

AI Assisted Coding

Assignment 5.3

Name: P. Vineeth Kumar

Hall ticket no: 2303A51256

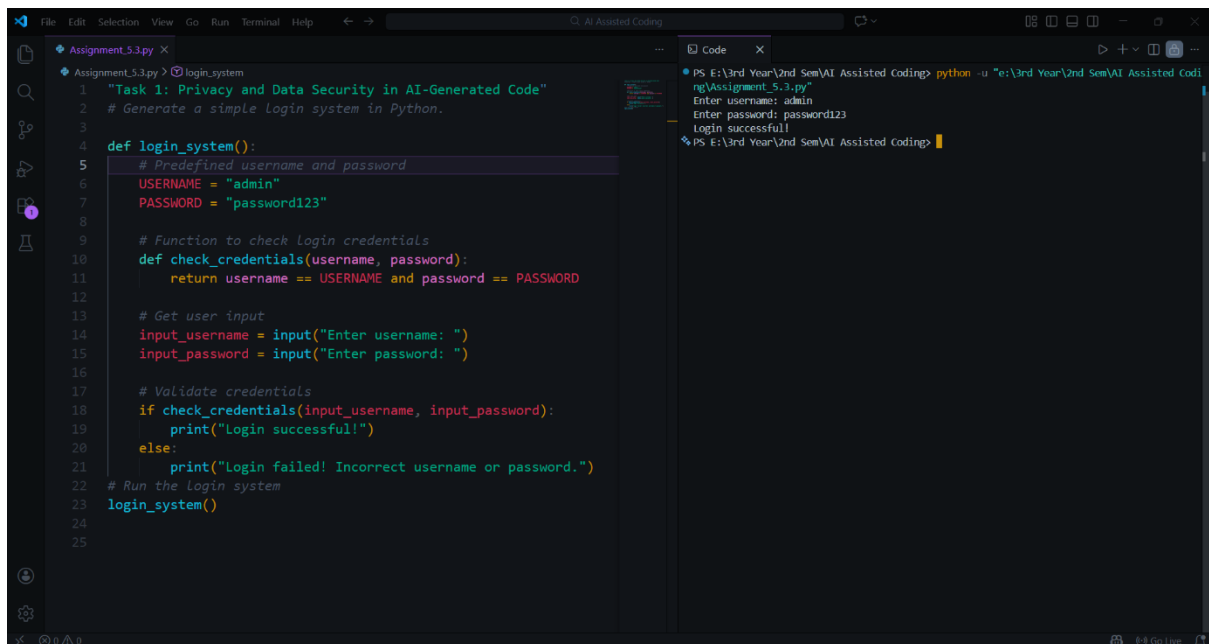
Batch no: 19

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

Simple Prompt:

Generate a simple login system in Python.

Code & Output:



```
File Edit Selection View Go Run Terminal Help ← → AI Assisted Coding
Assignment_5.3.py X
  Assignment_5.3.py > login_system
1 "Task 1: Privacy and Data Security in AI-Generated Code"
2 # Generate a simple Login system in Python.
3
4 def login_system():
5     # Predefined username and password
6     USERNAME = "admin"
7     PASSWORD = "password123"
8
9     # Function to check Login credentials
10    def check_credentials(username, password):
11        return username == USERNAME and password == PASSWORD
12
13    # Get user input
14    input_username = input("Enter username: ")
15    input_password = input("Enter password: ")
16
17    # Validate credentials
18    if check_credentials(input_username, input_password):
19        print("Login successful!")
20    else:
21        print("Login failed! Incorrect username or password.")
22
23    # Run the Login system
24    login_system()
25
Code
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment_5.3.py"
Enter username: admin
Enter password: password123
Login successful!
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

Explanation:

In this task, the AI generated a basic login system based on the given prompt. However, the code contains multiple security risks. The username and password are hardcoded directly in the source code, making them easily accessible to anyone who reads the program. Additionally, the password is stored and compared in plain text, which is unsafe and can lead to credential theft. The code also lacks input validation and does not follow secure authentication practices. This shows that AI-generated code must be carefully reviewed before use.

Revised Prompt:

Generate a secure login system in Python with hashed passwords. While Hardcoding use hashed values.

Code & Output:

The screenshot displays a Windows desktop environment. The primary focus is the Visual Studio Code (VS Code) editor, which is open to a file named `Assignment_5.3.py`. The script is a Python program designed to create a secure login system. It includes comments explaining its purpose: "Generate a secure login system in Python with hashed passwords. While Hardcoding use hashed values." The code defines a `login_system()` function that uses the `hashlib` module to hash usernames and passwords. It prompts the user to enter a username and password, checks if they match the stored hashes, and prints a success or failure message. The script is executed using the command `python .\Assignment_5.3.py`.

To the right of the editor, a terminal window is open, showing the execution of the script. It displays the prompt `PS E:\3rd Year\2nd Sem\VAI Assisted Coding>` followed by the command `python -u "e:\3rd Year\2nd Sem\VAI Assisted Coding\Assig\rmnt_5.3.py"`. The terminal output shows the program's execution, including prompts for "Enter username:" and "Enter password:", and the resulting "Login failed! Incorrect username or password." message.

The desktop background is a solid blue color. In the bottom right corner, there is a small icon for "Go Live" with a green play button.

Explanation:

The revised prompt guided the AI to generate a more secure version of the login system. Instead of storing passwords in plain text, the password is hashed using SHA-256 before comparison. Input validation is also added by removing unnecessary spaces. These improvements reduce security risks and demonstrate responsible handling of sensitive data. This task highlights that humans must refine prompts and improve AI-generated code to ensure privacy and security.

Task 2: Bias Detection in AI-Generated Decision Systems

Prompt:

Create a loan approval system in Python.

Code & Output:

The screenshot shows a VS Code editor with a file named `Assignment_5.3.py`. The code is a Python script for a loan approval system. It includes a function `approve_loan` that takes `income`, `credit_score`, and `gender` as arguments. The function uses conditional logic to approve or deny a loan based on these factors. The `gender` parameter is used in a way that introduces bias, as it requires female applicants to have a higher income than male applicants. The script also includes a `try` block to handle `ValueError` exceptions for invalid input.

```
53
54 "Task 2: Bias Detection in AI-Generated Decision Systems"
55 #Create a loan approval system in Python.
56 def loan_approval_system():
57     # Function to determine loan approval
58     def approve_loan(income, credit_score, gender):
59         if gender.lower() == "female":
60             return income >= 50000 and credit_score >= 600
61         else:
62             return income >= 60000 and credit_score >= 700
63
64     # Get user input
65     try:
66         income = float(input("Enter your annual income: "))
67         credit_score = int(input("Enter your credit score: "))
68         gender = input("Enter your gender (male/female): ") # This
69         # input is not used in decision-making
70     except ValueError:
71         print("Invalid input. Please enter numeric values.")
72         return
73
74     # Determine loan approval
75     if approve_loan(income, credit_score, gender):
76         print("Loan approved!")
77     else:
78         print("Loan denied. You do not meet the criteria.")
79
80 # Run the loan approval system
81 loan_approval_system()
```

The terminal output shows the execution of the script. It prompts the user for annual income, credit score, and gender. For a male applicant with an income of 75000 and a credit score of 850, the loan is approved. For a female applicant with an income of 55000 and a credit score of 650, the loan is denied.

```
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment_5.3.py"
Enter your annual income: 75000
Enter your credit score: 850
Enter your gender (male/female): male
Loan approved!
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assignment_5.3.py"
Enter your annual income: 55000
Enter your credit score: 650
Enter your gender (male/female): female
Loan denied!
```

Explanation:

The AI-generated loan approval system uses gender as a decision factor. Female applicants are required to have a higher income than male applicants, which introduces bias. Gender is an irrelevant personal attribute for loan approval. This demonstrates how AI-generated logic can unintentionally lead to unfair and discriminatory decisions.

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

Prompt:

Generate a recursive binary search program in Python with explanation.

Code & Output:

```
File Edit Selection View Go Run Terminal Help
Assignment_5.3.py X
Task 3: Transparency and Explainability in AI-Generated Code
(Recursive Binary Search)
# Generate a recursive binary search program in Python with
# explanation.
def binary_search(arr, target, low, high):
    if low > high:
        return -1 # Target not found

    mid = (low + high) // 2
    print(f"Checking middle index {mid}, value {arr[mid]}") #
    # Explanation step

    if arr[mid] == target:
        return mid # Target found
    elif arr[mid] < target:
        print(f"Target {target} is greater than {arr[mid]}, searching
        right half") # Explanation step
        return binary_search(arr, target, mid + 1, high)
    else:
        print(f"Target {target} is less than {arr[mid]}, searching
        left half") # Explanation step
        return binary_search(arr, target, low, mid - 1)

# Get user input
try:
    arr = list(map(int, input("Enter a sorted list of numbers
    (space-separated): ").split()))
    target = int(input("Enter the target number to search for: "))
    result = binary_search(arr, target, 0, len(arr) - 1)
    if result != -1:
        print(f"Target {target} found at index {result}.")
    else:
        print(f"Target {target} not found in the list.")
except ValueError:
    print("Invalid input. Please enter numeric values.")
110
```

```
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "E:
\3rd Year\2nd Sem\AI Assisted Coding\Assignment_5.3.py"
Enter a sorted list of numbers (space-separated): 1 2 4 5 8 9
Enter the target number to search for: 5
Checking middle index 2, value 4
Target 5 is greater than 4, searching right half
Checking middle index 4, value 8
Target 5 is less than 8, searching left half
Checking middle index 3, value 5
Target 5 found at index 3.
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

Explanation:

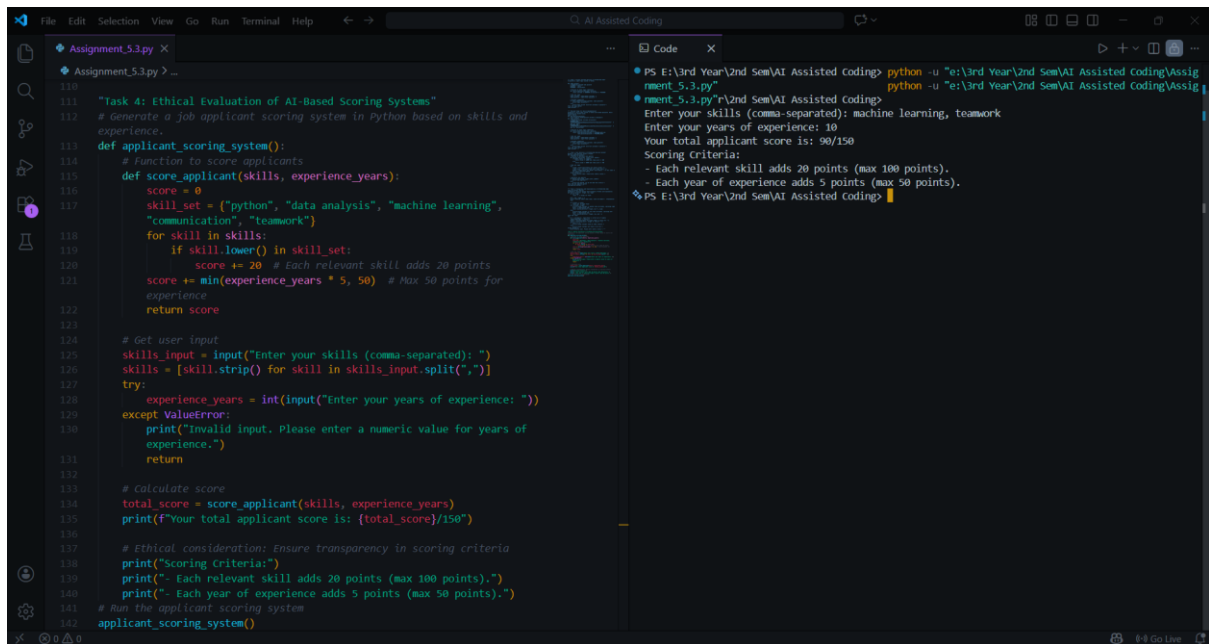
The recursive binary search works by dividing the sorted list into halves. The base case occurs when the search range becomes invalid, meaning the element is not present. In the recursive case, the function compares the target with the middle element and searches the appropriate half. The code is clear, well-structured, and easy for beginners to understand.

Task 4: Ethical Evaluation of AI-Based Scoring Systems

Prompt:

Generate a job applicant scoring system in Python based on skills and experience.

Code & Output:



```
110
111 "Task 4: Ethical Evaluation of AI-Based Scoring Systems"
112 # Generate a job applicant scoring system in Python based on skills and
    experience.
113 def applicant_scoring_system():
114     # Function to score applicants
115     def score_applicant(skills, experience_years):
116         score = 0
117         skill_set = ("python", "data analysis", "machine learning",
118                     "communication", "teamwork")
119         for skill in skills:
120             if skill.lower() in skill_set:
121                 score += 20 # Each relevant skill adds 20 points
122             score += min(experience_years * 5, 50) # Max 50 points for
123                 experience
124             return score
125
126 # Get user input
127 skills_input = input("Enter your skills (comma-separated): ")
128 skills = [skill.strip() for skill in skills_input.split(",")]
129 try:
130     experience_years = int(input("Enter your years of experience: "))
131 except ValueError:
132     print("Invalid input. Please enter a numeric value for years of
133     experience.")
134     return
135
136 # Calculate score
137 total_score = score_applicant(skills, experience_years)
138 print(f"Your total applicant score is: {total_score}/150")
139
140 # Ethical consideration: Ensure transparency in scoring criteria
141 print("Scoring Criteria:")
142 print("- Each relevant skill adds 20 points (max 100 points).")
143 print("- Each year of experience adds 5 points (max 50 points).")
144
145 # Run the applicant scoring system
146 applicant_scoring_system()
```

```
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "E:\3rd Year\2nd Sem\AI Assisted Coding\Assign
ment_5.3.py"
python -u "E:\3rd Year\2nd Sem\AI Assisted Coding\Assign
ment_5.3.py"
Enter your skills (comma-separated): machine learning, teamwork
Enter your years of experience: 10
Your total applicant score is: 90/150
Scoring Criteria:
- Each relevant skill adds 20 points (max 100 points).
- Each year of experience adds 5 points (max 50 points).
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

Explanation:

The AI-generated scoring system adds extra points based on gender, which introduces bias. Gender has no relation to job performance, and using it in scoring logic can lead to unfair hiring decisions. This highlights the importance of ethically evaluating AI-generated systems used in recruitment.

Task 5: Inclusiveness and Ethical Variable Design

Simple Prompt:

Generate a Python code snippet that processes employee details for their uniform.

Code & Output:

The screenshot shows a VS Code editor with a file named `Assignment_5.3.py`. The code is a Python script that takes an employee's name and gender as input and assigns a uniform based on the gender. The code is as follows:

```
143
144 "Task 5: Inclusiveness and Ethical Variable Design"
145 # Generate a Python code snippet that processes employee details for their
    uniform.
146
147 def employee_uniform():
148     name = input("Enter Name: ")
149     gender = input("Enter Gender(male/female): ").strip().lower()
150     if gender == "male":
151         uniform = "Shirt and Pants"
152     else:
153         uniform = "Skirt"
154     print(f"Employee Name: {name}")
155     print(f"Assigned Uniform: {uniform}")
156 # Run the employee uniform assignment
157 employee_uniform()
```

The output of the script is shown in the terminal window:

```
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assign
ment_5.3.py"
Enter Name: Ravi
Enter Gender(male/female): male
Employee Name: Ravi
Assigned Uniform: Shirt and Pants
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assign
ment_5.3.py"
Enter Name: Ria
Enter Gender(male/female): female
Employee Name: Ria
Assigned Uniform: Skirt
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

Explanation:

This code uses gender-specific variables and makes assumptions about employee roles and preferences. It enforces stereotypes and limits inclusiveness by considering only binary gender options. Such logic is not suitable for ethical and inclusive software design.

Revised Prompt:

Generate inclusive and gender-neutral Python code to process employee details for their uniform.

Code & Output:

The screenshot shows a VS Code editor with a file named `Assignment_5.3.py`. The code is a Python script that takes an employee's name as input and assigns a uniform based on the name. The code is as follows:

```
157
158 "Revised Prompt for Neutral Employee Uniform Assignment"
159 # Generate inclusive and gender-neutral Python code to process
    employee details for their uniform.
160
161 def employee_uniform():
162     name = input("Enter Name: ")
163     uniform = "Formal Uniform"
164     print(f"Employee Name: {name}")
165     print(f"Assigned Uniform: {uniform}")
166 # Run the employee uniform assignment
167 employee_uniform()
168
```

The output of the script is shown in the terminal window:

```
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assign
ment_5.3.py"
Enter Name: Ravi
Employee Name: Ravi
Assigned Uniform: Formal Uniform
PS E:\3rd Year\2nd Sem\AI Assisted Coding> python -u "e:\3rd Year\2nd Sem\AI Assisted Coding\Assign
ment_5.3.py"
Enter Name: Ria
Employee Name: Ria
Assigned Uniform: Formal Uniform
PS E:\3rd Year\2nd Sem\AI Assisted Coding>
```

Explanation:

The revised code removes gender-based variables and uses neutral naming. This avoids assumptions and promotes inclusiveness and fairness. Ethical variable design ensures that software respects diversity and treats all users equally.