**Building an RAG System for VAT Compliance**

### **Context**

Attached in the email is data related to invoice extraction from a Vision Language Model (VLM). This data includes a Doc Transcript field, which represents the text of some invoices, as extracted from the image or PDF document.

Additionally, find the attached PostgreSQL data dump containing VAT legislation we scraped from HMRC. Your task is to utilise this information to build a Retrieval-Augmented Generation (RAG) system that allows searching through the HMRC VAT legislation data (vat\_legislation table) and extracting the necessary insights to assist in VAT categorization.

### **Task Requirements**

1. **RAG System Design: Invoice Search & Compliance Assistance**
   * Build a RAG system that can efficiently use the Doc Transcript of invoices to search for relevant VAT legislation from the vat\_legislation table in the PostgreSQL dump.
   * The goal is to facilitate the classification of the following key fields for each invoice:
     + **VAT % (Tax Code)**
     + **Chart of Account (Category)**
   * Use **Llamaindex** to speed up the process of building the RAG system.
2. **Implementation of GL Code Agent**
   * Create a simple General Ledger (GL) Code agent that can predict the **VAT %** and the **Chart of Account (Category)** for each invoice.
   * This agent should leverage the RAG setup to provide the predictions, utilising the extracted invoice data and the HMRC VAT legislation as reference.
3. **FastAPI Endpoint**
   * Build a FastAPI endpoint that accepts a JSON input in the following format:

  
{

"data": "<Doc Transcript of Invoice>"

}

* + Upon receiving the input, the endpoint should return:
    - **VAT % Prediction**
    - **Chart of Account (Category) Prediction**
    - **ROUGE-1 Score** for each prediction
    - **Reference** to the relevant parts of the HMRC VAT legislation that influenced the predictions, including a reference to the specific source.

1. **MLFlow Integration**
   * Integrate a simple **MLFlow** setup to track the performance of the GL Code agent.
   * The system should accept input in the following format:

  
{

"VAT %": {

"original": "<Actual VAT %>",

"prediction": "<Predicted VAT %>"

},

"Chart of Account": {

"original": "<Actual Category>",

"prediction": "<Predicted Category>"

}

}

* + The MLFlow integration should track:
    - The **ROUGE-1 Score** for the predictions.
    - Metrics comparing the **original** and **predicted** values for both **VAT %** and **Chart of Account (Category)**.

### **Resources to be used**

* PostgreSQL data dump containing VAT legislation data (vat\_legislation table) from HMRC.
* Sample JSON data of invoice extraction from the VLM.
* Llamaindex, FastAPI and MLFlow

### **Submission Guidelines**

* Submit your solution in a private GitHub repository with a clear README.md explaining how to set up and use the solution with an invite extended to tosi-n on github as collaborator.
* Ensure that the code is well documented and follows standard best practices.
* Provide a demo video (maximum 5 minutes) explaining the workflow and demonstrating the FastAPI endpoint in action.

### **Evaluation Criteria**

* **Correctness**: Does the system correctly predict VAT % and Chart of Account (Category)?
* **Scalability**: How well does the RAG system handle different invoices and legislation queries?
* **Accuracy**: Are the predictions accurate, as evidenced by ROUGE-1 scores and references to VAT legislation?
* **Code Quality**: Is the code modular, clean, and maintainable?
* **Documentation**: Is the setup process clear, and is the purpose of each part of the system well explained?
* **Creativity**: Innovative approaches to the problem are highly valued.