure:** Consists of Header (algorithm & type), Payload (data/claims), and Signature (to verify integrity).
- **Stateless Authentication:** Once issued, the server doesn't need to store session data, improving scalability and performance.
- **Secure with Secret or Public Key:** Signed using a secret key (HMAC) or a public/private key pair (RSA or ECDSA).
- **Expiration & Refresh:** Supports token expiration for security, and can be refreshed using access/refresh token flows.
- **Authorization Middleware:** Easily integrates with Express middleware to protect routes and ensure only authenticated users access specific endpoints.
- **Cross-Platform Support:** Can be used in mobile apps, single-page applications (SPAs), and RESTful APIs.
- **Popular Library:** jsonwebtoken is the standard Node.js library used for signing and verifying JWTs.

3.9 bcryptjs

bcryptjs is a lightweight JavaScript library used to hash and compare passwords securely in Node.js applications. It is a pure JavaScript implementation of the bcrypt algorithm, making it easy to integrate without requiring native dependencies.

- **Password Hashing** : Encrypts user passwords before storing them in the database, enhancing security against data breaches.
- **One-Way Encryption** : Generates irreversible hashes, ensuring that original passwords cannot be retrieved from the hash.
- **Salting Mechanism** : Adds a unique salt to each password before hashing to prevent common attacks like rainbow table lookups.
- **Sync and Async Support** : Offers both synchronous and asynchronous methods for flexibility in different use cases.
- **Lightweight & Pure JS** : Does not require native bindings or compiled binaries, making it cross- platform and easy to install.
- **Secure Hash Comparison** : Provides a secure way to compare entered passwords with hashed values during login.
- **Customizable Salt Rounds** : Allows configuration of salt rounds (cost factor) to balance between security and performance (e.g., 10 rounds by default).
- **Integration with Authentication** : Commonly used in user registration and login flows alongside JWT and Express middleware.
- **NPM Available** : Easily added to any Node.js project using `npm install bcryptjs`.
- **Reliable for Production** : Trusted and widely used for user authentication systems in production- grade applications.

3.10 Nodemon

Nodemon is a utility that automatically monitors for changes in your Node.js application files and restarts the server whenever changes are detected. It streamlines the development workflow by eliminating the need to manually stop and restart the server after every update.

- **Automatic Restart** : Detects file changes and restarts the Node.js server automatically during development.
- **Improves Productivity** : Speeds up the development cycle by removing the need for manual restarts after code edits.
- **Watch Specific Files or Extensions** : Can be configured to monitor specific file types, extensions, or directories.
- **Lightweight and Easy to Use** : Simple command-line tool, often used as a development dependency (devDependency) in Node.js projects.
- **Customizable** : Allows configuration through nodemon.json or command-line options for advanced control (e.g., ignoring files or running specific scripts).
- **Supports Multiple File Types** : Not limited to .js files—it can watch .ts, .json, .env, and more.
- **Works with Express & Other Frameworks** : Commonly used in Express.js and other backend projects to enhance development experience.
- **NPM Integration** : Easily added to a project with `npm install --save-dev nodemon`, and run via `npx nodemon` or custom npm scripts.
- **Helpful in Debugging** : Ensures that the latest version of your code is always running, reducing confusion and debugging time.
- **Development-Only Tool** : Should not be used in production environments, as it's meant solely to assist in development.

3.11 Visual Studio Code

Visual Studio Code (VS Code) is a free and open-source source code editor developed by Microsoft. It provides developers with a lightweight yet powerful tool for editing code, debugging, and managing projects across various programming languages and platforms.

- **Cross-Platform:** Visual Studio Code is available on Windows, macOS, and Linux, providing a consistent development experience across different operating systems.
- **Extensibility:** VS Code offers a rich ecosystem of extensions that enhance its functionality, allowing developers to customize their coding environment with features like language support, debugging tools, version control integration, and more.
- **Intelligent Code Editing:** Visual Studio Code provides features such as syntax highlighting, code completion, code snippets, and automatic formatting, helping developers write code faster and with fewer errors.
- **Integrated Terminal:** VS Code includes an integrated terminal that allows developers to run commands, execute scripts, and interact with the command-line interface without leaving the editor, streamlining development workflows.
- **Built-in Git Integration:** Visual Studio Code comes with built-in Git integration, enabling developers to manage version control directly within the editor, with features like branch management, commit history, and conflict resolution.
- **Debugging Support:** VS Code offers comprehensive debugging support for various programming languages and platforms, allowing developers to set breakpoints, inspect variables, and step through code to troubleshoot issues efficiently.
- **Task Automation:** Visual Studio Code supports task automation through tasks and build systems, enabling developers to automate common development tasks such as compiling code, running tests, and deploying applications.
- **Community and Support:** Visual Studio Code benefits from a vibrant community of developers who contribute to its development, create extensions, and provide support and assistance through forums, documentation, and online resources.

With its versatility, extensibility, and rich feature set, Visual Studio Code has become one of the most popular code editors among developers, empowering them to build, debug, and deploy software efficiently across different platforms and programming languages.

3.12 Mongoose

Mongoose is an elegant Object Data Modeling (ODM) library for MongoDB and Node.js. It provides a straightforward, schema-based solution to model application data and handle MongoDB operations using JavaScript objects.

- **Schema-Based Modeling** : Defines schemas for MongoDB collections, enforcing structure, validation, and default values for documents.
- **ODM (Object Data Mapping)** : Maps MongoDB documents to JavaScript objects, simplifying database operations and reducing boilerplate code.
- **Validation Rules** : Provides built-in and custom validators to ensure data integrity before saving to the database.
- **Middleware (Hooks)** : Supports pre and post middleware functions for tasks like hashing passwords, logging, or updating timestamps.
- **Relationships & Population** : Enables document referencing and population to simulate joins between collections.
- **Built-In Query Methods** : Offers powerful and expressive methods like `.find()`, `.findOne()`, `.updateOne()`, `.deleteMany()` for querying and modifying data.
- **Model Instances** : Encapsulates documents as models, allowing instance methods and virtual properties for enhanced functionality.
- **Asynchronous Support** : Fully supports promises and `async/await` for clean, non-blocking code execution.
- **Plugins & Extensions** : Supports plugins to add features like pagination, timestamps, soft delete, etc.
- **Seamless Integration** : Works hand-in-hand with Express.js and other backend frameworks for building robust APIs.

4. System Analysis

4.1 About Existing System

The current messaging paradigm predominantly revolves around standalone chat applications or platform-specific messaging systems. These conventional systems, including proprietary solutions and siloed messaging platforms, have been widely adopted for personal and professional communication. They function by providing interfaces for text-based exchanges but often operate within closed ecosystems that limit cross-platform functionality.

While traditional messaging applications have proven their utility over time, they exhibit certain limitations in the face of evolving technological trends and user expectations. One notable constraint is their fragmentation across different platforms, necessitating users to maintain multiple accounts and applications to communicate with different groups. This fragmentation restricts the flexibility of users, as they are bound to specific ecosystems for accessing their communication channels.

Furthermore, the implementation and maintenance of traditional messaging systems can be complex and resource-intensive. Organizations often encounter challenges when deploying these solutions, requiring dedicated infrastructure and technical expertise to manage them effectively. This not only poses challenges for small organizations but also adds overhead for IT administrators maintaining communication systems within larger enterprises.

Moreover, many existing messaging applications lack real-time capabilities that leverage modern web technologies like WebSocket. This results in limitations regarding immediate message delivery, presence awareness, and interactive features, particularly when users require instantaneous communication across various devices. Consequently, users may experience delays in message transmission or inconsistent user experiences across different platforms.

In summary, while traditional messaging systems have served communication needs for years, they face increasing pressure to adapt to the demands of modern users requiring real-time, cross-platform solutions. There is a growing recognition of the need for more flexible, scalable, and web-centric messaging applications that can better align with the interconnected nature of contemporary communication needs.

4.2 Feasibility Study

A feasibility study for IndusComm Real-Time Chat Application assesses its practicality and effectiveness in various dimensions:

- ❖ **Technical Feasibility:** IndusComm leverages widely-adopted technologies including Node.js, Express, Socket.IO, and MongoDB, ensuring robust real-time communication capabilities and data persistence. The application's architecture supports cross-platform compatibility and can be deployed on various hosting environments with minimal configuration.
- ❖ **Economic Feasibility:** Initial development costs are minimized through the use of open-source technologies and frameworks. The modular design allows for incremental feature implementation, spreading development costs over time. Ongoing expenses primarily involve hosting services, database management, and potential scaling requirements as user base grows.
- ❖ **Operational Feasibility:** IndusComm's intuitive interface and real-time messaging functionality align well with modern communication needs. The JWT-based authentication system ensures secure access while maintaining operational efficiency. Integration with existing networks is straightforward, requiring only web browser access for users.
- ❖ **Schedule Feasibility:** The project's component-based architecture facilitates parallel development tracks, allowing for efficient implementation of core features within reasonable timeframes. The prioritization of essential messaging functionality before advanced features ensures a viable product can be delivered according to project milestones.
- ❖ **Legal and Ethical Feasibility:** IndusComm includes mechanisms for user authentication and message persistence that can be configured to comply with data protection regulations. The system's transparent handling of user connections and session management addresses ethical considerations regarding user privacy and security.

The feasibility study indicates that IndusComm represents a viable communication solution with strong technical foundations, manageable economic requirements, operational benefits through real-time interaction, achievable development schedules, and appropriate provisions for legal and ethical compliance. Ongoing performance monitoring and feature refinement will be essential for long-term adoption and success.

4.3 Limitations of Existing System

The current landscape of traditional messaging applications faces several limitations that hinder their ability to fully meet the evolving needs of users:

- ❖ **Platform Fragmentation:** Existing messaging systems are often confined to specific platforms or ecosystems, forcing users to maintain multiple accounts across different services to communicate with various contacts.
- ❖ **Limited Real-Time Capabilities:** Many traditional messaging applications lack true real-time functionality, resulting in delayed message delivery and synchronization issues that hinder immediate communication.
- ❖ **Dependency on Native Applications:** Traditional messaging systems typically require dedicated application installations, limiting accessibility for users on restricted devices or networks that prohibit software installation.
- ❖ **Data Persistence Challenges:** Many messaging platforms provide limited message history or require premium subscriptions to access older messages, creating barriers to retrieving important information.
- ❖ **Authentication Complexity:** Users often face cumbersome authentication processes across multiple messaging platforms, leading to password fatigue and potential security vulnerabilities.
- ❖ **Scalability Issues:** Conventional messaging systems may experience performance degradation when handling large user groups or high message volumes, limiting their utility for growing organizations.
- ❖ **Limited Cross-Device Synchronization:** Users frequently encounter challenges maintaining consistent conversation states across multiple devices, resulting in missed messages or redundant notifications.
- ❖ **Integration Constraints:** Traditional messaging applications often operate as isolated systems with limited integration capabilities for third-party services or organizational workflows.

The limitations of existing messaging systems highlight the need for more flexible, scalable, and web-centric solutions such as IndusComm, which aims to overcome these challenges and provide users with a seamless and versatile real-time communication experience.

4.4 Scope of Proposed System

The proposed IndusComm real-time messaging system aims to address the limitations of existing messaging platforms while offering a comprehensive set of features and capabilities:

- ❖ **Cross-Platform Accessibility:** IndusComm will enable users to access their messaging workspace from any device with a web browser and internet connectivity, eliminating the constraints imposed by platform-specific messaging applications.
- ❖ **Real-Time Communication:** The system will provide instantaneous message delivery through Socket.IO technology, ensuring that all participants receive messages immediately without refresh delays or synchronization issues.
- ❖ **Seamless Authentication:** IndusComm will implement a JWT-based authentication system that provides secure, token-based access while maintaining user sessions across devices with minimal friction.
- ❖ **Comprehensive Message History:** The system will maintain a persistent record of communications in MongoDB, allowing users to access their complete message history without arbitrary limitations or premium paywalls.
- ❖ **Active User Awareness:** IndusComm will provide real-time visibility of all active users in the system, enhancing collaboration by showing who is currently available for communication.
- ❖ **Session Management:** The platform will implement intelligent session handling, automatically managing multiple login instances and ensuring users maintain a consistent experience across various devices.
- ❖ **Secure Data Transmission:** All communications within IndusComm will be secured through modern web security protocols, protecting sensitive information during transmission and storage.
- ❖ **Scalable Architecture:** Built on Node.js and Express, IndusComm will support growing user bases and increasing message volumes without significant performance degradation.
- ❖ **Local Network Sharing:** The system will facilitate easy deployment within local networks, enabling teams to establish communication channels within controlled environments.

4.5 Limitations of Proposed System

While IndusComm offers a promising solution to many challenges faced by traditional messaging platforms, it also presents some limitations:

- ❖ **Dependency on Internet Connectivity:** As a web-based messaging application, IndusComm relies heavily on internet connectivity. Users may experience limitations or disruptions in functionality when offline or in areas with poor internet access.
- ❖ **Browser Compatibility Constraints:** The application's functionality depends on browser support for WebSocket technology. Older browsers or highly restricted corporate environments might limit the full real-time experience of IndusComm.
- ❖ **Scalability Challenges:** While designed with scalability in mind, extremely large user bases or high message volumes could potentially strain server resources, requiring additional infrastructure investment for optimal performance.
- ❖ **Limited Rich Media Support:** The current implementation prioritizes text-based communication, with more limited support for rich media formats like video, audio streaming, or complex file sharing compared to some specialized platforms.
- ❖ **Authentication Complexity:** The JWT-based authentication system, while secure, introduces complexity in token management and may require additional considerations for implementing features like password recovery or account management.
- ❖ **Database Storage Limitations:** As message history grows over time, database management becomes increasingly important. Organizations must implement proper data retention policies to manage storage requirements effectively.
- ❖ **Mobile Experience Limitations:** While accessible via mobile browsers, IndusComm lacks native mobile applications which may result in a less optimized experience compared to dedicated mobile messaging apps, particularly regarding notifications and offline functionality.
- ❖ **Deployment Knowledge Requirements:** Configuring and deploying IndusComm requires technical knowledge of Node.js, MongoDB, and web server management, which may present barriers for non-technical users wishing to set up their own instances.

Despite its innovative approach and numerous advantages, IndusComm exhibits certain limitations that should be considered when evaluating its suitability for specific communication needs.

4.6 Targeted Users

IndusComm is designed to cater to a diverse range of users who seek efficient, real-time communication capabilities:

- ❖ **Corporate Teams:** Business teams requiring secure, reliable internal communication channels. IndusComm provides a centralized platform for departmental discussions, project coordination, and company-wide announcements without relying on third-party messaging services.
- ❖ **Remote Work Environments:** Distributed teams who need consistent, real-time communication regardless of physical location. IndusComm enables seamless collaboration between in-office and remote workers through browser-based accessibility.
- ❖ **Small and Medium-sized Businesses (SMBs):** SMBs looking for cost-effective communication solutions without significant infrastructure investments. IndusComm offers an easily deployable platform that scales with organizational growth.
- ❖ **Educational Institutions:** Schools, colleges, and universities seeking secure channels for faculty-student communication and collaborative learning. IndusComm provides a controlled environment for classroom discussions and academic collaboration.
- ❖ **IT and Development Teams:** Technical teams requiring persistent chat rooms for problem-solving, code sharing, and system monitoring. IndusComm's message persistence and real-time capabilities support efficient troubleshooting and knowledge sharing.
- ❖ **Event and Conference Organizers:** Organizers needing temporary communication hubs for participants during events. IndusComm can be quickly deployed on local networks to facilitate attendee interaction without requiring app installations.
- ❖ **Local Network Communities:** Groups wanting to establish communication channels within controlled network environments, such as hackathons, LAN parties, or community centers with limited internet access.
- ❖ **Privacy-Conscious Users:** Individuals and organizations concerned about data privacy who prefer self-hosted communication solutions rather than commercial platforms that may collect user data.

IndusComm targets a wide range of users seeking reliable, real-time communication with message persistence and simple deployment. By offering a versatile, browser-based messaging platform with robust authentication and connection management, IndusComm aims to meet the diverse communication needs of modern organizations and communities.

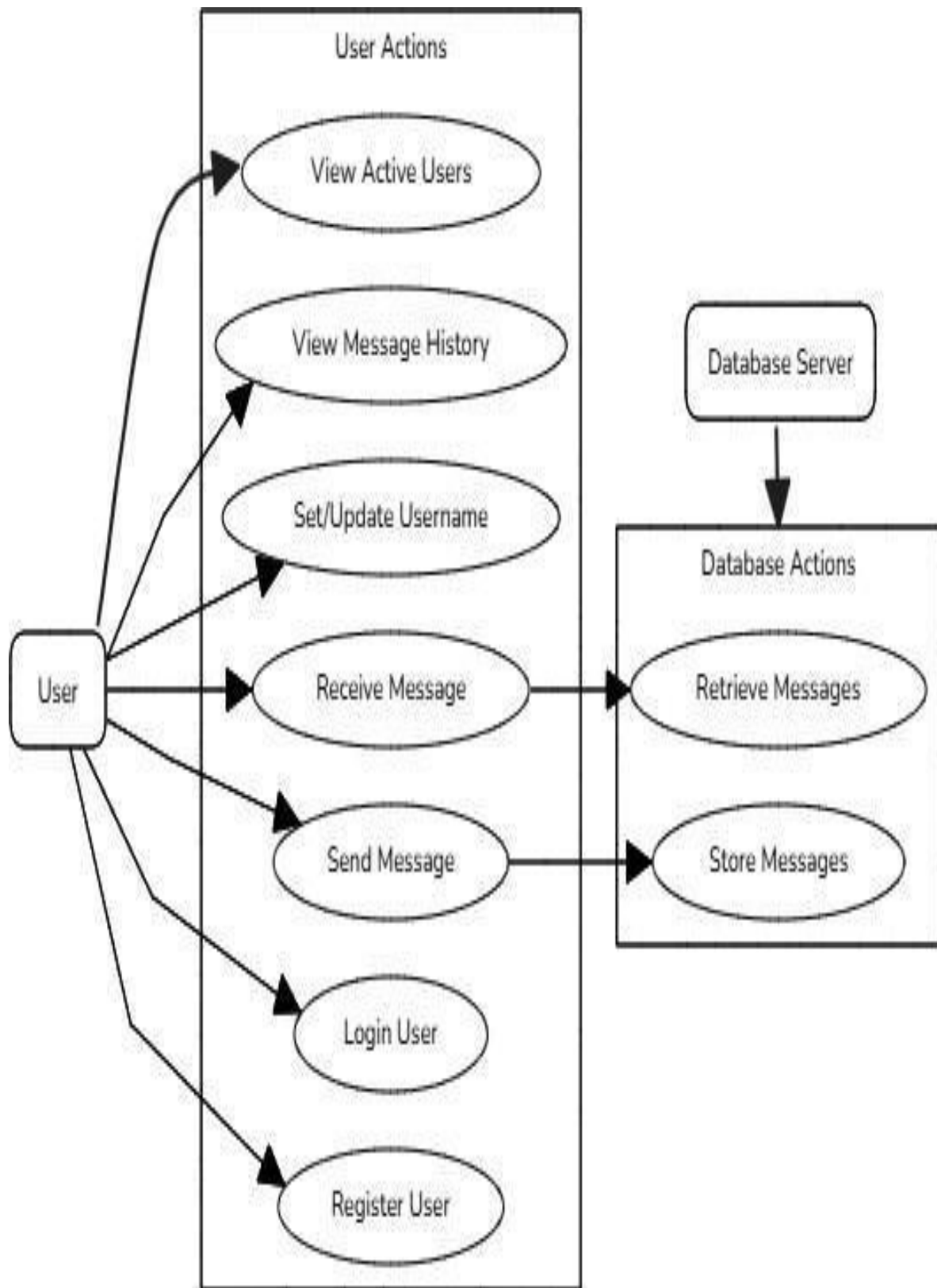
4.7 Project Module

The IndusComm real-time messaging application consists of several modules, each serving a specific function to ensure effective communication and system reliability:

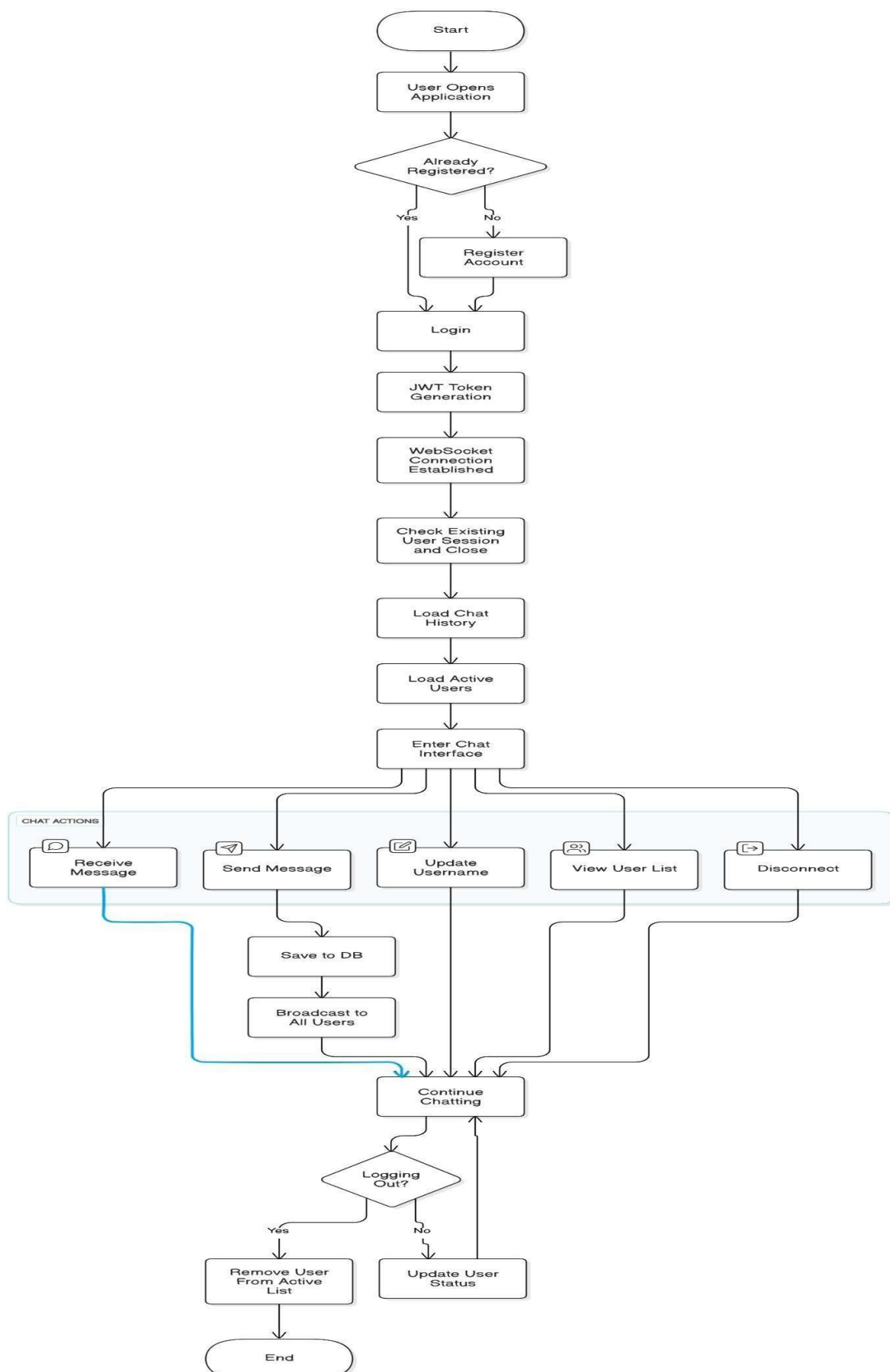
- ❖ **Authentication Module:** Handles user registration, login, and session management using JWT-based authentication. This module ensures secure access to the messaging platform and maintains user identity across sessions, preventing unauthorized access while facilitating seamless reconnection.
- ❖ **Real-Time Communication Module:** Leverages Socket.IO to establish and maintain WebSocket connections between clients and server. This core module manages real-time message transmission, user presence updates, and ensures immediate delivery of all communication across connected clients.
- ❖ **User Management Module:** Tracks active users, manages user sessions across multiple devices, and handles user status updates. This module maintains the roster of online participants, prevents duplicate sessions, and broadcasts user join/leave events to all connected clients.
- ❖ **Message Persistence Module:** Stores and retrieves messages using MongoDB, maintaining a comprehensive history of communications. This module handles database operations for saving new messages and fetching historical conversations for users joining the conversation.
- ❖ **Server Information Module:** Provides network configuration details to clients, enabling local network discovery and connection sharing. This module detects and broadcasts the server's IP address and port, facilitating easy access for users on the same network.
- ❖ **Error Handling Module:** Manages exceptions, connection failures, and operational errors throughout the application. This module implements recovery mechanisms, graceful degradation when services are unavailable, and user notification about system status.
- ❖ **API Module:** Exposes RESTful endpoints for authentication and message retrieval operations. This module provides HTTP interfaces for operations that don't require real-time capabilities, complementing the WebSocket-based communication channels.
- ❖ **Static Content Module:** Serves the web-based client interface and associated resources. This module delivers the HTML, CSS, JavaScript, and media assets that constitute the user interface for the messaging application.

5. System Design

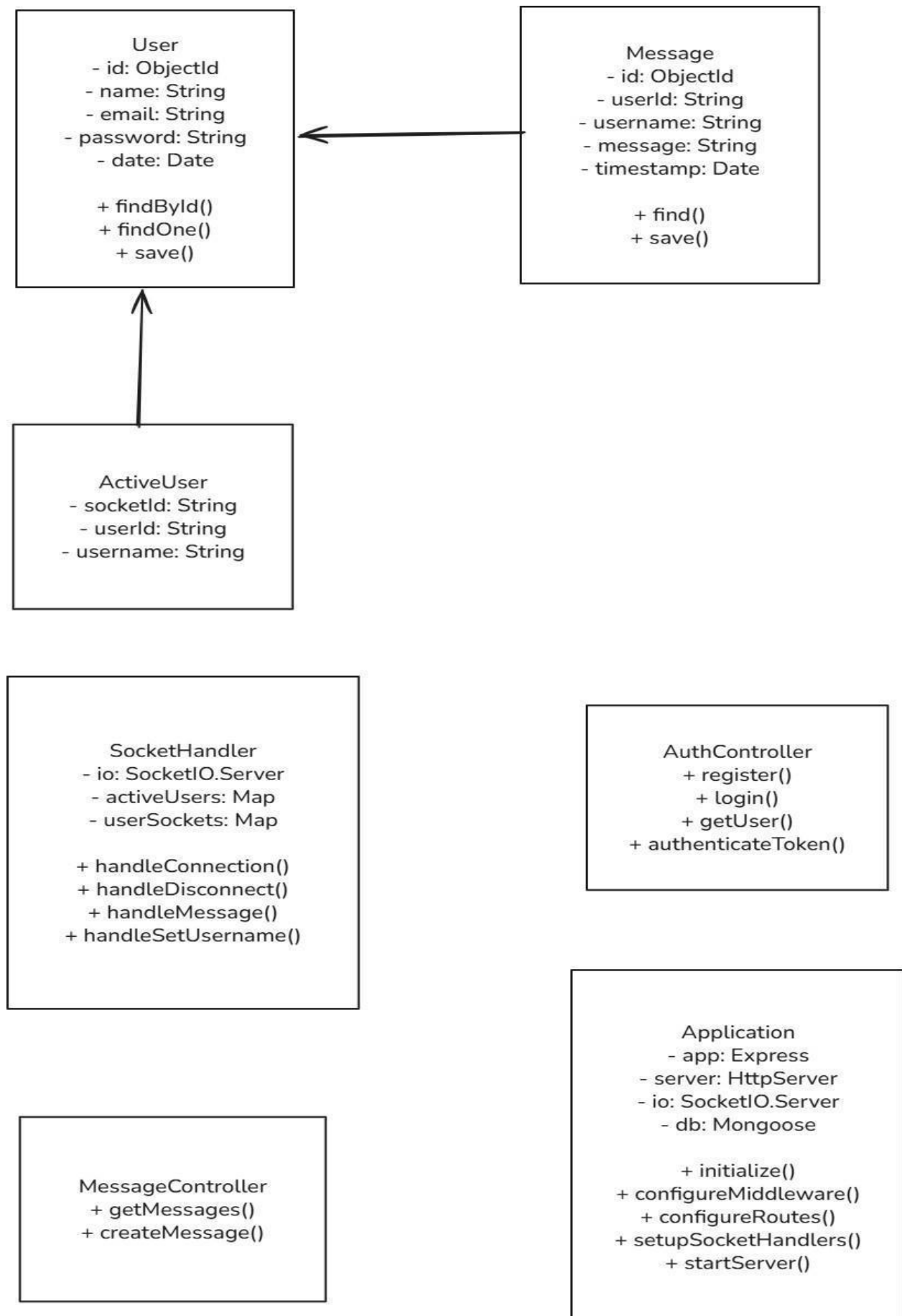
5.1 Use Case Diagram



5.2 Activity Diagram

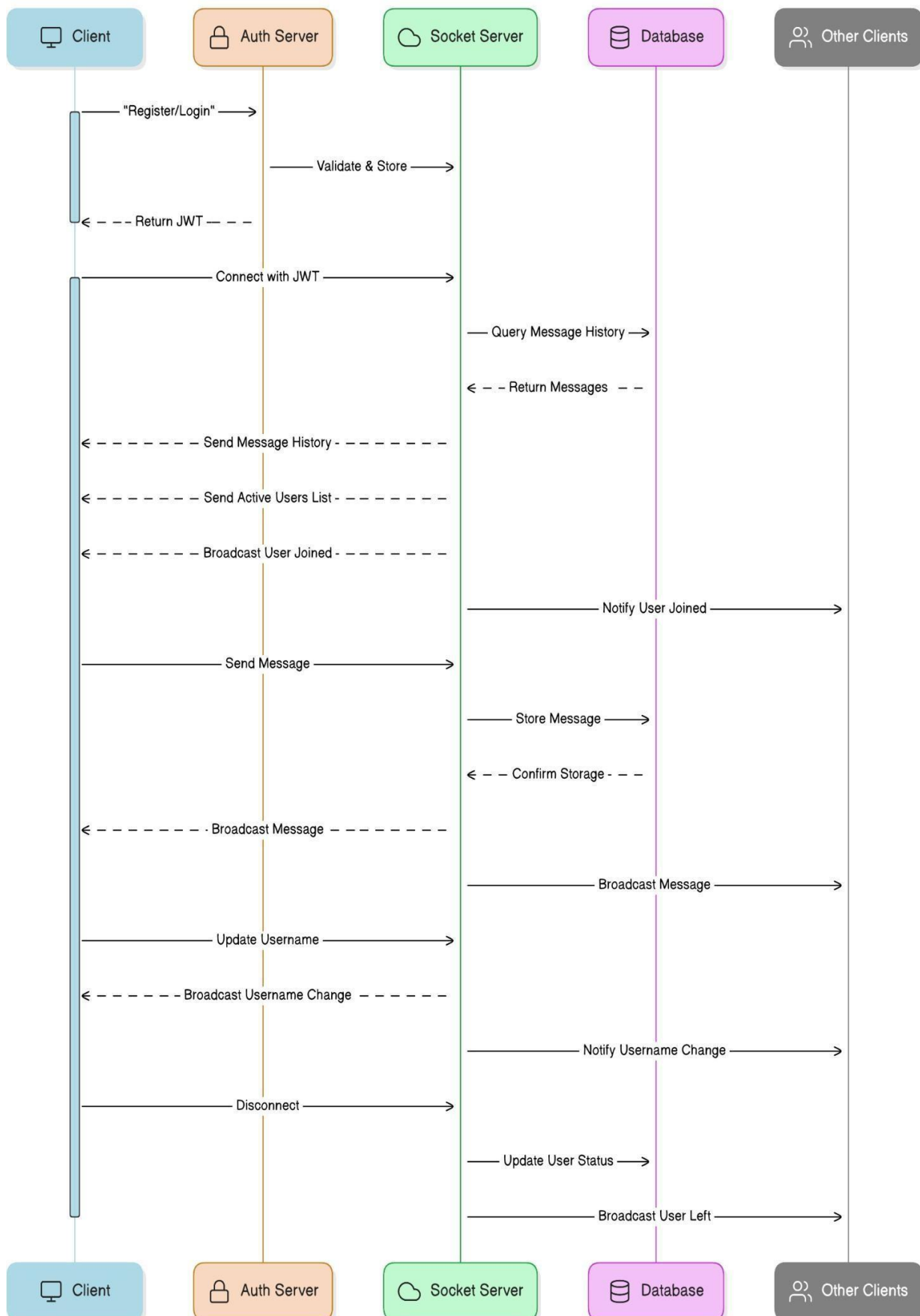


5.3 Class Diagram



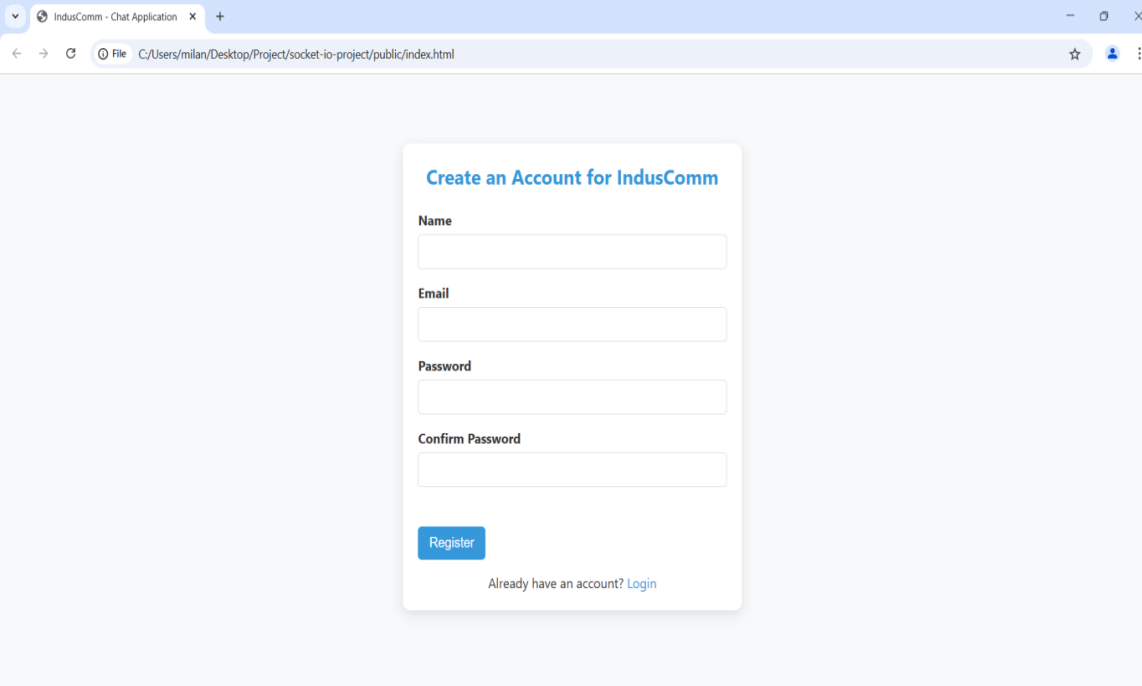
5.4 Sequence Diagram

Chat Application Sequence



5.5 Screenshots

1. Register



The screenshot displays a web browser window with a single tab titled 'IndusComm - Chat Application'. The address bar shows the file path 'C:/Users/milan/Desktop/Project/socket-io-project/public/index.html'. The main content area features a registration form with the heading 'Create an Account for IndusComm'. The form contains four input fields: 'Name', 'Email', 'Password', and 'Confirm Password'. Below these fields is a blue 'Register' button. At the bottom of the form, there is a link that reads 'Already have an account? Login'.

Welcome to the IndusComm registration page, where you begin your journey towards seamless real-time communication. Our streamlined registration process ensures quick and secure account creation, enabling you to connect with team members instantly through our robust messaging platform.

Begin by entering your essential details: username, email address, and a secure password. The intuitive form validates your inputs in real-time, ensuring all required information meets our security standards while maintaining a straightforward user experience.

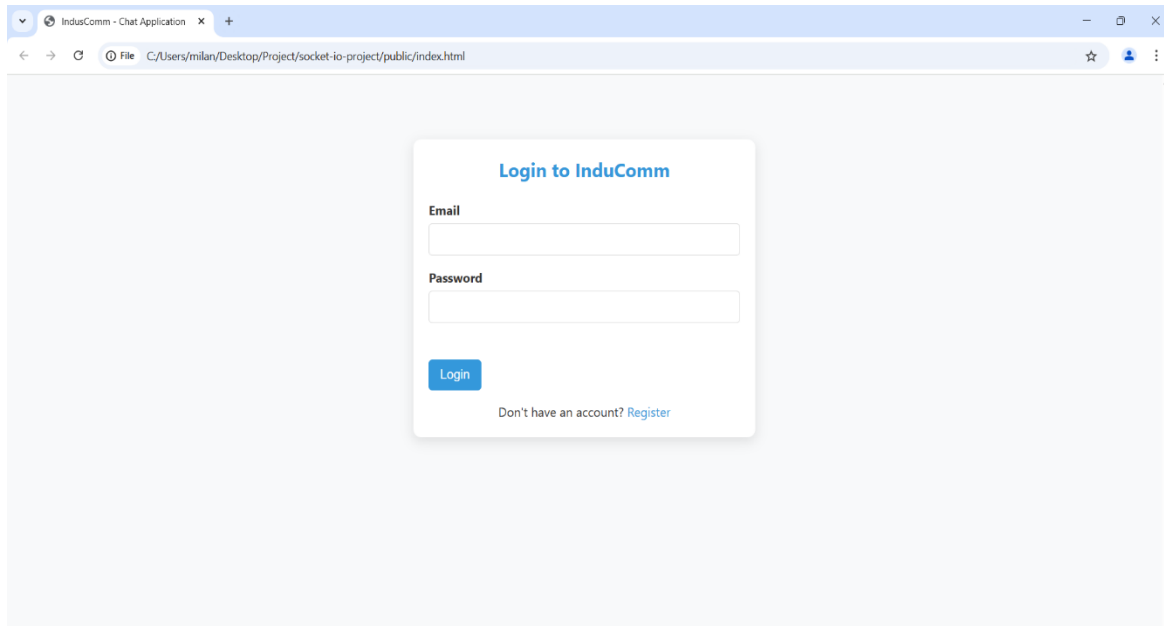
Our password field incorporates a robust strength indicator, guaranteeing the security of your account while providing guidance on creating strong credentials. The registration form is designed with simplicity in mind, allowing new users to complete the process quickly and without unnecessary complexity.

For added security, IndusComm implements email verification to confirm account ownership, preventing unauthorized access and ensuring the integrity of your communications.

Join the IndusComm network today and experience the power of secure, real-time messaging designed for modern teams. Connect, collaborate, and communicate with confidence in a platform built for today's fast-paced digital environment.

Experience the future of team communication with IndusComm – where real-time connectivity meets enterprise-grade security.

2. Login



Welcome back to IndusComm! Access your secure messaging environment with ease through our streamlined login interface. Whether you're returning to continue team conversations or joining new discussions, our intuitive design ensures a frictionless authentication experience.

Simply enter your registered email address and password to access your IndusComm account. The clean, focused layout eliminates distractions and guides you directly to your communication workspace with minimal effort.

Our login form features real-time validation, providing immediate feedback as you enter your credentials. This helps prevent common errors and ensures a smooth authentication process, getting you connected to your team faster.

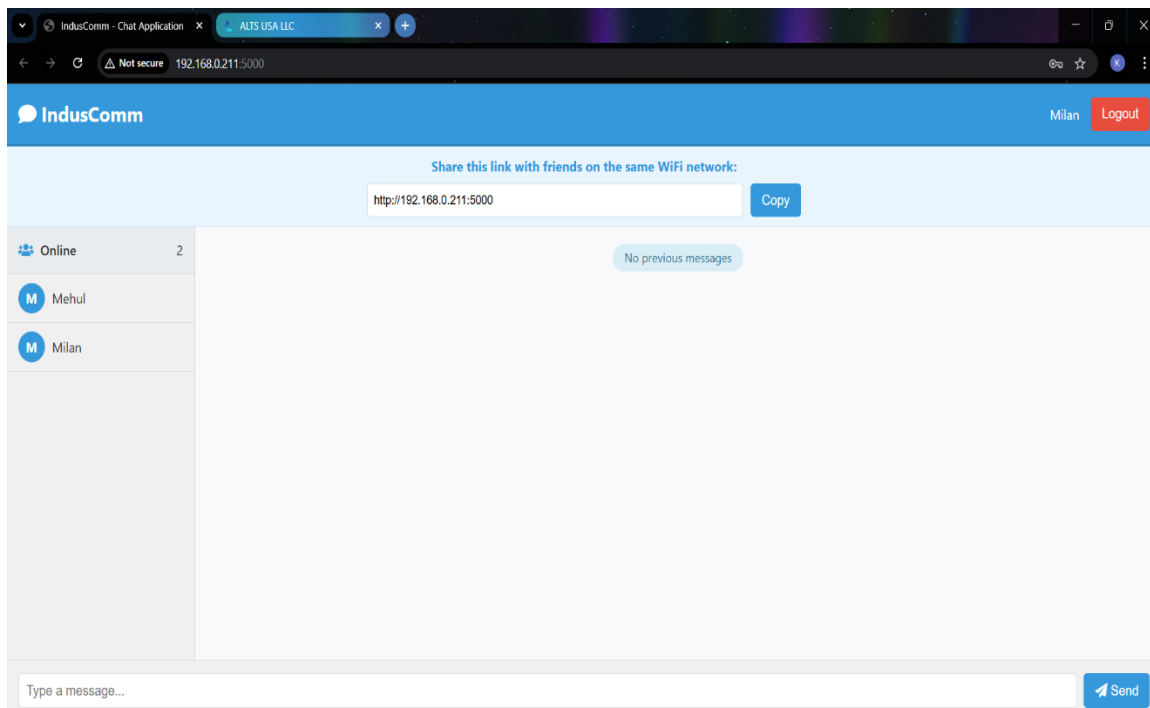
Forgot your password? No worries – IndusComm's account recovery feature makes it simple to reset your credentials securely through your verified email address, maintaining the security of your communications while providing convenient access recovery.

IndusComm's commitment to security means your login credentials are protected with industry-standard encryption and JWT-based authentication. Experience peace of mind knowing that your conversations and data remain private and secure within the IndusComm platform.

The responsive design ensures a consistent login experience across all devices, from desktop workstations to mobile phones, allowing you to stay connected wherever you work.

Connect with your team instantly through IndusComm – your gateway to efficient, real-time communication in today's collaborative digital workspace.

3. Dashboard



Welcome to the IndusComm chat dashboard, the central hub for real-time team communications. This carefully designed interface combines functionality with clarity, providing an intuitive workspace for productive conversations with your team.

The main interface is thoughtfully organized into three functional zones:

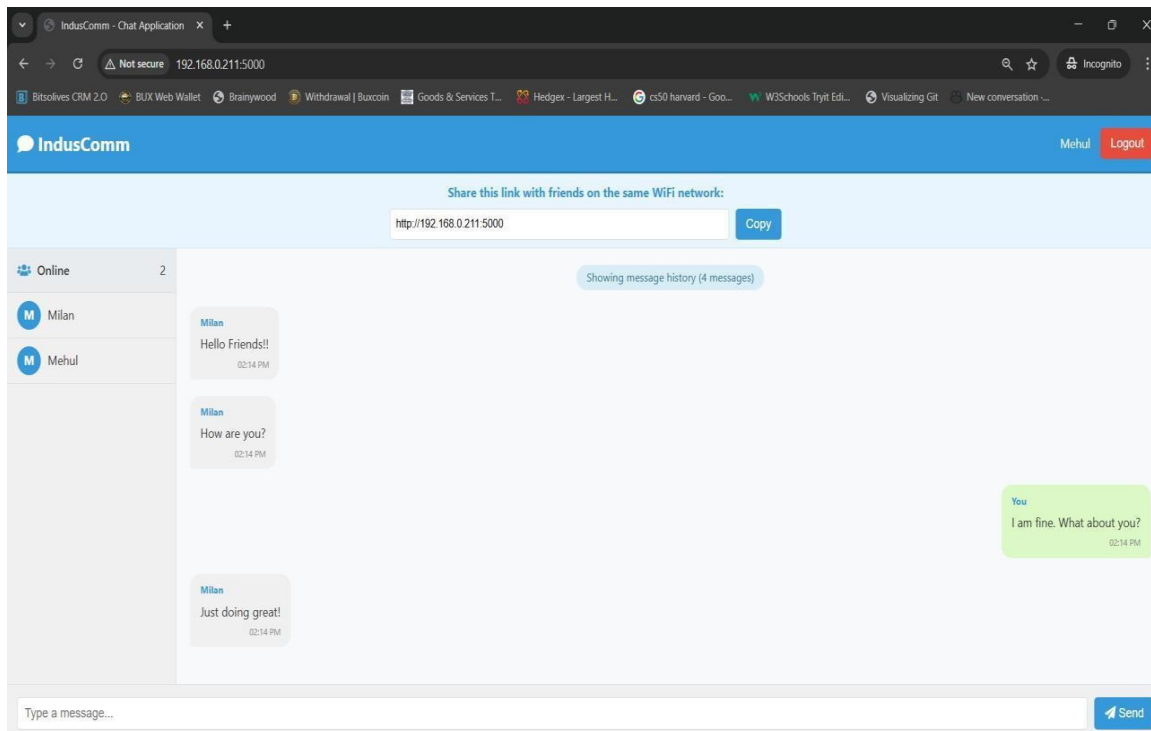
The **user sidebar** provides instant visibility of your team's online status, with a clear counter showing the total number of active participants. Each connected user appears in the roster below, giving you immediate awareness of who's available for collaboration. This real-time presence information eliminates the uncertainty of wondering which team members are currently accessible.

The **message display area** constitutes the central and largest portion of the interface, where conversations unfold chronologically in real-time. Each message includes clear attribution, timestamps, and visual styling to distinguish between your messages and those from colleagues. The scrollable container maintains a complete conversation history while keeping recent communications in focus.

The **message composition area** features a streamlined input field and send button at the bottom of the screen, allowing you to craft and dispatch messages efficiently. The intuitive paper plane icon reinforces the sending action, making the process visually intuitive.

IndusComm's responsive design ensures this dashboard maintains its functionality across all device sizes, from desktop workstations to mobile phones, allowing you to stay connected regardless of your working environment.

4. Historical Messages



Welcome to the IndusComm message history, your comprehensive record of team communications. This thoughtfully designed feature ensures no important information is lost and provides context for both ongoing and new conversations.

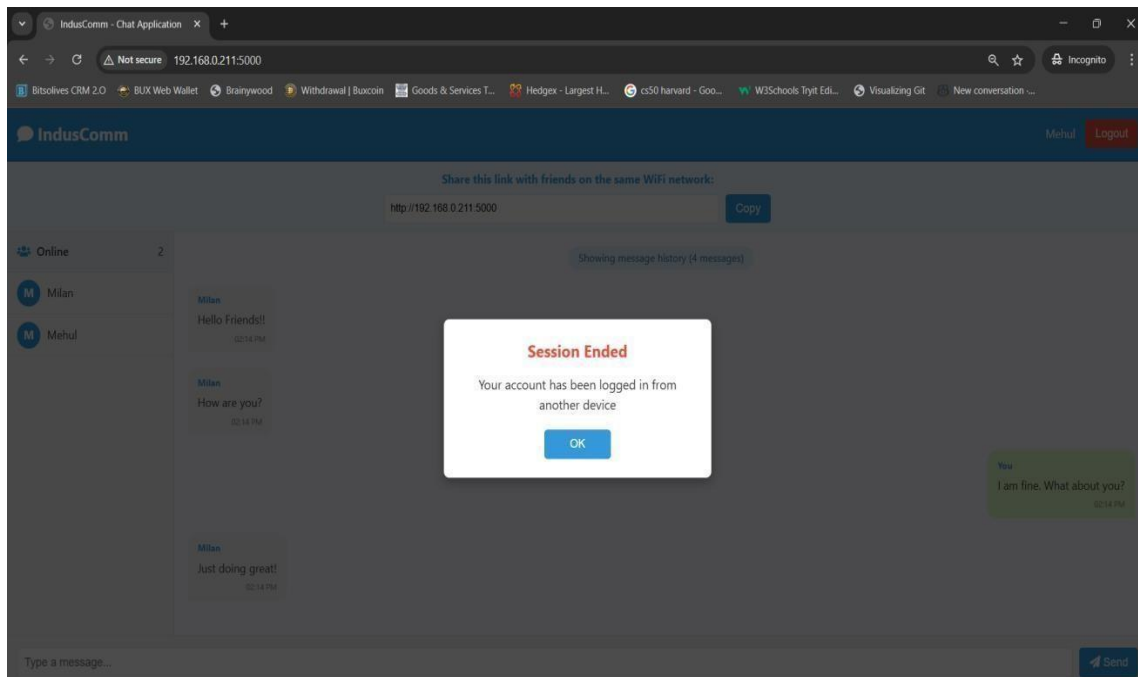
The message history displays in the central chat panel, presenting a chronological record of all communications within your workspace. Each message is clearly attributed to its sender with consistent visual styling that distinguishes between your messages and those from teammates, making conversation flow easy to follow.

Messages are displayed with precise timestamps, allowing you to track when information was shared and maintaining an accurate timeline of your team's communication. This chronological arrangement helps establish context and continuity for discussions that span multiple sessions.

When you first log in or reconnect to IndusComm, the system automatically retrieves your recent message history from secure database storage, ensuring you never miss important information shared while you were offline. The seamless loading process brings you up to speed instantly, eliminating the need to ask colleagues for updates.

The message display area features smooth scrolling functionality, allowing you to navigate through extensive conversation history with ease. As new messages arrive in real-time, they appear at the bottom of the panel, while older messages remain accessible with a simple scroll upward. Experience communication continuity with IndusComm's message history – where your team's valuable exchanges are preserved in a clear, accessible format that enhances collaboration and information retention.

5. Session Timed Out



Welcome to IndusComm's multi-session management notification, a crucial security feature designed to protect your account integrity while providing clear information about your session status. This carefully implemented alert ensures your communications remain secure by preventing simultaneous access from multiple locations.

When you attempt to log in while having an active session elsewhere, a prominent notification appears in the center of your screen. The message is presented in a clean, focused dialog box that clearly communicates the situation without creating unnecessary alarm.

The notification informs you that your account is currently active in another session, explaining that your previous session has been terminated to maintain security. The message is written in straightforward language that explains the policy of allowing only one active session per account at any time.

A single "OK" button accompanies the notification, acknowledging that you understand the previous session has been closed and that you can now proceed with your current session. This streamlined approach maintains strong security while providing a simple, friction-free user experience.

6. Agile Documentation

6.1 Project Charter

- ❖ **Vision** : Create IndusComm, a secure real-time messaging application that revolutionizes team communication through instant connectivity and reliable message delivery.
- ❖ **Objectives** :
 - Develop an intuitive messaging platform with high user satisfaction rates.
 - Ensure cross-browser compatibility and robust security measures.
 - Implement real-time message delivery with minimal latency.
 - Create a scalable architecture supporting both small teams and larger organizations.
- ❖ **Scope** : Core features include Socket.IO WebSocket integration, MongoDB message persistence, JWT authentication, active user tracking, session management, and responsive web interface.
- ❖ **Stakeholders** : Development Team, Project Manager, End Users (Businesses, Remote Teams, Educational Institutions), System Administrators.
- ❖ **Requirements** : User registration and authentication system, real-time message broadcasting, persistent message storage, active user tracking, session management across devices, responsive interface design.
- ❖ **Constraints** : Browser compatibility with WebSocket technology, reliance on stable internet connectivity, proper server configuration for optimal performance.
- ❖ **Risks** : Scaling challenges with high user counts, potential security vulnerabilities in authentication implementation, performance degradation with large message history.
- ❖ **Success Criteria** : Successful message delivery with <500ms latency, positive user feedback on interface usability, reliable message persistence across sessions, successful deployment in varied network environments.

This Project Charter guides the development of IndusComm, ensuring alignment with real-time communication goals, stakeholder needs, and modern web application principles while maintaining focus on security and reliability.

6.2 Roadmap

❖ Phase 1 : Core Development

- **Sprint 1** : Set up Node.js environment, implement Express server, define MongoDB schema
- **Sprint 2** : Build authentication system with JWT implementation and user registration/login
- **Sprint 3** : Implement Socket.IO integration for real-time communication
- **Sprint 4** : Develop message persistence with MongoDB and user tracking system

❖ Phase 2 : Interface and User Experience

- **Sprint 5** : Design and implement responsive frontend interface
- **Sprint 6** : Develop active user display and session management
- **Sprint 7** : Implement message history loading and display
- **Sprint 8** : Add user notification system and connection status indicators

❖ Phase 3 : Testing and Optimization

- **Sprint 9** : Conduct security testing and implement vulnerability fixes
- **Sprint 10** : Performance optimization for message delivery and history loading
- **Sprint 11** : Cross-browser compatibility testing and fixes
- **Sprint 12** : Load testing and scaling optimizations

❖ Phase 4 : Deployment and Feedback

- **Sprint 13**: Beta deployment and initial user testing
- **Sprint 14**: Address feedback from beta testing
- **Sprint 15**: Production deployment and documentation
- **Sprint 16+**: Continuous improvements based on user feedback

❖ Key Milestones :

- Working prototype with basic messaging (end of Phase 1)
- Feature-complete beta version (end of Phase 2)
- Production-ready version 1.0 (end of Phase 3)
- Continuous enhancement releases (Phase 4 and beyond)

❖ Dependencies :

- Node.js and Express implementation for server foundation
- MongoDB availability for message and user data persistence
- Socket.IO integration for real-time capabilities
- Browser compatibility with WebSocket technology

This roadmap outlines the development trajectory for IndusComm, ensuring a structured approach to building, testing, and deploying the real-time messaging system with clear milestones and responsibilities.

6.3 User Story

Title : As a user, I want to communicate in real-time with my team members securely.

Story : As a team member, I want to exchange messages instantly with my colleagues through IndusComm, regardless of where I'm working from. I need to see who's currently online and available for communication, and I want my messages to be delivered immediately without having to refresh the page. When I'm away and return, I should be able to see the messages I missed, maintaining continuity in our conversations. I expect the interface to be intuitive and responsive, working well on both my office computer and mobile device when I'm on the go.

Acceptance Criteria :

- ❖ **Real-time Messaging** : Messages should appear in the chat window for all connected users immediately after being sent, without requiring page refresh.
- ❖ **User Presence** : The system should display a list of currently active users and update this list in real-time when users connect or disconnect.
- ❖ **Message Persistence** : Previous messages should be loaded automatically when joining a conversation, allowing users to catch up on communications they missed.
- ❖ **Cross-Device Compatibility** : The interface should function properly across desktop browsers and mobile devices, adapting to different screen sizes.
- ❖ **Secure Authentication** : Users should be able to register, login securely, and maintain their session across page reloads.
- ❖ **Session Management** : The system should prevent concurrent logins from multiple devices, ensuring account security.

6.4 Release Plan

Release 1.0: Core Features :

- ❖ **Features:** Authentication system, real-time messaging, active user tracking, message persistence, responsive interface, session management.
- ❖ **Testing :** Security testing, cross-browser compatibility, real-time delivery verification, basic load testing.
- ❖ **Deployment :** Internal beta release, deployment guide for local networks.
- ❖ **Success Criteria :** Message delivery within 500ms, successful authentication flow, accurate user tracking, interface compatibility across browsers.

Release 2.0: Enhanced Features :

- ❖ **Features :** Extended message history, read receipts, typing indicators, file sharing, enhanced security (2FA), message search, private messaging.
- ❖ **Testing :** Stress testing with larger groups, security penetration testing, file transfer reliability.
- ❖ **Deployment :** Phased rollout, database migration support, enhanced monitoring.
- ❖ **Success Criteria :** Maintained performance under load, high file transfer success rate, improved user satisfaction metrics.

6.5 Sprint Backlog

Sprint 1: Authentication System :

- ★ Develop user login functionality with JWT
- ★ Implement user registration process with form validation
- ★ Create secure password storage with bcrypt
- ★ Design responsive authentication interfaces
- ★ Implement session persistence across page refreshes

Sprint 2: Real-Time Connection :

- ★ Set up Socket.IO server configuration
- ★ Implement WebSocket connection with authentication
- ★ Create user connection/disconnection handlers
- ★ Develop active user tracking system
- ★ Design real-time connection status indicators

Sprint 3: Messaging Core :

- ★ Create message sending functionality
- ★ Implement real-time message broadcasting
- ★ Develop message display with proper formatting
- ★ Design message input interface with submission handling
- ★ Implement timestamp display for messages

Sprint 4: Message Persistence :

- ★ Create MongoDB schema for message storage
- ★ Implement database operations for message saving
- ★ Develop message history loading on connection
- ★ Design scrollable message container for history
- ★ Add message retrieval optimization

Sprint 5: User Experience :

- ★ Implement user list display with online status
- ★ Create username updating functionality
- ★ Develop session management across multiple devices
- ★ Design responsive layout for various screen sizes
- ★ Implement user feedback for connection status

Note : Each sprint will include development, testing, and refinement phases to ensure quality and functionality. Daily stand-ups will track progress, and sprint reviews will evaluate completed features.

6.6 Agile Test Plan

Objective :

The objective of this Agile Test Plan is to ensure the quality, security, and reliability of IndusComm through iterative testing and validation of its real-time communication features and functionality.

Testing Approach :

- ❑ **Iterative Testing** : Testing will be conducted in short iterations aligned with development sprints to continuously verify the evolving application.
- ❑ **User Feedback** : Gather feedback from pilot users during each iteration to identify usability issues, performance bottlenecks, and areas for improvement.
- ❑ **Socket Testing** : Utilize specialized WebSocket testing tools to verify real-time messaging capabilities and connection handling.
- ❑ **Cross-Browser Testing** : Test IndusComm on various browsers and devices to ensure consistent behavior and responsiveness.
- ❑ **Security Testing** : Conduct thorough security testing to identify and address vulnerabilities in authentication and data transmission.

Test Strategy :

- ❑ **Unit Testing** : Developers will write unit tests for individual components including authentication, message handling, and user management functions.
- ❑ **Integration Testing** : Test the integration between Socket.IO, Express, and MongoDB to verify seamless data flow across components.
- ❑ **Real-Time Testing** : Validate message delivery, active user updates, and session management under various network conditions.
- ❑ **User Acceptance Testing (UAT)** : Involve end-users to validate the application meets requirements for intuitive real-time communication.

Test Cases :

- ❑ **Authentication** : Verify user registration, login, JWT validation, and session management work correctly.
- ❑ **Real-Time Messaging** : Test message sending, receiving, and persistence under normal and degraded network conditions.
- ❑ **Active User Tracking** : Verify correct display of online users and real-time updates when users connect or disconnect.
- ❑ **Session Management** : Test handling of multiple login attempts, session timeouts, and browser refresh scenarios.
- ❑ **Message History** : Verify correct loading and display of historical messages when users join conversations.

Reporting :

- ❑ **Sprint Demos** : Conduct demonstrations at the end of each sprint to showcase completed features and gather feedback.
- ❑ **Defect Tracking** : Use GitHub Issues to log and track bugs identified during testing phases.
- ❑ **Test Documentation** : Maintain comprehensive documentation of test scenarios, results, and performance metrics.
- ❑ **Security Audit Reports** : Document findings from security testing with remediation steps.

Roles and Responsibilities :

- ❑ **Development Team** : Responsible for writing unit tests and fixing identified issues.
- ❑ **QA Tester** : Responsible for designing test cases, conducting integration and system testing.
- ❑ **Security Analyst** : Responsible for performing security assessments and vulnerability testing.

7. Future Enhancements

As IndusComm evolves, several enhancements are envisioned to further enrich the user experience and expand its capabilities:

- **Rich Media Support:** Implement file sharing capabilities allowing users to exchange documents, images, and other media files directly within conversations, eliminating the need for external sharing platforms.
- **End-to-End Encryption:** Enhance security by implementing end-to-end encryption for all messages, ensuring that only intended recipients can access sensitive communications and protecting user privacy.
- **Channel/Group Messaging:** Introduce structured conversation channels allowing teams to organize communications by topic, project, or department, improving information organization and accessibility.
- **Message Threading:** Implement conversation threading capabilities that allow users to create focused sub-conversations within the main message flow, keeping related responses organized.
- **Mobile Application:** Develop native mobile applications for iOS and Android, providing optimized experiences for mobile users with push notifications and offline message queueing.
- **API Integration:** Create a robust API system allowing integration with project management tools, calendar applications, and other productivity platforms to streamline workflow between systems.
- **Advanced User Permissions:** Implement role-based access controls allowing organizations to designate administrators, moderators, and users with varying levels of system privileges.
- **Message Search:** Introduce comprehensive search functionality with filters for dates, users, and content, enabling users to quickly locate specific information in conversation history.
- **Typing Indicators:** Add real-time typing notifications showing when other users are composing messages, improving conversation flow and reducing duplicate responses.
- **Read Receipts:** Implement message status indicators showing when messages have been delivered and read by recipients, providing assurance of communication receipt.

8. Bibliography & References

Node.js : <https://nodejs.org/>

Socket.IO : <https://socket.io/>

GitHub : <https://github.com/>

Stack Overflow: <https://stackoverflow.com/>

Copilot : <https://github.com/features/copilot>

ChatGPT : <https://chat.openai.com/>

YouTube : <https://www.youtube.com/>

References:

- ❖ **Node.js** provided the foundation for building the server-side application with its event-driven, non-blocking I/O model.
- ❖ **Socket.IO** was instrumental in implementing real-time, bidirectional communication between web clients and the server.
- ❖ **GitHub** facilitated version control and collaborative development throughout the project lifecycle.
- ❖ **Stack Overflow** was consulted for technical guidance and problem-solving during development challenges.
- ❖ **GitHub Copilot** assisted with code generation and suggestions during the development process, improving productivity.
- ❖ **ChatGPT** provided guidance on architectural decisions and helped troubleshoot implementation challenges.
- ❖ **YouTube** offered tutorials and educational content on Socket.IO implementation and real-time web application development.

9. Suggestions & Feedback

[illegible]