Swig-Chat (Swiggy Chat Bot)

- ## Problem Understanding

  Food delivery is very time sensitive, and the biggest point of concern to the customers is the question of "Where is my order?" Customers demand constant presence and confirmation during the delivery process. When such unforeseen situations arise as unexpected delays, traffic jams, rain disturbances, overcrowded restaurants or a mistaken payment, customers require information immediately and not when they have been put on hold or through a series of help desk screens.
  In case the Swiggy app does not have a chatbot, users experience the following problems:
    - They are not provided with immediate and customised support within the app.
    - This is because they get stuck when the ETA shifts abruptly, and they are not aware of the reason.
    - They file unreasonable complaint tickets on petty problems that can be addressed automatically. Even basic ones, like "It shows delivered, but I did not get it," are difficult to resolve.
    - Customers are made to feel disrespected or powerless, resulting in a bad brand experience.

  Thus, it is not necessary to answer the questions, but to make it less nervous, open it up, and automate the repetitive support functions to ensure that the customer experience is smooth
  Introducing a chatbot in the Swiggy app directly addresses these gaps.
  A Chatbot :

    - Provides instant status lookup for any order
    - Explains delays clearly (traffic, rain, restaurant backlog, high order value)
    - Offers next-step guidance (track rider, wait time, escalation options)
    - Manages angry situations professionally
    - Escalates only when necessary, reducing dependency on human support.

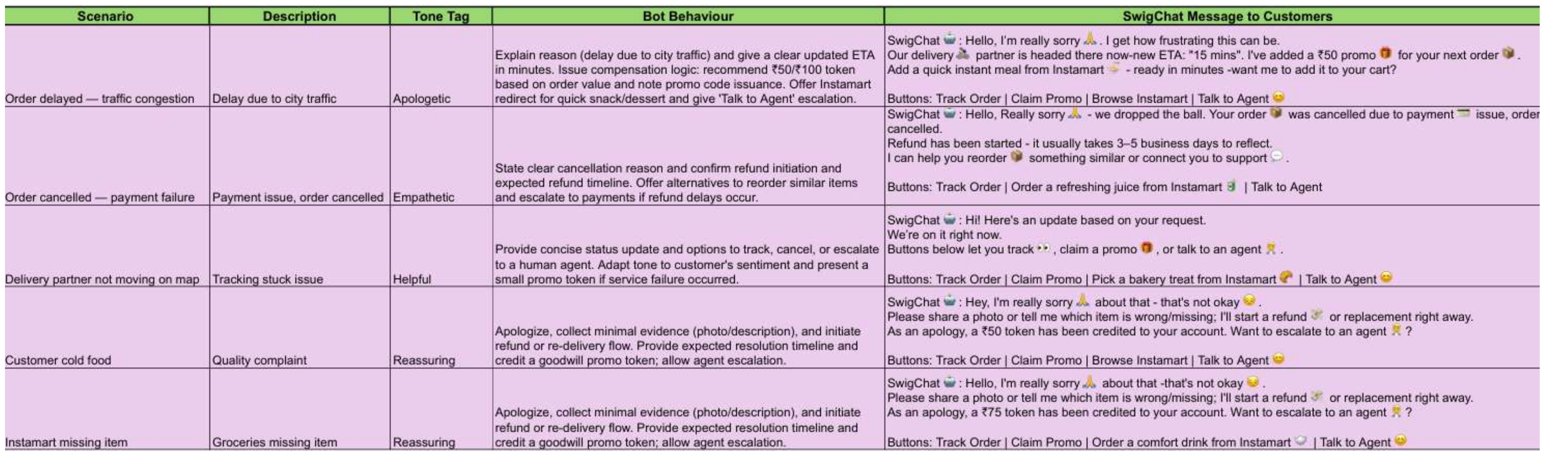
  In summary, a chatbot serves as a critical bridge between user emotions (stress about food delays) and brand service quality, ensuring seamless communication and real-time support within the app.

- ## Data Sets Used
    - Order data - Data Set
      The dataset includes 29 mock Swiggy orders covering both Food and Instamart categories, simulating real user scenarios such as delays, cancellations, refunds, and payment failures. Each record contains details like order ID, restaurant name, status, ETA, delay reason, payment method, delivery partner ID, and complaints. This structured data enables the chatbot to generate realistic, context-specific responses for queries like "Where is my order?", "Why is it delayed?", or "I didn't receive my food." It helps the bot mimic an actual backend system, improving accuracy and user experience during conversation.
    - Scenarios data - Data Set
      The scenario dataset showcases how Swig-Chat should respond to different real-world customer situations, such as delays, refunds, Instamart issues, missing items, and emotional tones like frustration or appreciation. Each entry specifies the tone, bot behaviour, and message template, ensuring that responses stay natural, empathetic, and aligned with Swiggy's brand voice. It helps the chatbot understand user sentiment and context, so replies vary appropriately, whether the user is angry, polite, or confused. This dataset essentially acts as the emotional and conversational engine behind Swig-Chat's personality and response accuracy.

- ## Rationale behind prompt design and conversation logic
    - The design of Swig-Chat was based on the idea of making support faster, more human and genuinely helpful. The entire design focuses on three major things: providing accurate updates, responding with empathy, and resolving issues efficiently, just like a real Swiggy representative would. The chatbot is built to understand user emotions, analyse order data and deliver clear, conversational responses that feel personal and trustworthy.
    - **Data Training**: Swig-Chat's backend logic works on structured data training using two data sets that act as a mock Swiggy backend system.
      Whenever a user asks a query, the bot searches the Order dataset to find details like order status, delay reason, ETA, or payment info, and uses the Scenarios Dataset to decide how to reply.
      For example:
      If a user says "Where's my order 1012?", the backend logic checks the dataset, finds status Delayed, reason =Traffic congestion, and automatically replies:
      "Order 1012 from McDonald's is delayed due to traffic.
      Updated ETA: 40 mins.
      We're really sorry for the wait!"
    - **Sentiment Detection:** A sentiment detection layer was also added to help the bot pick the emotional tone of the user's message, like identifying when someone is angry, frustrated or upset.
      The system scans for emotion-indicative keywords like "what is this", "throw", "terrible", "late again", etc and adjusts tone accordingly.
      Example:
      User: "I have a spillage issue. What is this? I had to throw the food."
      (Swig-Chat detects frustration and responds empathetically):
      "I'm really sorry to hear that. It looks like your food got spilt. Let me initiate a refund and add a ₹50 Instamart coupon as a small gesture."
    - **Issue Resolution**: The chatbot's main goal is to solve the user's problem instantly and avoid unnecessary escalation to human agents.
      Its decision logic mainly focuses on:

1) What kind of answer to give - status update, refund, delay reason, etc.
2) When to issue refunds or coupons - triggered by failed payments, spills, or delays.
3) How many coupons to offer- ₹50-₹200, depending on the order value or delay length.
4) Which tone to use- based on detected emotion.
5) When to escalate- if the user remains unsatisfied after two messages.

- **Scenario-based Logic Mapping:** The scenario logic connects different data points - like Order Status, Delay Reason, and User Emotion- to choose the right action. To showcase the mapping, here are a few scenario cases (for the rest of the scenarios, look up **Scenario dataset**



Scenario	Description	Tone Tag	Bot Behaviour	SwigChat Message to Customers
Order delayed — traffic congestion	Delay due to city traffic	Apologetic	Explain reason (delay due to city traffic) and give a clear updated ETA in minutes. Issue compensation logic: recommend ₹50/₹100 token based on order value and note promo code issuance. Offer Instamart redirect for quick snack/dessert and give 'Talk to Agent' escalation.	SwigChat 🍲 : Hello, I'm really sorry 🙏 . I get how frustrating this can be. Our delivery 🛵 partner is headed there now-new ETA: "15 mins". I've added a ₹50 promo 🎁 for your next order 📦 . Add a quick instant meal from Instamart 👉 - ready in minutes -want me to add it to your cart? Buttons: Track Order | Claim Promo | Browse Instamart | Talk to Agent 😊
Order cancelled — payment failure	Payment issue, order cancelled	Empathetic	State clear cancellation reason and confirm refund initiation and expected refund timeline. Offer alternatives to reorder similar items and escalate to payments if refund delays occur.	SwigChat 🍲 : Hello, Really sorry 🙏 - we dropped the ball. Your order 📦 was cancelled due to payment 💳 issue, order cancelled. Refund has been started - it usually takes 3–5 business days to reflect. I can help you reorder 📦 something similar or connect you to support 💬 . Buttons: Track Order | Order a refreshing juice from Instamart 🧃 | Talk to Agent
Delivery partner not moving on map	Tracking stuck issue	Helpful	Provide concise status update and options to track, cancel, or escalate to a human agent. Adapt tone to customer's sentiment and present a small promo token if service failure occurred.	SwigChat 🍲 : Hi! Here's an update based on your request. We're on it right now. Buttons below let you track 👀 , claim a promo 🎁 , or talk to an agent 🧑‍💼 . Buttons: Track Order | Claim Promo | Pick a bakery treat from Instamart 🥐 | Talk to Agent 😊
Customer cold food	Quality complaint	Reassuring	Apologize, collect minimal evidence (photo/description), and initiate refund or re-delivery flow. Provide expected resolution timeline and credit a goodwill promo token; allow agent escalation.	SwigChat 🍲 : Hey, I'm really sorry 🙏 about that - that's not okay 😞 . Please share a photo or tell me which item is wrong/missing; I'll start a refund 💸 or replacement right away. As an apology, a ₹50 token has been credited to your account. Want to escalate to an agent 🧑‍💼 ? Buttons: Track Order | Claim Promo | Browse Instamart | Talk to Agent 😊
Instamart missing item	Groceries missing item	Reassuring	Apologize, collect minimal evidence (photo/description), and initiate refund or re-delivery flow. Provide expected resolution timeline and credit a goodwill promo token; allow agent escalation.	SwigChat 🍲 : Hello, I'm really sorry 🙏 about that -that's not okay 😞 . Please share a photo or tell me which item is wrong/missing; I'll start a refund 💸 or replacement right away. As an apology, a ₹75 token has been credited to your account. Want to escalate to an agent 🧑‍💼 ? Buttons: Track Order | Claim Promo | Order a comfort drink from Instamart 🥛 | Talk to Agent 😊

- Limitations
  - Even though Swig-Chat works quite well with the mock data I've created, there are still a few limitations that I noticed while testing it:
    - **Language Support:** Right now, the bot only understands and replies in English. This can be difficult for users who prefer using Hindi or any other regional language.
    - **No Live Integration Yet:** Since this is a prototype, it isn't connected to Swiggy's live backend system. All the order details, payment updates, and delivery statuses are simulated from my dataset, not from real-time data.
    - **Basic Sentiment Detection:** The emotion detection works on keywords, so it picks up words like "angry," "bad," or "late" to understand the user's mood. It works okay, but sometimes it misses the real context or tone.
    - **No Voice Commands:** The chatbot only works through text right now. It doesn't support voice inputs, which could have made the experience more natural and convenient.
- Future Improvements
  - There's a lot of scope to make Swig-Chat more advanced and closer to a real customer support system. Here's what I plan to improve in the next version:
  - **Multi-Language Support:** I want to add Hindi and a few regional languages using translation APIs so that users can chat in whichever language they're most comfortable with.
  - **Connecting to Live Data:** The next step would be integrating it with Swiggy's actual backend through APIs so that all order and refund updates come directly from the live system instead of mock data.
  - **Better Emotion Detection:** I plan to replace keyword-based sentiment checks with an AI model that can understand tone and intent more accurately - for example, detecting sarcasm or subtle frustration.
  - **Voice Interaction:** Adding voice-based chat support would make it feel more interactive and accessible, especially for users who prefer speaking instead of typing.