import openai

def get\_mca\_questions(context: str, api\_key: str):

# Validate input

if not isinstance(context, str):

raise TypeError("Input context must be a string.")

# Validate API key

if not isinstance(api\_key, str) or not api\_key:

raise ValueError("Invalid API key provided.")

# Step 1: Preprocess the text (optional)

# You can preprocess the text here if required.

# Step 2: Generate questions using OpenAI GPT-3.5

mca\_questions = []

prompt = f"Generate only 3 multiple-choice questions which have multiple correct answers for the following passage:\n{context}"

try:

response = openai.Completion.create(

engine="text-davinci-002",

prompt=prompt,

max\_tokens=150, # Set the desired length of the generated questions

n=10, # Number of questions to generate

stop=None, # Set a stop token if necessary to control question length

api\_key=api\_key, # Use the provided API key

)

# Extract the generated questions from the API response

for choice in response["choices"]:

mca\_questions.append(choice["text"].strip())

except Exception as e:

# Handle API errors

raise Exception(f"Error generating questions: {str(e)}")

return mca\_questions

def format\_question\_options(input\_list):

questions = []

options = []

for entry in input\_list:

q\_and\_a = entry.split('\n\n')

question = q\_and\_a[0].strip()

options\_and\_answers = q\_and\_a[0].split('\n')

question\_number = question[0] # Assuming the question always starts with a number followed by a period.

formatted\_question = f"Q{question\_number}: {question[3:]}" # Removing the question number and period.

questions.append(formatted\_question)

question\_options = [option[3:] for option in options\_and\_answers[:-1]]

options.append(question\_options)

return questions, options

# Given list of questions and options

given\_list = get\_mca\_questions(context, api\_key)

questions, options = format\_question\_options(given\_list)

# Printing the formatted questions and options

for i, question in enumerate(questions):

print(question)

for j, option in enumerate(options[i]):

print(f"{chr(j + 97)}. {option}")

print()

Description:

This code generates multiple-choice questions from a given context using OpenAI GPT-3.5. The code first validates the input and then generates the questions. The questions are then formatted and printed.

Usage:

To use the code, you will need to have an OpenAI API key. You can get an API key from the OpenAI website. Once you have an API key, you can run the code by providing the context and API key as arguments. For example, to generate questions from the context "This is a test", you would run the code like this:

Python

get\_mca\_questions("This is a test", "YOUR\_API\_KEY")

Use code with caution

content\_copy

Output:

The output of the code will be a list of formatted questions and options. For example, if the context is "This is a test", the output might be:

Q1: What is this?

A. A test.

B. A sentence.

C. A paragraph.

D. A document.

Q2: What is the purpose of this text?

A. To test the code.

B. To provide an example of a multiple-choice question.

C. To demonstrate the use of OpenAI GPT-3.5.

D. All of the above.

Notes:

The code uses the openai library to interact with the OpenAI API.

The code is designed to be run from a Python interpreter