**Procedure Document**

**Problem Code**: **5.1**

**Problem Statement**: Explore the potential of LLM and generative AI models like for Natural Language Processing Tasks Such as Text Generation, Summarization, Q&A.

**Objective**: The objective is to provide a detailed guide for implementing a project that utilizes a RAG-based approach with an LLM to perform Q&A, sentiment analysis, and summarization on uploaded PDFs, ensuring seamless integration and optimal functionality.

**Importance**: This procedure document ensures accurate execution, standardizes the development process, and facilitates maintenance and troubleshooting, ultimately leading to a robust and efficient system for analysing and summarizing PDF content.

**Features**:

* Perform Q & A on uploaded PDF maintaining conversation history
* Perform Aspect Based Sentimental Analysis on uploaded PDF
* Perform Summarization on uploaded PDF

**Libraries Used with versions**:

* Langchain: 0.2.0
* Langchain\_openai: 0.1.7
* Langchain\_core: 0.2.0
* Langchain\_community: 0.2.0

**Dataset**: Tata Safari Reviews

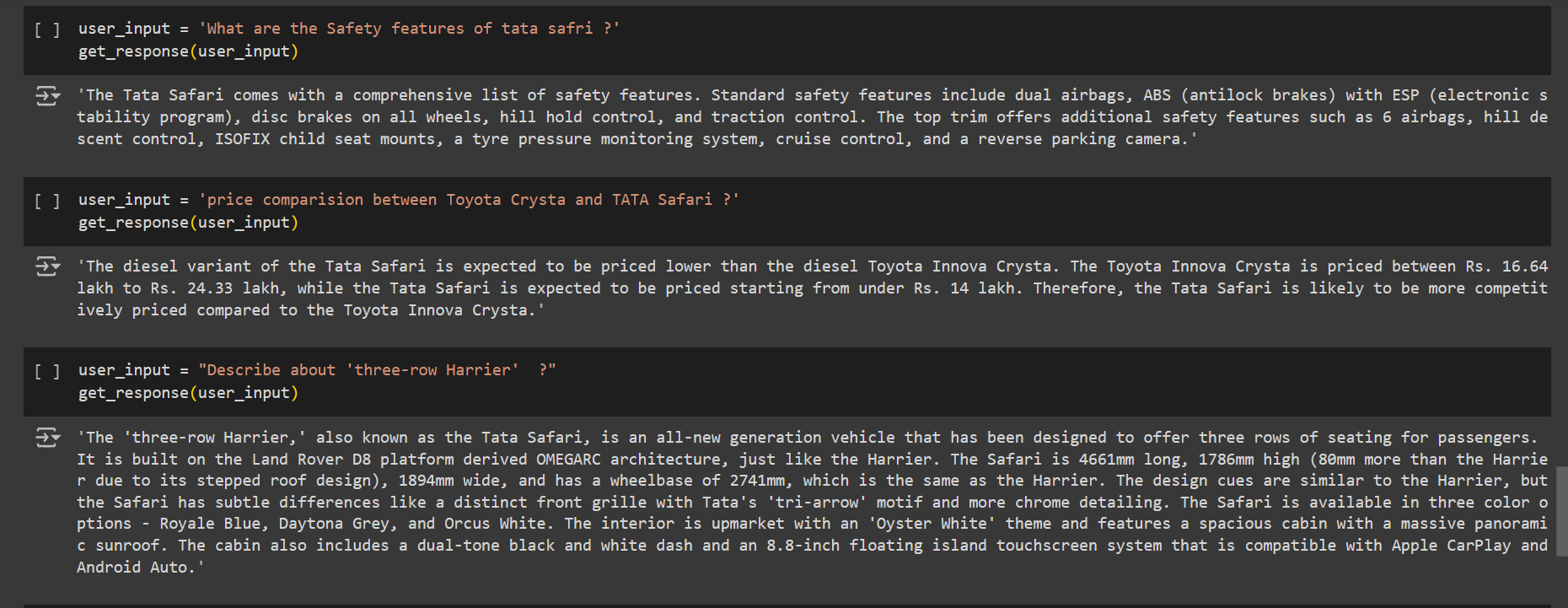
**Input**:

* PDF (can be optimized for .docx, CSV, Excel, URL)
* Queries (String)

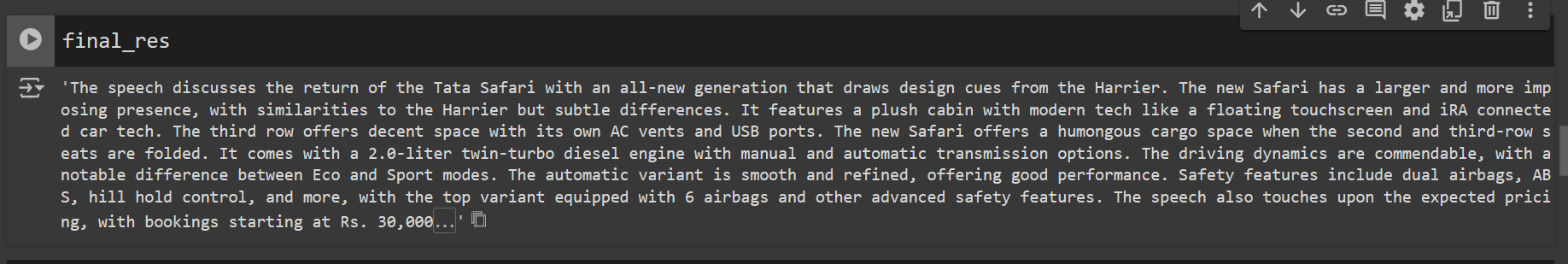
**Output**:

* Answers generated (String)
* Summary(string)
* Aspect-Feedback pair(string)

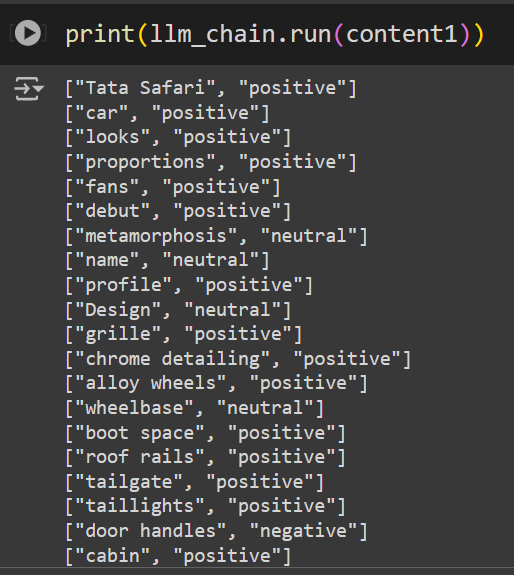
**Q &A:**



**Summary:**



**Aspect Based Sentiment Analysis:**



**Procedure**:

**Q & A**:

* Install and import of all required packages.
* Set up the python environment with OpenAI API.
* Load and extract the PDF by using “PyPDFLoader”.
* Split the extracted document with “RecursiveCharacterTextSplitter” with chunk size 1000, chunk overlap 100.
* Convert the text to vector by using “OpenAIEmbeddings”.
* Store the vectors in Vector database (Chroma).
* Create a history aware retriever by using LLM(ChatGPT 3.5 Turbo) and a clear Prompt Template.
* Pass the query through final conversational RAG chain and get the output.

**Aspect Based Sentiment Analysis and Summarization**:

* Prepared a pipeline which will take the extracted text of the PDF as input and perform ABSA and Summarization by using 2 different potential prompt templates and a LLM (ChatGPT 3.5 Turbo. It gives a precise summary and Aspect: Feedback pairs as output.

Gathered all the tasks into a final Pipeline.

**Further Extension**:

**Evolution Metrics**: Now we have a pipe line. We can also compare different opensource models like Mistral and Llama by using Playground or Gradio. We can check the bellow performance metrics to finalize the model selection.

* Text generation (language Model): Perplexity
* Summarization: ROGUE
* Sentiment analysis: Precision, Recall, F1 Score

**Api Integration**: We can integrate the final pipeline with Fast API.

* We should use Post method
* We should Enable the CORS (for UI integration)
* We can define a proper path and organise the input and output format for seamless integration.
* Finally, we should define the Lambda Handler, which will help us for the AWS Lambda deployment.

**Dockerization**: We should dockerise all the dependencies like .py files, requirement.txt, into a Docker File. Which can start with;

* Base docker image like Ubuntu or Python 3
* Copy all required python scripts and requirement.txt
* Define the run function for python files.
* Define a route or expose path and command to run all through CMD
* Create a Docker Image out of it.

**Deployment in AWS**:

* We can push the image to AWS ECR after logging in to AWS CLI.
* Push the ECR ID to AWS lambda function, change the path in 3 places in Lambda function code and test for the result.
* With successful deployment we can take the API from AWS and integrate with WEB UI.

**Conclusion**: In conclusion, this procedure document is essential for the successful implementation and maintenance of the RAG-based project using an LLM. It ensures consistent performance across features, aids in troubleshooting, and promotes seamless integration, ultimately enhancing the system's reliability and efficiency in analyzing and summarizing PDF content.