Name: ADENIYI ADEBOYIN TOLUWALOPE
Matric No: 23CG034021

NLP Question-and-Answering System Using an LLM API CLI + Web GUI Application

INTRODUCTION

This project is a modern, responsive AI chat web application that allows users to ask any question and receive real-time streaming answers powered by Google Gemini Pro Latest, the highest-quality reasoning model in the Gemini family. The application features perfect Markdown rendering, animated loading indicators, a clean chat-style interface that refreshes with every new question, and a stunning purple gradient design.

Key Features Implemented

1. Real-time word-by-word streaming responses (TextDecoderStream API)
2. Perfect Markdown formatting using marked.js (bold, headings, lists, code blocks)
3. Animated "Thinking..." with dots and blinking cursor during streaming
4. Clean chat bubbles: user question (right, purple) + AI answer (left, light)
5. Fresh start on every new question (no conversation history clutter)
6. Button text smoothly changes from “Get Answer” → “Loading...”
7. All styling in separate style.css (no inline styles)
8. Mobile-responsive design with animations (fadeInUp, slideIn)
9. Secure deployment: .env and queries.db ignored via .gitignore
10. Live deployment on Render with custom URL format as required

Technology Stack

1. Backend: Flask (Python)
2. AI Model: Google Gemini Pro Latest (`gemini-pro-latest`)
3. Frontend: HTML, CSS, vanilla JavaScript
4. Streaming: Fetch API + TextDecoderStream
5. Markdown: marked.js CDN
6. Deployment: Render.com (free tier) + GitHub
7. Process Manager: gunicorn (via Procfile)

Major Challenges & Solutions

1. Raw ** and ### appearing → Fixed with marked.js
2. Slow loading feel → Implemented real-time streaming + instant “Thinking...”
3. Button text overlapping → Proper show/hide logic
4. Previous conversations persisting → Cleared chat container on new question
5. Render deployment failures → Added Procfile + correct Start Command
6. Environment variable confusion → Used standard Environment Variables section
7. Git safety → Created comprehensive .gitignore

Deployment Process

1. Created public GitHub repository: AI_QUERY_ASSISTANT
2. Added .gitignore, Procfile, requirements.txt
3. Pushed code safely (no API key or database uploaded)
4. Connected to Render.com
5. Set service name: adeniyi-23cg034021-ai-query-assistant
6. Added GEMINI_API_KEY as environment variable
7. Successful deployment at: https://adeniyi-23cg034021-ai-query-assistant-wy7l.onrender.com/

Conclusion

This project evolved from a basic form-based AI app into a professional-grade, real-time streaming AI assistant powered by Gemini Pro Latest, delivering the highest possible answer quality with excellent reasoning capabilities. The final application is fast, beautiful, secure, and fully exceeds all CSC331 project requirements.

Live Application: https://adeniyi-23cg034021-ai-query-assistant-wy7l.onrender.com/
Source Code: https://github.com/Adeboyin1/AI-Query-Assistant