**AUTOMATING VBA MACRO DOCUMENTATION AND TRANSFORMATION**

**Team Members:**

K R Navin Aananthan (20PT18)

S Niveditha (20PT22)

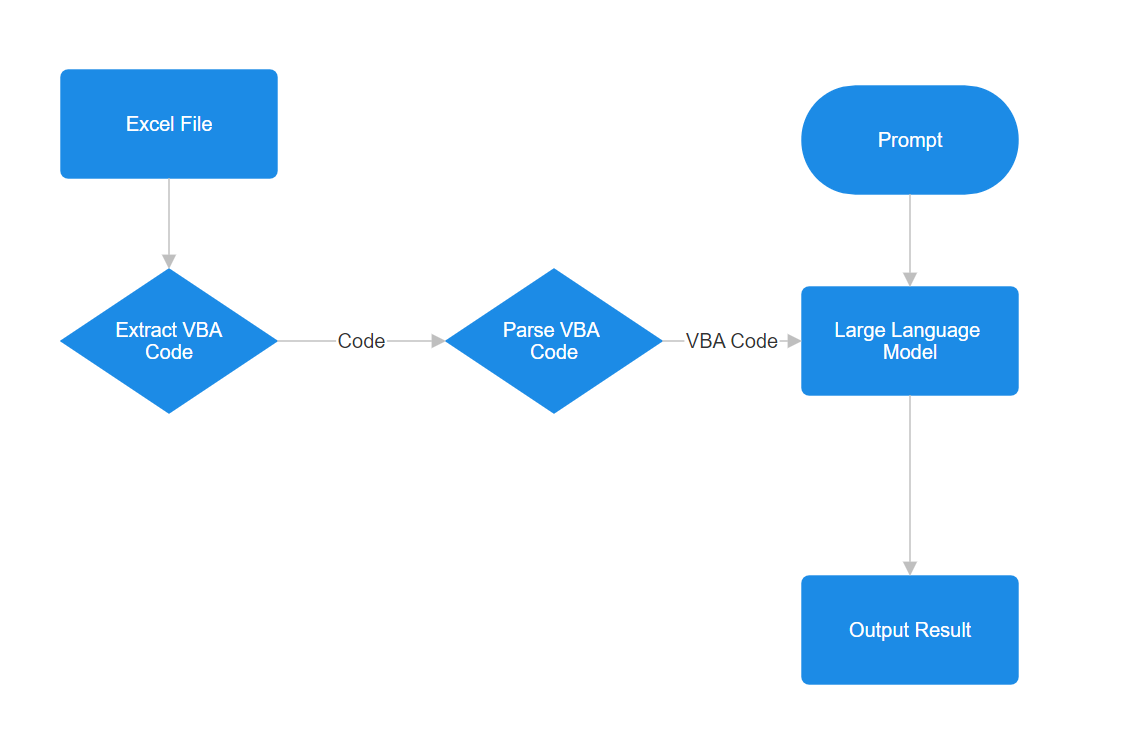
**GitHub Link:**

**Objective:**

The goal is to develop a solution that automates the documentation and understanding of legacy VBA macros used in critical processes across DFIN, facilitating their transformation into modern technologies and IT platforms.

**Architectures:**

* The platform allows users to upload Excel files containing macros.
* It utilizes the **oletools** Python library to parse these files and extract VBA code.
* Extracted VBA code is then passed to **Mistral**, a Large Language Model, to generate responses based on predefined prompts.
* For VBA code security, the platform incorporates oletools for malicious keyword detection and pattern analysis.
* Additionally, a Deep Learning algorithm is integrated to handle **code** **obfuscation**, enhancing security measures within the platform.



**Algorithm Analysis:**

* **Excel File Parsing and VBA Extraction**:
* When users upload Excel files containing macros, the platform initiates a parsing process facilitated by the oletools Python library.
* oletools is utilized to extract Visual Basic for Applications (VBA) code embedded within these Excel files.
* This extraction process ensures that all VBA code segments, crucial for subsequent analysis and processing, are efficiently retrieved from the uploaded Excel documents.
* **Preprocessing of Extracted VBA Code**:
  + Once the VBA code is extracted, it undergoes preprocessing stages within the platform.
  + Preprocessing involves cleaning the code to eliminate non-essential artifacts such as comments, white spaces, and formatting inconsistencies.
  + The cleaned VBA code is then formatted into a structured format suitable for subsequent analysis and input into the platform's algorithmic components.
* **Utilization of Mistral Large Language Model:**
  + The core of the analysis algorithm involves leveraging Mistral, a state-of-the-art Large Language Model (LLM), for processing the extracted VBA code.
  + Mistral is employed to receive prompts from predefined prompt and generate contextually relevant responses based on the analyzed VBA code.
  + Its integration enables advanced natural language processing (NLP) capabilities, allowing the platform to interpret user queries and provide intelligent insights or actions related to the uploaded VBA code.
* **Security Checks with oletools for Malicious Detection**:
  + To ensure the security and integrity of the processed VBA code, the platform utilizes specific functionalities provided by oletools.
  + oletools includes built-in mechanisms for detecting malicious keywords, patterns, and structures within the VBA code.
  + This security check is crucial for identifying potential threats such as malware, suspicious commands, or harmful coding practices embedded within the uploaded Excel macros.
* **Deep Learning Algorithm for Code Obfuscation**:
  + Using obfuscation-detection Library: The obfuscation-detection Python library detects potential malware or malicious intent in VBA code by analyzing code structures and patterns.
  + Code Classification: It classifies VBA scripts as either malicious or non-malicious, based on syntax, semantic patterns, and known malicious signatures, enhancing system security against sophisticated attacks.

**User Interface design:**

