

## ASSIGNMENT-5.1

**Name:** Nandini Reddy

**HT. No:** 2303A51896

**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.

```
1  #Task1
2  #Generate code to fetch weather data securely without exposing API keys in the
   code.
3
4  import os
5  import requests
6  API_KEY = os.getenv("WEATHER_API_KEY")
7  CITY = "London"
8  url = f"https://api.weatherapi.com/v1/current.json?key={API_KEY}&q={CITY}"
9  response = requests.get(url)
10
11 if response.status_code == 200:
12     print(response.json())
13 else:
14     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.

```

1  #Task2
2  #generate a Python script that stores user data (name, email, password) in a
   file
3
4  import hashlib
5
6  name = input("Enter name: ")
7  email = input("Enter email: ")
8  password = input("Enter password: ")
9
10 hashed_password = hashlib.sha256(password.encode()).hexdigest()
11
12 with open("users.txt", "a") as file:
13     file.write(f"{name},{email},{hashed_password}\n")
14
15 print("User data stored securely")

```

Enter name: ROSE  
Enter email: [kani@fmail.com](mailto: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.

```

1  #Task3
2  #to generate an Armstrong number checking function
3  def is_armstrong(number):
4      total = 0
5      digits = len(str(number))
6      temp = number
7      while temp > 0:
8          digit = temp % 10
9          total += digit ** digits
10         temp //= 10
11     return total == number
12 num = int(input("Enter a number: "))
13 if is_armstrong(num):
14     print("Armstrong Number")
15 else:
16     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."

```

1 #Task4
2 #Generate Python code for QuickSort and BubbleSort
3 def bubble_sort(arr):
4     n = len(arr)
5     for i in range(n):
6         for j in range(0, n - i - 1):
7             if arr[j] > arr[j + 1]:
8                 arr[j], arr[j + 1] = arr[j + 1], arr[j]
9     return arr
10 def quick_sort(arr):
11     if len(arr) <= 1:
12         return arr
13     pivot = arr[len(arr) // 2]
14     left = [x for x in arr if x < pivot]
15     middle = [x for x in arr if x == pivot]
16     right = [x for x in arr if x > pivot]
17     return quick_sort(left) + middle + quick_sort(right)
18 data = [64, 34, 25, 12, 22, 11, 90]
19 print("Bubble Sort:", bubble_sort(data.copy()))
20 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."

```

1 #Task5
2 #Generate a recommendation system that also provides reasons for each
  suggestion.
3
4 def recommend_products(user_history, products):
5     recommendations = []
6     for product in products:
7         if product not in user_history:
8             reason = f"Recommended because you liked {user_history[-1]}"
9             recommendations.append((product, reason))
10    return recommendations
11 user_history = ["Mobile", "Headphones"]
12 products = ["Mobile", "Headphones", "Smart Watch", "Bluetooth Speaker"]
13 results = recommend_products(user_history, products)
14 for product, reason in results:
15     print(product, "-", reason)

```

Smart Watch - Recommended because you liked Headphones

Bluetooth Speaker - Recommended because you liked Headphones