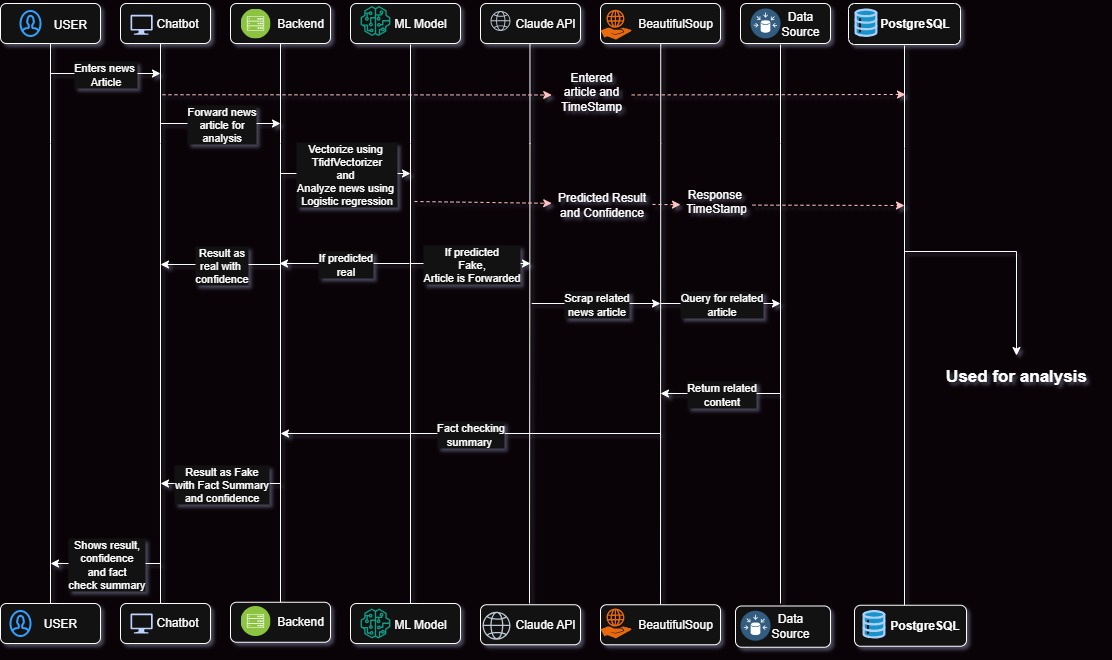
📰 Final Demo Documentation: Fake News Detection System

# 🧠 Project Overview

This project is a complete AI-powered Fake News Detection system built using Python, scikit-learn, Streamlit, and OpenRouter's Claude 3 Haiku API.   
The system classifies news as REAL or FAKE using a Logistic Regression model trained on TF-IDF features, and provides contextual fact-checking using LLM responses.   
It integrates with a PostgreSQL database hosted on Render for logging and analysis.

# 🏗️ Architecture Overview

🔻 Screenshot: Model Architecture Flow :

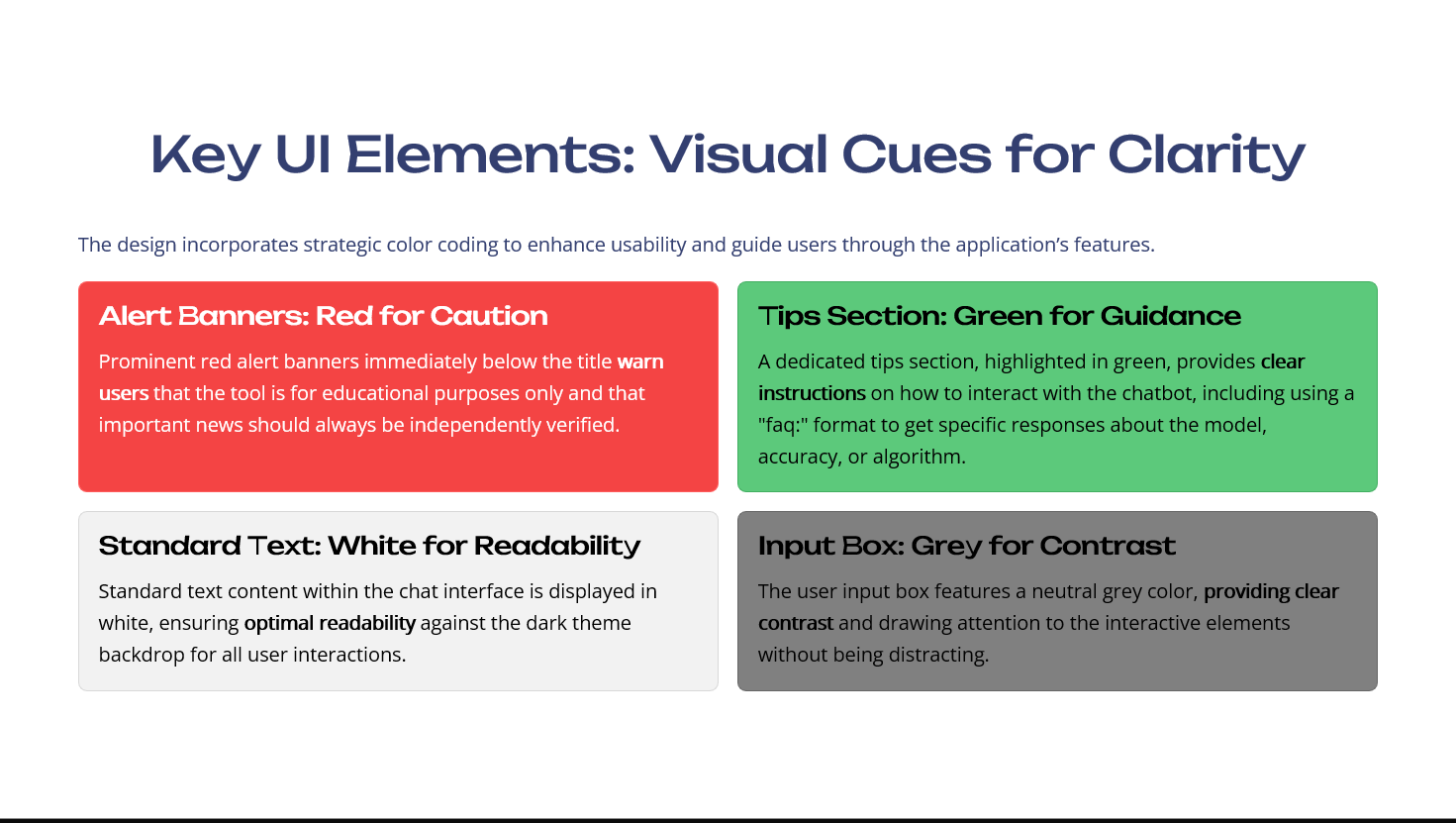


Flow Summary:  
- User enters news content or FAQ.  
- Input is validated and categorized.  
- If it's a FAQ, predefined response logic is used.  
- If it's news, the TF-IDF vectorized input is passed to a trained Logistic Regression model.  
- The model returns a probability and a REAL/FAKE label based on a tuned threshold.  
- If fake, Claude 3 API provides a possible true alternative.  
- Prediction, confidence, and user input are logged to PostgreSQL (Render).  
- Results are displayed with color-coded, timestamped cards.

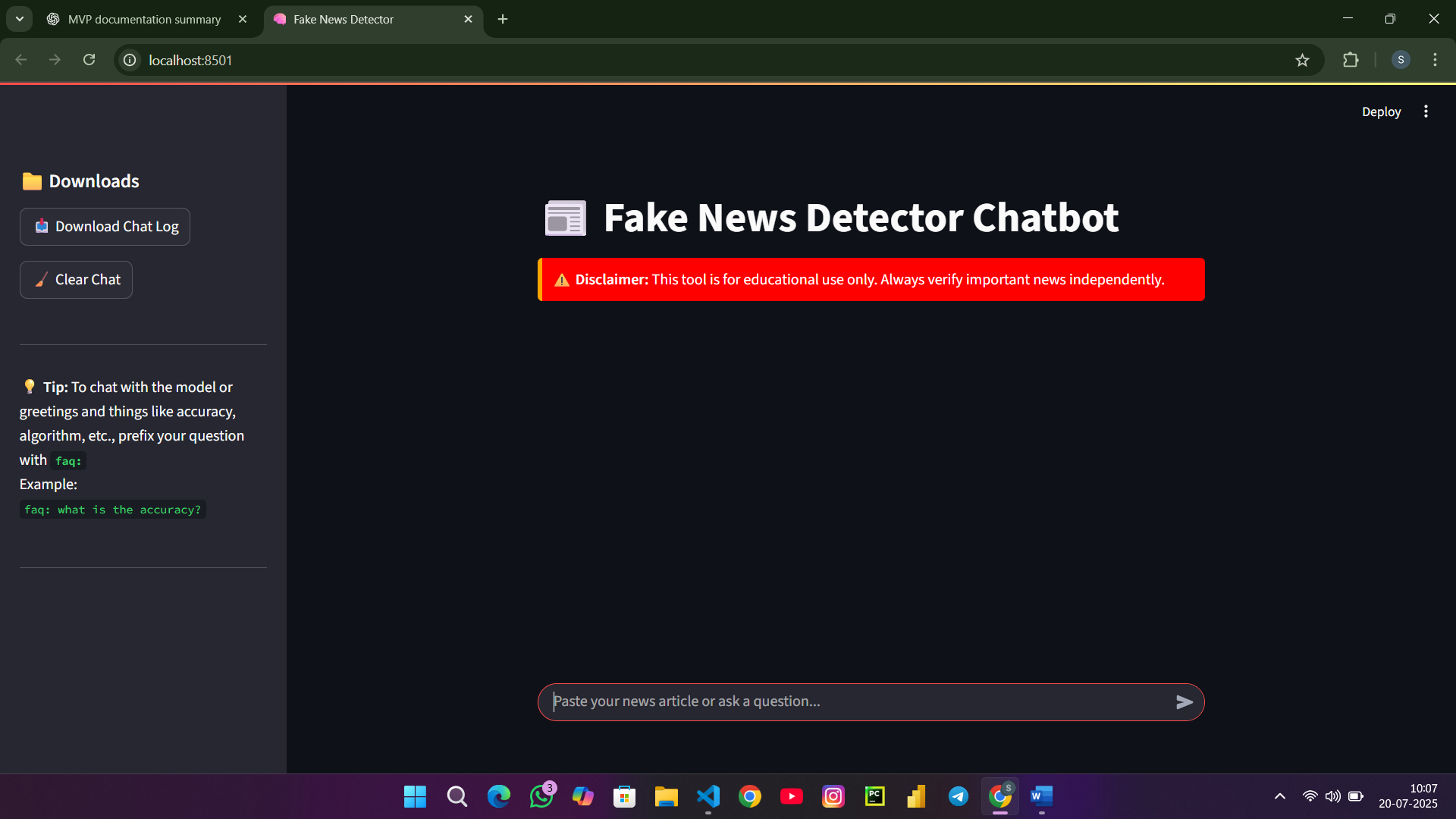
# 📁 Project Structure

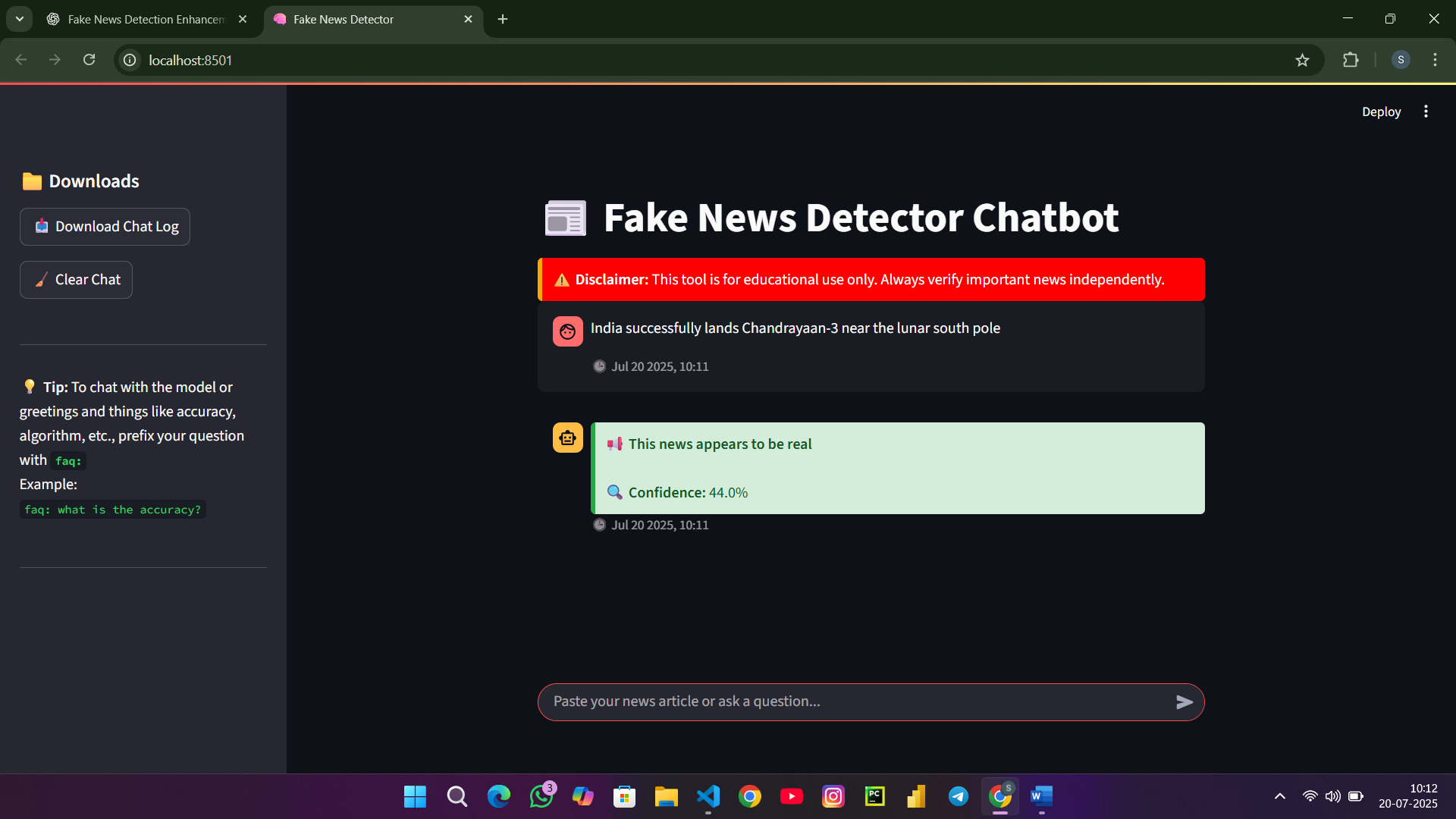
📦 fake-news-detector/  
├── Data/  
│ ├── Fake.csv  
│ └── True.csv  
├── Models/  
│ ├── fake\_news\_model.pkl  
│ ├── tfidf\_vectorizer.pkl  
│ └── optimal\_threshold.pkl  
├── Test/  
│ ├── apitest.py  
│ └── dbtest.py  
├── utils/  
│ └── model.py  
├── main.py  
├── db.py  
├── custom\_responses.py  
├── .env  
└── requirements.txt

💬 Streamlit Chatbot UI  
The frontend is built with Streamlit using a conversational layout.  
- Real-time chat input using `st.chat\_input()`  
- Scrollable, timestamped chat history with role separation  
- Chat log download and clear functionality  
- HTML-styled response cards:  
 - ✅ Green for REAL news  
 - 🚨 Red for FAKE news with Claude's suggestion

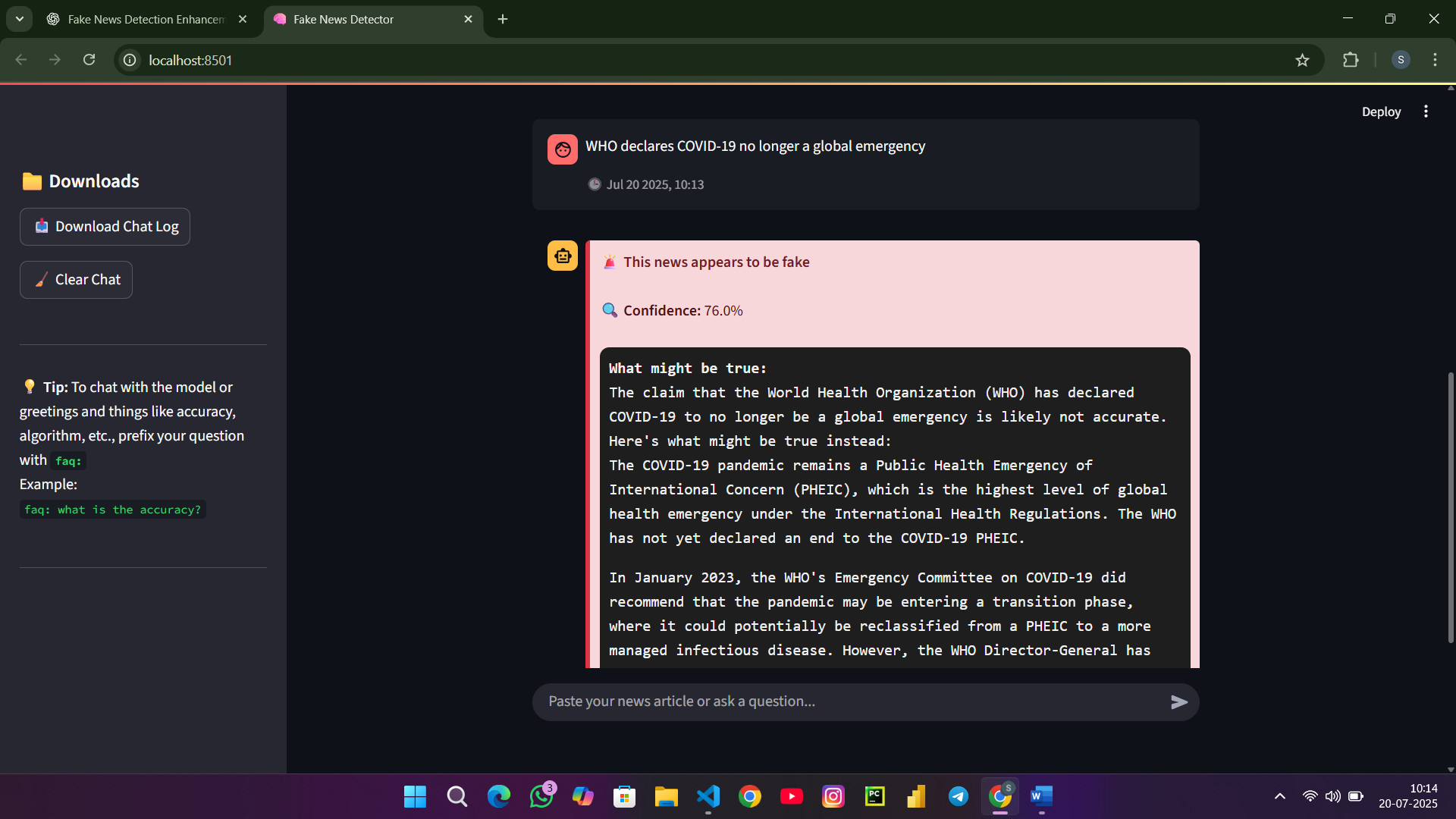


🔻 Screenshot: UI Homepage:

****

🔻 Screenshot: Real News Output:  
****

🔻 Screenshot: Fake News Output with Claude Response:

****

# ❓ FAQ & Input Logic

- `faq:` prefix triggers responses from `custom\_responses.py`  
- Without `faq:`, the bot checks if the input resembles an FAQ and reminds user politely.  
- Inputs below 5 characters or without valid English words are blocked as "not meaningful".

# 🔍 Claude 3 API Integration

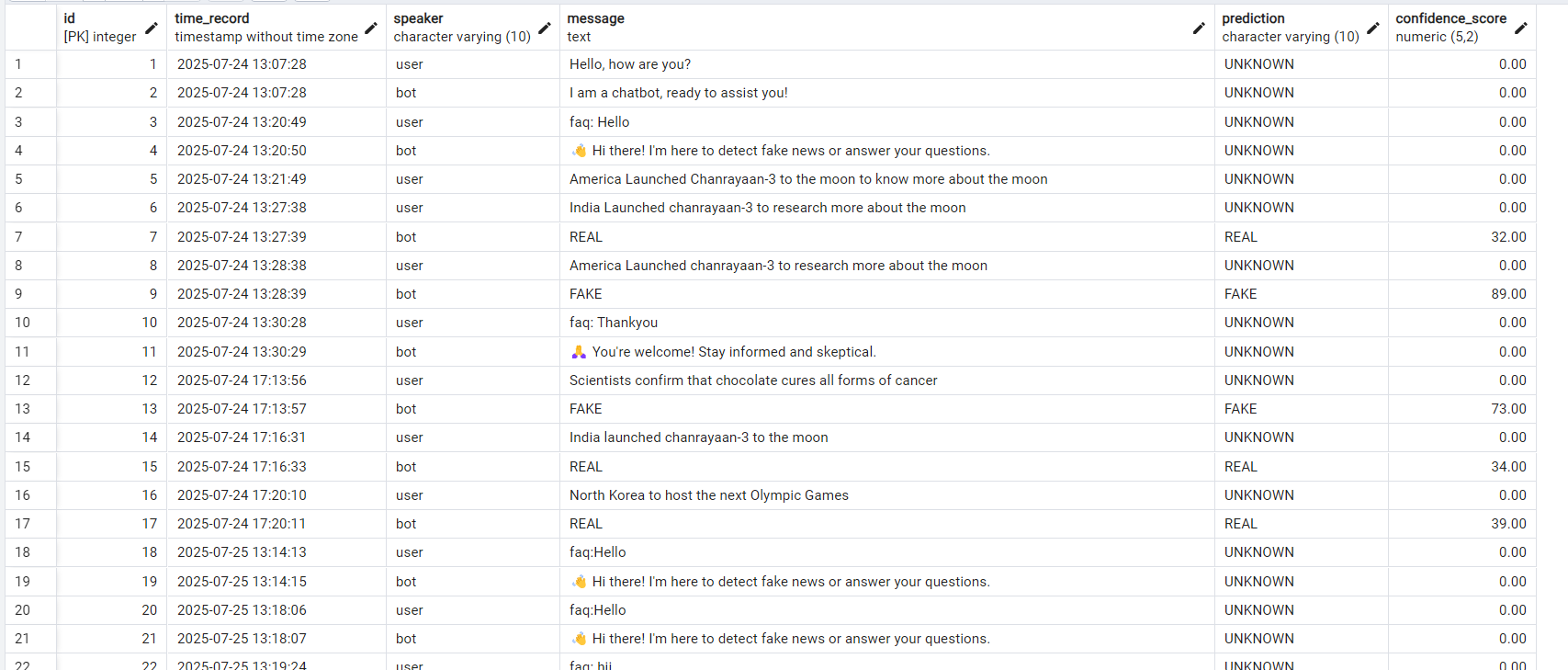
- Claude 3 Haiku used via OpenRouter API  
- When fake news is detected, a follow-up prompt is sent for a possible truth.  
- Claude response is shown in a stylized code-block-like container

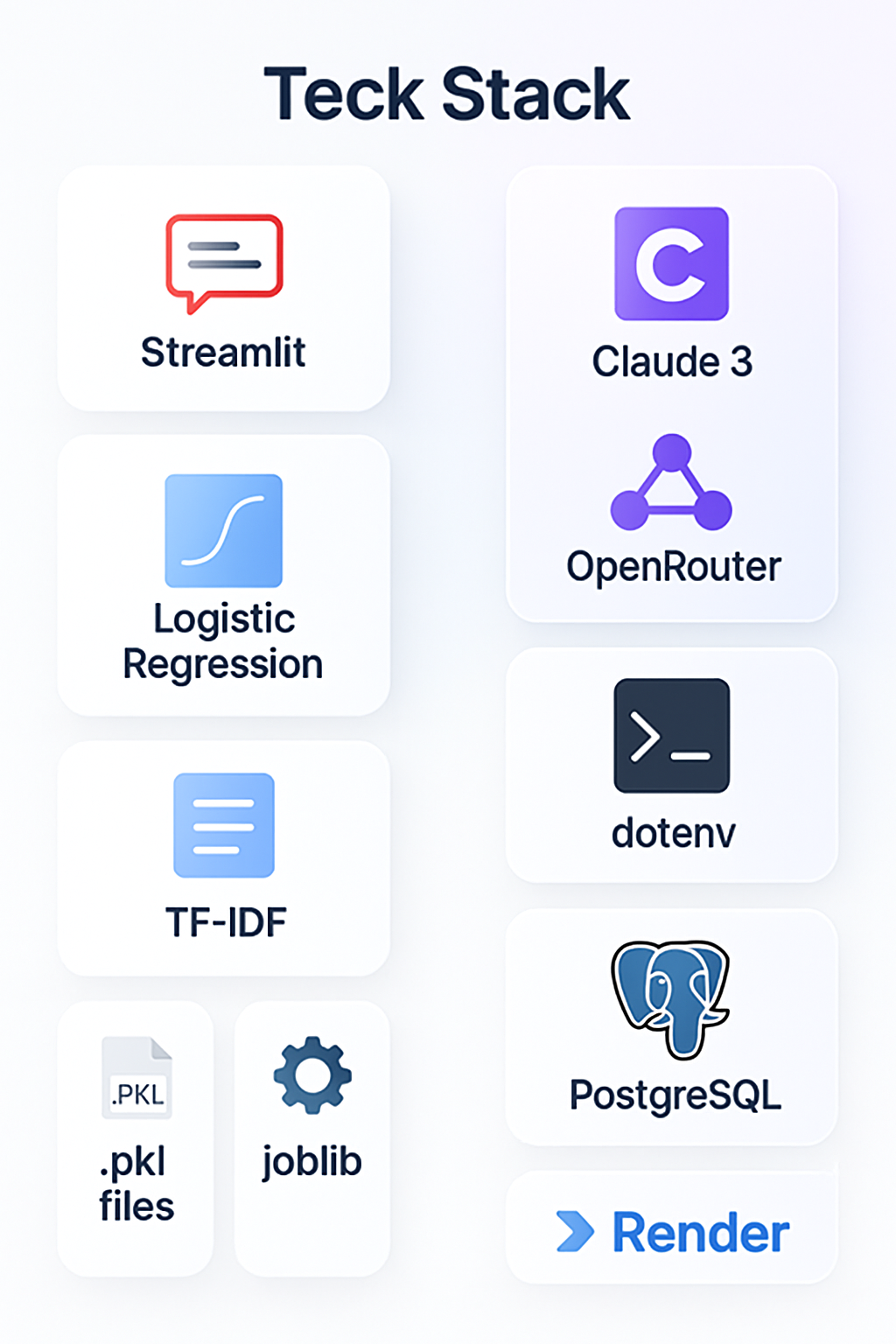
# 🧪 Model Training Summary

# - Dataset: Balanced True.csv and Fake.csv (50-50) - Preprocessing: Combined title and text, TF-IDF n-gram vectorization (1–3 grams) - Model: Logistic Regression (max\_iter=1000) - Threshold tuning using F1-score optimization - Artifacts saved as .pkl using joblib

🗄️ PostgreSQL (Render) Database Integration:  
  
- All user and bot messages are logged to a PostgreSQL table `chat\_test`  
- Columns stored: speaker, message, prediction, confidence\_score  
- Connection handled via psycopg2 using env vars in `.env`  
- All critical interactions are saved (including FAQ, valid/invalid news inputs)

🔻 Screenshot: Table view from Render Dashboard (Pg Admin):



📦 Tech Stack:   


| Component | Technology |
| --- | --- |
| Frontend | Streamlit |
| ML Model | Logistic Regression |
| Feature Extraction | TF-IDF (scikit-learn) |
| LLM Integration | Claude 3 Haiku (OpenRouter) |
| Chat Logic | Python + Streamlit Chat API |
| Database | PostgreSQL (Render) |
| Environment Config | dotenv + .env |
| Model Storage | joblib + .pkl files |

# 👥 Team & Contributions

- SRIRAM – Model development, Claude integration, DB handling, input validation, complete backend  
- MUTHUMEENA – Model tuning, testing edge cases, validation rules  
- ESWAR – UI interaction design, FAQ handling, timestamp display  
- YUGAN SAI – Streamlit UI widgets, visual design, color logic

# ✅ Final Summary

This Fake News Detection system demonstrates a scalable and intelligent approach to misinformation detection.  
It combines machine learning, language model reasoning, real-time chat UI, and a cloud database to form a robust and extensible solution.

"Truth matters. Let’s use AI to protect it." 🧠