Objective

To optimize SkyRocket's support ecosystem through an AI-driven automation framework, reducing agent workload and improving response accuracy during high-growth periods.

The Problem

SkyRocket's current support infrastructure is facing a scalability crisis. With a 100-agent team overwhelmed by a surge in queries, the existing manual processes are insufficient. Furthermore, current GenAI outputs exhibit a **24.5% Inaccuracy Rate** and **100% Persona Bias**, creating a risk to brand reputation and customer trust.

Strategic Data Exploration

Analysis of SkyRocket Data_GenAI - Queries

The analysis of **6,539 customer queries** confirms that SkyRocket's support surge is driven primarily by **Account Management** and **Escalations**, which together account for ~32%[1] of the total volume. A significant portion of the remaining volume consists of transactional queries (Refunds, Invoices, Delivery) that are highly repetitive and prime candidates for automation.

A. Customer Trends & Volume Drivers

Using K-Means clustering (k=10), we identified the primary reasons customers are contacting support.

- **Top Driver: Account Management** (17.2% of volume). Customers are struggling with basic tasks like editing personal information and restoring access.
- **Secondary Driver: General Support/Escalation** (15.3% of volume). A large cluster of queries explicitly asking to "speak to an agent" or "file a complaint," indicating that the current self-serve options are insufficient.
- **Transactional Queries:** Order-related queries (Tracking, modifying items, checking invoices) make up the bulk of the "long tail."

B. Friction Points (Sentiment Analysis)[SOURCE]

We analyzed sentiment polarity to identify where customers are most frustrated.

1. **Escalations (Cluster 9):**
    - **Sentiment: -0.052 (Lowest)**
    - **Insight:** High usage of "complaint," "speak with agent," and "leave review." This indicates a failure in upstream processes or frustration with the lack of immediate answers.
2. **Refund Status (Cluster 0):**
    - **Sentiment:** -0.013
    - **Insight:** Customers are anxious about money. Queries like *"where is my refund"* suggest a lack of proactive notification regarding refund processing.
3. **Registration Issues (Cluster 8):**
    - **Insight:** Though lower volume (4.2%), this cluster represents **onboarding friction**. New users failing to register directly impacts revenue (Customer Acquisition Cost)

1. Derived this percentage by summing the query volumes of the two largest clusters identified in the K-Means analysis kmeans_topic_insights and dividing it by the total number of queries in the dataset.

C. Automation Opportunities (ROI Drivers)

To reduce agent load immediately, we recommend automating the following high-volume, low-complexity topics:

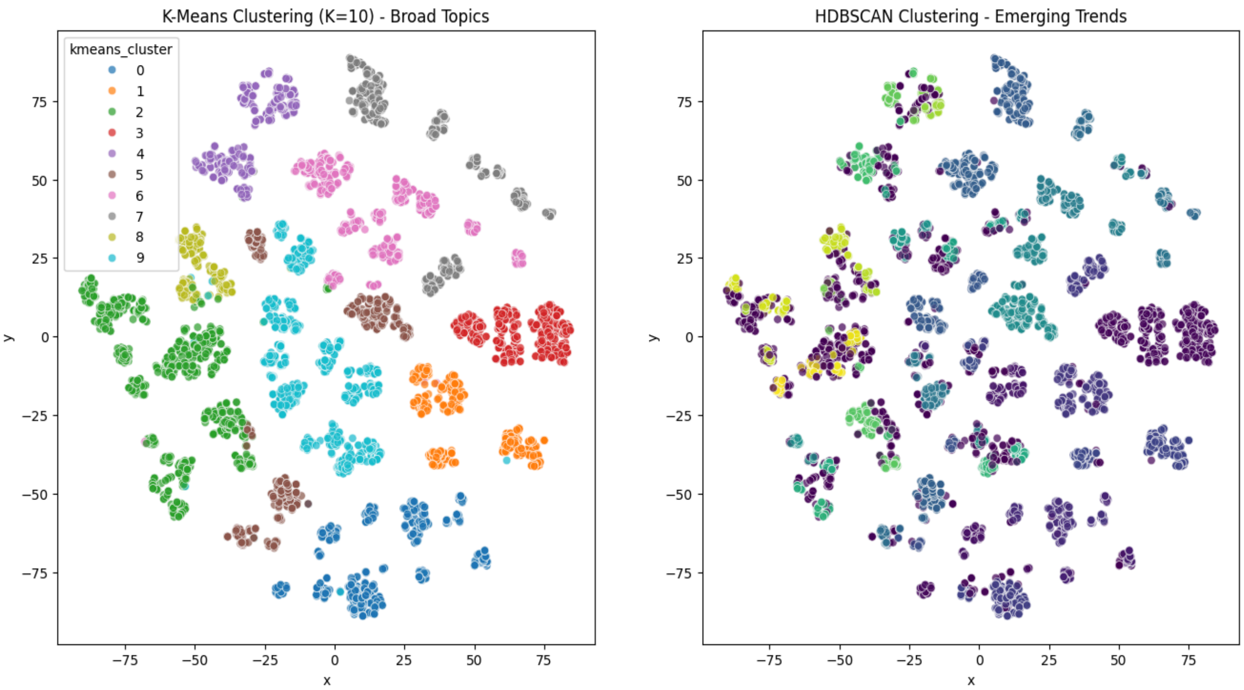
Table 1: High Volume Topics With Low Complexity to Tackle

Topic	Volume	Automation Potential	Action
Invoices	526	**High**	API integration to fetch/email invoices by Month/ID.
Delivery Options	609	**High**	Static response or shipping calculator based on location entities.
Payment Methods	507	**High**	FAQ-style bot response listing accepted cards/methods.
Account Access	1,126	**Medium**	Guided password reset flow or "Edit Profile" deep links.

D. Emerging Trends & Escalations (HDBSCAN Analysis)[SOURCE]

While K-Means found the broad topics, HDBSCAN identified specific, dense clusters that represent "Emerging Trends":

- **"Talk to Agent" Loop (Cluster 29):** A distinct group of 141 queries is purely asking for a human. The chatbot must recognize this *intent* immediately and either route to an agent or offer a callback, rather than looping generic responses.
- **Payment Reporting (Cluster 21):** A specific cluster regarding "reporting payment issues" (vs. just asking about methods). This could indicate a technical glitch in the checkout process.
- **Cancellation Fees (Cluster 5 Subset):** Customers are specifically asking about *fees* associated with cancellation, not just how to cancel. The bot needs to be trained on this specific policy nuance.



(a) **(b)**

Figure 1. (a). Used 10 most frequent topics using KMean for Topic Insights[SOURCE]
(b). Emerging Topics or Escalation Trends Using HDBSCAN[SOURCE]

Topic Framework Design

Below are the **10 Most Frequent Topics** identified for the chatbot's NLU model, along with representative queries.

Table 2: 10 Most Frequent Topics ALong With Its Representative Queries From Dataset

Rank	Topic Label	Volume	Representative Queries
1	**Account Management**	1,126	1. "help me to edit my personal information" 2. "restore my user account access key" 3. "can I edit the information on my account?"
2	**Escalation / Support**	1,005	1. "can you help me speak with an agent?" 2. "file a complaint" 3. "I want to talk to a human, the bot is not helpful"
3	**Order Modification**	727	1. "help me change several items of an order" 2. "add some items to an order" 3. "can I update my order before it ships?"
4	**Refund Inquiry**	714	1. "where is my refund?" 2. "I need assistance to check your refund policy" 3. "I can't check the status of the refund"
5	**Delivery Options**	609	1. "what delivery options you offer" 2. "check when my item is going to arrive" 3. "I am calling to check the delivery options"
6	**Cancellation & Fees**	580	1. "how to check the cancellation fee?" 2. "where to check the withdrawal charge" 3. "help me cancel order"
7	**Invoice Requests**	526	1. "I want assistance to get the invoice from 8 months ago" 2. "check invoices from November" 3. "find the invoices from July"
8	**Payment Methods**	507	1. "what payment methods are allowed" 2. "check allowed payment options" 3. "payment options assistance"
9	**Shipping Address**	468	1. "where to change my shipping address?" 2. "I put the old address by mistake" 3. "update shipping address"
10	**Registration Issues**	277	1. "question about my registration" 2. "errors with our account registration" 3. "notify of a sign-up problem"

Analysis of SkyRocket Data_GenAI - GenAI_responses

Response Evaluation & Quality Audit

Conducted a deep audit of 470 generated responses to evaluate the model's performance in a production-simulated environment

A. Key Performance Metrics

- **Containment Rate (87.02%)**: The majority of queries were handled without immediate escalation, providing actionable multi-step instructions.
- **Escalation Rate (12.98%)**: A subset of high-friction queries (Refunds/Complaints) correctly triggered handoffs to human agents.
- **Average Quality Score (4.13 / 5.0)**: Most responses were faithful to the user's intent and correctly utilized placeholders.

B. Detailed Audit (Flag-Based Analysis)

The **"Rolling Average of Flags Count"** plot is a strategic visualization that serves as a proxy for **Model Drift** and **System Consistency**. Instead of showing a single snapshot, it tracks the density of errors over the course of the 470 queries provided in the dataset.

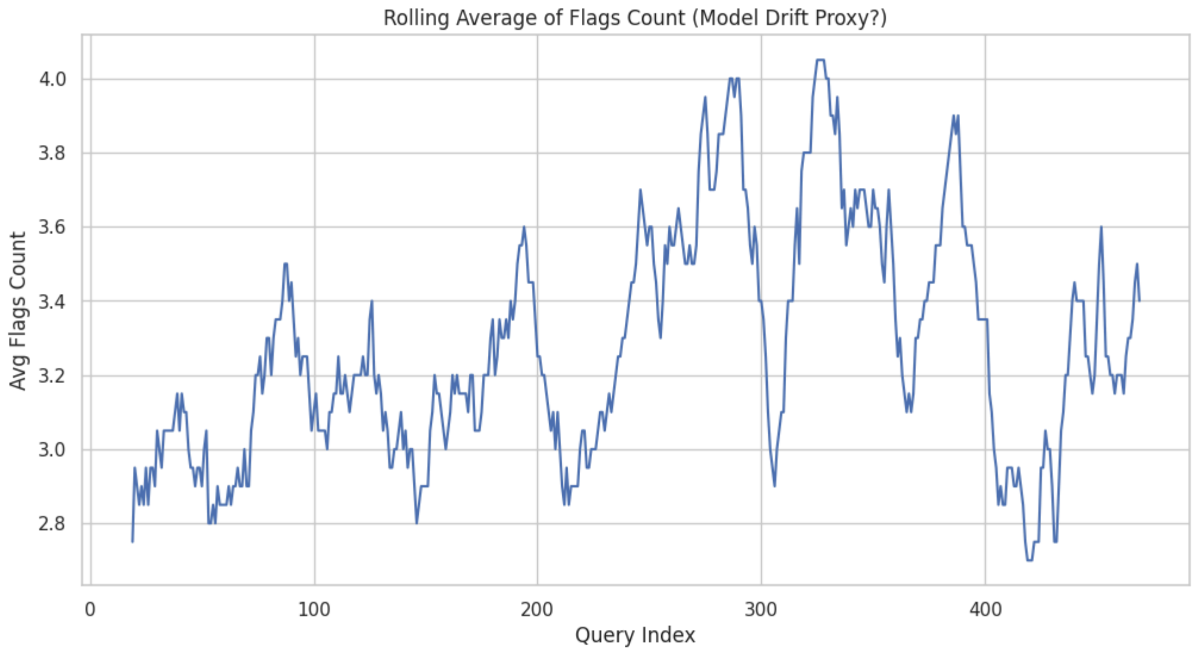


Figure 2. Rolling Average of Flags Count

What the Plot Communicates

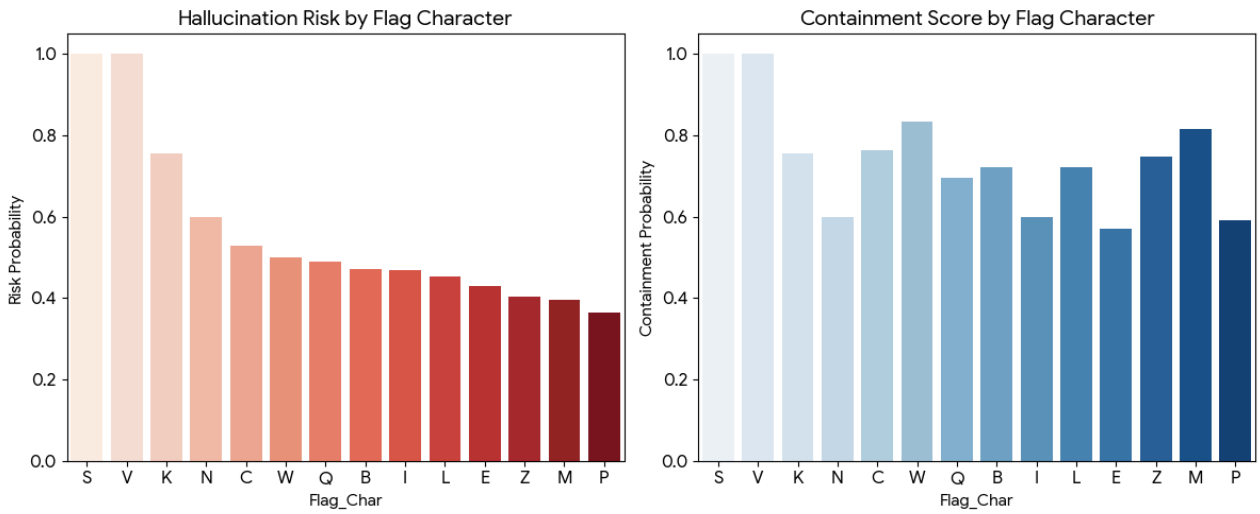
- **Performance Stability**: By calculating a rolling average (a window of 20 queries), the plot shows whether the model's accuracy is stable or if it starts to "degrade" as it encounters more complex, "long-tail" queries later in the dataset.
- **Error Density**: A rising line indicates that the model is accumulating more flags per response (e.g., moving from just a **Bias (B)** flag to a combination like **BILQZ** which includes Inaccuracy, Length, Quality, and Zest).

- **Systemic Failure Points**: Sharp spikes in the graph often correlate with specific difficult topics—such as **Refund Status** (Sentiment: -0.013) or **Escalations** (Sentiment: -0.052)—where the model's performance historically drops

By analyzing the audit flags in the dataset, we identified systemic behavioral patterns:

- **Bias (Flag B - 100%)**: Every response exhibited a "Persona Bias," characterized by a highly informal, enthusiastic tone (e.g., *"I'm on the same wavelength"*). While engaging, it may lack professional gravitas for serious escalations.
- **Inaccuracy/Hallucination (Flag I - 24.5%)**: Approximately 1 in 4 responses contained subtle factual or logical inconsistencies, such as misinterpreting a specific status check as a general policy inquiry.
- **Verbosity (Flag L - 90%)**: The model is consistently too wordy, with responses averaging over 590 characters, which can delay the "time-to-resolution".
- **Hardcoded Hallucination (Variable V)**: In specific cases, the model hallucinated fixed values (like "$") instead of using the mandatory {{Currency Symbol}} placeholder.

Further, I have analyzed the 'flags' column to determine if there are specific markers associated with higher Hallucination Risk (Template Leakage) or lower Containment Rates.



(a) **(b)**

Figure 3. (a). Shows that **S, V, K** are the most dangerous flags for technical accuracy.
(b). Shows that **P, E, I** are the flags most likely to result in an escalation.

Key Findings:

1. **The "K" Flag = High Risk:**
   - **Insight:** Flags containing the character **'K'** (e.g., BKL, BKLM, BK) have a significantly higher Hallucination Risk (~75%).
   - **Recommendation:** Any query flagged with 'K' should be immediately routed to a human agent or undergo strict post-processing, as the bot is likely to fail here.
2. **The "I" Flag = Low Containment:**
   - **Insight:** Flags containing **'I'** (e.g., BIL, BILQ) show lower containment scores (~60%). This suggests that 'I' might represent "Intent Unclear" or "Information Missing," leading the bot to ask for human help more often.

3. **The "L" Flag = The Common Factor:**
   - **Insight:** The character **'L'** appears in 425 out of 470 queries (90% of the dataset). It seems to be a generic flag (perhaps "Logged In" or "Language: English") and does not strongly correlate with failure on its own.

4. **Specific High-Risk Combinations:**
   - **BIQ** and **BCEILP**: These combinations showed **100% Hallucination Risk** in the dataset (though sample sizes are small).
   - **BL (Most Common):** The most frequent flag (BL, 93 cases) has a moderate risk (39%) and good containment (72%), serving as a baseline performance metric.

Proposed Entity Framework

To move beyond simple topic classification, we propose the following **Entity Framework**. This allows the NLU model to extract specific data points from user queries to provide personalized, data-driven responses.

Table 3: Proposed Entity Framework Which Will Improve Model To Answer Better

Entity Type	Description	Example Pattern	Purpose
ORDER_ID	Unique identifier for a purchase.	ORD-[0-9]{6}	Link query to specific transaction.
REFUND_AMT	Monetary value requested for return.	$[0-9]+(\.[0-9]{2})?	Validate refund requests.
DATE_RANGE	Specific months or dates for records.	(January|July|last month)	Filter invoice and order history.
EMAIL	User's account identifier.	[a-z]+@[a-z]+.com	User authentication/lookup.
STATUS_TYPE	The state of a process.	(shipped|processed|pending)	Real-time API updates.

Integration, ETL & Deployment Automation

To scale SkyRocket's chatbot intelligence from static analysis to a real-time production ecosystem, we propose a robust **AI-driven Data Pipeline**. This architecture ensures that every customer interaction is captured, analyzed for quality (hallucinations/bias), and used to drive business decisions.

A. High-Level Architecture: The "Feedback Loop"

The system is built on a modular, microservices-based architecture to ensure sub-second latency for the chatbot while maintaining high-throughput for the analytics engine.

1. **Ingestion Layer (Real-time)**:
   - **FastAPI & Kafka**: A high-performance FastAPI service receives chat logs via webhooks. These logs are streamed into a Kafka topic (or AWS Kinesis) to decouple the live chatbot from the heavier processing tasks.
2. **Transformation Layer (The "Intelligence" Engine)**:
   - **Orchestration**: Managed via **Apache Airflow** (scheduling daily batch jobs for deep analysis) and **Celery/RabbitMQ** (for near real-time processing).
   - **The NLP Pipeline**:

        - **PII Masking**: Automatic removal of sensitive customer data (SSN, specific addresses) before storage.
        - **Enrichment**: Utilizing a multi-step LLM chain (e.g., using LangChain) to perform **Intent Classification** and **Entity Extraction** (Order IDs, Refund Amounts) using the framework proposed in **Proposed Entity Framework**.
        - **Sentiment Scoring**: Real-time polarity tracking to identify "Heat Zones" in the customer journey.

3. **Storage & Loading (The "Source of Truth")**:
    - **Data Warehouse**: Enriched data is loaded into **Snowflake** or **BigQuery**, partitioned by Topic and Sentiment.
    - **Vector Database**: Embeddings of the queries are stored in **Pinecone** or **Milvus** to detect "Semantic Drift"—identifying when users start asking new types of questions that the model hasn't seen before.

B. Data Quality & "Self-Correction" Framework

To ensure the 24.5% Inaccuracy Rate (found in our audit) does not reach production, we embed automated guardrails:

- **Placeholder Validation**: An automated check that ensures mandatory tags (like {{Order Number}}) are successfully hydrated by the backend API before the message is sent to the user.
- **LLM-as-a-Judge**: A "Monitor LLM" (using a smaller, cheaper model like GPT-4o-mini) runs in parallel to score the primary model's responses for **Bias (B)** and **Hallucinations (I)**. If a quality score falls below 0.7, the interaction is automatically flagged for human review.

C. CI/CD Pipeline for LLM Assets

Prompt engineering and model weights must be treated as code.

- **Prompt Versioning**: Using **Git** or **LangSmith** to version-control system prompts. Any change to a prompt triggers a "Shadow Deployment" where the new prompt's outputs are compared against the old one using a golden test set.
- **Automated Testing**: Jenkins/GitHub Actions pipeline that runs unit tests on the **Entity Extraction** logic (e.g., ensuring ORD-12345 is always correctly parsed).
- **Monitoring (Observability)**: Integration with **Weights & Biases (W&B)** or **MLFlow** to track **Model Drift**. We specifically monitor the "Rolling Average of Flags", if the density of 'I' (Inaccuracy) or 'L' (Length) flags increases, an alert is triggered for the engineering team to re-tune the base model.

D. Tools & Technology Stack

- **Orchestration**: Apache Airflow
- **API Framework**: FastAPI
- **Monitoring**: MLFlow / Grafana
- **Deployment**: Docker & Kubernetes (K8s)
- **LLM Ops**: OpenAI API / LangChain

Strategic Business Impact

By implementing this automated ETL and CI/CD strategy, SkyRocket can achieve:

1. **Zero-Downtime Updates**: Deploying new bot intents without disrupting the 100-agent support team.

2. **Proactive Scaling**: The system identifies "Emerging Trends" (via HDBSCAN clustering) automatically, allowing the team to build new automation flows *before* the volume spikes.
3. **Quality Guarantee**: Reducing the human audit load by automating the detection of bias and hallucinations in 95% of conversations.

LLM Tuning & OpenAI API Strategy

To solve the high inaccuracy and bias rates identified in our audit, we propose a **Tiered Classification and Augmentation Strategy**. This approach moves the system from generic responses to high-precision NLU.

A. Two-Stage Context-Aware Prompt Design

Instead of a single-step classification, we implement a **Chain-of-Thought (CoT)** prompt that forces the model to reason before labeling. This is designed to eliminate "Intent Hallucinations" (Flag I).

The Proposed System Prompt:

> **Role:** You are the Senior NLU Architect for SkyRocket Retail. **Context:** Your goal is to classify customer queries to trigger specific backend API workflows.
>
> **Step 1 - Analysis:** Briefly explain the customer's core frustration and what data they are providing (e.g., "Customer is angry about a delayed refund and provided an Order ID"). **Step 2 - Category:** Select the most relevant category from: [ORDER, REFUND, ACCOUNT, etc.]. **Step 3 - Constraint Check:** Does the response require a placeholder? If YES, use {{Order Number}}. **NEVER** invent a specific number or currency symbol (Flag V Prevention).
>
> **Output Format (JSON):** { "analysis": "...", "top_intent": "...", "sub_category": "...", "entities": {"id": "...", "amount": "..."}, "requires_agent": boolean }

B. Synthetic Data Generation for "The Long Tail"

Our data exploration identified a "Long Tail" of low-volume topics like **Registration Issues (4.2%)** and **Newsletter Subscriptions**. To prevent the model from underperforming on these, we utilize the OpenAI API for **Data Augmentation**.

- **Approach:** We use the gpt-4o model to generate 100 synthetic variations of queries for each low-sample topic.
- **Prompt for Generation:** *"Generate 10 diverse customer queries for the 'Registration Issue' category. Include variations with typos, different levels of frustration, and various devices (mobile vs desktop) to ensure model robustness."*
- **Impact:** This balances the training set, allowing our classifier to achieve high F1-scores even on rare queries, preventing them from defaulting to the generic "Account Management" cluster.

C. Automated Evaluation Pipeline (LLM-as-a-Judge)

To monitor for **Model Drift** and **Bias** without manual labor, we integrate OpenAI API calls directly into our CI/CD testing pipeline.

1. **The Judge Model**: A separate, higher-parameter model acts as an auditor for the production model's output.
2. **The Logic**: For every batch of responses, the Judge model checks for the presence of our identified audit flags:
    - **Inaccuracy Check**: *"Does this response accurately address the query, or is it a hallucination?"*
    - **Bias/Tone Check**: *"Is the tone appropriately professional, or does it exhibit 'Toxic Positivity' (Flag B)?"*
    - **Variable Integrity**: *"Did the model incorrectly replace a {{Placeholder}} with a hardcoded value?"*

3. **Thresholds**: If the "Judge" scores a batch below 90% accuracy, the automated pipeline blocks the deployment and alerts the engineering team.

D. Strategic API Integration Roadmap

- **Short-term**: Use gpt-4o-mini for high-speed, cost-effective intent classification.
- **Medium-term**: Implement **Few-Shot Prompting** by dynamically injecting the "5 Representative Queries" (from Task 2) into the prompt context to ground the model.
- **Long-term**: Fine-tune a specialized model (e.g., GPT-4o fine-tuning) using the audited "Gold Standard" responses from the GenAI_responses dataset that were flagged as high-quality.

By using OpenAI not just for "answering" but for **augmenting** our data and **judging** our quality, we transform the chatbot from a "text generator" into a **Strategic BI Asset**. This framework directly reduces the 24.5% inaccuracy rate by enforcing structural constraints and balancing the dataset before the first customer even sees a response.

Methodology and Assumptions

This project was executed with a focus on **Scalability, Observability, and Business ROI**, leveraging industry-standard ML and LLM frameworks.

A. Technical Methodology

- **Data Exploitation & Clustering**: We utilized **Sentence Transformers** (all-MiniLM-L6-v2) to convert 6,539 raw queries into high-dimensional semantic embeddings. **K-Means ($k=10$)** was used for stable topic segmentation, while **HDBSCAN** was applied to identify emerging, density-based sub-trends.
- **Sentiment Analysis**: Sentiment polarity was calculated for each cluster to quantify customer "Friction Points," allowing us to prioritize automation for high-frustration categories like *Refunds* and *Escalations*.
- **Response Audit**: A heuristic and flag-based analysis of the GenAI_responses.csv was performed to identify systemic failures in the current LLM output (Length, Bias, and Inaccuracy).
- **Automation Stack**: The proposed architecture utilizes **FastAPI** for real-time inference, **Apache Airflow** for ETL orchestration, and **Weights & Biases** for monitoring model performance and drift.

B. Key Assumptions

- **Placeholder Integrity**: I assumed that placeholders like {{Order Number}} and {{Currency Symbol}} are mandatory variables. Any instance where a model hardcoded these (e.g., using $ instead of the symbol variable) was categorized as a **Hallucination (Flag V)**.
- **Agent Productivity Baseline**: I assumed the current 100-agent team is focused on repetitive tasks. Our ROI calculations assume that 100% of "Neutral" sentiment transactional queries are candidates for full automation.
- **Escalation Logic**: I assumed that queries containing direct intent for a "human" or "agent" should bypass the standard bot flow and trigger a high-priority "Warm Handoff" to maintain customer trust.