Gatsby QA System Report

Introduction:

The Gatsby QA System is a proof-of-concept interview project designed to demonstrate a full end-to-end Retrieval-Augmented Generation (RAG) pipeline. Built around The Great Gatsby PDF, the system allows users to pose natural-language questions about the novel and receive concise, grounded answers displayed as interactive flashcards.

Architecture and Data Flow

At startup in in_p.py, it extracted the full text of The Great Gatsby using the pdfminer.six library. The text is segmented into overlapping chunks of approximately 500 tokens with 100 tokens of overlap to retain context. Each chunk is encoded into a vector embedding using a sentence-transformer model (all-mpnet-base-v2) and indexed with FAISS (Facebook AI Similarity Search) for efficient similarity retrieval. The resulting index (gatsby_index.faiss) and the serialized list of chunks (chunks.pkl) enable sub-second nearest-neighbor lookups.

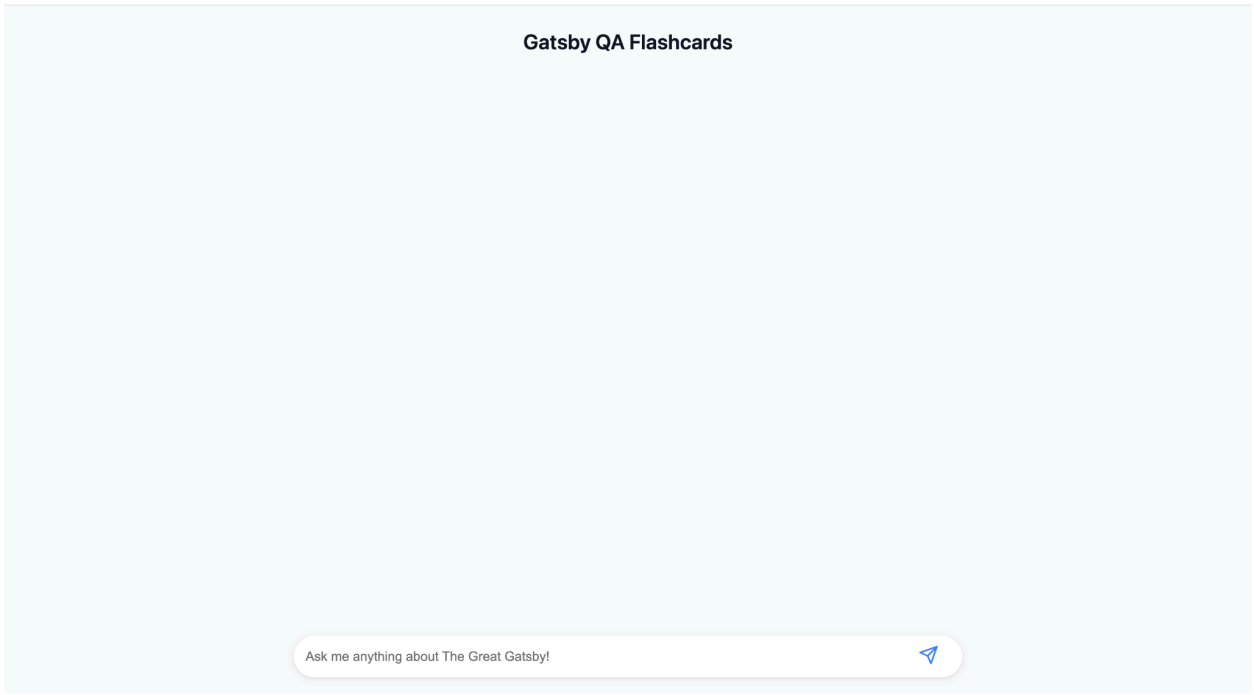
A FastAPI backend (main.py) exposes a single POST endpoint, /ask, which accepts a JSON payload containing the user's question. When a request arrives, the backend retrieves the top-K most relevant chunks from FAISS, re-ranks them with a lightweight CrossEncoder to improve precision, and feeds the top passages into a quantized LLaMA-2 7B model via Hugging Face's transformers pipeline (text-generation). To bound latency, generation is capped at 60 new tokens. The model's 1-2 sentences' output is then split on periods into a small set of micro-answers (each roughly 20–30 words) that form the contents of the flashcard.

Frontend Design and User Interaction:

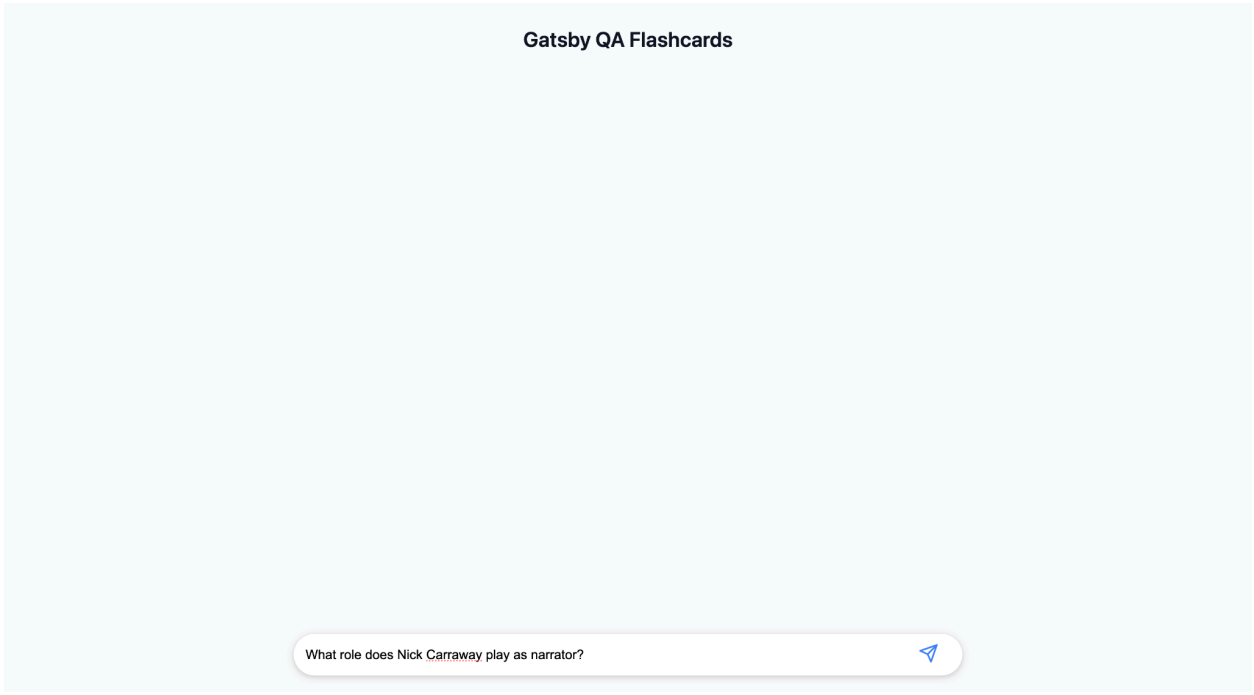
The user interface is a single static HTML page styled with Bootstrap and custom CSS, and powered by plain JavaScript. Upon loading, the page displays a header bearing the title "Gatsby QA Flashcards," followed by a scrollable conversation area. A fixed input bar at the bottom lets users type questions and submit via a send-button click. When submission occurs, the app immediately appends the user's question in a gray chat bubble at the top of the conversation, then displays a "skeleton card" loader—three animated gray bars that shimmer to indicate processing.

Behind the scenes, a fetch('/ask') call carries the question to the backend. When the JSON response arrives, the loader is removed and replaced by a single flashcard containing a bulleted list of the model's micro-answers. Each list item is prefixed with a green check mark, and the card itself is styled to grow vertically as needed, with a subtle hover effect for interactivity.

Gatsby QA Flashcards



Gatsby QA Flashcards



Gatsby QA Flashcards

What role does Nick Carraway play as narrator?

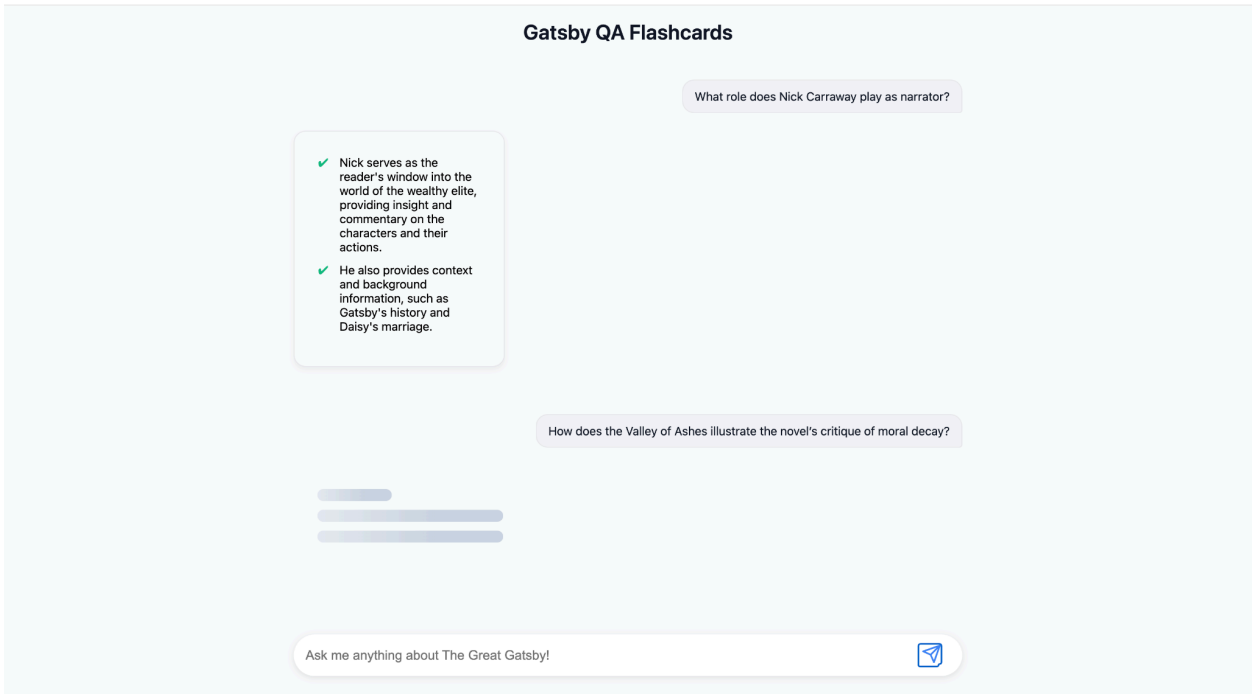
Ask me anything about The Great Gatsby!

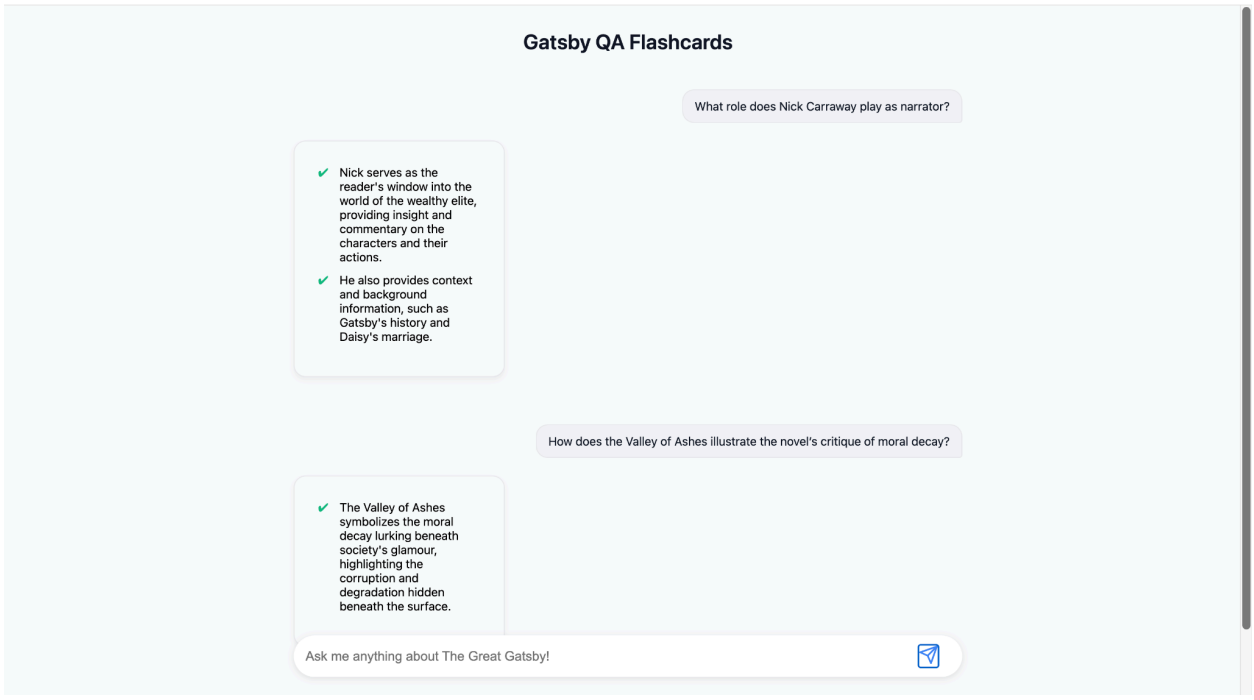
Gatsby QA Flashcards

What role does Nick Carraway play as narrator?

- ✔ Nick serves as the reader's window into the world of the wealthy elite, providing insight and commentary on the characters and their actions.
- ✔ He also provides context and background information, such as Gatsby's history and Daisy's marriage.

Ask me anything about The Great Gatsby!


Gatsby QA Flashcards
What role does Nick Carraway play as narrator?
✔ Nick serves as the reader's window into the world of the wealthy elite, providing insight and commentary on the characters and their actions.
✔ He also provides context and background information, such as Gatsby's history and Daisy's marriage.
How does the Valley of Ashes illustrate the novel's critique of moral decay?
Ask me anything about The Great Gatsby!


Gatsby QA Flashcards
What role does Nick Carraway play as narrator?
✔ Nick serves as the reader's window into the world of the wealthy elite, providing insight and commentary on the characters and their actions.
✔ He also provides context and background information, such as Gatsby's history and Daisy's marriage.
How does the Valley of Ashes illustrate the novel's critique of moral decay?
✔ The Valley of Ashes symbolizes the moral decay lurking beneath society's glamour, highlighting the corruption and degradation hidden beneath the surface.
Ask me anything about The Great Gatsby!

Performance Evaluation and Iteration:

Initial experiments on a local Intel Core i7 CPU without GPU proved impractically slow: embedding with all-mpnet-base-v2, indexing with FAISS, and generation with google/flan-t5-base each took 10–20 seconds per query, and the quality of answers was unsatisfactory. Migrating to Google Colab's T4 GPU reduced latency to around 5–6 seconds, but results remained off-topic or vague. Downsizing to flan-t5-small improved speed modestly yet offered no accuracy gains; scaling up to flan-t5-large did not resolve the grounding issues.

Ultimately, I adopted a quantized LLaMA-2 7B model (4-bit NF4) for generation. On the T4 GPU, end-to-end response times averaged 5–6 seconds, with clearly more accurate and context-relevant answers. Upgrading to Colab's A100 GPU further cut latency to 2.5–3.5 seconds per query. Setting max_new_tokens to 60 prevented uncontrolled token growth and kept the service responsive.