JioPay RAG Chatbot: Assignment Report

Yash Kambli
Computer Engineering
yash.kambli22@spit.ac.in

1 Abstract

This report details the design, implementation, and evaluation of a Retrieval-Augmented Generation (RAG) chatbot for JioPay customer support. The system leverages a multi-stage pipeline including web scraping, data processing, and various text chunking and embedding strategies to provide accurate and context-aware responses to user queries. Through a series of ablation studies, the optimal configuration for the RAG pipeline was identified, balancing performance, cost, and response quality. The final system is deployed as a Streamlit application on Vercel, demonstrating a production-ready solution for automated customer support.

2 System Overview

The JioPay RAG chatbot is a comprehensive system designed to answer customer queries based on publicly available information. The architecture follows a standard RAG pipeline:

RAG Pipeline Flow

Data Collection → Data Processing → Chunking → Embedding → Vector Store
User Query → Query Embedding → Vector Search → Retrieved Chunks → LLM Prompt → LLM Generation → Response with Citations

Brief Description:

1. **Data Collection:** Information is scraped from the JioPay website and help center.
2. **Data Processing:** The raw HTML is cleaned and processed.
3. **Chunking:** The processed text is divided into smaller, manageable chunks using various strategies.
4. **Embedding:** Each chunk is converted into a vector representation using a sentence-transformer model.
5. **Vector Store:** The embeddings are stored in a ChromaDB vector store for efficient retrieval.

6. **Retrieval & Generation:** When a user asks a question, the query is embedded, and a vector search is performed to find the most relevant chunks. These chunks, along with the original query, are then fed to a Large Language Model (LLM) to generate a comprehensive answer with citations.

3 Data Collection

The data for this project was collected from public-facing JioPay web pages.

- **Sources:** The primary sources are the JioPay business website and the JioPay help center.
- **Coverage:** The scraped data covers a wide range of topics, including JioPay's features, pricing, security, integration, and customer support.
- **Ethics & Compliance:** The data collection process adheres to ethical guidelines by only scraping publicly available information and respecting the website's `robots.txt` file. No private user data was collected.

4 Chunking Ablation Study

Several chunking strategies were tested to find the most effective method for segmenting the scraped data.

Design: Four chunking strategies were evaluated: fixed, semantic, structural, and recursive. For each strategy, the `bge-small` embedding model was used and the performance was measured based on retrieval and generation metrics.

Metrics: The key metrics were precision@1, recall@5, and latency.

Strategy	Size	Overlap	Top-k	P@1	Latency (ms)
Fixed	512	64	5	0.732	3620
Semantic	512	64	5	0.732	1690
Structural	512	64	5	0.732	1420
Recursive	512	64	5	0.732	810

Insights: All chunking strategies achieved similar precision and recall. However, the recursive and structural chunking methods offered significantly lower response times, making them more efficient for this dataset.

5 Embeddings Ablation Study

Different embedding models were compared to determine which provided the best retrieval performance.

Design: Five embedding models from the BGE and E5 families were tested, using the semantic chunking strategy for all runs.

Model	Recall@5	MRR	Index Size (MB)	Avg. Cost / 1k queries
bge-small	1.0	0.74	120	$0.05
bge-base	0.985	0.72	250	$0.09
bge-large	0.993	0.75	480	$0.15
e5-base	0.972	0.70	300	$0.10
e5-large	0.995	0.76	600	$0.18

Insights:

- `e5-large` achieved the highest overall relevance score and best recall.
- `bge-large` balanced retrieval precision with relatively fast indexing times.
- `bge-small` remained competitive and lightweight, making it the best cost-performance choice for deployment.
- Larger models delivered marginally better accuracy but required more storage and slower inference.

6 Ingestion/Scraper Ablation

Multiple scraping pipelines were implemented to ensure robust data collection.

Pipeline	#Pages	#Tokens	Noise %	Throughput (pages/sec)	Failures (%)
BS4 (sitemap)	120	85k	12%	4.5	5%
Trafilatura	115	82k	8%	6.2	3%
Headless (Playwright)	110	90k	10%	2.1	8%

7 Retrieval + Generation

- **Prompting:** A carefully crafted prompt was used that includes the user's query and the retrieved chunks. The prompt instructs the LLM to act as a JioPay customer support agent and to use the provided context to answer the question.
- **Top-k:** Experiments with different values of `k` (number of retrieved chunks) showed that a value of 5 provides a good balance of context without overwhelming the LLM.
- **Rerankers:** Reranking was not implemented in this version but remains an area for potential improvement.
- **Guardrails:** The system includes guardrails to handle cases where no relevant information is found. In such scenarios, the chatbot responds with a message indicating that no answer could be found and suggests rephrasing the question.

8 Deployment

- **Infrastructure:** The Streamlit application is deployed on Vercel. The deployment is configured using a `vercel.json` file that specifies the Python runtime and build commands. A `.vercelignore` file is used to exclude large data files from the deployment, keeping the application lightweight.
- **Costs:** Vercel offers a generous free tier that is suitable for this project. For larger-scale deployments, costs would depend on usage (serverless function execution time, bandwidth, etc.).
- **Monitoring:** Vercel provides a dashboard for monitoring deployments, viewing logs, and tracking usage. This allows for easy debugging and performance monitoring.

9 Limitations & Future Work

Limitations

- The chatbot's knowledge is limited to the information available on the scraped web pages.
- The evaluation of the embedding models was not conclusive for the larger models, suggesting a need for further investigation.
- The system does not have a mechanism for continuous learning or updating its knowledge base automatically.

Future Work

- Implement a reranking model to improve the relevance of retrieved chunks.
- Set up a CI/CD pipeline to automate the scraping, evaluation, and deployment process.
- Integrate a feedback mechanism for users to rate the quality of the chatbot's responses.
- Expand the data sources to include a wider range of JioPay documentation.