Product Case Study: "JoelGPT"

Problem

In November 2025, major cloud providers like Cloudflare and AWS experienced an outage that caused globally-used applications — especially Large Language Models (LLMs) like ChatGPT and Claude— to suddenly become unavailable.

This exposed a critical issue:

> **We are over-dependent on cloud-hosted AI for essential tasks like writing, learning, and productivity**.

Users were left without access to tools they rely on daily, even for simple tasks.
There needs to be an alternative that:

- Runs fully offline
- Is cost-efficient
- Supports basic LLM capabilities
- Doesn't stop working just because the internet does

The Mission

Design and deploy a locally hosted chatbot with real-time search functionality and a lightweight RAG (Retrieval-Augmented Generation) pipeline, optimized for minimal compute usage while ensuring reliable and efficient performance.

Target Users

User Persona	What They Need	How JoelGPT Helps
Individual Creators	Reliable writing assistant anytime	Offline text generation & summaries

Fresh Graduates & Job Seekers	Tools for resume writing, emails, interview prep	Instant content creation without paid services
College Students	Research help & personalized knowledge access	RAG for personal notes + Web updates command

No GPU farms. No cloud bills. Everything executed locally.

Requirements

Functional Requirements:

- Local LLM inference (offline text generation)
- Ability to answer user-specific context using a RAG pipeline
- Ability to search internet on demand using a /search <topic> command
- PDF ingestion to build personal knowledge base
- Automatic fallback to local model if online search fails

Non-Functional Requirements:

- Low computational overhead — run on a consumer laptop
- Full privacy — data never leaves device
- Modular architecture for future model upgrades
- Simple CLI interface for quick interaction

Execution & Architecture

1) Local LLM Model (Core Brain)

- Uses lightweight models with Ollama (e.g., Llama 3:1B)
- Handles general queries, writing, and creative tasks

2) Retrieval-Augmented Generation (Memory Layer)

- Users upload personal PDFs (like notes, resumes, docs)

- System extracts text into a knowledge store
- When queries match uploaded content → it retrieves relevant chunks
- Model produces responses based on both:
    - User query
    - Matching personal context

3) Web Search & Scraping (Live Knowledge)

- Activated with command: /search <topic>
- Uses DuckDuckGo Search + BeautifulSoup to:
    - Fetch live info (sports results / stock price / breaking news)
    - Summarize into concise readable format
- Appends sources into LLM prompt for accuracy

System Workflow

User Input
↓
Check if `/search` command?
↓ Yes → Web Search → Scrape → Summary
↓ No
PDF Context (RAG)
↓
LLM Prompt Construction
↓
Local Inference via Ollama
↓
Response (Text + Optional Voice)

Outcome

- A self-reliant AI assistant that never goes down due to cloud outages
- Secure and data-private, ideal for personal and academic use
- Customizable and extendable for future improvements (UI, embeddings DB, model upgrade)

Future Scope

Feature	Value
GUI (Streamlit / Electron)	Wider usability
Voice activation	Hands-free interaction
On-device speech recognition	Full offline conversation
Model scheduling & power optimization	Longer use on lower hardware

---