ChatDOC - Chat with Document

The demo is designed to streamline your workflow by allowing you to upload documents and ask questions directly related to the content of those files. Leveraging advanced Optical Character Recognition (OCR) technology, the system can accurately extract text from images, transforming your physical and digital documents into machine-readable text.

Once your file is uploaded, the interface integrates with a cutting-edge GPT-based chat model to understand and analyze the content of your documents. This powerful combination enables you to ask specific, contextual questions about your documents and receive instant, accurate answers. Whether you're querying data points, seeking clarification on document content, or need summaries of lengthy reports, the Intelligent Document Assistant is here to help.

User Interface

The user interface is built mainly by Flask and it allow you to chat with a gpt, upload the file, and restart the conversation.

Optical Character Recognition (OCR)

Upon file upload, the application utilizes Optical Character Recognition (OCR) technology to extract text from the uploaded documents. This process involves scanning the document image, recognizing the characters, and converting them into machine-encoded text.

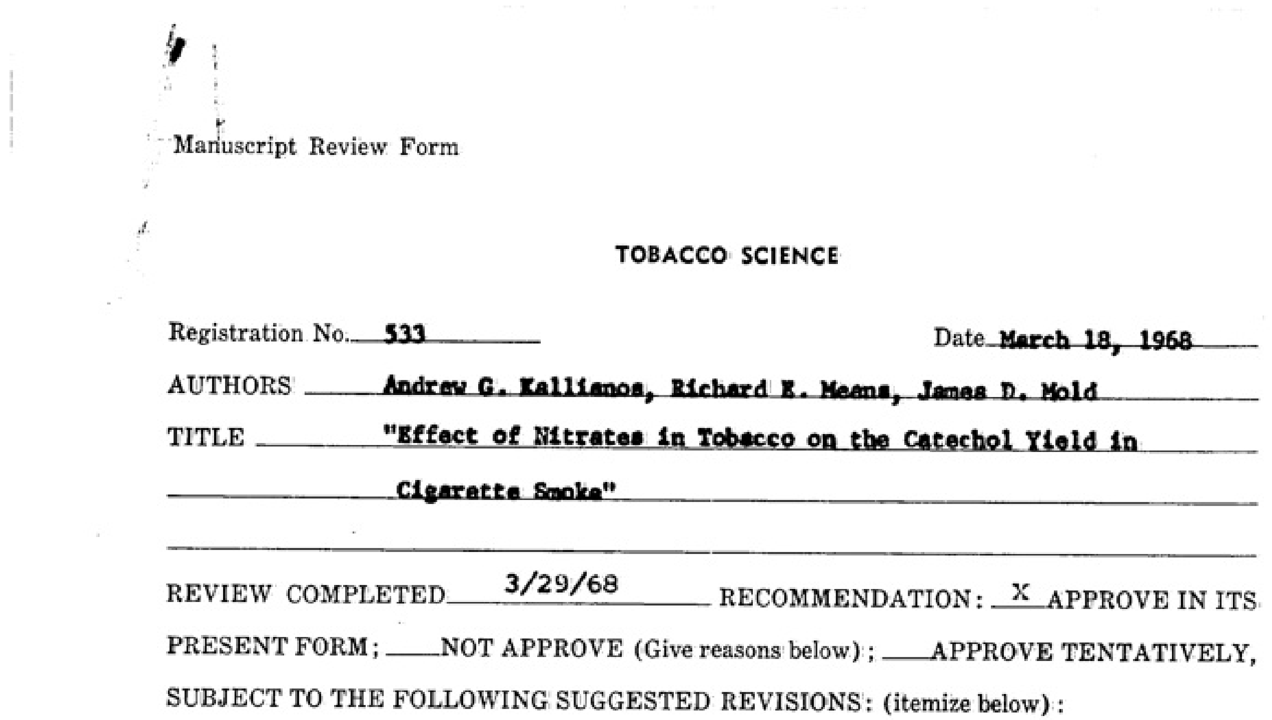
In [2]:

```
from PIL import Image
import pytesseract
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

image_path = 'test1.png'

img = Image.open(image_path)
img
```

Out[2]:



Manuscript Review Form

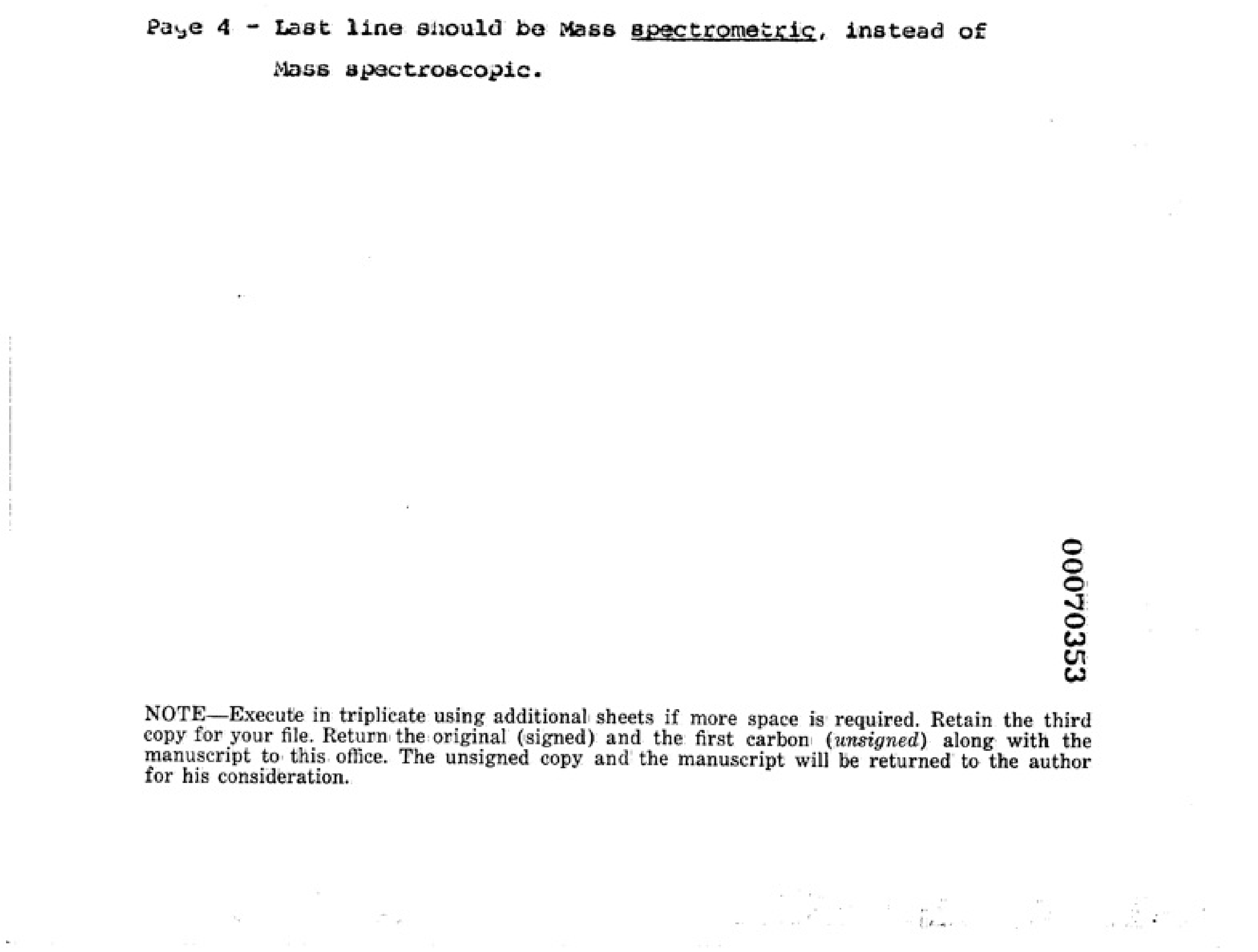
TOBACCO SCIENCE

Registration No. 533 Date March 18, 1968

AUTHORS Andrew G. Kallianos, Richard E. Means, James D. Mold

TITLE "Effect of Nitrates in Tobacco on the Catechol Yield in Cigarette Smoke"

REVIEW COMPLETED 3/29/68 RECOMMENDATION: X APPROVE IN ITS PRESENT FORM; ___NOT APPROVE (Give reasons below); ___APPROVE TENTATIVELY, SUBJECT TO THE FOLLOWING SUGGESTED REVISIONS: (itemize below):



Paye 4 - Last line should be Mass spectrometric, instead of
Mass spectroscopic.

00070353

NOTE—Execute in triplicate using additional sheets if more space is required. Retain the third copy for your file. Return the original (signed) and the first carbon (*unsigned*) along with the manuscript to this office. The unsigned copy and the manuscript will be returned to the author for his consideration.

In [3]:

```
text = pytesseract.image_to_string(img)
print(text)
```

```
,

'Maniuseript Review Form

TOBACCO: SCIENCE

Registration No.3 ___ Date_March 18,1968 ___
AUTHORS ____Andraw G. Kallianoa, Richard R. Means, Jones D. Mold
TITLE "Rffect of Nit:

REVIEW COMPLETED. RECOMMENDATION: _X_APPROVE IN ITS.

PRESENT FORM; ___NOT APPROVE (Give reasons below); ___APPROVE TENTATIVELY,
SUBJECT TO THE FOLLOWING SUGGESTED REVISIONS: (itemize below) :

Paye 4 - Last line should be Mass spectrometric, instead of
Mass spectroscopic.

€S£04000

NOTE—Execute in triplicate using additional sheets if more space is required. Retain the
third
copy for your file. Return the original (signed) and the first carbon: (unsigned) along w
ith the

manuscript to this office. The unsigned copy and the manuscript will be returned to the a
uthor
for his consideration.
```

GPT API

After text extraction, the GPT chat model takes over. This AI-driven component is designed to understand and analyze the extracted text and the user's question. Utilizing natural language processing (NLP) and machine learning algorithms, the chat model generates contextually relevant answers. The responses are then passed back to the front end, where they are displayed to the user. This part of the backend is responsible for the intelligent interaction, providing users with informative and accurate answers based on the content of their uploaded files.

In [4]:

```
import openai
openai.api_key = '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'
model_name = "gpt-3.5-turbo"


conversation = [
    {"role": "system", "content": "You are a helpful assistant."},
    {"role": "user", "content": f"Based on the following text \"{text}\", I will ask you
so questions."},
    {"role": "assistant", "content": "Of course!"}
    # Include the initial messages and responses here
]

# Function to continue the conversation
def continue_conversation(new_user_message):
    # Add the new user message to the conversation
    conversation.append({"role": "user", "content": new_user_message})

    # Send the updated conversation to the API
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=conversation
    )

    # Extract and add the assistant's response to the conversation
    assistant_message = response.choices[0].message['content']
    conversation.append({"role": "assistant", "content": assistant_message})

    # Return or print the assistant's response
    return assistant_message

# Example usage
print(continue_conversation("Who is the auther?"))
print(continue_conversation("what is the note?"))
```

```
The authors listed in the text are Andrew G. Kallianoa, Richard R. Means, and Jones D. Mo
ld.
The note at the end of the text provides instructions to execute in triplicate using addi
tional sheets if more space is required. It also advises to retain the third copy for you
r file, return the original (signed) and the first carbon (unsigned) along with the manus
cript to the office. The unsigned copy and the manuscript will be returned to the author
for his consideration.
```

Summary and Further Steps

This demo showcases the integration of Optical Character Recognition (OCR) and AI chat technology, providing a user experience that is both interactive and responsive. Utilizing Flask for web development and HTML for content presentation, the application serves as an effective tool for users to extract information from documents and gain immediate understanding through natural language questions.

The focus should now shift towards enhancing the OCR model. Presently, it has not been fine-tuned for specific types of data, such as medical notes from doctors, leading to less accurate results and potential loss of information from the original documents. Furthermore, the current output does not effectively represent non-text elements like marks, tick boxes, separators, and diagrams, nor does it accurately capture the document's layout, including page structure, forms, and tables, or style elements such as font, colors, and highlighting. Enhancements to the OCR's output format could significantly improve its performance and accuracy.