A **mini chat website** is a perfect project to practice the basics of **Node.js**, **MongoDB**, and **Vue.js**. Here's a suggested roadmap to help you achieve your goal with a small but complete real-time chat app:

## 🧠 ****Core Features for Your Mini Chat Project****

### ✅ Backend (Node.js + Express + MongoDB):

* User registration and login (simple auth)
* Store chat messages in MongoDB
* Use **Socket.IO** for real-time messaging
* API to get previous messages (chat history)

### ✅ Frontend (Vue.js):

* Register/Login UI
* Chat interface (message input + chat history)
* Real-time message updates using **Socket.IO client**

## 📁 Suggested Folder Structure

chat-app/

├── backend/ # Node.js backend

│ ├── server.js

│ ├── routes/

│ ├── models/

│ └── controllers/

├── frontend/ # Vue.js frontend (Vite or Vue CLI)

│ ├── src/

│ └── public/

└── README.md

## ⚙️ Key Technologies

| **Layer** | **Stack** |
| --- | --- |
| Frontend | Vue 3, Socket.IO client, Axios |
| Backend | Node.js, Express, Socket.IO, Mongoose |
| Database | MongoDB (local or MongoDB Atlas) |

## 🗓️ Project Plan (Milestones)

### Day 1–2: Setup Project

* Setup backend (Node + Express)
* Setup MongoDB and test connection
* Setup Vue project

### Day 3–4: Basic User System

* User model (username only or simple password)
* Registration/Login routes
* JWT or session auth (optional for simple project)

### Day 5–6: Real-time Messaging

* Integrate Socket.IO (send/receive messages)
* Save messages in MongoDB
* Fetch message history on page load

### Day 7: Final Touches

* Improve UI (basic styles)
* Display usernames and timestamps
* Test and deploy (optional)

Would you like a starter template or code snippet for the backend or frontend setup?