# Node.js Mini Project Report

Project Title: Real-Time Chat Application using Node.js and Socket.io

Submitted by: Igwegbe Emmanuel Chibueze

Course: EPE 411

Date: May 2025

## 1. Project Description

This project is a real-time chat application built using Node.js, Express.js, and Socket.io. The app allows multiple users to connect and exchange messages in real-time through a web interface. It serves as a demonstration of Node.js’s asynchronous, event-driven capabilities and real-time communication via WebSockets.

## 2. Objectives

- To build a scalable web application using Node.js.  
- To implement real-time two-way communication with Socket.io.  
- To understand the asynchronous behavior of Node.js.  
- To practically demonstrate Node.js's use in modern web development.

## 3. Technologies Used

|  |  |
| --- | --- |
| Technology | Purpose |
| Node.js | Server-side JavaScript runtime |
| Express.js | Simplifies HTTP server creation and routing |
| Socket.io | Real-time, bidirectional communication |
| HTML/CSS | Frontend user interface |
| JavaScript | Client-side interactivity |

## 4. Features

- Real-time message broadcasting to all connected clients.  
- Simple user interface for sending and receiving messages.  
- Uses WebSockets for low-latency communication.  
- Scalable server setup with Express and Socket.io.

## 5. How It Works

1. When a user visits the site, they are served an HTML page by Express.  
2. A WebSocket connection is opened between the client and the server.  
3. When a user sends a message:  
 - The client emits a `chat message` event.  
 - The server listens for the event and broadcasts it to all clients.  
4. All clients receive the message in real-time without reloading the page.

## 6. How to Run

1. Install Node.js and npm.  
2. Extract the project zip and open terminal inside the folder.  
3. Run:  
 npm install express socket.io  
 node server.js  
4. Visit http://localhost:3000 in your browser.

## 7. Scalability Potential

- User authentication  
- Chat rooms  
- Storing chat history in a database  
- Mobile responsive design  
- Deployment to cloud platforms

## 8. Conclusion

This real-time chat application demonstrates the power of Node.js and Socket.io for building modern, real-time web applications. It highlights the simplicity and efficiency of asynchronous I/O and event-driven programming, core to Node.js.