Reddit LLM Pipeline Optimization Report

🛠️ Problem Statement

You have a daily data pipeline that collects approximately 700 Reddit posts related to automotive issues. These posts are passed to an LLM cleaning phase that extracts structured problem–solution pairs in JSON format. The LLM is run locally via Ollama, hosted on a GitHub-connected device without access to a GPU.

Currently, the process runs serially and takes over 30 hours to complete a single day's batch, rendering the system unusable for real-time or near-real-time production deployment.

🔍 Root Causes

- • Serial Processing: Posts are sent to the LLM one by one, without any batching or parallelization.
- • No GPU Acceleration: The local LLM runs on CPU, significantly slowing down token generation and response time.
- • No External API: Cost constraints prevent the use of paid LLM APIs like OpenAI or Anthropic.
- • Limited Hosting Options: The system avoids cloud GPU platforms due to budget limits.
- • Captchas/Bot Blockers: Attempting automation via unofficial browser automation could face anti-bot protections (e.g., Cloudflare, CAPTCHA).
- • Resource Bottlenecks: Limited threads, memory, or CPU on the hosting machine may further throttle performance.

🧩 Solution Options Comparison Table

Solution Option	💰 Monetary Cost	🖥️ Hardware Requirements	🕒 Dev Time	🔧 Maintenance Overhead	✅ Feasibility	✅ Pros	⚠️ Cons	🔍 Notes / Hidden Risks
1. Use Official APIs	$20–$100+/mo	No hardware needed	Low	Low	High	Stable, reliable	Ongoing cost in USD	Budget constraints if prices rise
2. Lightweight Queue System + Micro-batching	$0	Existing machine	Medium	Medium	High	Modular and scalable	Still bound by CPU-only limits	Needs smart batching + error handling
3. Deploy to a More Powerful Local Machine	$200–$1000 one-time	New device (more RAM/CPU or GPU)	Low	Low	Medium	Fastest local option	Upfront cost	Hardware must be well configured for Ollama
4. Browser Automati	$0–$20/mo (for	No GPU needed	High	High	Medium	Can leverage powerful	Captchas, bot blocking	Proxy rotation, potential

on for Web LLMs (Playwright)	proxies)					LLMs for free		legal risk
5. Use Reverse-Engineered APIs (⚠️ Risky)	$0	Internet connection only	Medium	High	Low	Access to high-quality models for free	High ethical, legal, and stability risks	May get blocked anytime
6. Hybrid Model Design (Local + Remote)	$0–$10/mo	Small local model + Ollama	High	Medium	Medium	Offload easy cases, optimize cost	Complex to implement	Needs intelligent routing between models
7. Parallel Local Execution with Ollama	$0	Multi-core CPU	Medium	Low	High	Fast improvement without new hardware	Limited by CPU speed	Needs careful multiprocessing to avoid overloading

📌 Conclusion

After evaluating all the above options, **Solution 1: Use Official APIs** emerges as the most feasible for this project. It offers a stable and scalable pathway forward with minimal hardware dependencies and development time. Although it incurs a monetary cost, the reliability and speed gains far outweigh the expenses, especially if real-time performance is critical.