V.HRK SRI HARSHA VIGNESH

BATCH-42

2303A52304

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

Prompt:

Generate a simple login system which can register and login users details manually and runs until we quit

Code:

```
users = {}
def register(username, password):
    if username in users:
        return "Username already exists."
    users[username] = password
    return "User registered successfully."
def login(username, password):
    if username not in users:
        return "Username does not exist."
    if users[username] != password:
        return "Incorrect password."
    return "Login successful."
```

```
while True:
    action = input("Do you want to (register/login/quit)? ").strip().lower()
    if action == "register":
        username = input("Enter username: ")
        password = input("Enter password: ")
        print(register(username, password))
    elif action == "login":
        username = input("Enter username: ")
        password = input("Enter password: ")
        print(login(username, password))
    elif action == "quit":
        print("Exiting the system. Goodbye!")
        break
    else:
        print("Invalid action. Please choose register, login, or quit.")
```

Output:

```
Do you want to (register/login/quit)? register
Enter username: nandu
Enter password: 123
User registered successfully.
Do you want to (register/login/quit)? login
Enter username: nandu
Enter password: 123
Login successful.
Do you want to (register/login/quit)? quit
Exiting the system. Goodbye!
```

Justification:

By the above code, we have created login page which also includes register page in it. By this code we are checking whether the details are matching accurately or not which is crucial to be done.

Task 2: Bias Detection in AI-Generated Decision Systems

Prompt:

Create a loan applroval system based on user input

Code:

```
def loan_approval_system(age, income, credit_score, loan_amount):
    if age < 18:
        return "Loan Denied: Applicant must be at least 18 years old."
    if income < 20000:
        return "Loan Denied: Income must be at least $20,000."
    if credit_score < 600:
        return "Loan Denied: Credit score must be at least 600."
    if loan_amount > income * 5:
        return "Loan Denied: Loan amount exceeds allowable limit."
    return "Loan Approved!"
name = input("Enter your name: ")
age = int(input("Enter your age: "))
income = float(input("Enter your annual income: "))
credit_score = int(input("Enter your credit score: "))
loan_amount = float(input("Enter desired loan amount: "))
result = loan_approval_system(age, income, credit_score, loan_amount)
print(result)
```

Output:

```
Enter your name: vignanand
Enter your age: 18
Enter your annual income: 250000
Enter your credit score: 256
Enter desired loan amount: 30000
Loan Denied: Credit score must be at least 600.
PS C:\Users\VIGNANAND RAO\OneDrive\Desktop\AI_coding>
```

Justification:

By this code we can get to know whether the loan is approved or denied, we can clearly see that loan approval decision is not at all depending on name, caste, gender and any other irrelavent aspects which shows the transparency and fairness of the code generated by A.I.

Task 3: Transparency and Explainability in AI-Generated Code (Recursive Binary Search)

Prompt:

Binary search on a sorted list using recursion with comment

Code:

```
def binary_search(arr, target, low, high):
    if low > high:
        return -1  # Target not found
    mid = (low + high) // 2  # Find the middle index
    if arr[mid] == target:
        return mid  # Target found at mid index
    elif arr[mid] < target:
        return binary_search(arr, target, mid + 1, high)  # Search in the right half
    else:
        return binary_search(arr, target, low, mid - 1)  # Search in the left half
print("Enter sorted numbers separated by spaces:")
sorted_list = list(map(int, input().split()))
target = int(input("Enter the number to search for: "))
result = binary_search(sorted_list, target, 0, len(sorted_list) - 1)
if result != -1:
    print(f"Element found at index {result}.")
else:
    print("Element not found in the list.")
```

Output:

```
Enter sorted numbers separated by spaces:
2 3 4 5 6 7 8 9 20 36 254
Enter the number to search for: 32
Element not found in the list.
PS C:\Users\VIGNANAND RAO\OneDrive\Desktop\AI_coding>
```

Justification:

In the above generated code we are searching the number in a sorted array using binary search on a sorted list. We are performing by recursion which helps in reusing of code and also having step by step explanation(comments in code) which plays a key role in easy understanding of the code.

Task 4: Ethical Evaluation of AI-Based Scoring Systems

Prompt:

Job application scoring system based on users input

Code:

```
def job_application_score(experience_years, skills_count, education_level):
    score = 0
    score += experience_years * 5
    score += skills_count * 3
    education_scores = {
        "high_school": 5,
        "bachelor": 10,
        "master": 15,
        "phd": 20
    }
    score += education_scores.get(education_level.lower(), 0)
    return score
name = input("Enter your name: ")
experience_years = int(input("Enter years of experience: "))
skills_count = int(input("Enter number of relevant skills: "))
education_level = input("Enter highest education level (high_school/bachelor/master/phd): ")
total_score = job_application_score(experience_years, skills_count, education_level)
print(f"Total Application Score for {name}: {total_score}")
```

Output:

```
Enter your name: Vignanand
Enter years of experience: 5
Enter number of relevant skills: 5
Enter highest education level (high_school/bachelor/master/phd): bachlor
Total Application Score for Vignanand: 40
PS C:\Users\VIGNANAND RAO\OneDrive\Desktop\AI_coding>
```

Justification:

In the above code we are able to find or calculate the application score value which helps in jobs selection process. We can clearly see that only the required things or parameters like experience, skills and education is only taken into consideration and unimportant parameters like name, age, gender and other is not effecting which shows the fairness of the code generated by A.I. which is most important.

Task 5: Inclusiveness and Ethical Variable Design

Prompt:

Create a Python employee management system that stores employee details including name, gender (male/female), job title, and salary. Add a function that gives different benefits: maternity leave for females and paternity leave for males. Use Mr./Mrs. titles based on gender when displaying employee information. user input should be taken for adding employees and displaying their details along with the benefits they are entitled to.

Code:

```
employees = []
def add_employee(name, gender, job_title, salary):
    employees.append({
        "name": name,
        "gender": gender,
        "job_title": job_title,
        "salary": salary
    })
def display_employees():
    for emp in employees:
        title = "Mr." if emp["gender"].lower() == "male" else "Mrs."
        benefits = "Paternity Leave" if emp["gender "].lower() == "male" else "Maternity Leave"
        print(f"{title} {emp['name']}, Job Title: {emp['job_title']}, Salary: ${emp['salary']}, Benefits: {benefits}")
```

```
while True:
    action = input("Do you want to (add/display/quit)? ").strip().lower()
    if action == "add":
        name = input("Enter employee name: ")
        gender = input("Enter employee gender (male / female): ")
        job_title = input("Enter job title: ")
        salary = float(input("Enter salary: "))
        add_employee(name, gender, job_title, salary)
    elif action == "display":
        display_employees()
    elif action == "quit":
        print("Exiting the system. Goodbye!")
        break
```

Output:

```
Do you want to (add/display/quit)? add
Enter employee name: vignanand
Enter employee gender (male / female): male
Enter job title: web dev.
Enter salary: 5000000
Do you want to (add/display/quit)? quit
Exiting the system. Goodbye!
PS C:\Users\VIGNANAND RAO\OneDrive\Desktop\AI_coding>
```

Justification:

In the above code, the application score is calculated to support the job selection process. The model considers only relevant parameters such as experience, skills, and education, which directly reflect a candidate's suitability for the role. Non-essential and potentially biased attributes like name, age, gender, or personal identity are intentionally excluded from the logic. This ensures that the decision-making process remains fair, unbiased, and inclusive. Such an approach demonstrates ethical AI design by focusing on merit rather than personal characteristics.