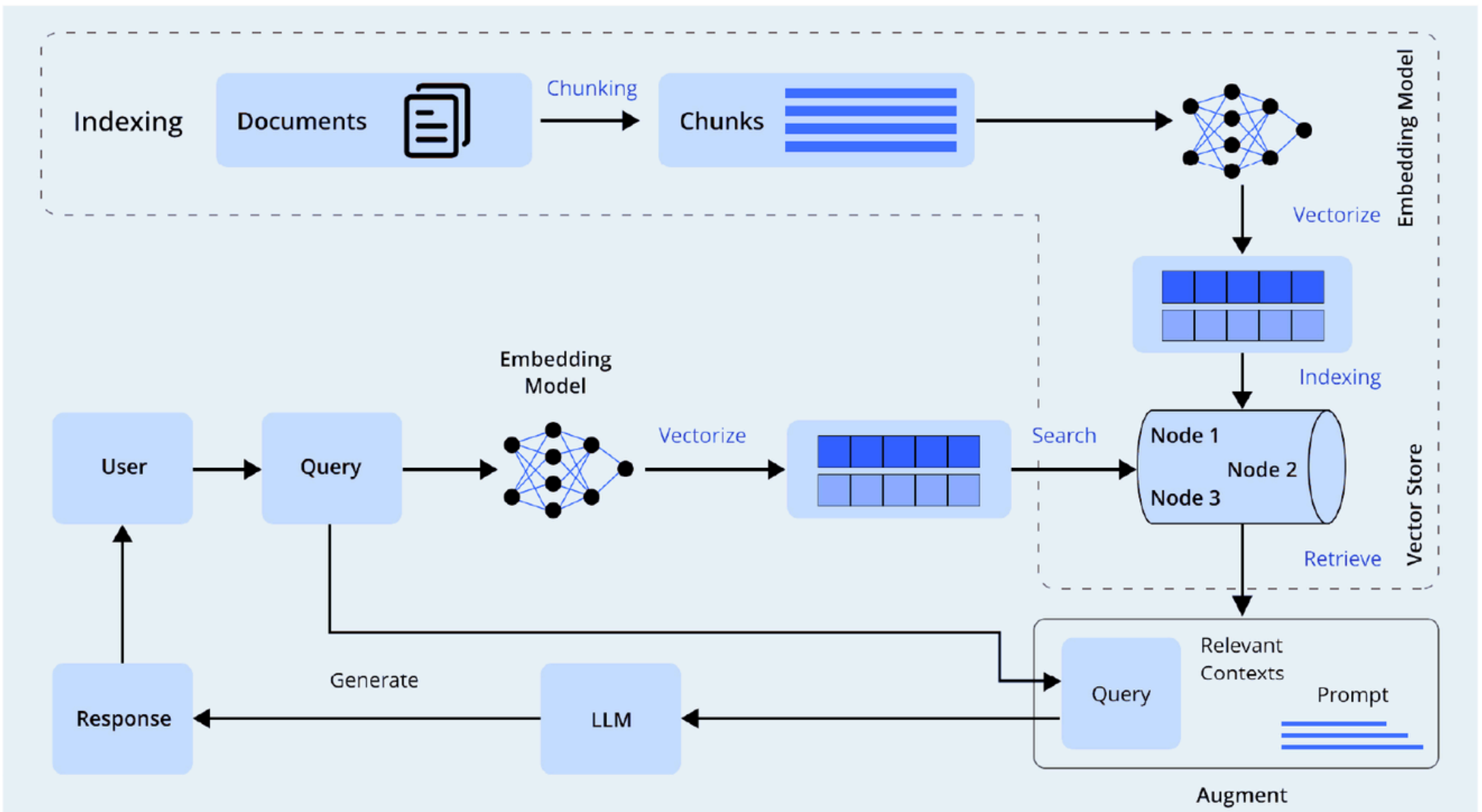
Analytics Vidhya

Query Resolution with a RAG System and AI Agents




```python
# Define retrieval tool with metadata filtering
def retrieve_course_materials(query: str, course = course):
    """Retrieves course materials filtered by course name."""
    filter_dict = {"course": course}
    results = vectorstore.similarity_search(query, k=3, filter=filter_dict)
    return "\n\n".join([doc.page_content for doc in results])
```

Selecting the Right Data for Query Resolution

Before building a RAG-based query resolution system, the most important factor to consider is data – specifically, the types of data required for effective retrieval. A well-structured knowledge base is essential, as the accuracy and relevance of responses depend on the quality of the data available. Below are the key data types that should be considered for different purposes:

- **Customer Support Data**: FAQs, troubleshooting guides, product manuals, and past customer interactions.
- **Sales & Marketing Data**: Product catalogs, pricing details, competitor analysis, and customer inquiries.
- **Internal Knowledge Base**: Company policies, training documents, and standard operating procedures (SOPs).
- **Financial & Legal Documents**: Compliance guidelines, financial reports, and regulatory policies.
- **User-Generated Content**: Forum discussions, chat logs, and feedback forms that provide real-world user queries.

Selecting the right data sources was crucial for our learner query resolution system, to ensure accurate and relevant responses. Initially, I experimented with different types of data to determine which provided the best results.

First, I used PowerPoint slides (PPTs), but they didn't yield comprehensive answers as expected. Next, I incorporated common queries, which improved response accuracy but lacked sufficient context.

Then, I tested past discussions, which helped in making responses more relevant by leveraging previous learner interactions. However, the most effective approach turned out to be using subtitles from course videos, as they provided structured and detailed content directly related to learner queries.

This approach helps in providing quick and relevant answers, making it useful for e-learning platforms and educational support systems.

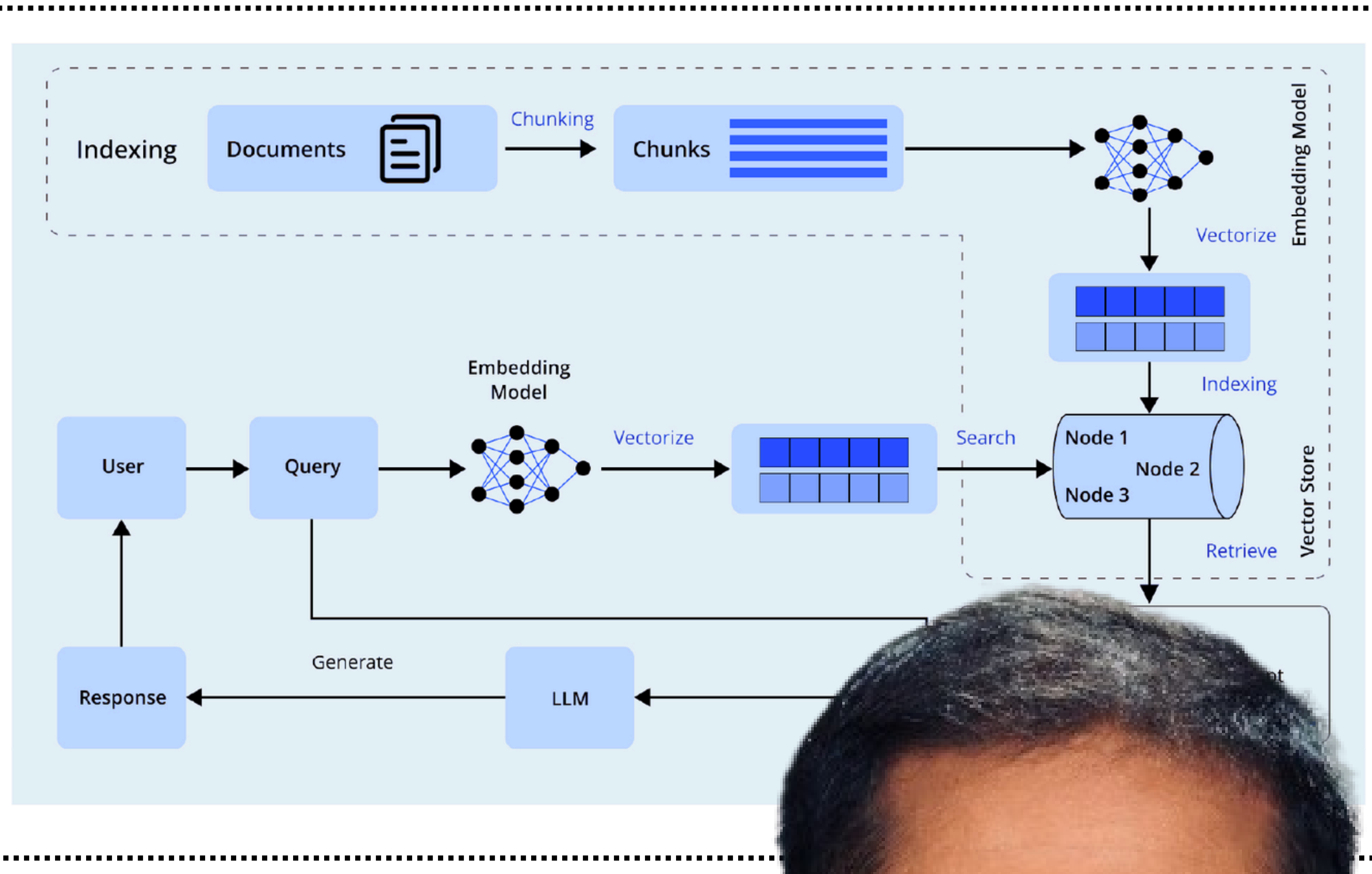
Structuring the Query Resolution System

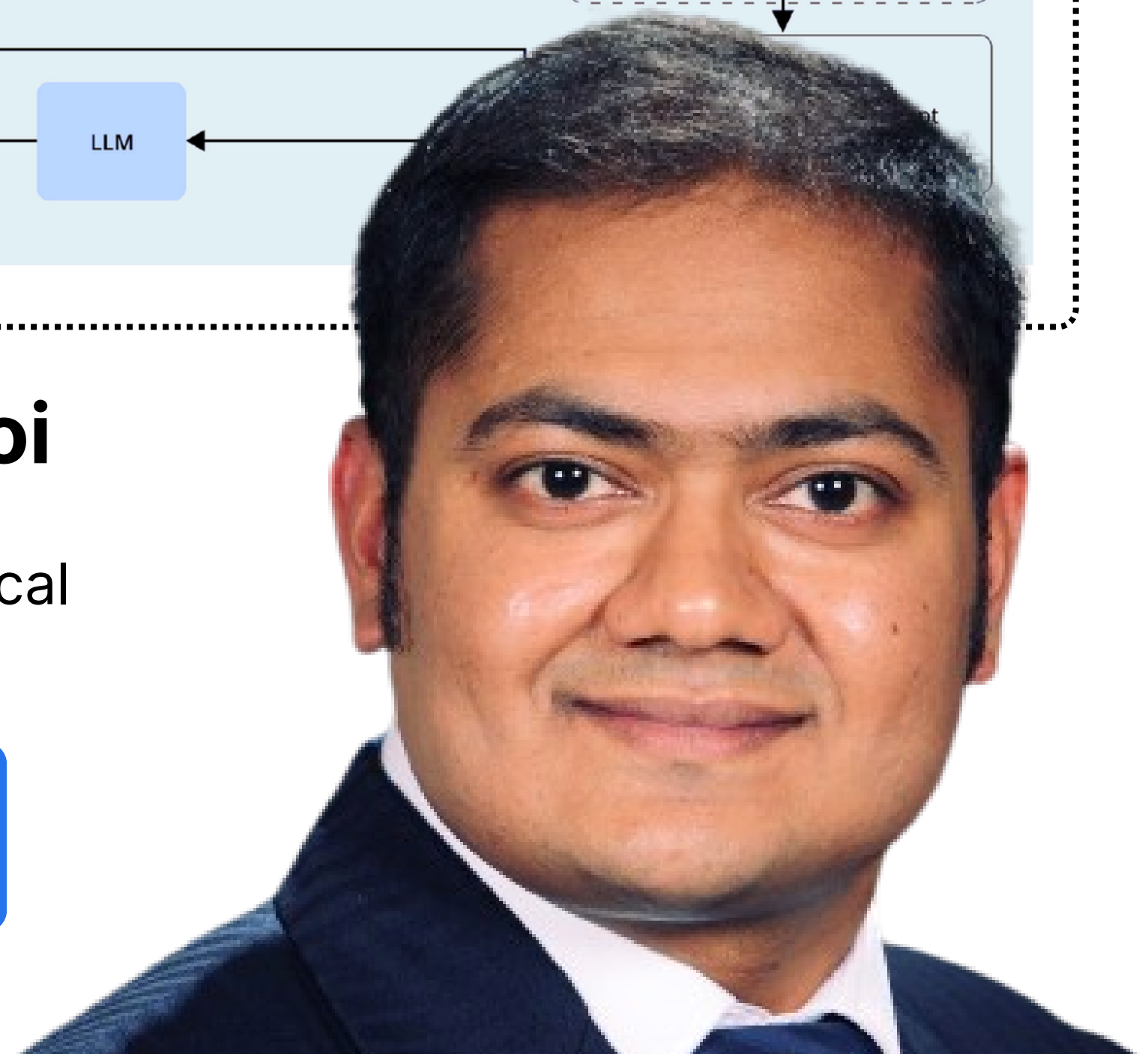
Before coding, it is important to structure the Query Resolution System. The best way to do this is by defining the key tasks it needs to perform.

Analytics Vidhya

Moreover, we are offering a Free Course on

Revolutionizing Query Resolution with a RAG System and AI Agents





Apoorv Vishnoi

Head - Training Vertical
Analytics Vidhya

Enroll Now

The system will handle three main tasks:

1. Extract and store course content from subtitles (SRT files).
2. Retrieve relevant course materials based on learner queries.
3. Use an AI-powered agent to generate structured responses.

To achieve this, the system is divided into three components, each handling a specific function. This ensures efficiency and scalability. The system consists of:

- **Subtitle Processing** – Extracts text from SRT files, processes it, and stores embeddings in ChromaDB.
- **Retrieval** – Searches and retrieves relevant course materials based on learner queries.
- **Query Answering Agent** – Uses CrewAI to generate structured and accurate responses.

Each component ensures efficient query resolution, personalized responses, and smooth content retrieval. Now that we have our structure, let’s move on to implementation.