PeerSupportBot — AI Agent Architecture

1. Introduction

When I designed **PeerSupportBot**, I wanted it to feel less like a “content filter” and more like a **supportive agent** inside a community. That meant the architecture couldn’t just be a single classifier spitting out yes/no answers. It needed **multiple agents**, each with a role, working together in a flow that balances **precision**, **empathy**, and **safety**.

---

2. Core Components

🛡️ Sentinel (Detection Agent)

- **Purpose:** First line of defense. Watches every incoming message and runs it through the AI models.
- **Models Used:**
    - **Toxicity Detector** (DistilBERT + LoRA fine-tuned on Jigsaw) → picks up slurs, harassment, threats, obscene language.
    - **Sarcasm Detector** (fine-tuned BERTweet) → catches irony and playful tone so the system doesn’t overreact.
- **Why this choice:** I wanted a lightweight but sharp sentinel—something that can quickly read the “temperature” of a message and hand off enriched information (toxicity scores, sarcasm probability, seriousness estimate).

⚖️ Triage (Decision Agent)

- **Purpose:** Takes the raw signals from Sentinel and decides the *type* of case: harmless, toxic, serious, or crisis.
- **Logic Applied:**

    - Weighted **seriousness score** (toxicity labels + sarcasm relief + user/channel context).
    - **Policy overrides** for crisis/self-harm phrases and extreme words like *rape, kill, n-word*.
- **Why this choice:** I didn't want the bot to act impulsively. Triage ensures each decision is **context-aware** and errs on the side of caution in high-risk cases.

💬 Responder (Interaction Agent)

- **Purpose:** Shapes the **human-facing response** once an action is chosen.
- **Actions:**
    - **Redact message** (delete or replace with "[message redacted]").
    - **DM user** with an empathetic explanation or crisis resources.
    - **Final warning** in both DM and channel if violations >5.
- **Why this choice:** A moderation system isn't useful if it just deletes content. I wanted it to **communicate with empathy**—explaining the "why" behind actions so users don't feel silenced by a black box.

📜 Archivist (Logging & Reporting Agent)

- **Purpose:** Keeps the long-term memory of incidents and generates accountability artifacts.
- **Functions:**
    - Logs every incident in SQLite (user hash, channel, message excerpt, scores, action).
    - Generates **daily reports** (23:59 IST), **rolling reports** (every 50 incidents), and **special reports** for repeat violators.

- **Why this choice:** In sensitive communities, transparency matters. The Archivist ensures moderators have data they can review, not just actions taken silently.

---

3. Interaction Flow

Here's how a single message moves through the system:

```
flowchart TD
    A[Incoming Message] --> B[Sentinel: Run toxicity + sarcasm models]
    B --> C[Triage: Compute seriousness + apply crisis/extreme rules]
    C -->|Crisis detected| D[Responder: Redact + DM with crisis
resources]
    C -->|Serious toxicity| E[Responder: Redact + DM warning]
    C -->|Moderate toxicity| F[Responder: Redact + DM softer warning]
    C -->|No action| G[Archivist: Log only]
    D --> H[Archivist: Log incident + update DB + report]
    E --> H
    F --> H
    G --> H
    H --> I[Report Generation (daily/rolling/special)]
```

---

4. Models and Their Roles

Model	Role in System	Why Chosen
DistilBERT + LoRA (fine-tuned on Jigsaw)	Multi-label toxicity (toxic, severe_toxic, obscene, threat, insult, identity_hate)	Lightweight, fast, adaptable to custom thresholds, captures multiple flavors of toxicity.
BERTweet (fine-tuned sarcasm detector)	Sarcasm probability (0–1)	Sarcasm is common in peer groups; without this, benign jokes would be over-flagged.
Policy Rules (regex + heuristics)	Crisis/self-harm detection, extreme toxic words	No ML model is perfect; hard-coded safety nets guarantee coverage for high-stakes cases.

---

5. Why This Architecture Works

- **Separation of concerns:** Each agent has a single responsibility—detection, decision, response, or logging. This makes the system more transparent and debuggable.
- **Hybrid AI + Rules:** I didn't want to rely blindly on models. The regex safety overrides catch things ML sometimes misses.
- **Human-centred design:** Redaction + empathetic DM feels like guidance, not punishment. Reports empower moderators with oversight.
- **Scalable foundation:** Because it's agent-based, I can add new agents (e.g., a Wellbeing agent with `/pause` commands, or a Dashboard agent) without breaking the core flow.

---

6. Conclusion

The architecture reflects a balance: **AI for nuance, rules for reliability, and agents for empathy + accountability**. The result is a bot that doesn't just "moderate"—it **supports**. That was the north star guiding every design choice.