ASSIGNMENT-5.1

Name: B.Shravya

HT. No: 2303A51492

Batch: 08

Lab 5: Ethical Foundations – Responsible AI Coding Practices

Task Description #1 (Privacy in API Usage)

Task: Use an AI tool to generate a Python program that connects to a weather API.

```python
#Task1
#Generate code to fetch weather data securely without exposing API keys in the code.

import os
import requests
API_KEY = os.getenv("WEATHER_API_KEY")
CITY = "London"
url = f"https://api.weatherapi.com/v1/current.json?key={API_KEY}&q={CITY}"
response = requests.get(url)

if response.status_code == 200:
    print(response.json())
else:
    print("Error fetching data")
```

```
Error fetching data
```

Task Description #2 (Privacy & Security in File Handling)

Task: Use an AI tool to generate a Python script that stores user data (name, email, password) in a file.

Analyze: Check if the AI stores sensitive data in plain text or without encryption.

```python
#Task2
#generate a Python script that stores user data (name, email, password) in a file

import hashlib

name = input("Enter name: ")
email = input("Enter email: ")
password = input("Enter password: ")

hashed_password = hashlib.sha256(password.encode()).hexdigest()

with open("users.txt", "a") as file:
    file.write(f"{name},{email},{hashed_password}\n")

print("User data stored securely")
```

```
Enter name: ROSE
Enter email: kani@fmail.com
Enter password: adnjobd
User data stored securely
```

Task Description #3 (Transparency in Algorithm Design)

Objective: Use AI to generate an Armstrong number checking function.

```python
#Task3
#to generate an Armstrong number checking function
def is_armstrong(number):
    total = 0
    digits = len(str(number))
    temp = number
    while temp > 0:
        digit = temp % 10
        total += digit ** digits
        temp //= 10
    return total == number
num = int(input("Enter a number: "))
if is_armstrong(num):
    print("Armstrong Number")
else:
    print("Not an Armstrong Number")
```

```
Enter a number: 153
Armstrong Number
```

Task Description #4 (Transparency in Algorithm Comparison) Task:

Use AI to implement two sorting algorithms (e.g., QuickSort and BubbleSort).

Prompt:

"Generate Python code for QuickSort and BubbleSort, and include comments explaining step-by-step how each works and where they differ."

```python
#Task4
#Generate Python code for QuickSort and BubbleSort
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
    return arr
def quick_sort(arr):
    if len(arr) <= 1:
        return arr
    pivot = arr[len(arr) // 2]
    left = [x for x in arr if x < pivot]
    middle = [x for x in arr if x == pivot]
    right = [x for x in arr if x > pivot]
    return quick_sort(left) + middle + quick_sort(right)
data = [64, 34, 25, 12, 22, 11, 90]
print("Bubble Sort:", bubble_sort(data.copy()))
print("Quick Sort:", quick_sort(data.copy()))
```

```
Bubble Sort: [11, 12, 22, 25, 34, 64, 90]
Quick Sort: [11, 12, 22, 25, 34, 64, 90]
```

Task Description #5 (Transparency in AI Recommendations) Task:

Use AI to create a product recommendation system.

Prompt:

"Generate a recommendation system that also provides reasons for each suggestion."

```python
#Task5
#Generate a recommendation system that also provides reasons for each
suggestion.

def recommend_products(user_history, products):
    recommendations = []
    for product in products:
        if product not in user_history:
            reason = f"Recommended because you liked {user_history[-1]}"
            recommendations.append((product, reason))
    return recommendations
user_history = ["Mobile", "Headphones"]
products = ["Mobile", "Headphones", "Smart Watch", "Bluetooth Speaker"]
results = recommend_products(user_history, products)
for product, reason in results:
    print(product, "-", reason)
```

```
Smart Watch - Recommended because you liked Headphones
Bluetooth Speaker - Recommended because you liked Headphones
```