ASSIGNMENT - 5.4

2303A51355

Batch-10

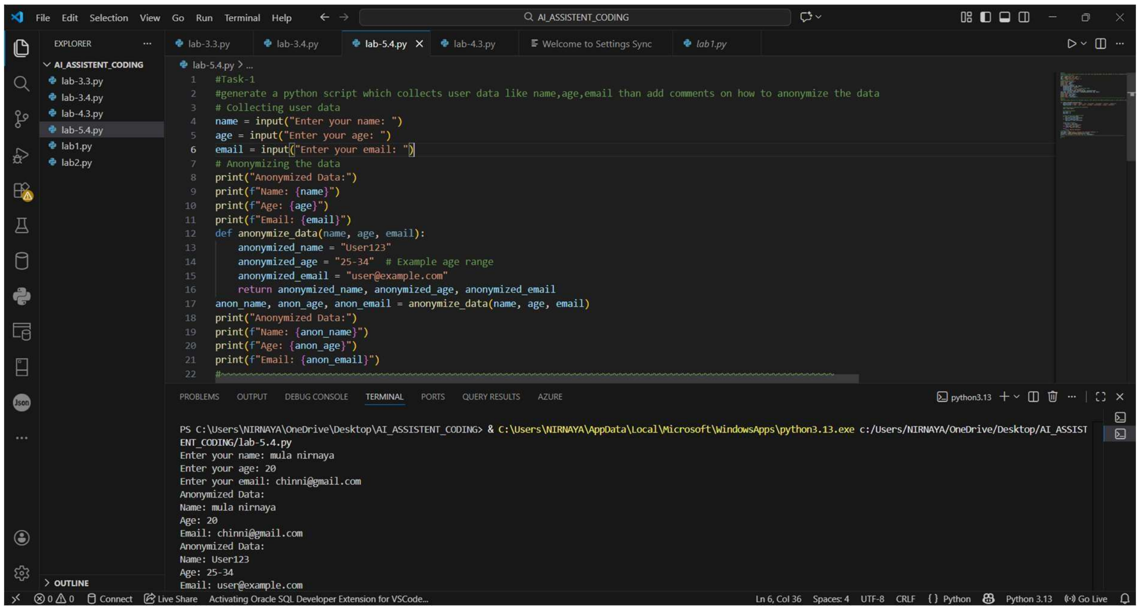
Task-1

Prompt: generate a Python script that collects user data like name, age, and email, then add comments on how to anonymise the data

Code :

```
# Collec ng user data name =
input("Enter your name: ") age =
input("Enter your age: ") email =
input("Enter your email: ") #
Anonymizing the data
print("Anonymized Data:")
print(f"Name: {name}") print(f"Age:
{age}") print(f"Email: {email}") def
anonymize_data(name, age, email):
    anonymized_name = "User123"     anonymized_age = "25-34"  #
Example age range     anonymized_email = "user@example.com"
return anonymized_name, anonymized_age, anonymized_email
anon_name, anon_age, anon_email = anonymize_data(name, age, email)
print("Anonymized Data:")    print(f"Name: {anon_name}")
print(f"Age: {anon_age}")
print(f"Email: {anon_email}")
```

Output :



```python
#Task-1
#generate a python script which collects user data like name,age,email than add comments on how to anonymize the data
# Collecting user data
name = input("Enter your name: ")
age = input("Enter your age: ")
email = input("Enter your email: ")
# Anonymizing the data
print("Anonymized Data:")
print(f"Name: {name}")
print(f"Age: {age}")
print(f"Email: {email}")
def anonymize_data(name, age, email):
    anonymized_name = "User123"
    anonymized_age = "25-34"  # Example age range
    anonymized_email = "user@example.com"
    return anonymized_name, anonymized_age, anonymized_email
anon_name, anon_age, anon_email = anonymize_data(name, age, email)
print("Anonymized Data:")
print(f"Name: {anon_name}")
print(f"Age: {anon_age}")
print(f"Email: {anon_email}")
#
```

```
PS C:\Users\NIRNAYA\OneDrive\Desktop\AI_ASSISTENT_CODING> & C:\Users\NIRNAYA\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/NIRNAYA/OneDrive/Desktop/AI_ASSIST
ENT_CODING/lab-5.4.py
Enter your name: mula nirnaya
Enter your age: 20
Enter your email: chinni@gmail.com
Anonymized Data:
Name: mula nirnaya
Age: 20
Email: chinni@gmail.com
Anonymized Data:
Name: User123
Age: 25-34
Email: user@example.com
```

Code Analysis :

- The program first asks the user to enter personal details like name, age, and email using input().
- These values are stored in variables so they can be processed later.
- The anonymize_data() function replaces real data with dummy values to protect privacy.
- This shows how personal data can be hidden or masked before sharing or storing it.

Task-2

Prompt: generate python func on for sen ment analysis than iden fy and handle poten al biases in data used for analysis without using modules

Code :

def simple_sen ment_analysis(text):

   posi ve_words = ['good', 'happy', 'joy', 'excellent', 'fortunate', 'correct', 'superior']

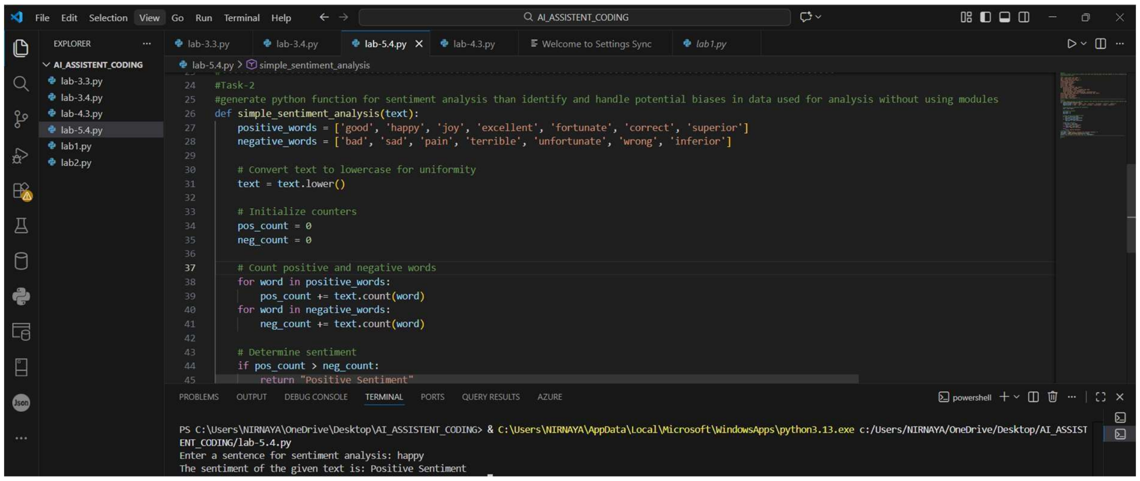
nega ve_words = ['bad', 'sad', 'pain', 'terrible', 'unfortunate', 'wrong', 'inferior']

   # Convert text to lowercase for uniformity

text = text.lower()

```
    # Ini alize counters
pos_count = 0
neg_count = 0


    # Count posi ve and nega ve words
for word in posi ve_words:
        pos_count += text.count(word)
for word in nega ve_words:
        neg_count += text.count(word)


    # Determine sen ment
    if   pos_count   >   neg_count:
return "Posi ve Sen ment"      elif
neg_count        >        pos_count:
return "Nega ve Sen ment"
    else:
        return "Neutral Sen ment"
# Example usage user_input = input("Enter a sentence for sen
ment     analysis:     ")          sen      ment      =      simple_sen
ment_analysis(user_input)
print(f"The sen ment of the given text is: {sen ment}")
```

Output :



```python
#Task-2
#generate python function for sentiment analysis than identify and handle potential biases in data used for analysis without using modules
def simple_sentiment_analysis(text):
    positive_words = ['good', 'happy', 'joy', 'excellent', 'fortunate', 'correct', 'superior']
    negative_words = ['bad', 'sad', 'pain', 'terrible', 'unfortunate', 'wrong', 'inferior']

    # Convert text to lowercase for uniformity
    text = text.lower()

    # Initialize counters
    pos_count = 0
    neg_count = 0

    # Count positive and negative words
    for word in positive_words:
        pos_count += text.count(word)
    for word in negative_words:
        neg_count += text.count(word)

    # Determine sentiment
    if pos_count > neg_count:
        return "Positive Sentiment"
```

```
PS C:\Users\NIRNAYA\OneDrive\Desktop\AI_ASSISTENT_CODING> & C:\Users\NIRNAYA\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/NIRNAYA/OneDrive/Desktop/AI_ASSIST
ENT_CODING/lab-5.4.py
Enter a sentence for sentiment analysis: happy
The sentiment of the given text is: Positive Sentiment
```

Code Analysis :

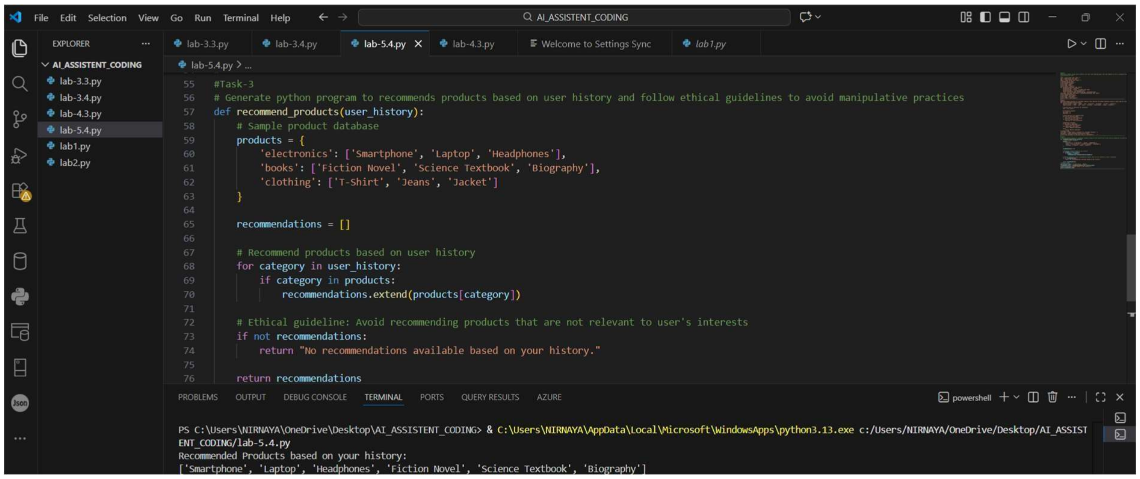
- ☐ The function checks the text for positive and negative words using predefined lists.
- ☐ The input text is converted to lowercase to avoid case-sensitive errors.
- ☐ It counts how many positive and negative words are present in the sentence.
- ☐ Based on the count, the program decides whether the sentiment is Positive, Negative, or Neutral.

Task-3

Prompt : Generate python program to recommends products based on user history and follow ethical guidelines to avoid manipula ve prac ces def recommend_products(user_history):

```
    # Sample product database
    products = {
        'electronics': ['Smartphone', 'Laptop', 'Headphones'],
        'books': ['Fic on Novel', 'Science Textbook', 'Biography'],
        'clothing': ['T-Shirt', 'Jeans', 'Jacket']
    }

    recommenda ons = []

    # Recommend products based on user history
    for category in user_history:
if category in products:
            recommenda ons.extend(products[category])


    # Ethical guideline: Avoid recommending products that are not relevant to user's interests
if not recommenda ons:
        return "No recommenda ons available based on your history."


    return recommenda ons   #
Example usage
user_history_input = ['electronics', 'books']
recommended_items = recommend_products(user_history_input)
print("Recommended Products based on your history:") print(recommended_items)
```

Output :



```
#Task-3
# Generate python program to recommends products based on user history and follow ethical guidelines to avoid manipulative practices
def recommend_products(user_history):
    # Sample product database
    products = {
        'electronics': ['Smartphone', 'Laptop', 'Headphones'],
        'books': ['Fiction Novel', 'Science Textbook', 'Biography'],
        'clothing': ['T-Shirt', 'Jeans', 'Jacket']
    }

    recommendations = []

    # Recommend products based on user history
    for category in user_history:
        if category in products:
            recommendations.extend(products[category])

    # Ethical guideline: Avoid recommending products that are not relevant to user's interests
    if not recommendations:
        return "No recommendations available based on your history."

    return recommendations
```

```
PS C:\Users\NIRNAYA\OneDrive\Desktop\AI_ASSISTENT_CODING> & C:\Users\NIRNAYA\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/NIRNAYA/OneDrive/Desktop/AI_ASSIST
ENT_CODING/lab-5.4.py
Recommended Products based on your history:
['Smartphone', 'Laptop', 'Headphones', 'Fiction Novel', 'Science Textbook', 'Biography']
```

Code Analysis :

- The program stores products in a dictionary based on categories like electronics and books.
- It checks the user’s past interests (user_history) to suggest related products.
- Only relevant items are recommended, avoiding unnecessary or misleading suggestions.

- This follows ethical guidelines by respecting user preferences and avoiding manipulation.

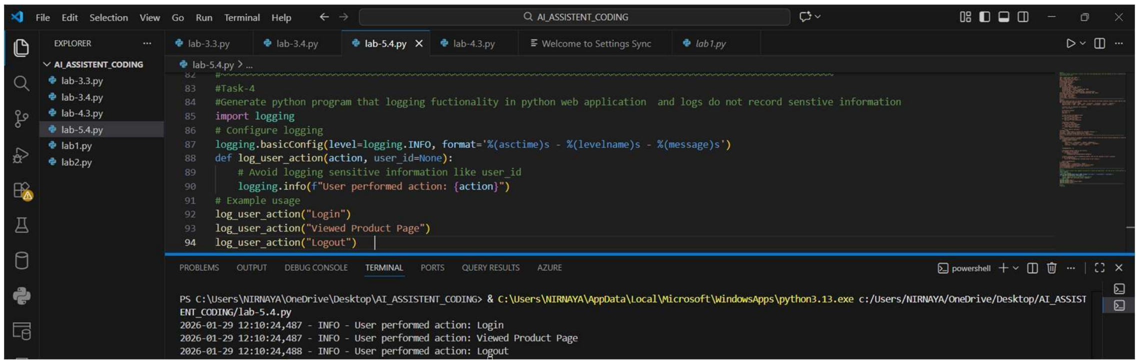
Task-4

Prompt: Generate python program that logging fuc onality in python web applica on and logs do not record sens ve informa on

Code :

```
import logging   #

Configure logging

logging.basicConfig(level=logging.INFO, format='%(asc me)s - %(levelname)s -
%(message)s') def log_user_ac on(ac on,

user_id=None):     # Avoid logging sensi

ve informa on like user_id

logging.info(f"User performed ac on: {ac

on}")

#      Example      usage      log_user_ac

on("Login")   log_user_ac   on("Viewed

Product          Page")          log_user_ac

on("Logout")
```

Output :



```
#Task-4
#Generate python program that logging fuctionality in python web application  and logs do not record senstive information
import logging
# Configure logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
def log_user_action(action, user_id=None):
    # Avoid logging sensitive information like user_id
    logging.info(f"User performed action: {action}")
# Example usage
log_user_action("Login")
log_user_action("Viewed Product Page")
log_user_action("Logout")
```

```
PS C:\Users\NIRNAYA\OneDrive\Desktop\AI_ASSISTENT_CODING> & C:\Users\NIRNAYA\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/NIRNAYA/OneDrive/Desktop/AI_ASSIST
ENT_CODING/lab-5.4.py
2026-01-29 12:10:24,487 - INFO - User performed action: Login
2026-01-29 12:10:24,487 - INFO - User performed action: Viewed Product Page
2026-01-29 12:10:24,488 - INFO - User performed action: Logout
```

Code Analysis :

- The program uses Python’s logging feature to record user actions.

- It logs only general actions like login or logout, not private data.
- Sensitive details such as user ID or passwords are intentionally avoided.
- This improves system monitoring while maintaining user privacy and security.

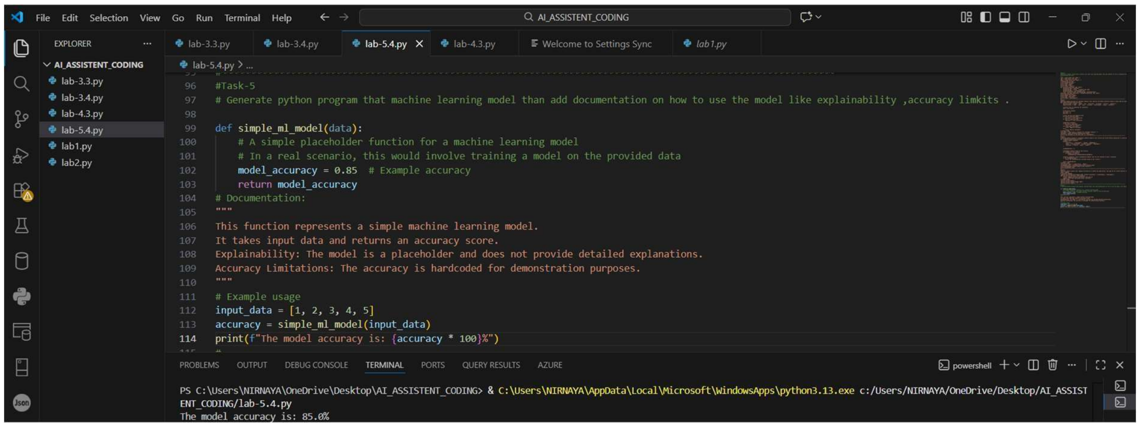
Task 5

Prompt : Generate python program that machine learning model than add documenta on on how to use the model like explainability ,accuracy limkits .

code :

```
def simple_ml_model(data):

    # A simple placeholder func on for a machine learning model

    # In a real scenario, this would involve training a model on the provided data

model_accuracy = 0.85  # Example accuracy

    return model_accuracy

# Documenta on:

"""

This func on represents a simple machine learning model.

It takes input data and returns an accuracy score.

Explainability: The model is a placeholder and does not provide detailed explana ons.

Accuracy Limita ons: The accuracy is hardcoded for demonstra on purposes.

"""

# Example usage

input_data = [1, 2, 3, 4, 5]

accuracy = simple_ml_model(input_data)

print(f"The model accuracy is: {accuracy * 100}%")
```

Output :



```python
#Task-5
# Generate python program that machine learning model than add documentation on how to use the model like explainability ,accuracy limkits .

def simple_ml_model(data):
    # A simple placeholder function for a machine learning model
    # In a real scenario, this would involve training a model on the provided data
    model_accuracy = 0.85  # Example accuracy
    return model_accuracy
# Documentation:
"""
This function represents a simple machine learning model.
It takes input data and returns an accuracy score.
Explainability: The model is a placeholder and does not provide detailed explanations.
Accuracy Limitations: The accuracy is hardcoded for demonstration purposes.
"""
# Example usage
input_data = [1, 2, 3, 4, 5]
accuracy = simple_ml_model(input_data)
print(f"The model accuracy is: {accuracy * 100}%")
```

```
PS C:\Users\NIRNAYA\OneDrive\Desktop\AI_ASSISTENT_CODING> & C:\Users\NIRNAYA\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/NIRNAYA/OneDrive/Desktop/AI_ASSIST
ENT_CODING/lab-5.4.py
The model accuracy is: 85.0%
```

Code Analysis :

- The func on represents a basic machine learning model using a placeholder.
- It returns a fixed accuracy value for demonstra on purposes.
- Comments explain that the model does not show real predic ons or explana ons.
- Documenta on clearly men ons limita ons in accuracy and explainability.