Lab 5: Ethical Foundations – Responsible AI Coding Practices

Week 3 – Monday

Name: P.Sathvika

Hall Ticket No: 2303A51497

Lab Objec2ves

• To understand ethical risks involved in AI-generated code.
• To identify issues related to privacy,security, and transparency.
• To analyzethe responsibility of developers when using AI tools.
• To promote responsible and ethical AI coding practices.

Lab Outcomes

After completing this lab, students will be able to:
• Identify insecure coding patterns generated by AI tools.
• Analyze privacy and security risks in AI-generated programs.
• Understand the importance of transparency and explainability.
• Recognize the role of human responsibility in ethical AI coding.

Task Descrip2on #1: Privacy in API Usage

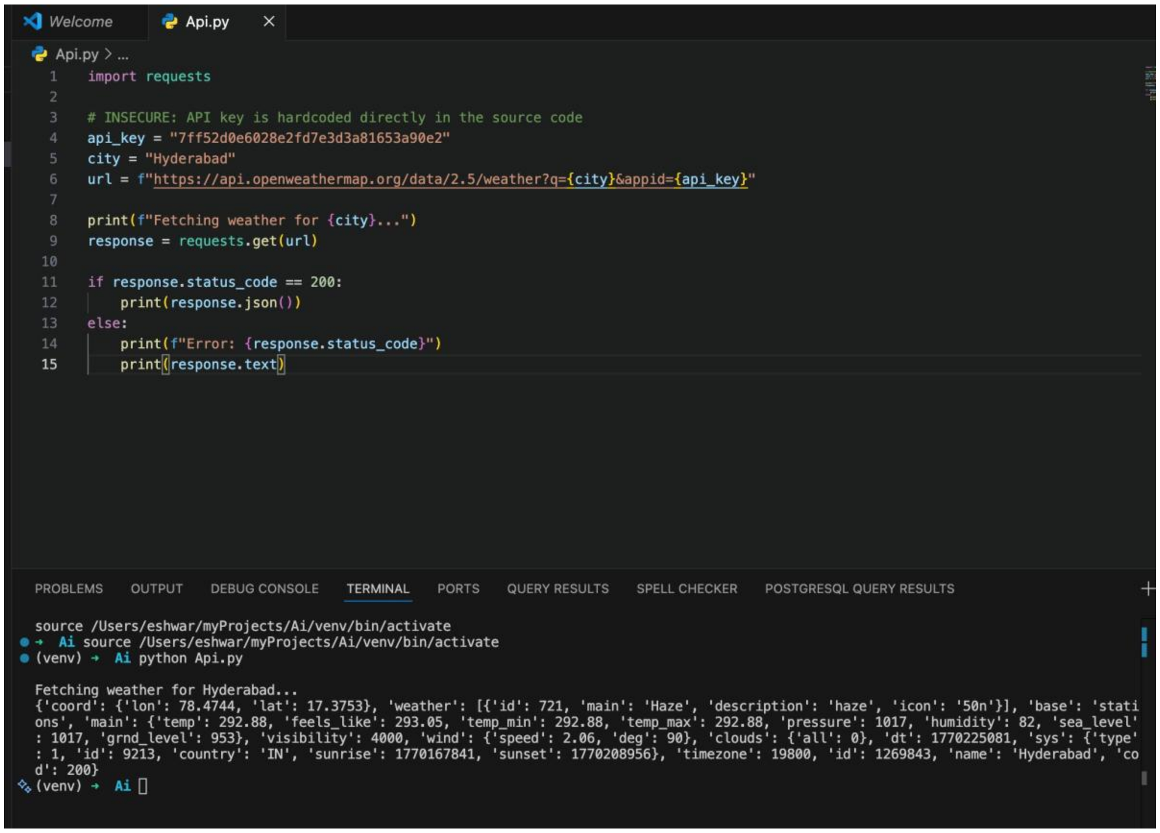
Objective:

To generate a Python program that fetches weather data securely without exposing API keys.

Risk Analysis:

AI-generated code may hardcode API keys directlyin the program. This is unsafe and may lead to security breaches.

Conclusion:
Using environment variables protects sensitive credentials and follows ethical security practices.



```python
import requests

# INSECURE: API key is hardcoded directly in the source code
api_key = "7ff52d0e6028e2fd7e3d3a81653a90e2"
city = "Hyderabad"
url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}"

print(f"Fetching weather for {city}...")
response = requests.get(url)

if response.status_code == 200:
    print(response.json())
else:
    print(f"Error: {response.status_code}")
    print(response.text)
```

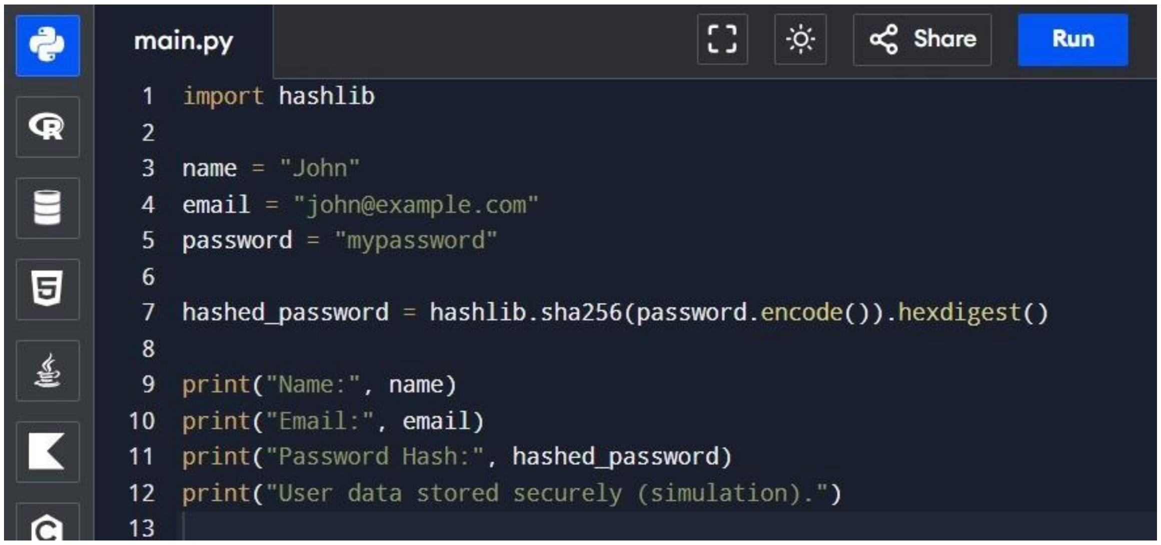
```
source /Users/eshwar/myProjects/Ai/venv/bin/activate
● → Ai source /Users/eshwar/myProjects/Ai/venv/bin/activate
● (venv) → Ai python Api.py

Fetching weather for Hyderabad...
{'coord': {'lon': 78.4744, 'lat': 17.3753}, 'weather': [{'id': 721, 'main': 'Haze', 'description': 'haze', 'icon': '50n'}], 'base': 'stati
ons', 'main': {'temp': 292.88, 'feels_like': 293.05, 'temp_min': 292.88, 'temp_max': 292.88, 'pressure': 1017, 'humidity': 82, 'sea_level'
: 1017, 'grnd_level': 953}, 'visibility': 4000, 'wind': {'speed': 2.06, 'deg': 90}, 'clouds': {'all': 0}, 'dt': 1770225081, 'sys': {'type'
: 1, 'id': 9213, 'country': 'IN', 'sunrise': 1770167841, 'sunset': 1770208956}, 'timezone': 19800, 'id': 1269843, 'name': 'Hyderabad', 'co
d': 200}
(venv) → Ai
```

Task Descrip2on #2: Privacy & Security in File Handling

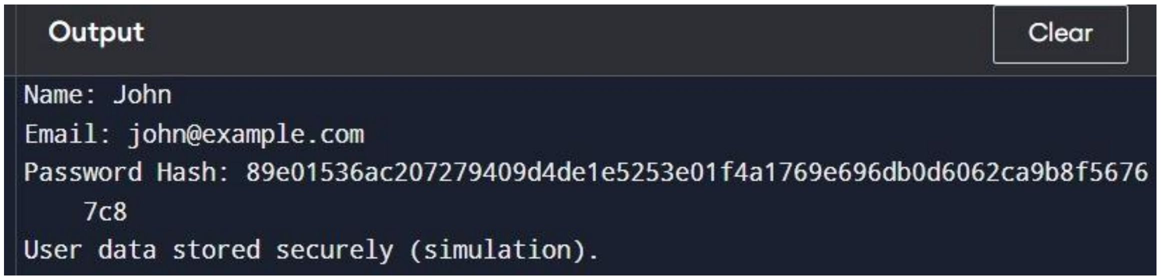
Objective:
To analyzehow AI-generated code stores user data and improve its security.

Privacy Risk IdentiUied:
Storing passwords in plain text can compromise user accounts.

Conclusion:
Hashing passwords ensures data privacy and security.



```python
import hashlib

name = "John"
email = "john@example.com"
password = "mypassword"

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

print("Name:", name)
print("Email:", email)
print("Password Hash:", hashed_password)
print("User data stored securely (simulation).")
```



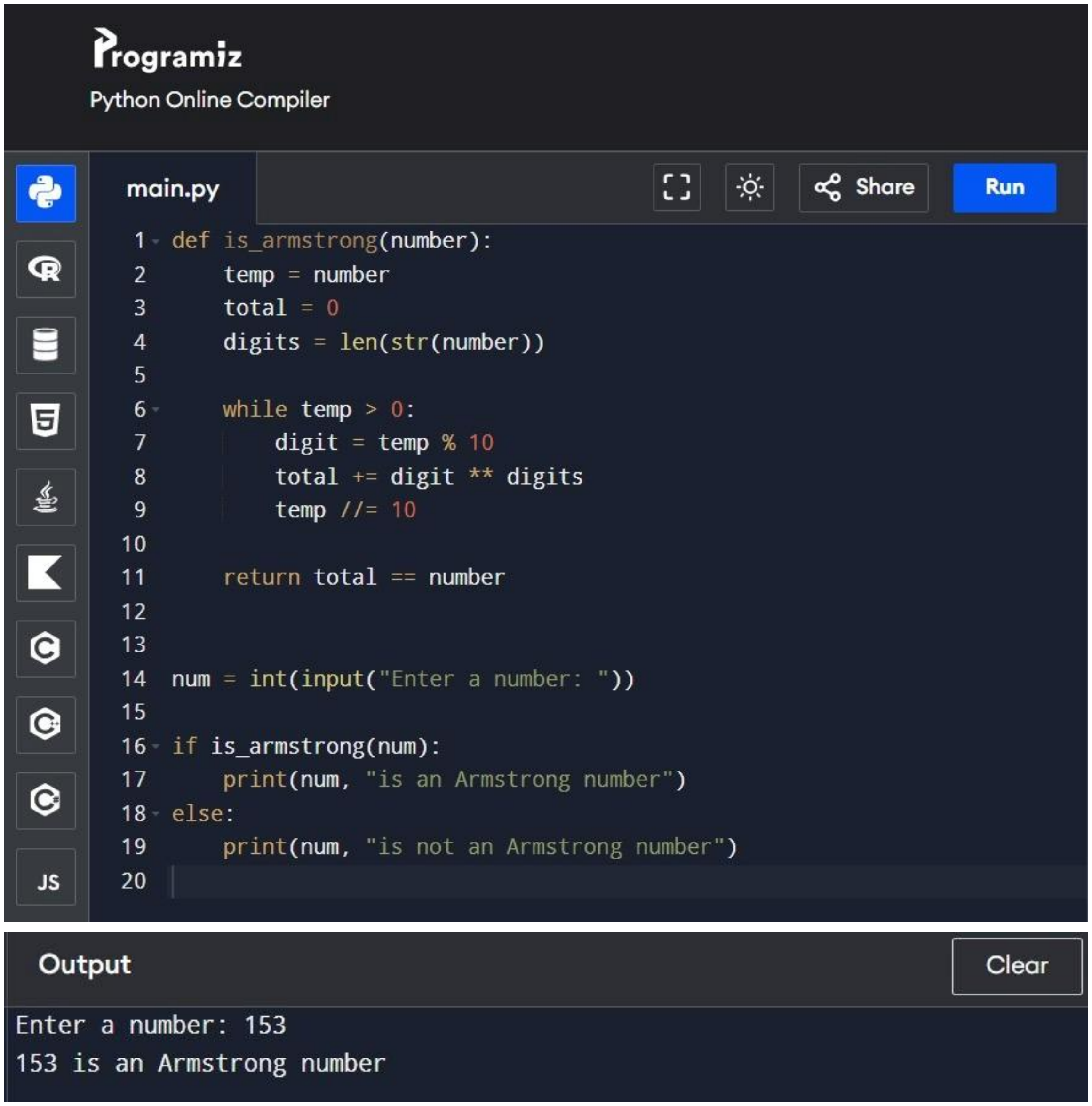
```
Name: John
Email: john@example.com
Password Hash: 89e01536ac207279409d4de1e5253e01f4a1769e696db0d6062ca9b8f5676
    7c8
User data stored securely (simulation).
```

Task Descrip2on #3: Transparency in Algorithm Design

Objective:
To create an Armstrong number checking program with clear explanation.

Explanation:
The program checks whether the sum of digits raised to the power of total digits equals the original number.

Conclusion:
The logic is simple, transparent, and easy to understand.



```python
def is_armstrong(number):
    temp = number
    total = 0
    digits = len(str(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(num, "is an Armstrong number")
else:
    print(num, "is not an Armstrong number")
```

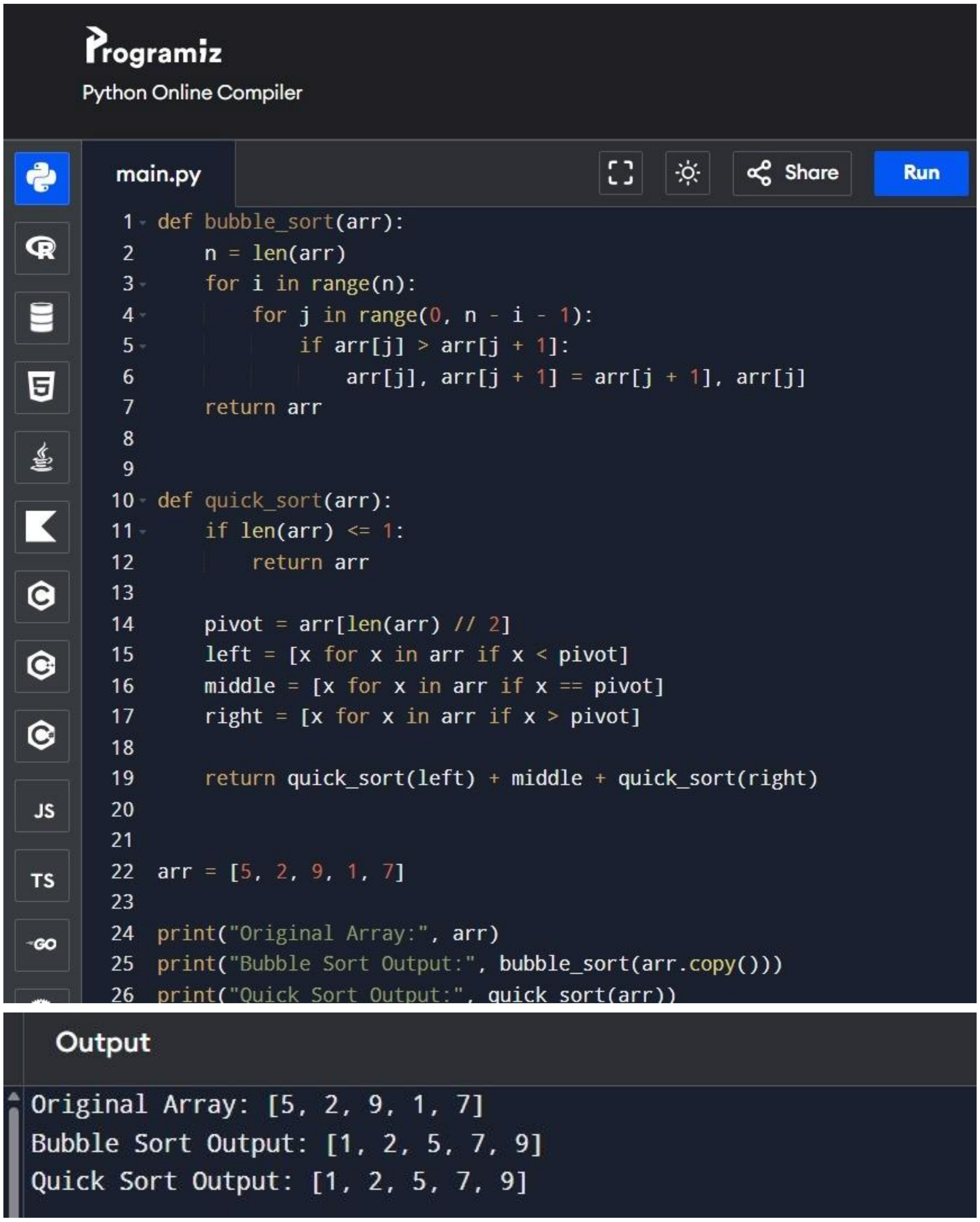
```
Enter a number: 153
153 is an Armstrong number
```

Task Descrip2on #4: Transparency in Algorithm Comparison

Objective:
To implement and compare Bubble Sort and Quick Sort algorithms.

Explanation:
Bubble Sort is easy to understand but slow, whereas Quick Sort is faster and efUicient for large datasets.

Conclusion:
Choosing the right algorithm improves performance and ethical decision-making.



```python
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)


arr = [5, 2, 9, 1, 7]

print("Original Array:", arr)
print("Bubble Sort Output:", bubble_sort(arr.copy()))
print("Quick Sort Output:", quick_sort(arr))
```

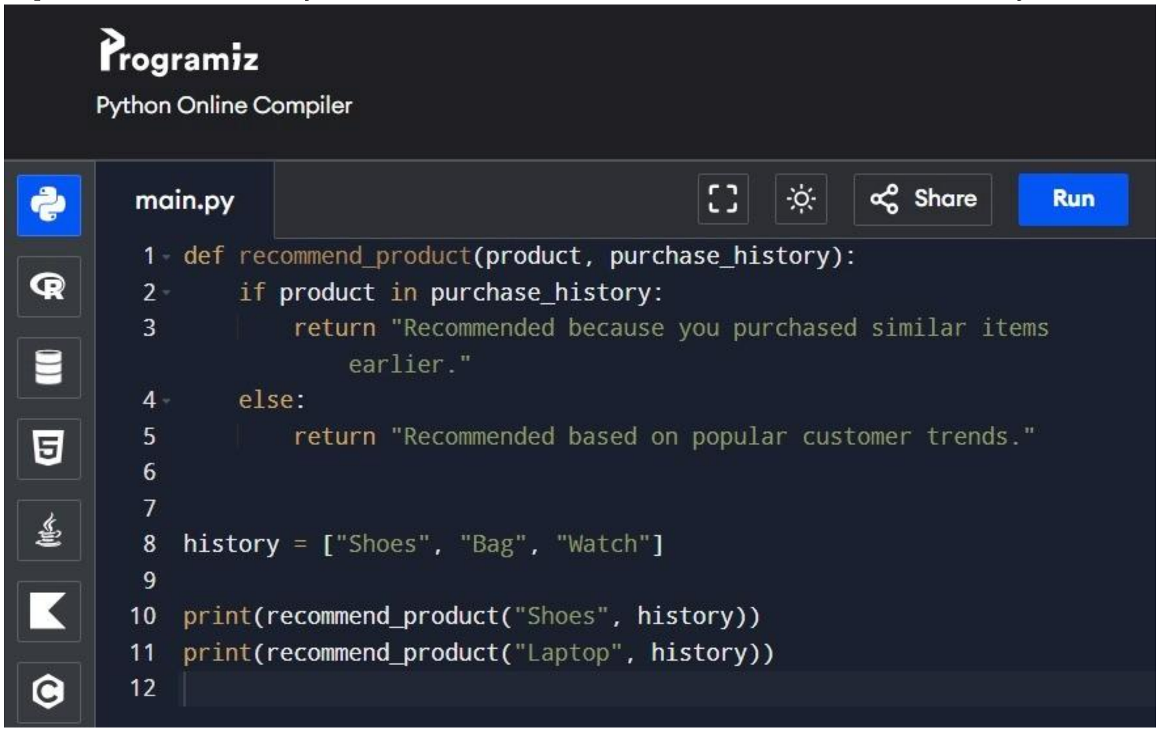
```
Original Array: [5, 2, 9, 1, 7]
Bubble Sort Output: [1, 2, 5, 7, 9]
Quick Sort Output: [1, 2, 5, 7, 9]
```

Task Descrip2on #5: Transparency in AI Recommenda2ons

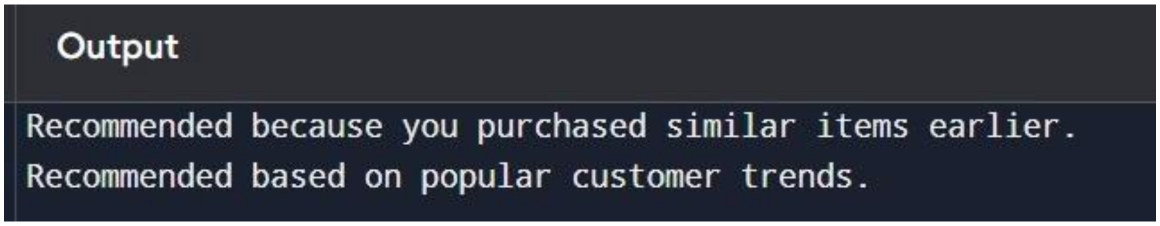
Objective:
To build a recommendation system that explains why items are suggested.

Explanation:
Providing reasons for recommendations improves transparency and trust.

Conclusion:
Explainable AI systems are more ethical and user-friendly.



```python
def recommend_product(product, purchase_history):
    if product in purchase_history:
        return "Recommended because you purchased similar items
            earlier."
    else:
        return "Recommended based on popular customer trends."


history = ["Shoes", "Bag", "Watch"]

print(recommend_product("Shoes", history))
print(recommend_product("Laptop", history))
```



```
Recommended because you purchased similar items earlier.
Recommended based on popular customer trends.
```