AI Assisted Coding LAB ASS-5.4

NAME: M BHARATH

BATCH:14

2303A510A2

Task Description #1:

- Prompt GitHub Copilot to generate a Python script that collects user data (e.g., name, age, email). Then, ask Copilot to add comments on how to anonymize or protect this data.

PROMPT:

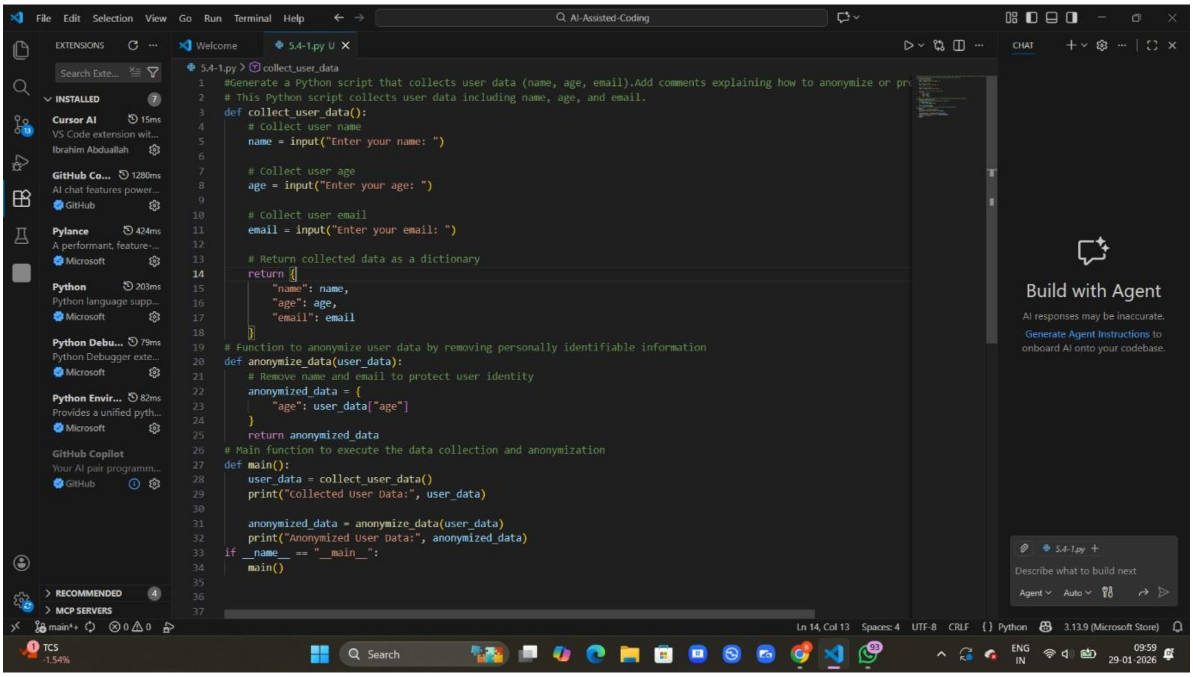
Generate a Python script that collects user data such as name, age, and email.

# Add inline comments explaining how to protect or anonymize this data,

# such as hashing email addresses, avoiding plain-text storage,

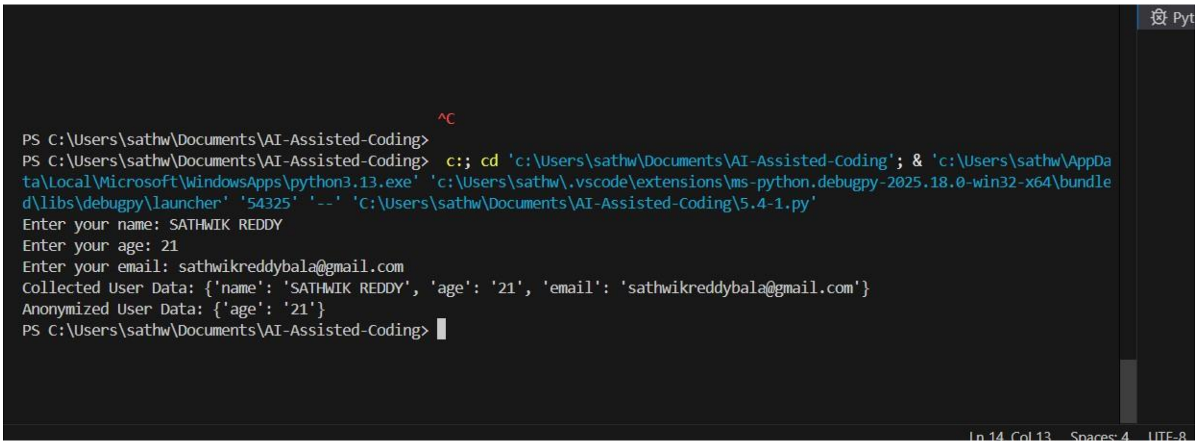
# and following basic privacy best practices.

CODE AND INPUT



```python
#Generate a Python script that collects user data (name, age, email).Add comments explaining how to anonymize or pro
# This Python script collects user data including name, age, and email.
def collect_user_data():
    # Collect user name
    name = input("Enter your name: ")

    # Collect user age
    age = input("Enter your age: ")

    # Collect user email
    email = input("Enter your email: ")

    # Return collected data as a dictionary
    return {
        "name": name,
        "age": age,
        "email": email
    }
# Function to anonymize user data by removing personally identifiable information
def anonymize_data(user_data):
    # Remove name and email to protect user identity
    anonymized_data = {
        "age": user_data["age"]
    }
    return anonymized_data
# Main function to execute the data collection and anonymization
def main():
    user_data = collect_user_data()
    print("Collected User Data:", user_data)

    anonymized_data = anonymize_data(user_data)
    print("Anonymized User Data:", anonymized_data)
if __name__ == "__main__":
    main()
```

OUTPUT:



```
                                                ^C
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding>  c:; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'; & 'c:\Users\sathw\AppDa
ta\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\sathw\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle
d\libs\debugpy\launcher' '54325' '--' 'C:\Users\sathw\Documents\AI-Assisted-Coding\5.4-1.py'
Enter your name: SATHWIK REDDY
Enter your age: 21
Enter your email: sathwikreddybala@gmail.com
Collected User Data: {'name': 'SATHWIK REDDY', 'age': '21', 'email': 'sathwikreddybala@gmail.com'}
Anonymized User Data: {'age': '21'}
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
```

Task Description #2:

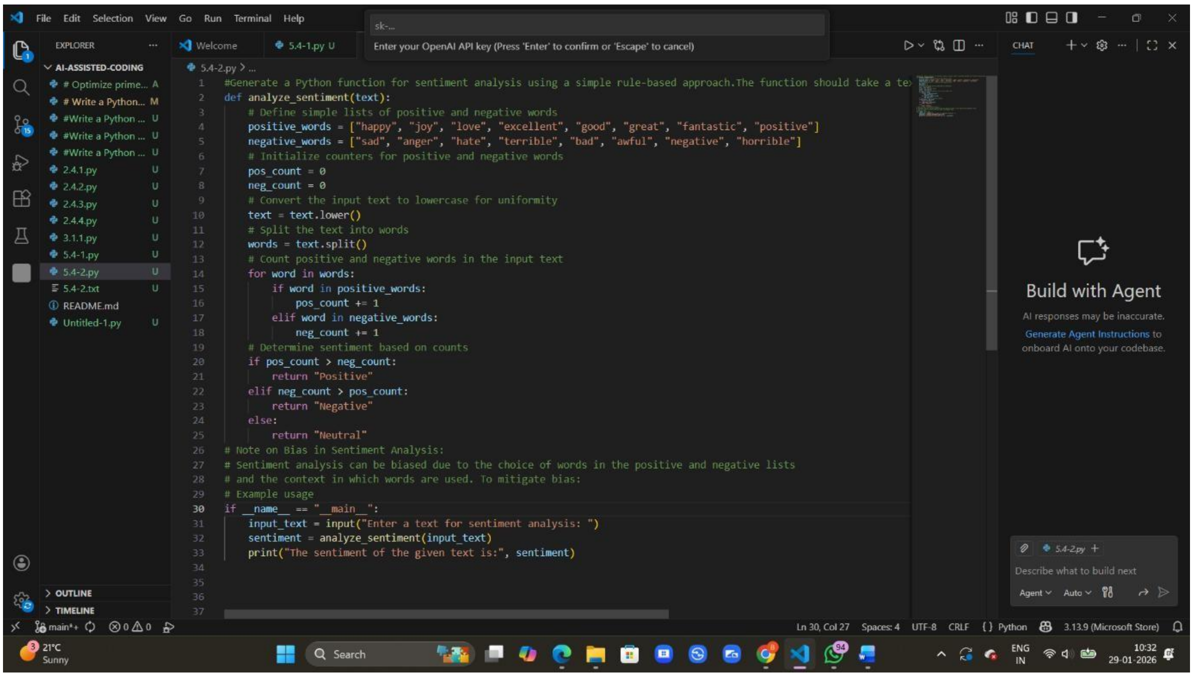
- Ask Copilot to generate a Python function for sentiment analysis. Then prompt Copilot to identify and handle potential biases in the data

PROMPT: **# Generate a Python function for sentiment analysis.**

# Add comments or code to identify and reduce potential biases in the data,

# such as removing offensive terms, balancing positive and negative samples,

# and avoiding biased language in predictions.

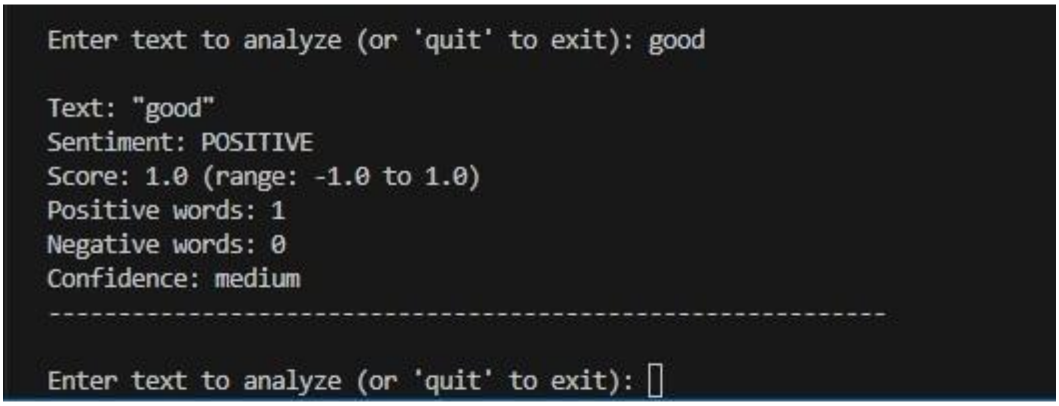


sk-...

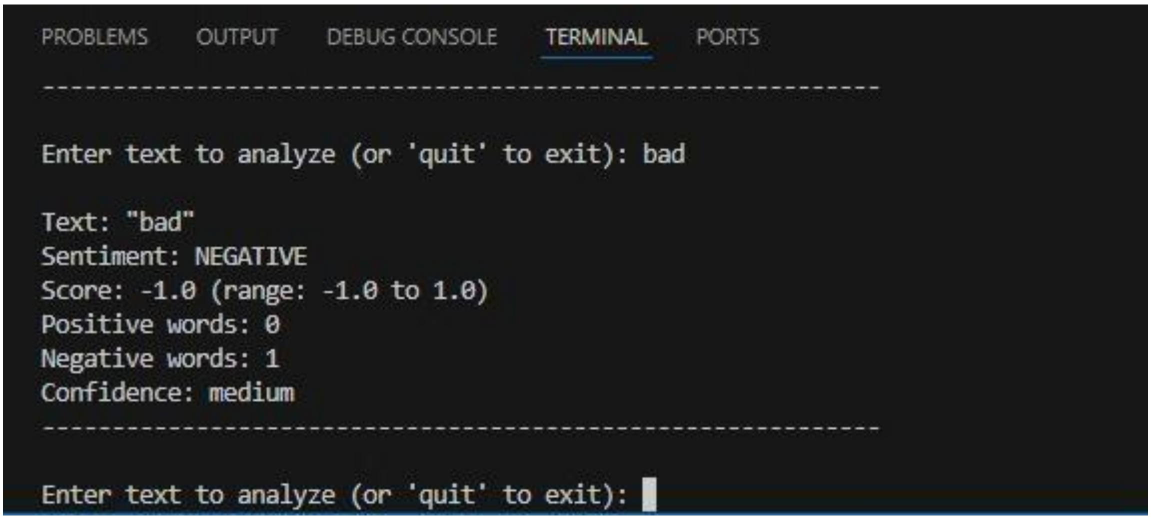
Enter your OpenAI API key (Press 'Enter' to confirm or 'Escape' to cancel)

```python
#Generate a Python function for sentiment analysis using a simple rule-based approach.The function should take a te
def analyze_sentiment(text):
    # Define simple lists of positive and negative words
    positive_words = ["happy", "joy", "love", "excellent", "good", "great", "fantastic", "positive"]
    negative_words = ["sad", "anger", "hate", "terrible", "bad", "awful", "negative", "horrible"]
    # Initialize counters for positive and negative words
    pos_count = 0
    neg_count = 0
    # Convert the input text to lowercase for uniformity
    text = text.lower()
    # Split the text into words
    words = text.split()
    # Count positive and negative words in the input text
    for word in words:
        if word in positive_words:
            pos_count += 1
        elif word in negative_words:
            neg_count += 1
    # Determine sentiment based on counts
    if pos_count > neg_count:
        return "Positive"
    elif neg_count > pos_count:
        return "Negative"
    else:
        return "Neutral"
# Note on Bias in Sentiment Analysis:
# Sentiment analysis can be biased due to the choice of words in the positive and negative lists
# and the context in which words are used. To mitigate bias:
# Example usage
if __name__ == "__main__":
    input_text = input("Enter a text for sentiment analysis: ")
    sentiment = analyze_sentiment(input_text)
    print("The sentiment of the given text is:", sentiment)
```

OUTPUT:



```
Enter text to analyze (or 'quit' to exit): good

Text: "good"
Sentiment: POSITIVE
Score: 1.0 (range: -1.0 to 1.0)
Positive words: 1
Negative words: 0
Confidence: medium
------------------------------------------------------------

Enter text to analyze (or 'quit' to exit): 
```



```
------------------------------------------------------------

Enter text to analyze (or 'quit' to exit): bad

Text: "bad"
Sentiment: NEGATIVE
Score: -1.0 (range: -1.0 to 1.0)
Positive words: 0
Negative words: 1
Confidence: medium
------------------------------------------------------------

Enter text to analyze (or 'quit' to exit): 
```

Task Description #3:

• Use Copilot to write a Python program that recommends products based on user history. Ask it to follow ethical guidelines like transparency and fairness.

PROMPT:

# Generate a Python program that recommends products based on user purchase history.

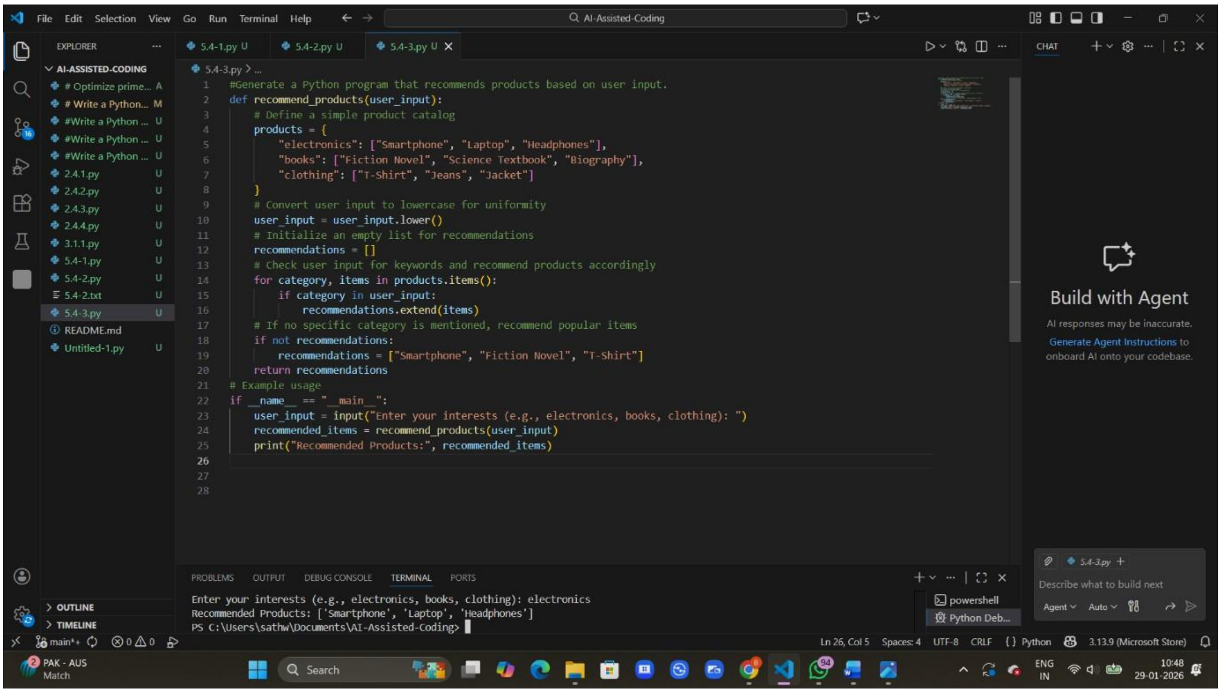
# Follow ethical AI guidelines such as transparency, fairness, and user control.

# Add comments explaining how recommendations are generated,

# Avoid favouritism toward only popular products,

# and allow users to give feedback or opt out of recommendations.

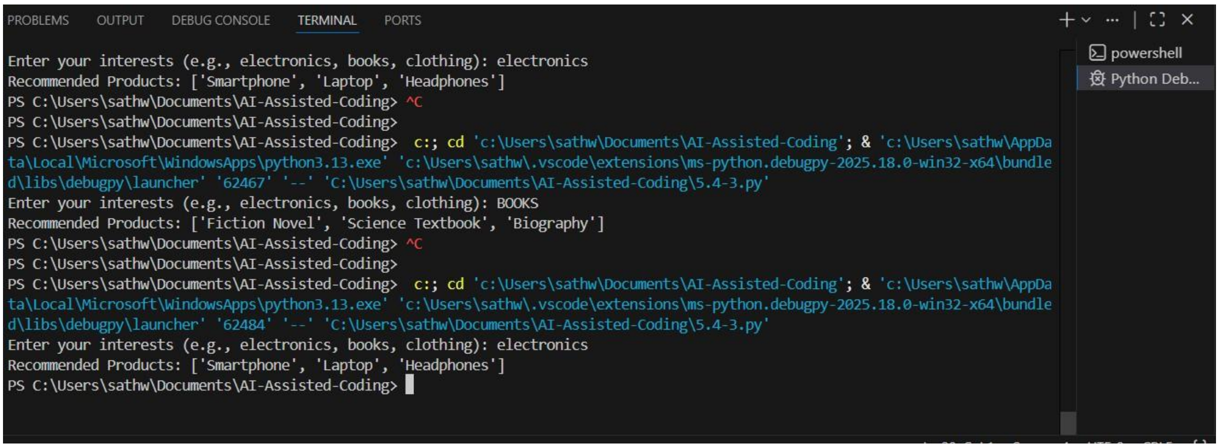
CODE AND INPUT:



```python
#Generate a Python program that recommends products based on user input.
def recommend_products(user_input):
    # Define a simple product catalog
    products = {
        "electronics": ["Smartphone", "Laptop", "Headphones"],
        "books": ["Fiction Novel", "Science Textbook", "Biography"],
        "clothing": ["T-Shirt", "Jeans", "Jacket"]
    }
    # Convert user input to lowercase for uniformity
    user_input = user_input.lower()
    # Initialize an empty list for recommendations
    recommendations = []
    # Check user input for keywords and recommend products accordingly
    for category, items in products.items():
        if category in user_input:
            recommendations.extend(items)
    # If no specific category is mentioned, recommend popular items
    if not recommendations:
        recommendations = ["Smartphone", "Fiction Novel", "T-Shirt"]
    return recommendations
# Example usage
if __name__ == "__main__":
    user_input = input("Enter your interests (e.g., electronics, books, clothing): ")
    recommended_items = recommend_products(user_input)
    print("Recommended Products:", recommended_items)
```

```
Enter your interests (e.g., electronics, books, clothing): electronics
Recommended Products: ['Smartphone', 'Laptop', 'Headphones']
PS C:\Users\sathw\Documents\AI-Assisted-Coding> 
```

OUTPUT:



```
Enter your interests (e.g., electronics, books, clothing): electronics
Recommended Products: ['Smartphone', 'Laptop', 'Headphones']
PS C:\Users\sathw\Documents\AI-Assisted-Coding> ^C
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding>  c:; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'; & 'c:\Users\sathw\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\sathw\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62467' '--' 'C:\Users\sathw\Documents\AI-Assisted-Coding\5.4-3.py'
Enter your interests (e.g., electronics, books, clothing): BOOKS
Recommended Products: ['Fiction Novel', 'Science Textbook', 'Biography']
PS C:\Users\sathw\Documents\AI-Assisted-Coding> ^C
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding>  c:; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'; & 'c:\Users\sathw\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\sathw\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62484' '--' 'C:\Users\sathw\Documents\AI-Assisted-Coding\5.4-3.py'
Enter your interests (e.g., electronics, books, clothing): electronics
Recommended Products: ['Smartphone', 'Laptop', 'Headphones']
PS C:\Users\sathw\Documents\AI-Assisted-Coding> 
```

Task Description #4:

- **Prompt Copilot to generate logging functionality in a Python web application. Then, ask it to ensure the logs do not record sensitive information.**

PROMPT:

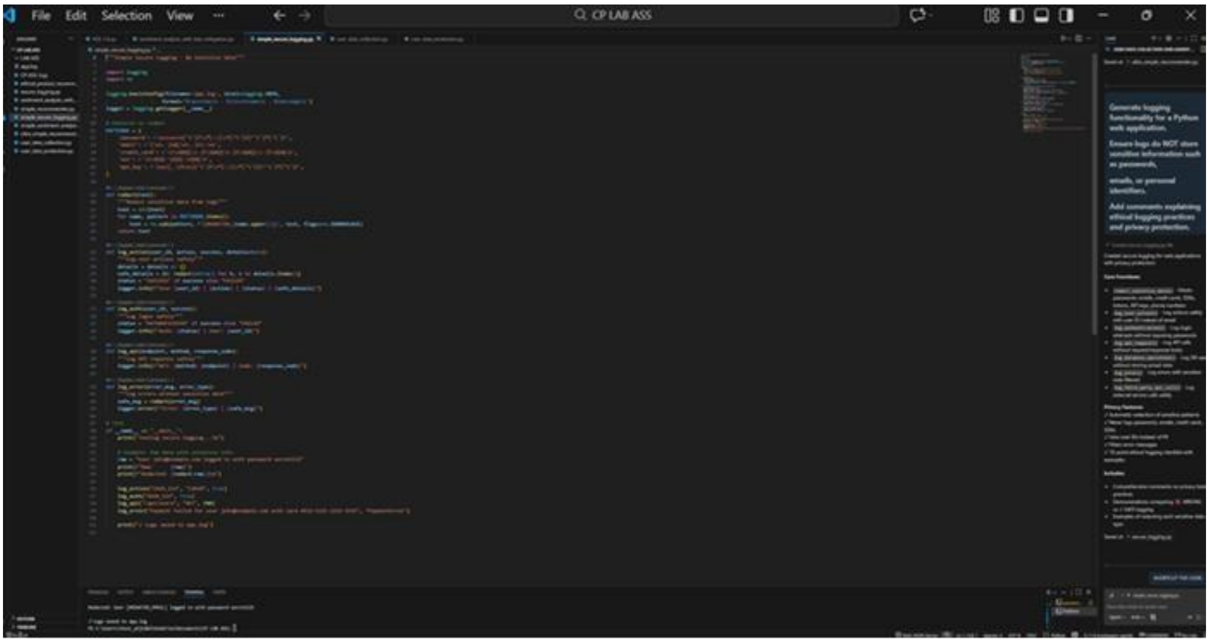
# Generate logging functionality for a Python web application.

# Ensure logs do NOT store sensitive information such as passwords,

# emails, or personal identifiers.

# Add comments explaining ethical logging practices and privacy protection.

CODE AND INPUT:



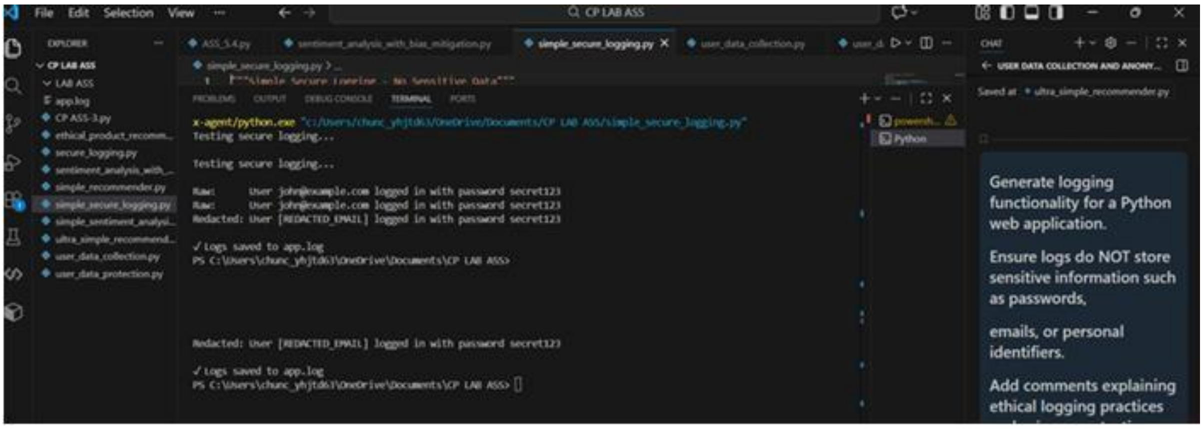
Generate logging functionality for a Python web application.

Ensure logs do NOT store sensitive information such as passwords,

emails, or personal identifiers.

Add comments explaining ethical logging practices and privacy protection.

OUTPUT:



```
"""Simple Secure Logging - No Sensitive Data"""
```

```
x-agent/python.exe "c:/Users/chunc_yhjtd63/OneDrive/Documents/CP LAB ASS/simple_secure_logging.py"
Testing secure logging...

Testing secure logging...

Raw:      User john@example.com logged in with password secret123
Raw:      User john@example.com logged in with password secret123
Redacted: User [REDACTED_EMAIL] logged in with password secret123

✓ Logs saved to app.log
PS C:\Users\chunc_yhjtd63\OneDrive\Documents\CP LAB ASS>




Redacted: User [REDACTED_EMAIL] logged in with password secret123

✓ Logs saved to app.log
PS C:\Users\chunc_yhjtd63\OneDrive\Documents\CP LAB ASS> []
```

Generate logging functionality for a Python web application.

Ensure logs do NOT store sensitive information such as passwords,

emails, or personal identifiers.

Add comments explaining ethical logging practices

Task Description #5:

• Ask Copilot to generate a machine learning model. Then, prompt it to add documentation on how to use the model responsibly (e.g., explainability, accuracy limits).

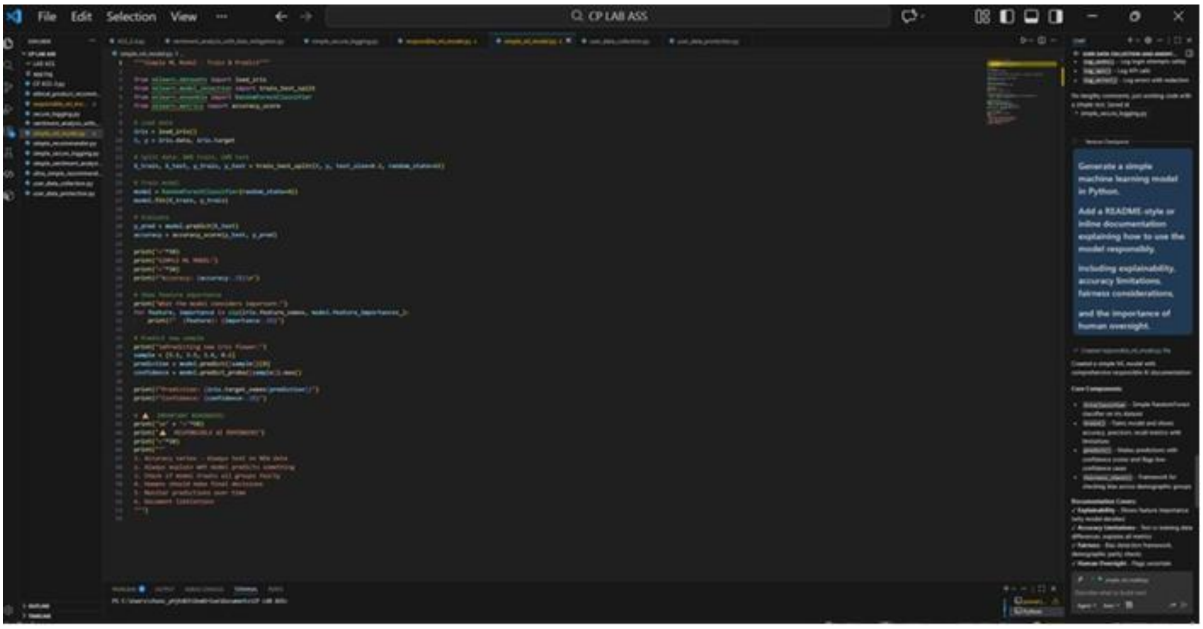
PROMPT: # Generate a simple machine learning model in Python.

# Add a README-style or inline documentation explaining how to use the model responsibly,

# including explainability, accuracy limitations, fairness considerations,

# and the importance of human oversight.

CODE AND INPUT:



Generate a simple machine learning model in Python.

Add a README-style or inline documentation explaining how to use the model responsibly.

including explainability, accuracy limitations, fairness considerations,

and the importance of human oversight.

OUTPUT: