■ ChatApp — Technical Documentation

Overview

ChatApp is a single-page web application built using React.js. It allows a user to send messages and receive simulated chatbot responses. The app demonstrates the fundamentals of React components, state management, and UI updates based on user interaction.

Technologies Used

1. Frontend Framework: React.js (Vite setup) — Component-based UI development
2. Styling: CSS (modular component styles) — Layout, colors, and responsive design
3. Language: JavaScript (ES6+) — Logic and event handling
4. Assets: PNG & GIF — User and bot avatars, loading animation

Architecture Overview

The app follows a component-based architecture, separating logic into reusable UI elements. App.jsx acts as the root controller, managing state and combining all UI parts.

Component Hierarchy:

App.jsx → ChatMessages.jsx → ChatMessage.jsx and ChatInput.jsx

Functional Flow

1. User types a message in ChatInput and sends it to App.jsx via props.
2. App.jsx stores messages in React state and triggers re-renders.
3. Bot response simulated with loading spinner and delayed reply.
4. ChatMessages dynamically renders all messages, styled differently for user/bot.

UI and Design

1. Minimalist layout with title, chat area, and input bar.
2. CSS modules for scoped and organized styles.
3. Smooth transitions for message updates.
4. Flexbox or grid layout for message alignment.

Core React Concepts Demonstrated

1. useState for message array and input handling.
2. Props drilling for passing data/functions between components.
3. Conditional rendering for loading and message types.

4 Reusability through modular component structure.

Future Scope

1 Integrate with real AI APIs (OpenAI, HuggingFace, Dialogflow).
2 Implement WebSocket or Socket.IO for real-time chat.
3 Add authentication for personalized chat sessions.
4 Store chat history in localStorage or backend database.