FleetBot (Student): A School Q&A Chatbot Shipped to 1,500 Students

A student-facing AI chatbot that answers everyday school questions, like the block rotation, who teaches what, and which clubs exist, from the school's own public documents. Launched school-wide through the AI Club Instagram.

Role: Sole builder, covering bot logic, prompt system, RAG data pipeline, and website **Timeline:** built through Jan 2026 · launched ~Jan 11, 2026 **Stack:** Botpress (deterministic flow) · Lovable (website) · GPT-4o Mini · GPT-5 Nano · RAG

- Ran a full public beta on **$2.76** of total AI cost across **227 messages** and **1,007 LLM calls**, with **0 errors**.
- Reached **112 unique visitors** (~7.5% of the school) in the launch window; **66** went on to chat with the bot.
- Cut runaway cost from **$5 per 20 messages** (autonomous agent) to cents per message by moving to a deterministic flow and tuning retrieval from **50 to 12 chunks**.

The Problem

At my school, everyday info like the daily block rotation, who the counsellors are, what clubs exist and the school map is buried in PDFs on the school website that nobody opens. Students dig through them or walk to the office and ask, and the office answers the same questions over and over. There was no fast, phone-friendly way to just ask a question and get an answer. Success meant a student could type "What's today's block order?" and get an instant, plain-English answer.

My Role

Built and owned the entire project: the bot logic, prompt system, RAG data pipeline, bug fixes, and the Lovable website it runs on. Solo build.

Key Decisions

1. Three platforms before landing on Botpress

The challenge: The bot had to be fast, cheap enough for a student budget, and private enough for a school.

Options considered: Relevance AI, rejected after a single answer took over 20 seconds. Voiceflow, where I built a working prototype, but when I showed it to the principal she flagged that it sent data to third-party servers the school couldn't control. Botpress, which gave full control over logic and data and was cheap to run.

What I chose and why: Botpress. The tradeoff was more manual wiring, but it was the only option that met the speed, cost, and privacy limits at once, and it became the platform for the later staff version too.

2. Deterministic flow over an autonomous agent

The challenge: The first version used an autonomous agent that kept looping until it "solved" a question. In a stress test it burned **$5 in 20 messages**, which is not sustainable on a student budget.

Options considered: Keep the autonomous agent, which is flexible but unaffordable and unpredictable. Or rebuild as a deterministic "if this, then that" flow, which is cheap and predictable, but every path has to be engineered by hand.

What I chose and why: The deterministic flow, plus cutting the RAG chunk limit from **50 to 12**, the point where accuracy held but the bot stopped reading far more text than it needed. Cost dropped to cents per message. To keep short follow-ups working ("What about yesterday?"), I added a 5-message memory window through a GPT-4o Mini extractor.

3. A system prompt built against the model's failure modes

The challenge: Left to its defaults, the model answered from its own training the moment it recognized a topic, invented a fake schedule on weekends, and tried to write code mid-conversation to count things. Each one produced a confident, wrong answer.

Options considered: Trust the model's defaults, which is fluent but unreliable. Or write explicit rules for each failure mode.

What I chose and why: Hard grounding, where every sentence has to trace back to a retrieved document chunk or it gets deleted. On top of that: an ordered schedule check (weekend → holiday → non-instructional day → early dismissal → regular day, stopping at the first match) that killed the weekend hallucination; a no-code rule; typed fallbacks instead of one generic "I don't know"; and style rules that ban the generic-AI tells. The bot refused 100% of unsafe questions in testing.

The Solution

A phone-friendly Q&A bot living on a Lovable site and shared through the AI Club Instagram, answering block orders, staff, and club questions from the school's own PDFs via live RAG search (see: end-to-end screen recording, node-architecture screenshot). The whole path is one funnel, an Instagram link to a landing page to the chatbot, so a student goes from curiosity to answer in a couple of taps.

The reliability came from specific fixes: a custom JavaScript card that resets the topic variable every turn (killing a bug where the bot got stuck on the previous subject), a graceful GPT-5 Nano fallback that offers a nearby answer instead of a dead-end "I don't know," and a crowd-control rule that caps long lists at five items and asks the user to narrow down.

Results & Impact

The launch window drew **112 unique visitors** (~7.5% of the ~1,500-student school): **77** from Instagram, **31** direct, and **2** from Facebook, at an 87% bounce rate and ~44-second average visit. Of those, **66** chatted with the bot, generating **227 messages** across **1,007 LLM calls**, for **$2.76** in total AI cost (see: Botpress dashboard, website analytics). The 112-to-66 step is the curiosity-to-use gap of a cold launch, where people clicked the link, glanced, and some left before typing.

The cost result is the one I'd point a skeptic to: the autonomous agent's **$5 per 20 messages** would have made a public beta impossible, and the deterministic rebuild plus a 12-chunk retrieval limit brought it down to cents per message and held it there for the entire run.

What Came Next

The live beta and its real usage numbers were the proof that unlocked the pitch for a bigger, staff-facing version, which became a system deployed on a physical terminal in the school office, built on the same platform.