**Project Title: Intelligent Document Finder with Llama Index**

**Overview of system:**

This system enables user to query on documents of any kind (ex. PDF, PPT, Word Documents, etc) which are uploaded in Google Drive and user can expect response of query along with the metadata like file name, page number, author etc.

**Setup and Installation:**

* Create Virtual Environment with the help of virtualenv library(steps are given below):
  + Run the following commands for creating virtualenvironment:

1. Pip install virtualenv
2. virtualenv venv
3. venv\Scripts\activate

* Install required libraries which are mentioned in requirments.txt
* Create following directory inside root of project folder: cache\_model, chroma\_db, model\_cache, pipeline\_store
* Create GoogleAI API key from here: <https://aistudio.google.com/app/apikey>
* Store the GoogleAI API key in .env file
* To setup Google Drive integration follow below steps:

1. Create a Google Cloud Platform (GCP) Project:

* Go to the Google Cloud Console (<https://console.cloud.google.com/>).
* Create a new project or select an existing project.

1. Enable the Google Drive API:

* In the Cloud Console, navigate to the "APIs & Services" > "Library" section.
* Search for "Google Drive API" and click on it.
* Click the "Enable" button to enable the API for your project.

1. Set Up OAuth 2.0 Credentials:

* In the Cloud Console, navigate to the "APIs & Services" > "Credentials" section.
* Click on "Create credentials" and select "OAuth client ID."
* Select "Web application" as the application type.
* Add the authorized redirect URIs (e.g., <http://localhost:8080/auth/google/callback>).
* Click "Create" and note down the client ID and client secret.

1. Configure Google Drive API Access:

* In your Google Drive, create a folder where users will upload documents.
* Share the folder with the service account email address (e.g., example@your-project-id.iam.gserviceaccount.com) generated for your GCP project.

1. Get the credentials.json file and add it in project folder(here is sample structure of the file.)

{

"type": "service\_account",

"project\_id": "your\_project\_id",

"private\_key\_id": "your\_private\_key\_id",

"private\_key": "your\_private\_key",

"client\_email": "your\_client\_email",

"client\_id": "your\_client\_id",

"auth\_uri": "https://accounts.google.com/o/oauth2/auth",

"token\_uri": "https://accounts.google.com/o/oauth2/token",

"auth\_provider\_x509\_cert\_url": "https://www.googleapis.com/oauth2/v1/certs",

"client\_x509\_cert\_url": "your\_client\_x509\_cert\_url"

}

**Instruction To Run Updated App:**

Once your done with above setup and installation process then run mai.py file to start backend server so signup and login and other functionality work properly. After that run app.py file with this command: “streamlit run app.py” and now you can see interface and in that you can select option from selectbox for SignUp, login, and search and you can use the app. Signup first if you are new user and then login and then select search from selectbox and provide your google drive folder ID and click on load data button, once it display success message then query whatever you want based on files you have provided.

**Module Information:**

load\_drive\_files.py:

* this file contains function for get files data from google drive.

setup\_embedding\_model.py:

* In this file huggingface’s opensource model is defined to use it for embeddings.

setup\_chromadb.py:

* this file contains function to create chromadb vector database so we can use that to store vectors.

setup\_ingestion\_pipeline.py:

* this file contains function to create ingestion pipeline for processing of input data, including text splitting, embedding, and storing vectors and documents.

app.py:

* This code sets up a streamlit interface for signup, login and providing google drive link and based on that querying a document and displaying the response along with metadata.

main.py:

* this file handles backend processing for SignUp, Login and Verifying user.

auth.py:

* API endpoints are defined in this file for SignUp, Login and Verifying user.

**System usage:**

once you have setup with project folder and add required dependencies in your system run the app.py file in terminal and then wait for some type for processing and you will get application URL something like this: <http://127.0.0.1:7860>, just click on that URL and now you can see interface for querying.

**Authentication working with the system:**

* Signup form takes two input username and password and in username user should provide his/her emailId and it do hashing on password and then username and hashed password will be stored in database.
* Login form takes two input same as signup and at backend provided username and password will be matched for authentication if it will successful then JWT encoded token will be generated.
* When user will provide google drive folder ID at that time encoded JWT token and Folder ID both will be send to server then at the backend emaiID will be extracted from folderID and token will be decoded and from token username(which in our case is emailid) will be fetched and it will be compared with emailID fetched from google drive folder ID and if it matches successfully then data will be fetched from user provided folder and now user can query.

**Resources:**

* For fetching files from Google drive:

1. <https://llamahub.ai/l/readers/llama-index-readers-google?from=readers>
2. <https://docs.llamaindex.ai/en/stable/examples/ingestion/ingestion_gdrive.html>

* for creation of ChromaDB:

1. <https://docs.llamaindex.ai/en/stable/examples/vector_stores/ChromaIndexDemo.html>

* for implementing ingestion pipeline:

1. <https://docs.llamaindex.ai/en/stable/examples/ingestion/ingestion_gdrive.html>

* for integrating gradio interface:

1. <https://www.gradio.app/docs/interface>