Clinical Entity Extraction & Medical Code Prediction Using RAG Approach

Objective:

You are tasked with designing and implementing a pipeline to automate the extraction of clinical information from medical reports and predict relevant **ICD-10** and **CPT** codes. Your solution should integrate **NLP models**, **vector databases**, and **retrieval-augmented generation (RAG)** techniques.

Assignment Overview:

You will be provided with sample medical reports. Your job is to:

1. **Design a Pipeline**: Build a pipeline that processes the report, extracts relevant clinical terms (such as diagnoses, procedures, anatomical locations), and predicts the corresponding **ICD-10** and **CPT** codes.
2. **Data Storage and Retrieval**: Store **ICD-10** and **CPT** codes along with their descriptions in a vector database, and retrieve the most relevant codes based on the extracted content from the reports.
3. **Model Integration**: Use a combination of **LLMs** (Large Language Models) and traditional **NLP models** to enhance the code prediction process.
4. **Extendibility and Scalability**: Focus on designing a scalable solution that can be extended for more complex use cases, additional medical codes (e.g., HCPCS, Modifiers), and broader datasets.

Approach to Solve the Problem:

You are expected to **design a solution**, not just follow a set of instructions. Here's a suggested approach you can take:

1. **Design the Pipeline**:
    - **Preprocessing**: The first step is to preprocess the provided medical reports. This involves extracting raw text, performing tokenization, and normalizing the data (removing irrelevant sections, handling different formats, etc.).
    - **Entity Extraction**: Use NLP techniques (like Named Entity Recognition or custom classifiers) to identify clinical entities such as diagnoses, procedures, and anatomical locations. You can choose to use pre-trained models or design custom models based on the task.
    - **Vector Database Design**: Design a vector database (e.g., using Pinecone) to store **ICD-10** and **CPT** codes, along with their official descriptions. These descriptions should be embedded as vectors for efficient similarity-based retrieval.
    - **Retrieval Process**: Develop a mechanism to retrieve relevant ICD-10 and CPT codes using a **retrieval-augmented generation (RAG)** approach. This should not be based on predefined queries; you need to develop a dynamic querying system to fetch the most relevant codes based on the extracted entities.

    - **Code Prediction**: After retrieving relevant codes from the vector database, use an **LLM** or other models to generate accurate code predictions for the given medical report. This step could include validation against known standards or heuristics.
2. **Handling Complex Cases**:
    - Design your pipeline to handle edge cases, such as when multiple diagnoses or procedures are mentioned in a report, or when the report lacks complete data.
    - Consider using **confidence scoring** to measure the reliability of the predicted codes and make sure your solution can flag uncertain predictions for review.
3. **Extending the Solution**:
    - **Scaling Up**: Once you have the basic pipeline in place, think about how you can scale it to handle larger volumes of medical reports. Consider optimizing the retrieval process for faster, more accurate results.
    - **Incorporating Additional Codes**: After working with **ICD-10** and **CPT**, think about how you could integrate additional medical codes such as **HCPCS** or **Modifiers**. How would you extend your vector database or model pipeline to handle these new types of codes?
    - **Human-in-the-loop Feedback**: Propose a system where human feedback could be incorporated into the pipeline, improving the predictions over time. This could involve an active learning setup or periodic model retraining based on new data.
4. **Add-ons & Improvements**:
    - **Visualization**: Create a mechanism to visualize the results, either through a web interface or a report summary. For example, show the extracted entities, predicted ICD-10 codes, and CPT codes with official descriptions.
    - **API Integration**: Build an API that can take in a medical report and return the extracted entities and predicted codes in a structured format. This could be further extended to work with other types of medical documents.

What to Focus On:

- **Designing a Modular, Scalable Solution**: Focus on how you design the architecture of your solution so that it can be easily extended to accommodate new codes, new document types, or more complex prediction workflows.
- **Using Advanced NLP & LLM Techniques**: Consider how you can leverage **state-of-the-art NLP models** (such as GPT, BERT, etc.) to improve accuracy and performance. Think about fine-tuning models for your specific use case if needed.
- **Optimizing the Pipeline for Performance**: Think about optimizing data flow, retrieval times, and model inference speeds. For example, how would you ensure your system is fast and scalable enough to handle real-world medical report data?

Submission Instructions:

- Submit a **.zip** file containing:
    - Python code files (`.py` or `.ipynb`).
    - A sample output file (in **JSON** or **CSV** format).

- Any additional components.