**AI Chat Interface**

A sophisticated React-based chat interface that supports both text and speech input, featuring a modern UI design and integration with AI capabilities through the Cohere API.

**Features**

* 🎙️ Speech-to-text input support
* 💬 Real-time AI responses via Cohere API
* 🌓 Light/Dark theme toggle
* ⚡ Quick suggestion buttons
* 🔄 Real-time typing indicators
* 📱 Responsive design

**Project Structure**

chat-interface/

├── src/

│ ├── assets/

│ ├── components/

│ │ ├── chat/

│ │ │ ├── ChatContainer.tsx

│ │ │ ├── InputArea.tsx

│ │ │ ├── MessageBubble.tsx

│ │ │ └── MessageThread.tsx

│ │ ├── settings/

│ │ │ └── ThemeToggle.tsx

│ │ ├── context/

│ │ │ └── ChatContext.tsx

│ │ ├── services/

│ │ │ └── api.ts

│ │ └── types/

│ │ └── chat.ts

│ ├── App.css

│ ├── App.tsx

│ ├── index.css

│ └── main.tsx

**Technical Implementation**

**Core Components**

**ChatContainer**

The root component that orchestrates the chat interface, containing:

* Header with branding and theme toggle
* Message thread display
* Input area for user interactions

**InputArea**

Handles user input through multiple channels:

* Text input with Enter key support
* Voice input using the Web Speech API
* Quick suggestion buttons
* Real-time typing indicators
* Message sending functionality

**MessageThread & MessageBubble**

* **MessageThread**: Renders the chat history
* **MessageBubble**: Individual message component with:
  + Type-based styling (user/AI)
  + Source indicator (text/speech)
  + Timestamp display

**State Management**

The application uses React Context API through ChatContext.tsx to manage global state:

interface ChatState {

messages: Message[];

isRecording: boolean;

isProcessing: boolean;

settings: {

theme: "light" | "dark";

language: string;

speechEnabled: boolean;

};

}

**Actions supported:**

* **ADD\_MESSAGE**: Append new messages
* **SET\_THEME**: Toggle light/dark theme
* **SET\_PROCESSING**: Update processing state

**API Integration**

The application integrates with Cohere's AI API through api.ts:

* Handles message sending and receiving
* Error handling and response formatting
* Environment-based configuration

**Speech Recognition**

Implements browser-native speech recognition:

const SpeechRecognition =

window.SpeechRecognition || window.webkitSpeechRecognition;

**Features:**

* Real-time transcription
* Visual feedback during recording
* Error handling for unsupported browsers

**Document submitted by Amsha J**