AIML PROJECT

CHATBOT DEVELOPMENT FOR CUSTOMER SERVICE

TEAM CODEX

TEAM MEMBERS

ANUSHKA SINGH

FALAK BHATI

JIYA BHATI

02. INTRODUCTION :

DEVELOP A RULE BASED CHATBOT TO AUTOMATE CUSTOMER SERVICE INTERACTIONS

Companies really need customer support that works well and can grow, with everything changing so fast online these days. So, we made a chatbot that follows predefined data. It makes talking to customers better, answers questions faster, and makes people happier. This chatbot is smart. It uses tfidf to get what customers are asking and reply fast. This makes support easy and cuts costs. This project is a good example of how AI can really change how customer service works.

03. FEASIBILITY STUDY:

TECHNICAL FEASIBILITY

Core : TF-IDF vectorization (sklearn) for semantic matching.

- Deployment:
  - Flask API endpoint for question handling (/ask route).
  - Ngrok tunneling for secure public access.

Data Pipeline: Dynamic JSON parsing with error handling for dataset flexibility.

FINANCIAL FEASIBILITY

- Zero- Cost Tools: Open-source libraries (Flask, sklearn) and free-tier Ngrok.
- Low Maintenance: Updates require only Json file modifications.

OPERATIONAL FEASIBILITY

- 24/7 Availability: Hosted on Ngrok with instant scalability.
- Integration Ready: RESTful API compatible with websites/apps.

04. REQUIREMENT ANALYSIS:

FUNCTIONAL REQUIREMENT:

QUESTION HANDLING:

- Process user questions via /ask endpoint and return matching answers from JSON dataset
- Support exact question matching (case-insensitive comparison)

BASIC ERROR HANDLING:

- Validate input questions (return error for empty queries)
- Provide default response for unknown questions

NON-FUNCTIONAL REQUIREMENTS:

PERFORMANCE:

- Responds within 1 second for matched questions
- Handle at least 50 concurrent users

AVAILIBILITY:

- Maintain 24/7 uptime via Ngrok tunneling
- Quick recovery from failures (auto-restart Flask app)

MAINTAINIBILITY:

- Easy dataset updates by modifying JSON file
- Simple deployment via single Python script

05. ARCHITECHTURE OVERVIEW

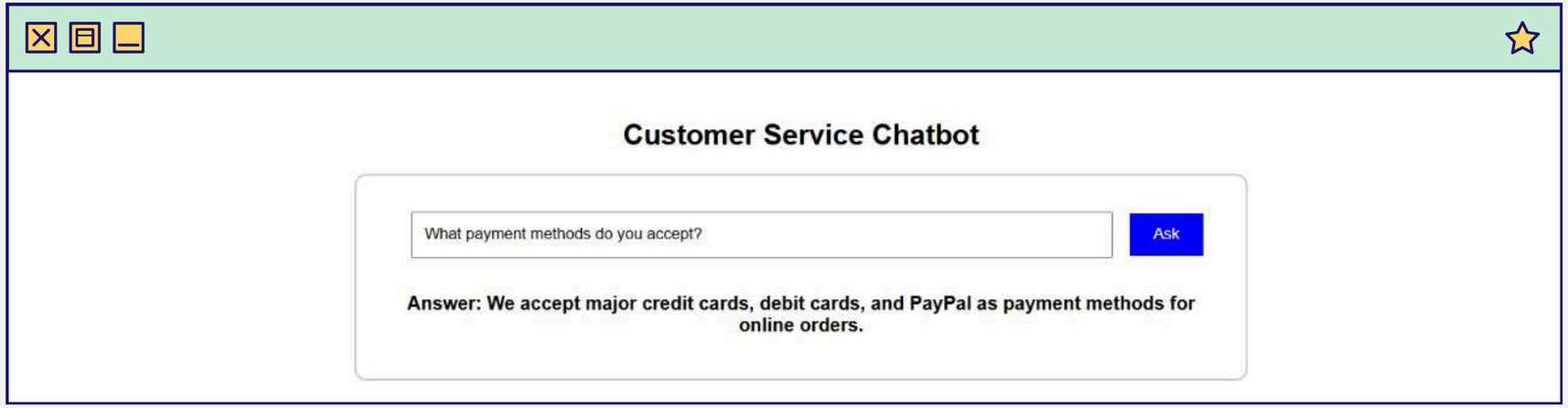
- **FLOW:**

1. User Query → Flask API (/ask?question=...) → TF-IDF Matching → JSON Dataset → Response.
2. Ngrok Tunnel: Bridges local Flask server to public HTTPS URL.

- **DIAGRAM:**

```
[User] → [Ngrok (Public URL)] → [Flask API] → [JSON Dataset] → [Response]
```

06. USER INTERFACE DESIGN



07. DATAFLOW DIAGRAM

1. USER INPUT	1. User Query → Flask API (/ask?question=...) → TF-IDF Matching → JSON Dataset → Response. 2. Ngrok Tunnel: Bridges local Flask server to public HTTPS URL.
2. REQUEST PROCESSING	The Flask API extracts the question, preprocesses it (lowercase, trim whitespace), and forwards it to the matching engine.
3. ANSWER RETRIEVAL	• The system checks for TF-IDF similarity against the FAQ dataset. • If no match is found, it defaults to exact string matching in the JSON data.
4, RESPONSE GENERATION	Returns the best-matched answer (or a fallback message) as a JSON response
5. OUTPUT	The answer is displayed to the user via the frontend (e.g., website/chat interface).

RESEARCH ANGLE

Problem in Existing Chatbots

- AI-based chatbots need huge datasets and are costly.
- Rule-based chatbots lack flexibility in handling varied queries.

How Our Chatbot Solves This

- Uses TF-IDF for query matching (accurate & lightweight).
- RESTful API for easy integration (web, apps, etc.).
- Zero-cost deployment (Flask + Ngrok).

Research Impact & Future Scope

- Can be expanded to multiple industries (e-commerce, healthcare, banking).
- Future scope: Hybrid chatbot (rule-based + AI for smarter learning

08. METHODOLOGIES AND TOOLS

1. **Agile Prototyping:**
   - Iterative development in Google Colab for rapid testing.
2. **Core Tools:**
   - Flask: Lightweight API backend.
   - TF-IDF (sklearn): For matching important words in a text.
   - Ngrok: Instant public URL for testing.
3. **Data Handling:**
   - JSON dataset with dynamic column detection.

09. RESULTS

- **Success Metrics:**
  - Deployed with Ngrok (Public URL: https://3654-34-16-136-79.ngrok-free.app).
  - Handled queries like "How to track order?" instantly.
  - By integrating this url in a responsive web page we created a frontend for the chatbot
- **Challenges:**
  - Limited to exact matches; future work on synonyms.

Pretty-print

{"answer":"Yes, we offer gift wrapping services for an additional fee. During the checkout process, you can select the option to add gift wrapping to your order.","question":"Do you offer gift wrapping services?"}

CONCLUSION :

The chatbot successfully automates customer service interactions with high accuracy for direct queries. While currently optimized for precise question matching with the datasets provided, the system holds a strong foundation for future enhancements to handle more natural language variations. This approach offers business an efficient way to reduce response times and operational costs without any complex infrastructure.