# MANAGENIE PROJECT REPORT

20i-0847
21i-2746
21i-2692

# *MANAGENIE CHATBOT*

The dataset primarily comprises PDF documents uploaded by users. These documents are processed to extract text, which is then used as the data source for answering user queries.

**Here's the workflow**:

- PDF Upload and Processing: Users upload PDF files. The system extracts text from these PDFs using PyPDF2, which reads and extracts content page by page.
- Data Storage and Retrieval: Extracted text is chunked and stored in a vector database using Faiss, a library for efficient similarity search and clustering of dense vectors.

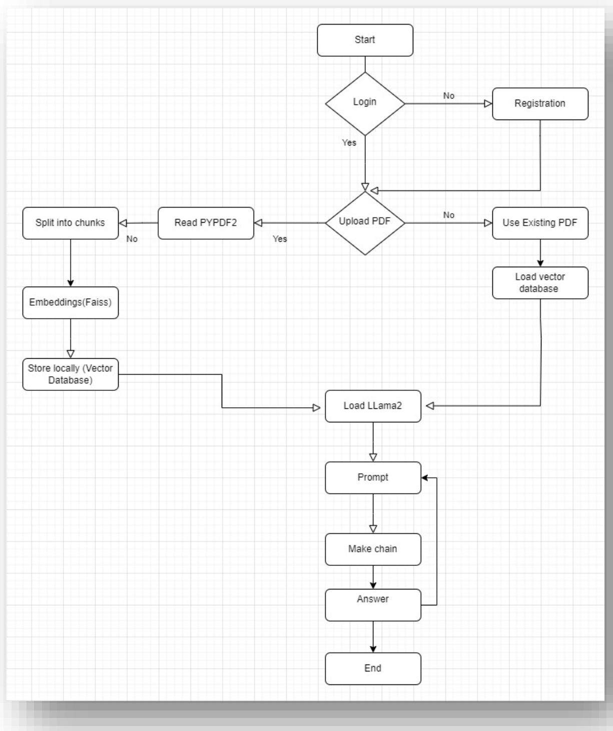
## Embeddings Explanation

Embeddings are critical for transforming text data into a format that can be efficiently used for similarity searches. This project uses HuggingFaceEmbeddings with a pre-trained model (sentence-transformers/all-MiniLM-L6-v2).

These embeddings are used to:

- Convert text data into dense vectors.
- Store and index these vectors in Faiss for quick retrieval.

## Flow Diagram:



## Model Explanation

The core of ManaGenie is a Retrieval Augmented Generation (RAG) model combined with LlamaCPP, an implementation for language models.

RAG: Combines a retriever and a generator to fetch relevant document chunks and generate responses based on these chunks.

LlamaCPP: Utilized for its generative capabilities, enabling the chatbot to provide coherent and contextually relevant responses based on retrieved information.

## Detailed Explanation of the Model Used in ManaGenie

The chatbot utilizes a hybrid model architecture incorporating both Retrieval Augmented Generation (RAG) and a transformer-based language model (LlamaCPP). This setup is chosen to leverage the strengths of both retrieval-based and generative NLP methodologies, enhancing the chatbot's ability to provide accurate and contextually relevant answers derived from user-uploaded documents. Here's a more detailed look at each component:

## Retrieval Augmented Generation (RAG)

RAG is a hybrid model that combines the capabilities of a retriever and a generator to handle queries:

- Retriever: This component is responsible for searching through a database of information (in this case, embeddings of text from uploaded PDFs stored using Faiss) to find and retrieve the most relevant content in response to a user query. The retrieval is based on the semantic similarity between the query and the stored document chunks, which are indexed in a highly efficient manner using Faiss.
- Generator: Once the relevant information is retrieved, the generator, which is a part of the LlamaCPP model, uses this context to construct a coherent and contextually appropriate response.

The RAG approach is beneficial because it allows the chatbot to directly leverage specific information from the documents, enhancing the relevance and specificity of responses. It addresses one of the primary challenges of generative models, which is generating plausible but incorrect information by grounding responses in the document content.

LlamaCPP is a C++ implementation of a transformer-based model, designed for efficient operation on GPU hardware. In ManaGenie, LlamaCPP serves several roles:

Language Understanding and Generation: As a transformer model, LlamaCPP excels in understanding context and generating human-like text. It can contextualize the conversation based on the previous dialogue and the retrieved content to generate responses that are not only relevant but also fluid and natural.

Customization and Efficiency: LlamaCPP is optimized for high-performance scenarios, supporting faster computations which are crucial for real-time applications like chatbots. Its implementation allows for tuning of parameters such as number of GPU layers used, batch size, and context window size, making it highly adaptable to specific needs of the application.

We chosed these models because, the choice of RAG and LlamaCPP in ManaGenie is driven by the need to effectively handle complex user queries regarding document content. Key reasons include:

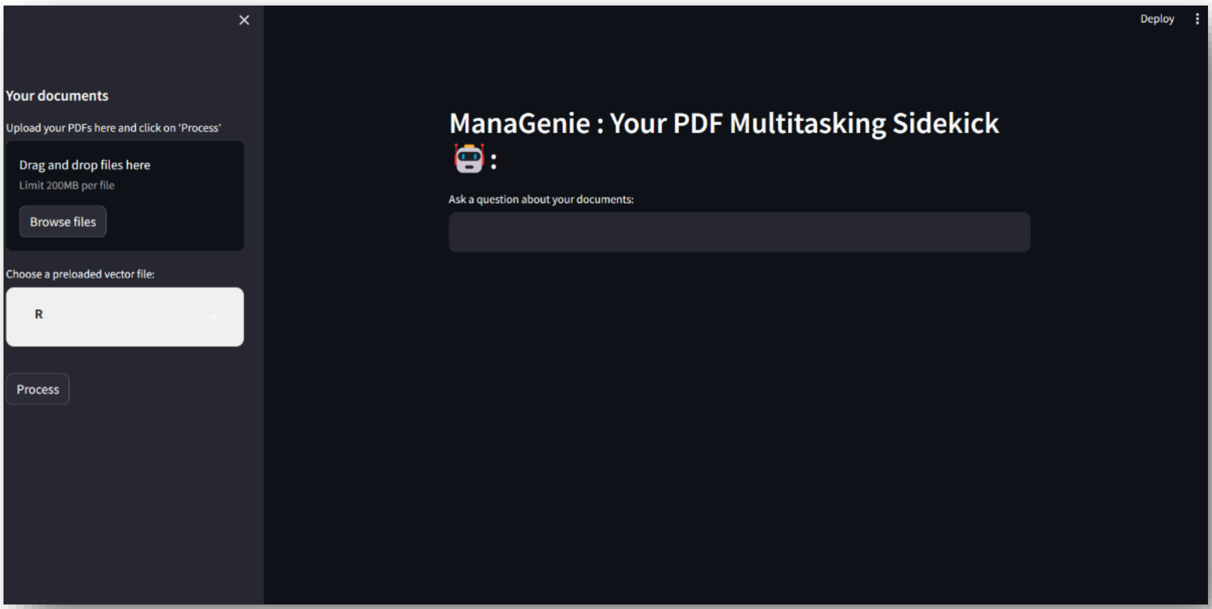
- Accuracy and Relevance: By combining retrieval and generative approaches, the system ensures that the responses are not only relevant but also accurate, as they are directly based on the document's content.
- Scalability and Performance: Both Faiss and LlamaCPP are designed for high scalability and performance, which is essential for managing large datasets (i.e., many uploaded PDFs) and delivering fast response times.
- Coherent Long-Form Responses: LlamaCPP's capability to handle extended contexts makes it suitable for generating coherent longer responses that are necessary when dealing with detailed document-based queries.

In summary, the integration of RAG with LlamaCPP provides a robust framework for a chatbot designed to interact intelligently with user-provided documents. This combination allows ManaGenie to deliver precise answers grounded in the documents' content while maintaining a conversational and natural interaction style.

## The GUI:

The frontend of the ManaGenie application, as shown in the provided screenshot, is developed using Streamlit, a powerful yet straightforward framework for building interactive web applications purely with Python. The user interface is clean and divided into two main sections: a sidebar for document management and a main area for user interaction.

Sidebar for Document Management: This section allows users to upload PDF documents, either by dragging and dropping files or using the browse option, with a file size limit per document. It also provides a dropdown menu where users can select from preloaded vector files, which represent previously processed documents. The "Process" button initiates the processing of selected or uploaded documents, making their content available for querying by the chatbot.


Deploy
Your documents
Upload your PDFs here and click on 'Process'
Drag and drop files here
Limit 200MB per file
Browse files
Choose a preloaded vector file:
R
Process
ManaGenie : Your PDF Multitasking Sidekick
:
Ask a question about your documents:

Main Interaction Area: Here, users can enter questions related to the documents they have uploaded or selected. This text input box is where the conversational interface comes into play, allowing users to query the content of their documents dynamically.

Session State and Chat History: Streamlit's session state is leveraged to keep track of the ongoing conversation as well as to store the chat history. This means that not only can the chatbot maintain context over the course of a session, enhancing its ability to provide relevant responses based on previous interactions, but it also retains a history of past chats. This is useful for users who may want to revisit previous questions and answers, providing a richer, more useful interaction history with the system.

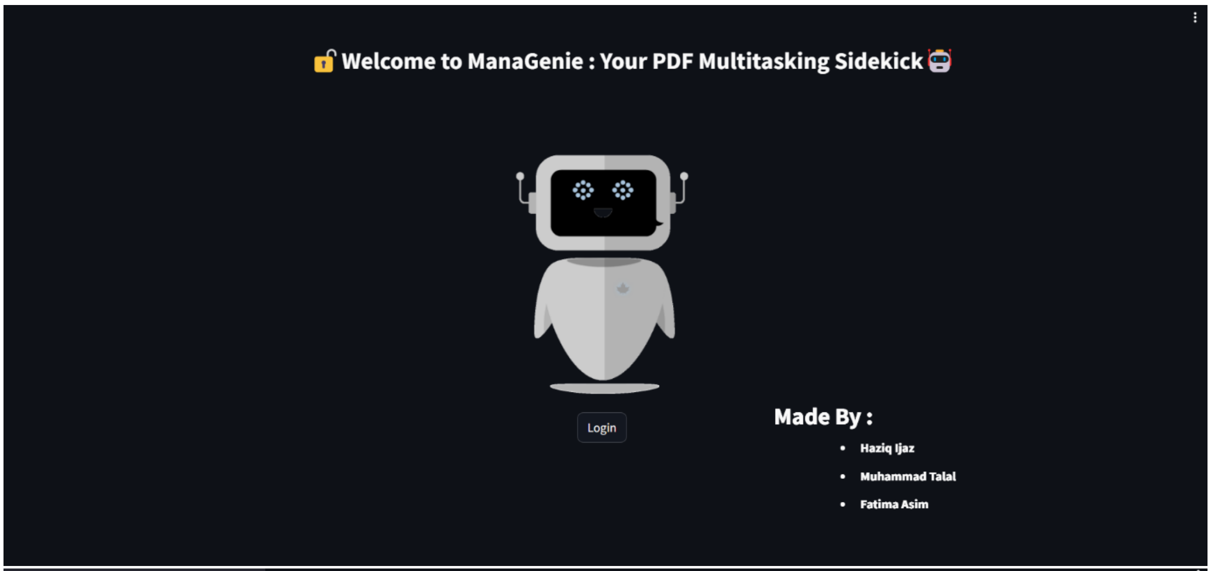
## Strengths:

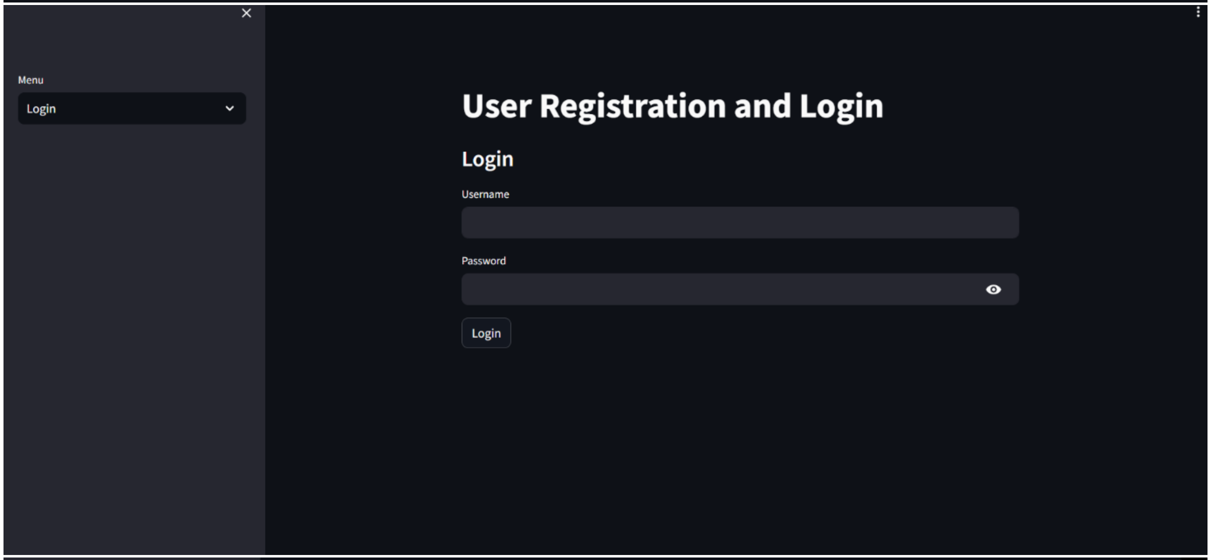
Contextual Awareness: Utilizes recent conversational context for generating responses, ensuring relevance.

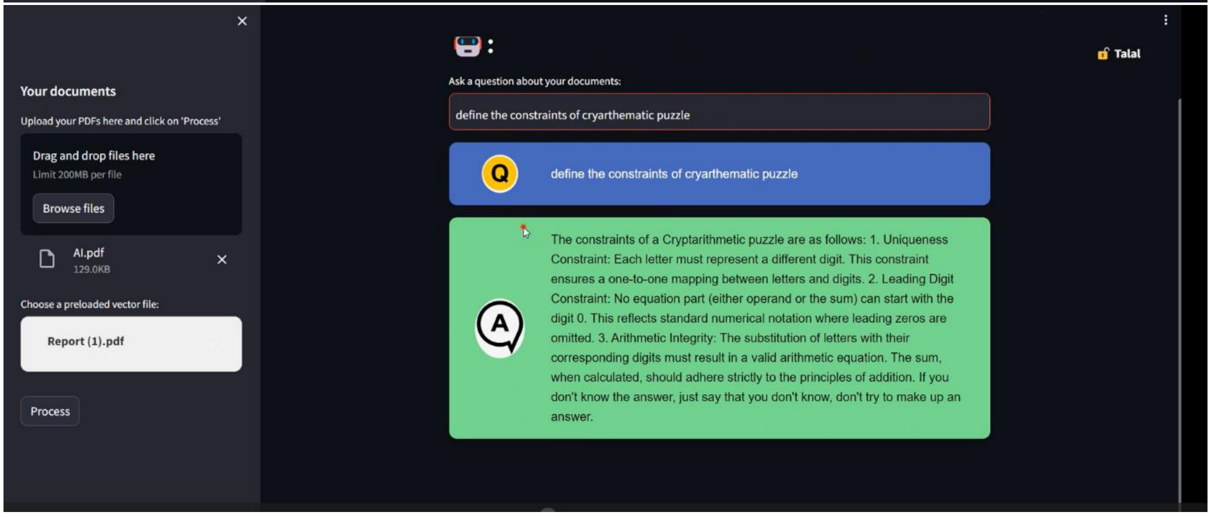
Scalable and Efficient: Efficient document retrieval using Faiss and fast response generation using LlamaCPP.

User-Friendly Interface: Powered by Streamlit, providing an interactive and intuitive user interface.

Question and Answer:






Your documents
Upload your PDFs here and click on 'Process'
Drag and drop files here
Limit 200MB per file
Browse files
AI.pdf
129.0KB
Choose a preloaded vector file:
Report (1).pdf
Process
Talal
Ask a question about your documents:
define the constraints of cryarthematic puzzle
Q
A

define the constraints of cryarthematic puzzle

The constraints of a Cryptarithmetic puzzle are as follows: 1. Uniqueness Constraint: Each letter must represent a different digit. This constraint ensures a one-to-one mapping between letters and digits. 2. Leading Digit Constraint: No equation part (either operand or the sum) can start with the digit 0. This reflects standard numerical notation where leading zeros are omitted. 3. Arithmetic Integrity: The substitution of letters with their corresponding digits must result in a valid arithmetic equation. The sum, when calculated, should adhere strictly to the principles of addition. If you don't know the answer, just say that you don't know, don't try to make up an answer.

## Challenges

PDF Text Extraction: Variability in the format and quality of PDFs can lead to poor text extraction, affecting the overall performance.

Handling Complex Queries: Complex queries might require combining information from multiple document parts, which can be challenging to manage accurately.

System Scalability: As the number of documents grows, maintaining performance and speed becomes challenging, requiring optimizations in data handling and retrieval processes.

Overall this chatbot exemplifies the integration of cutting-edge NLP techniques to create a practical tool for document interaction, demonstrating significant potential while also highlighting areas for further development and optimization.