Hackathon Challenge: ”Smart-Support” Ticket Routing Engine

System Design & NLP Track

48-Hour Implementation Window

1 Scenario Overview

Your company (a major SaaS provider) receives thousands of support tickets daily. Currently, manual triage is the bottleneck. You are tasked with building a high-throughput, intelligent routing engine that categorizes tickets, detects urgency, and manages agent load while remaining resilient to system failures and ”ticket storms.”

—

2 Milestone 1: The Minimum Viable Router (MVR)

Goal: Establish a functional end-to-end pipeline.

- **ML Component:** Implement a baseline classifier to route tickets into three categories: *Billing, Technical,* or *Legal.* Use a regex-based heuristic for urgency (e.g., flags for ”broken” or ”ASAP”).
- **System Component:** Create a REST API (FastAPI/Flask) that accepts a JSON payload. Store incoming tickets in an **in-memory priority queue** (e.g., Python’s `heapq`).
- **Concurrency:** Single-threaded execution is acceptable for this phase.

—

3 Milestone 2: The Intelligent Queue

Goal: Transition to production-grade reliability and deep learning.

- **ML Component:** Replace the baseline with a **Transformer-based model**. Implement a regression-based Sentiment Analysis model to generate a continuous urgency score $S \in [0, 1]$.
- **System Component:** Move to an **Asynchronous Broker** architecture using Redis or RabbitMQ. The API must return a `202 Accepted` immediately while background workers process the queue.
- **Concurrency:** The system must handle 10+ simultaneous requests at the exact same millisecond using **atomic locks** to prevent race conditions or duplicate ticket processing.
- **Integration:** Trigger a mock **Slack/Discord Webhook** for any ticket where $S > 0.8$.

—

4 Milestone 3: The Autonomous Orchestrator

Goal: Build a self-healing, "agent-aware" system capable of handling "Flash-Floods."

4.1 ML: Semantic Deduplication

To prevent "ticket storms" during outages, use **Sentence Embeddings** to calculate the **Cosine Similarity** between incoming tickets:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{1}$$

If similarity > 0.9 for more than 10 tickets in 5 minutes, the system must **suppress** individual alerts and create a single "Master Incident."

4.2 System: Circuit Breakers & Load Balancing

- **Circuit Breaker:** If the Transformer model latency exceeds 500ms, the system must automatically **failover** to the lightweight Milestone 1 model.
- **Skill-Based Routing:** Maintain a stateful registry of agents with **Skill Vectors** (e.g., Agent X is 90% Tech, 10% Billing). Solve a **Constraint Optimization** problem to route tickets to the best available agent based on their current capacity and skill match.

—