

## ASSIGNMENT-5.4

NAME:-N.ROHAN

BATCH-16

HTNO:-2303A51064

### Task 1: Ethical Handling of User Data (Privacy & Anonymization)

The screenshot shows the Thonny IDE interface. The top window displays a Python script named 'untitled'. The code prompts the user for their name, age, and email, then hashes the email using SHA-256. It includes a note about the ethical handling of user data, emphasizing that emails should never be stored in plain text and that hashing helps anonymize sensitive information.

```
Thonny - <untitled> @ 22:1
File Edit View Run Tools Help
 * 
1 # Program to collect basic user data
2 # NOTE: Sensitive data should always be protected or anonymized
3
4 import hashlib
5
6 name = input("Enter your name: ")
7 age = int(input("Enter your age: "))
8 email = input("Enter your email: ")
9
10 # Hashing email to protect user identity
11 hashed_email = hashlib.sha256(email.encode()).hexdigest()
12
13 print("\n--- User Data (Protected) ---")
14 print("Name:", name)
15 print("Age:", age)
16 print("Email (Hashed):", hashed_email)
17
18 # Ethical Note:
19 # - Emails should never be stored in plain text
20 # - Hashing helps anonymize sensitive information
21 # - Personal data should be stored securely or avoided if not required
22
```

The bottom window, titled 'Shell', shows the execution of the script. It prompts the user for their name, age, and email, then displays the protected user data, including the hashed email.

```
Shell >>> %RUN -C $EDITOR_CONTENT
Enter your name: ROHAN
Enter your age: 20
Enter your email: ROHANNAMUTHABAJI@GMAIL.COM

--- User Data (Protected) ---
Name: ROHAN
Age: 20
Email (Hashed): 7b86b08ec0b7222a0930d88b0f8b6a16ac73734e48555921c88b3cc14a57a480
>>> |
```

### Task 2: Sentiment Analysis with Bias Awareness

The screenshot shows the Thonny Python IDE interface. The top menu bar includes File, Edit, View, Run, Tools, and Help. Below the menu is a toolbar with icons for file operations like Open, Save, and Run, along with a Stop button. The main window has two tabs: 'untitled' and 'Shell'. The 'untitled' tab contains a Python script for sentiment analysis:

```
1 # Simple sentiment analysis with bias awareness
2
3 def analyze_sentiment(text):
4     positive_words = ["good", "happy", "excellent", "great"]
5     negative_words = ["bad", "sad", "terrible", "poor"]
6
7     score = 0
8     words = text.lower().split()
9
10    for word in words:
11        if word in positive_words:
12            score += 1
13        elif word in negative_words:
14            score -= 1
15
16    return "Positive" if score > 0 else "Negative" if score < 0 else "Neutral"
17
18
19 # Ethical considerations:
20 # - Avoid biased or offensive words in training data
21 # - Use balanced datasets
22 # - Regularly audit sentiment outputs for fairness
23
24 print(analyze_sentiment("This product is excellent and good"))
25 print(analyze_sentiment("This service is bad and terrible"))
26
```

The 'Shell' tab shows the output of running the script:

```
>>> %Run -c $EDITOR_CONTENT
Positive
Negative
>>>
```

### Task 3: Ethical Product Recommendation System

The screenshot shows the Thonny Python IDE interface. The top menu bar includes File, Edit, View, Run, Tools, and Help. Below the menu is a toolbar with icons for file operations like Open, Save, and Run, along with a Stop button and a flag icon.

The code editor window contains a script named 'untitled'. The code defines a function 'recommend\_products' that takes 'user\_history' and 'products' as parameters. It iterates through 'products', checking if each product's category is in 'user\_history'. If so, it adds the product's name to a list called 'recommendations'. Finally, it returns the list of recommendations. The script also includes example data for 'user\_history' and 'products'.

```
Thonny - <untitled> @ 31:1
File Edit View Run Tools Help
File Open Save Run Stop Flag
<untitled> * ×
5
6     for product in products:
7         # Avoid favoritism by recommending based on relevance only
8         if product["category"] in user_history:
9             recommendations.append(product["name"])
10
11     return recommendations
12
13
14 # Example data
15 user_history = ["electronics", "books"]
16
17 products = [
18     {"name": "Laptop", "category": "electronics"},
19     {"name": "Novel", "category": "books"},
20     {"name": "Shoes", "category": "fashion"}
21 ]
22
23 recommended = recommend_products(user_history, products)
24
25 print("Recommended Products:", recommended)
26
27 # Ethical Notes:
28 # - Recommendations should be transparent
29 # - Avoid unfair bias toward sponsored or preferred products
30 # - Allow user feedback to improve fairness
31
```

The shell window below shows the execution of the script. The command '%Run -c \$EDITOR\_CONTENT' is run, followed by the output 'Recommended Products: ['Laptop', 'Novel']'.

```
Shell ×
>>> %Run -c $EDITOR_CONTENT
Recommended Products: ['Laptop', 'Novel']
>>>
```

#### Task 4: Ethical Logging (Avoiding Sensitive Data)

The screenshot shows the Thonny Python IDE interface. The top window is titled "Thonny - <untitled> @ 23:1". It contains a code editor with the following Python script:

```
1 # Logging functionality with ethical considerations
2
3 import logging
4
5 logging.basicConfig(
6     filename="app.log",
7     level=logging.INFO,
8     format"%(asctime)s - %(levelname)s - %(message)s"
9 )
10
11 def login_user(username):
12     # Do NOT log passwords or sensitive personal data
13     logging.info(f"User login attempt: {username}")
14     print("Login successful")
15
16
17 login_user("student_user")
18
19 # Ethical Logging Guidelines:
20 # - Never log passwords or emails
21 # - Logs should be minimal and purpose-driven
22 # - Protect log files from unauthorized access
23
```

The bottom window is titled "Shell" and shows the output of running the script:

```
>>> %Run -c $EDITOR_CONTENT
Login successful
>>>
```

### Task 5: Responsible Machine Learning Model Usage

Thonny - <untitled> @ 25 : 1

File Edit View Run Tools Help



<untitled> \* ×

```
1 # Simple ML model example with responsible usage documentation
2
3 from sklearn.linear_model import LinearRegression
4 import numpy as np
5
6 # Sample training data
7 X = np.array([[1], [2], [3], [4], [5]])
8 y = np.array([2, 4, 6, 8, 10])
9
10 model = LinearRegression()
11 model.fit(X, y)
12
13 # Prediction
14 prediction = model.predict([[6]])
15 print("Predicted value:", prediction)
16
17 """
18 Responsible Usage Notes:
19 - This model is trained on a very small dataset
20 - Predictions may not generalize to real-world data
21 - Always validate accuracy using proper test data
22 - Ensure training data is unbiased and representative
23 - Provide transparency when deploying ML models
24 """
25 |
```