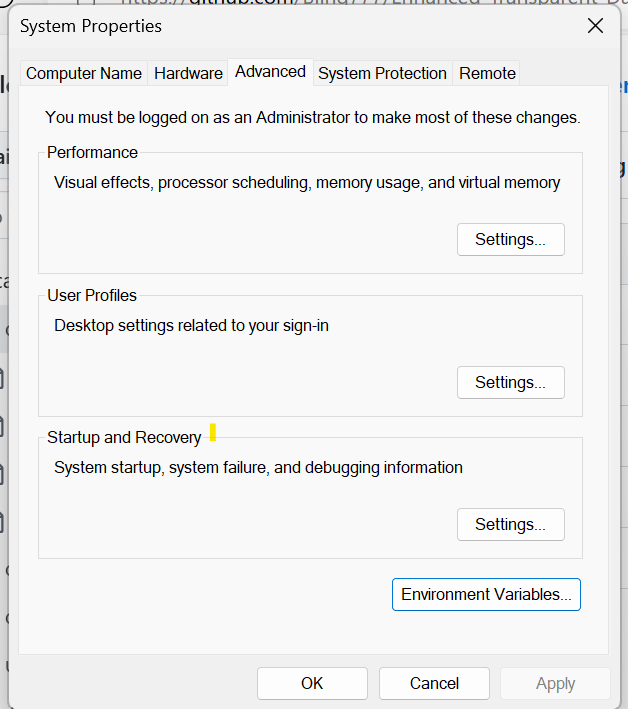
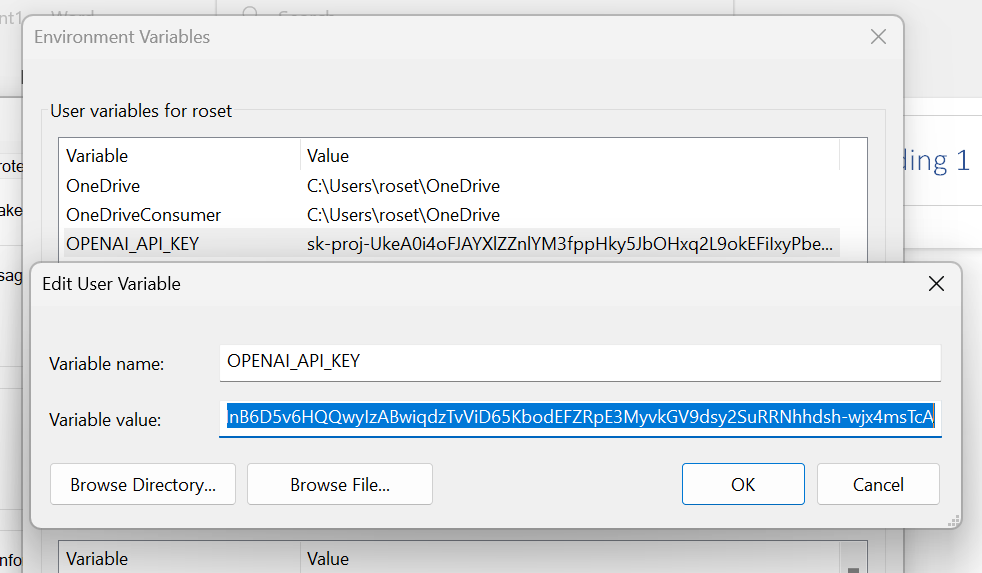
**Environment Variable Set Up to call OpenAI API**

1. Go to System Properties and Click Environment Variables



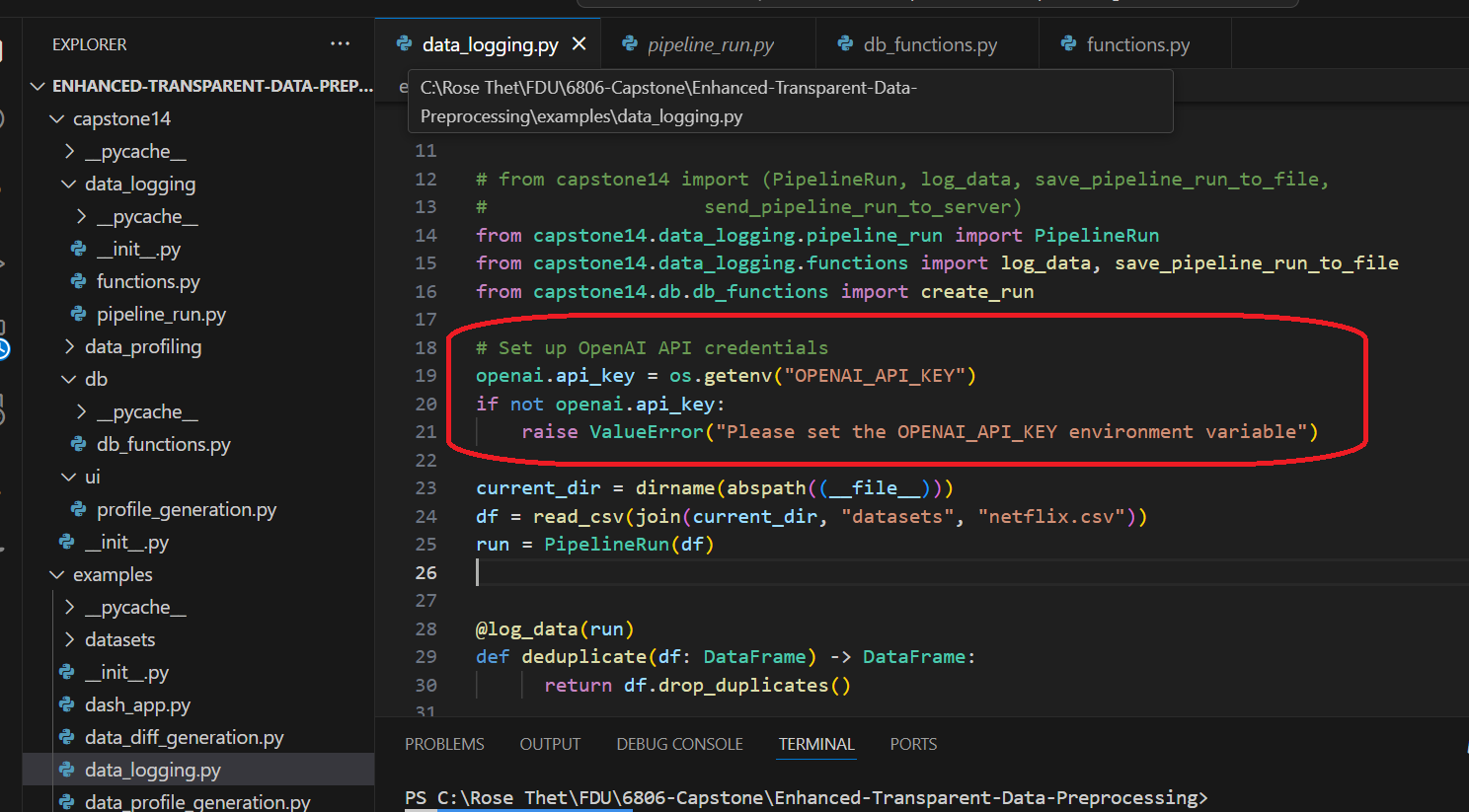
1. Create User Variable with variable name “OPENAI\_API\_KEY” and pass the Variable Value with

“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”



Explanation of Changes in Source Code

* Calling OPENAI\_API\_KEY in source code



* Before LLM’s description generation improvement, Manual Annotation in description



* After LLM’s description generation improvement, Auto Annotation in details from LLM in description



* Modified Data Logging Function to generate annotation from LLM, OpenAI

def generate\_description(func: Callable, args: tuple, kwargs: dict) -> str:

    """

    Generate a description of the processing step using the function's source code

    and context through LLM.

    """

    # Get the function's source code

    source = inspect.getsource(func)

    # Get the function's signature

    sig = inspect.signature(func)

    # Create a context string with argument values

    arg\_names = list(sig.parameters.keys())

    arg\_values = args[:len(arg\_names)]

    args\_context = dict(zip(arg\_names, arg\_values))

    args\_context.update(kwargs)

    # Prepare the prompt for the language model

    prompt = f"""

    Function name: {func.\_\_name\_\_}

    Function source code: {source}

    Arguments: {args\_context}

    Please provide a concise description of what this data processing step does.

    """

    try:

        #print("call openAI")

        # Use OpenAI API to generate description

        response = openai.chat.completions.create(

            model="gpt-3.5-turbo",

            messages=[{

                "role": "user",

                "content": prompt

            }]

        )

        #print(response)

        return response.choices[0].message.content.strip()

    except Exception as e:

        # Fallback to a basic description if LLM fails

        print(e)

        return f"Processing step: {func.\_\_name\_\_}"