Automating Student Data Insights and Report Generation for Enhanced Academic Support Using GPT

### Problem Statement:

Development of an AI-powered multimodal system for automating the retrieval, parsing, and generation of comprehensive reports using text and image data. The solution aims to support tutors by seamlessly integrating document indexing, query-based information retrieval, and visually enriched report creation using GPT, LlamaParse, and vector databases.

### Flow of the Software Solution:

### **. Input: File Upload**

* **User Action**: The user uploads a document (e.g., PDF, image, or text file) through the interface.
* **Purpose**: This is the starting point where the document enters the system for further processing.

### **2. File Parsing**

* **LlamaParse** is used to process the file.
  + **Text-based files (e.g., PDFs, TXT)**:
    - The content is extracted as plain text, often broken into manageable chunks.
    - Metadata like headings, page numbers, and sections may also be parsed.
  + **Image-based files (e.g., PNG, JPEG)**:
    - Optical Character Recognition (OCR) is applied to extract text from images.
    - Any diagrams or charts may be represented as image blocks.
  + **Tabular/Other Data**:
    - Tables or forms in the document are identified and parsed into structured data.

### 

### 

### 

### **3. Chunking for Efficient Storage**

* **Why Chunking?**
  + Large documents are divided into smaller, manageable "chunks" for better indexing and retrieval. Each chunk represents a portion of the document (e.g., a paragraph, a section, or an image).
* **How Chunking Works**:
  + **Text Chunks**: Sentences, paragraphs, or pages are grouped logically.
  + **Image Chunks**: Screenshots or diagrams are treated as separate blocks.

### **4. Embedding Generation**

* **Embedding Model**: BAAI/bge-large-en (or other embeddings model).
  + Each chunk (text or image) is passed through the embedding model to generate vector representations.
  + **Text Chunks**: Converted into dense vectors based on semantic meaning.
  + **Image Chunks**: Image embeddings are generated to capture visual features.

### **5. Vector Index Creation**

* **Why Use a Vector Index?**
  + A vector index stores embeddings in a format optimized for similarity searches (e.g., cosine similarity, Euclidean distance).
  + Enables efficient retrieval of relevant chunks based on queries.
* **How It’s Built**:
  + Each embedding (with associated metadata like chunk ID, page number, source file name) is stored in a **vector database** (e.g., Pinecone, Weaviate, or FAISS).
  + Metadata helps trace back results to the original document context.

### **6. Separate Indexing for Multimodal Data**

* **Text Chunks**: Indexed with their respective text embeddings.
* **Image Chunks**: Indexed with their respective image embeddings.
* Both are stored in a unified vector database, but with appropriate metadata tags to distinguish between types.

### **7. Storing the Index**

* **Storage in Vector Database**:
  + Each chunk's embedding and metadata is pushed into the vector database.
  + Example: If a document has 10 paragraphs and 5 images, 15 entries (vectors) will be stored, with appropriate metadata.

### 

### 

### **Architecture Summary**

#### **1. User Query/Upload**

* Files are uploaded via a user interface or API endpoint.

#### **2. Preprocessing and Parsing (LlamaParse)**

* Files are parsed into text chunks, image blocks, or other relevant data types.

#### **3. Embedding Generation**

* Text: Semantic embedding generated using BAAI/bge-large-en.
* Images: Visual feature embedding generated.

#### **4. Vector Indexing**

* Embeddings are created for each chunk (text or image) and stored in a vector database.

#### **5. Metadata Storage**

* Metadata for each chunk is stored alongside the embeddings, enabling efficient retrieval.

### **Benefits of This Flow**

1. **Scalability**: Handles multimodal data efficiently.
2. **Quick Retrieval**: Allows fast semantic searches based on user queries.
3. **Context Preservation**: Metadata ensures that retrieved chunks are placed back in context when generating reports.
4. **Multimodal Support**: Supports both text and images, enabling comprehensive report generation.

Can also use docker to simply containerize this solution if we are making it work offline as well.