AI-Powered PDF Knowledge Assistant Using Goggle PALM

DocuQuery: AI-Powered PDF Knowledge Assistant Using Google PALM

Project Flow

1. Prior Knowledge
2. Project Structure
3. Setting Up Google API Key
4. Generate PALM API
5. Installation and Importing of Libraries and Adding API Key
6. PDF Text Processing
7. Conversational Chain Setup
8. User Interaction and Response Handling
9. Application Setup and Integration

Prior Knowledge

To develop DocuQuery, you should have prior knowledge of:

1. Python programming language
2. Google Cloud Platform (GCP)
3. Google PALM (Platform for AI and Machine Learning)
4. Natural Language Processing (NLP)
5. Streamlit library for building web applications

Project Structure

The project structure will consist of the following directories and files:

1. app.py: The main application file
2. config.py: Configuration file for Google API key and other settings
3. pdf_processor.py: PDF text processing library
4. conversational_chain.py: Conversational retrieval chain library
5. user_interaction.py: User interaction and response handling library

6. streamlit_app.py: Streamlit web application file
7. requirements.txt: Dependencies file for the project

Setting Up Google API Key

To set up a Google API key, follow these steps:

1. Go to the Google Cloud Console and create a new project
2. Navigate to the API Library page and search for the Google Cloud Vision API
3. Click on the result, then click on the Enable button
4. Create a new API key by clicking on the Create Credentials button
5. Select API key and follow the prompts to create a new API key

Generate PALM API

To generate a PALM API, follow these steps:

1. Go to the Google Cloud Console and navigate to the PALM page
2. Click on the Create button to create a new PALM API
3. Follow the prompts to set up the PALM API

Installation and Importing of Libraries and Adding API Key

To install and import the necessary libraries, run the following command:

bash
pip install -r requirements.txt

Then, add the Google API key to the config.py file:

```
GOOGLE_API_KEY =
'YOUR_API_KEY_HERE'
```

PDF Text Processing

To extract text from PDF documents, use the pdf_processor.py library:

```
import pdfplumber

def extract_text_from_pdf(file_path):
    with pdfplumber.open(file_path) as pdf:
        text = ''
        for page in pdf.pages:
            text += page.extract_text()
        return text
```

Conversational Chain Setup

To set up the conversational retrieval chain, use the conversational_chain.py library:

```
import nltk
```

```
from nltk.tokenize import word_tokenize

def setup_conversational_chain(text):
    tokens = word_tokenize(text)
    chain = []
    for token in tokens:
        chain.append(token)
    return chain
```

User Interaction and Response Handling

To handle user input and generate responses, use the user_interaction.py library:

```
import streamlit as st

def handle_user_input(user_input):
    # Process user input and generate response
    response = 'Hello, how can I assist you?'
    return response
```

```
def generate_response(chain):
    # Generate response based on
conversational chain
    response = 'I understand you are looking
for information on {}.'.format(chain[0])
    return response
```

Application Setup and Integration

To set up the Streamlit web application, use the streamlit_app.py file:

```
import streamlit as st
from pdf_processor import
extract_text_from_pdf
from conversational_chain import
setup_conversational_chain
from user_interaction import
handle_user_input, generate_response

st.title('DocuQuery: AI-Powered PDF
Knowledge Assistant')
```

```
# Set up sidebar settings
st.sidebar.title('Document Processing')
st.sidebar.file_uploader('Upload PDF
document', type='pdf')

# Run the application
if st.sidebar.button('Run'):
    # Extract text from PDF document
    text =
extract_text_from_pdf(st.sidebar.file_upload
er('Upload PDF document', type='pdf'))

    # Set up conversational chain
    chain = setup_conversational_chain(text)

    # Handle user input and generate
response
    user_input = st.text_input('Enter your
question')
    response =
handle_user_input(user_input)
    st.write(response)
```

```
    # Generate response based on
conversational chain
    response = generate_response(chain)
    st.write(response)
```

Run the Application

To run the application, execute the following command:

bash
streamlit run streamlit_app.py

This will start the Streamlit web application, and you can interact with it by uploading a PDF document, entering your question, and viewing the response.