AI Assisted Coding

Assignment-5.3
Hall no : 2303A51835
Name:T.Shivalingeshwar

 Batch 27

Task 1: Privacy & Data Security in AI-Generated Code

◆ **Scenario (in simple words)**

AI may generate **unsafe login systems**. Your job is to:

1. Show **unsafe code**

2. Point out **security problems**

3. Fix it and **explain improvements**

AI-Generated Insecure Login Code

username = "admin"

password = "1234"

u = input("Enter username: ")

p = input("Enter password: ")

if u == username and p == password:

   print("Login successful")

else:

   print("Invalid credentials")

output:

```
•••    Enter username: admin
       Enter password: 1234
       Login successful
```

Security Issues Identified

- Credentials are **hardcoded**

- Password stored and compared in **plain text**

- No input validation

- Easy to hack

Revised Secure Login Code

```
import hashlib

stored_username = "admin"

stored_password_hash = hashlib.sha256("1234".encode()).hexdigest()

u = input("Enter username: ")

p = input("Enter password: ")

p_hash = hashlib.sha256(p.encode()).hexdigest()

if u == stored_username and p_hash == stored_password_hash:

    print("Login successful")
else:

    print("Invalid credentials")
```

output:

```
•••    Enter username: admin
       Enter password: 1234
       Login successful
```

📝 **Explanation of Improvements**

- Removed plain-text password storage

- Used **hashing (SHA-256)** for security

- Reduced risk of credential leakage

- Improved responsible coding practice

-

Task 2: Bias Detection in AI-Generated Decision Systems

◆ **Scenario**

AI may **favor or discriminate** based on name or gender.

AI-Generated Loan Approval Code:

```
def loan_approval(name, gender, income):

    if gender == "male" and income > 30000:

        return "Loan Approved"

    elif gender == "female" and income > 50000:

        return "Loan Approved"

    else:

        return "Loan Rejected"
```

Bias Identified

- Different income rules for males vs females

- Gender is **irrelevant** for loan approval

- This is **unfair and discriminatory**

Bias-Free Version

```
def loan_approval(income, credit_score):

    if income > 40000 and credit_score > 700:

        return "Loan Approved"

    else:

        return "Loan Rejected"
```

📝 **Fairness Discussion**

- Removed gender and name

- Decisions based only on **financial factors**

- Ensures equal treatment

Task 3: Transparency & Explainability

(Recursive Binary Search)

What they want

- Clear code

- Clear comments

- Easy explanation

 Recursive Binary Search Code

```
def binary_search(arr, low, high, target):

   if low > high:

      return -1

   mid = (low + high) // 2

   if arr[mid] == target:

      return mid

   elif arr[mid] > target:

      return binary_search(arr, low, mid - 1, target)

   else:

      return binary_search(arr, mid + 1, high, target)

arr = [2, 4, 6, 8, 10, 12]

target = 8

result = binary_search(arr, 0, len(arr)-1, target)

print("Element found at index:", result)
```

output:

```
···    Element found at index: 3
```

Student Assessment

- Base case clearly defined

- Recursive calls explained

- Code is beginner-friendly

- Transparent and readable

Task 4: Ethical Evaluation of AI-Based Scoring Systems

AI-Generated Scoring Code

```python
def applicant_score(name, gender, skills, experience):

    score = skills * 2 + experience * 3

    if gender == "male":

        score += 5

    return score
```

Ethical Issues

- Gender affects score

- Not job-related

- Biased hiring logic

Ethical Version

```python
def applicant_score(skills, experience, education_level):

    score = skills * 2 + experience * 3 + education_level * 2

    return score
```

Ethical Analysis

- Removed gender & name

- Focused on professional merit

- Fair and objective scoring

Task 5: Inclusiveness & Ethical Variable Design

Non-Inclusive Code

```
def employee_details(name, gender):

    if gender == "male":

        print(name, "is a strong leader")

    else:

        print(name, "is supportive")
```

Problems

- Gender stereotypes

- Non-inclusive logic

- Assumptions based on identity

Inclusive Version

```
def employee_details(name, role):

    print(name, "is performing well in the role of", role)
```

Inclusiveness Improvement

- Removed gender-based assumptions

- Used neutral variables

- Respectful and fair design