**Lab Evaluation -1**

**RollNo**: CB.SC.U4CSE23058

**Name**: AB JASHWANTH REDDY

**About the use case :**

The **"ASD Communication Assistant"** is a specialized web application designed to support children and individuals with Autism Spectrum Disorder (ASD) in navigating digital conversations. Children with ASD often struggle with interpreting implicit social cues, sarcasm, emotional undertones, and potential safety risks (like scams or bullying) in text-based chats. This application acts as an intelligent "**sidekick**" overlay on a simulated WhatsApp Web interface. When a user receives a confusing message, they can trigger an instant AI analysis that translates complex social language into literal, simple explanations, identifies the sender's emotion (e.g., "Sad" or "Angry"), and provides safe, context-appropriate reply suggestions.

**Purpose:**

**How this application will improve or provide comfort for autism kids?**

This application reduces social anxiety and cognitive load by removing the guesswork from digital communication. By providing concrete, literal interpretations of abstract concepts (like "lol" used passively-aggressively), it prevents misunderstandings that lead to social isolation. Furthermore, the built-in "Safety Guard" feature protects vulnerable users from online manipulation and cyberbullying by explicitly flagging risky messages (e.g., password requests) and advising them not to share personal info, fostering independence and digital safety.

**List of similar applications:**

While there are many apps for ASD, most focus on AAC (Augmentative and Alternative Communication) or gamified learning, rather than real-time social interpretation.

1. **Proloquo2Go:** A symbol-based AAC app that helps non-verbal children communicate. ***Difference****:* It generates speech from icons, whereas our app interprets incoming complex text.
2. **Otsimo / MITA:** Educational games focused on teaching emotions and facial expressions. ***Difference****:* These are passive learning games, while our app is an active tool for real-world conversation scenarios.
3. **Brain Power (Google Glass):** Uses AR to decode facial expressions in real-time. ***Difference****:* Our app focuses specifically on *text-based* nuances like sarcasm and scams on social media.

**Technology stack:**

* **Frontend:** React.js (Vite), Tailwind CSS v4 (for modern Glassmorphism UI).
* **AI/Backend Logic:** Google Gemini 2.5 Flash API (Cloud Intelligence).
* **Offline Fallback:** Local NLP Libraries (sentiment.js, compromise) for reliability without internet.
* **Icons/Assets:** Lucide React (Icons).
* **Environment Management:** Vite Environment Variables (.env).

**Responsiveness of the application:**

The application is built using ReactJS component architecture and Tailwind CSS utility classes to ensure a fluid and adaptive user interface.

* **Flexbox & Grid Layouts:** Used to center the chat interface dynamically regardless of the monitor size (w-screen, h-screen, flex-center).
* **Relative Units:** The Chat Bubbles and Analysis Cards use percentage-based widths (max-w-[70%]) and auto-margins to ensure they look correct on both small laptop screens and large monitors.
* **State-Driven UI:** The "Analysis Card" and "Chatbot Sidebar" respond dynamically to user interaction, sliding in or expanding without breaking the layout flow (using conditional rendering in React).

**6. Workflow of the application:**

1. **Message Reception:** The user views the chat interface (simulating WhatsApp). Incoming messages are rendered via the ChatBubble component.
2. **Trigger Analysis:** The user clicks the "Sparkles" icon/button next to a confusing message.
3. **Decision Engine (Hook):** The useMessageAnalysis hook checks for an active internet connection/API Key.
   * *If Online:* It sends the text to **Google Gemini** for deep context analysis.
   * *If Offline:* It seamlessly switches to **Local NLP** to provide basic emotion/keyword safety checks.
4. **Result Display:** The AnalysisCard component expands inline, showing the Simplified Meaning, Emotional Tone, and Safety Alerts.
5. **Contextual Help:** Optionally, the user can open the "Chatbot Sidebar" to ask specific questions about the conversation history.

**7. List of files with its purpose?**

|  |  |
| --- | --- |
| **File Name** | **Purpose** |
| src/App.jsx | The main container that renders the Chat Interface and manages global state (like the sidebar visibility). |
| src/components/ChatBubble.jsx | Renders individual messages. Contains the logic to trigger analysis and display the "Analysis Card" overlay. |
| src/components/AnalysisCard.jsx | The UI component that displays the AI results: Simplified Meaning, Emotion Badge, Safety Alerts, and Reply Chips. |
| src/components/ChatbotSidebar.jsx | A slide-out panel that acts as a conversational assistant, allowing the user to ask questions about the chat history. |
| src/components/SafetyAlert.jsx | A reusable component that specifically highlights high-risk warnings (Red Box) for scams or bullying. |
| src/hooks/useMessageAnalysis.js | A Custom React Hook that manages the "Hybrid Strategy." It attempts Cloud API first and catches errors to trigger Local Fallback. |
| src/services/geminiService.js | Handles the connection to Google's Gemini API. Sends prompts and parses the JSON response. |
| src/services/localNlpService.js | The "Offline Brain." Uses sentiment and compromise libraries to analyze text when the internet is down. |
| src/index.css | Contains the Tailwind CSS imports and global font resets to ensure the app looks modern and clean. |