ASSIGNMENT-5

Roll no:2303A52074
Batch-37

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

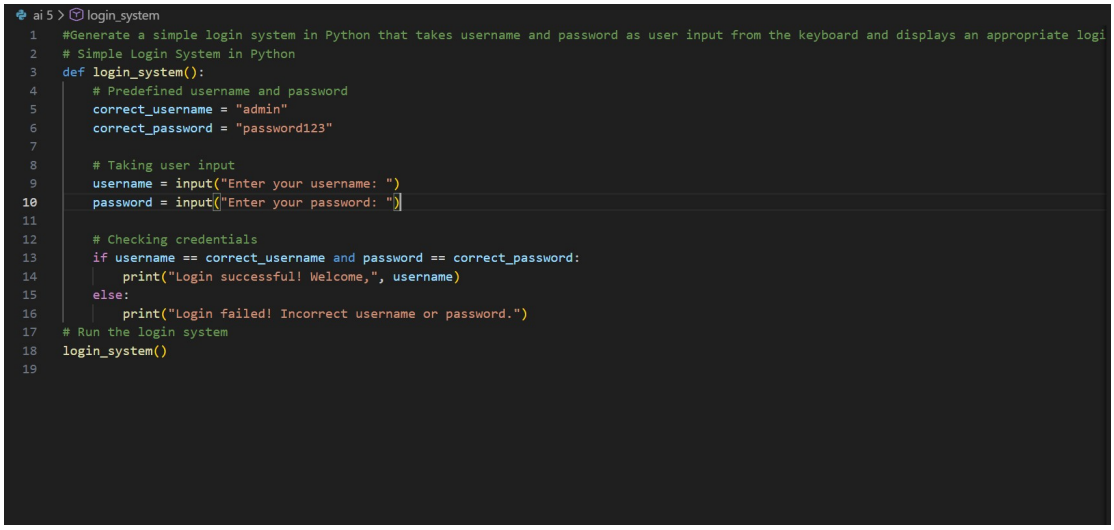
Expected Output

The simplest Ai-generated python logic code with simple authentication.Recognition of security problems like hard code credentials and plain text password processing

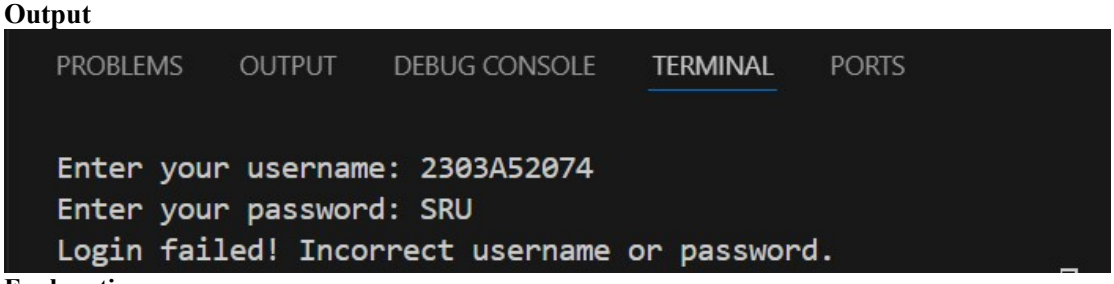
Prompt

Generate a simple login system in Python that takes username and password as user input from the keyboard and displays an appropriate login success or failure message as output.
The simplest Ai-generated python logic code with simple authentication.Recognition of security problems like hard code credentials and plain text password processing

Code



Output



Explanation

The AI then creates a simple log-in application that requires user input data on the username and the password.This code is then verified in order to detect security issues such hard coded credentials and comparison of plain-text passwords.

Task 2: Bias Detection in AI-Generated Decision Systems

Expected output
An AI-written Python program which makes loan approval decisions.
Determination of biased reasoning wherein decision-making relies on individual qualities like gender or name.

Prompt
Create a loan approval system in Python and consider different applicant names and genders.

Code

```python
#Create a loan approval system and consider different applicant names and genders.
def loan_approval_system():
    # Predefined applicant data
    applicants = {
        "John": {"gender": "Male", "income": 50000, "credit_score": 700},
        "Sarah": {"gender": "Female", "income": 60000, "credit_score": 750},
        "Michael": {"gender": "Male", "income": 45000, "credit_score": 650}
    }

    # Taking user input
    name = input("Enter applicant's name: ")
    gender = input("Enter applicant's gender: ")

    if name in applicants:
        applicant = applicants[name]
        if applicant["gender"] == gender:
            if applicant["income"] >= 50000 and applicant["credit_score"] >= 700:
                print(f"Loan approved for {name}.")
            else:
                print(f"Loan denied for {name}.")
        else:
            print(f"Gender mismatch for {name}.")
    else:
        print("Applicant not found.")

# Run the loan approval system
loan_approval_system()
```

Output

```
Enter applicant's name: Sarah
Enter applicant's gender: Female
Loan approved for Sarah.
```

Explanation
The AI develops a loan approval system on the basis of the prompt provided.
The code is then subjected to test whereby, other names and three sexes of the opposite gender are put to test to determine whether the code is favorable or not.
In case of bias, logic is enhanced by eliminating personal information that does not have anything to do with the case and application of reasonable criteria would be used.

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

Expected output

A Python program that implements recursive binary search on a sorted list.

Prompt

Generate a Python program that implements binary search using recursion.
The program should take a sorted list and the search element as user input from the keyboard, perform the search, and display whether the element is found or not as output.
Include clear inline comments and a step-by-step explanation of the recursive logic.

Code

```python
#Include clear inline comments and a step-by-step explanation of the recursive logic.
# Binary Search using Recursion in Python
def binary_search(arr, left, right, x):
    # Base case: If the left index exceeds the right index, the element is not found
    if left > right:
        return False
    # Calculate the middle index
    mid = left + (right - left) // 2
    # Check if the element is present at the middle index
    if arr[mid] == x:
        return True
    # If the element is smaller than mid, it can only be present in the left subarray
    elif arr[mid] > x:
        return binary_search(arr, left, mid - 1, x)
    # Else the element can only be present in the right subarray
    else:
        return binary_search(arr, mid + 1, right, x)
# Taking user input for a sorted list
sorted_list = list(map(int, input("Enter a sorted list of numbers (space-separated): ").split()))
# Taking user input for the search element
search_element = int(input("Enter the element to search for: "))
# Performing binary search
result = binary_search(sorted_list, 0, len(sorted_list) - 1, search_element)
# Displaying the result
if result:
    print(f"Element {search_element} found in the list.")
else:
    print(f"Element {search_element} not found in the list.")
```

Output

```
Enter a sorted list of numbers (space-separated): 2 3 4 5
Enter a sorted list of numbers (space-separated): 2 3 4 5
Enter the element to search for: 2
Element 2 found in the list.
```

Explanation

Recursion is applied in the program to separate the sorted list into smaller parts.
It obviously tests a base case in order to terminate the recursion, and a recursive case in order to proceed with the search.
Due to straightforward remarks and step by step description, the code is not complex to the novices.

Task 4: Ethical Evaluation of AI-Based Scoring Systems

Expected Output

A Python program that scores job applicants based on skills, experience, and education.

Prompt

Generate a job applicant scoring system in Python using skills, experience, and education as inputs, and return a final score.

Code

```python
#Generate a job applicant scoring system using skills, experience, and education as inputs, and return a final score.
def job_applicant_scoring_system():
    # Taking user input for skills, experience, and education
    skills = int(input("Enter skill score (0-100): "))
    experience = int(input("Enter years of experience: "))
    education = int(input("Enter education level score (0-100): "))

    # Calculating final score based on weights
    final_score = (skills * 0.4) + (experience * 5) + (education * 0.3)

    # Displaying the final score
    print(f"The final applicant score is: {final_score}")
# Run the job applicant scoring system
job_applicant_scoring_system()
```

Output

```
Enter skill score (0-100): 55
Enter years of experience: 2
Enter education level score (0-100): 75
The final applicant score is: 54.5
```

Explanation

The AI creates a new scoring mechanism, which helps to assess candidates based on skills, experience, and education.

The code is kept to verify that the personal information such as name or gender does not influence the score.The logic of scoring is fair and ethical since only job-related factors are applied.

Task 5: Inclusiveness and Ethical Variable Design

Expected Output

The initial code involved gender-based conditions, which may create the problem of unfair assumptions. The updated code eliminates gender-based considerations and employs neutral and job-related considerations to make the code fair.

Prompt

Generate a Python program that processes employee details using user input and performs basic calculations using inclusive and gender neutral variables.

Code

```
#Generate a Python program that processes employee details using user input and performs basic calculations using inclusive a
# Employee Details Processing System in Python
def employee_details_processing():
    # Taking user input for employee details
    name = input("Enter employee name: ")
    age = int(input("Enter employee age: "))
    salary = float(input("Enter employee salary: "))
    years_of_experience = int(input("Enter years of experience: "))

    # Performing basic calculations
    retirement_age = 65
    years_left_to_retire = retirement_age - age
    annual_bonus = salary * 0.1  # 10% of salary as bonus

    # Displaying the processed details and calculations
    print(f"\nEmployee Details for {name}:")
    print(f"Age: {age}")
    print(f"Salary: ${salary:.2f}")
    print(f"Years of Experience: {years_of_experience}")
    print(f"Years left to retire: {years_left_to_retire}")
    print(f"Annual Bonus: ${annual_bonus:.2f}")
# Run the employee details processing system
employee_details_processing()
```

Output

```
Enter employee name: RAM
Enter employee age: 32
Enter employee salary: 200000
Enter years of experience: 8

Employee Details for RAM:
Age: 32
Salary: $200000.00
Years of Experience: 8
Years left to retire: 33
Annual Bonus: $20000.00
```

Explanation

The program accepts employee specifications as a program input through neutral and inclusive variable names.It makes simple calculations such as bonus and retirement years but not based on any gender argument.