

School of Computer Science and Artificial Intelligence

Lab Assignment # 5.2

Program	: B. Tech (CSE)
Specialization	: --
Course Title	: AI Assisted coding
Course Code	:
Semester	II
Academic Session	: 2025-2026
Name of Student	: Akash Reddy
Enrollment No.	: 2403A51L30
Batch No.	: 51
Date	: 20-01-2026

Task Description-1

- **Secure API Usage:** Use an AI tool to generate a simple REST API for user registration. Review the generated code for improper handling of API keys, missing authentication, or lack of input validation.

Code for Improper handling of API keys, missing authentication, or lack of input validation.

The screenshot shows a code editor with a sidebar of icons. The top bar has tabs for 'lab_1.2.py', 'lab_4.2.py', and 'lab_5.2.py X'. The current file is 'lab_5.2.py' which contains the following code:

```
1  # Task Description-1
2  # [Secure API Usage: Use an AI tool to generate a simple REST API for user registration.
3  # Review the generated code for improper handling of API keys, missing authentication, or lack of input validation.
4  from flask import Flask, request, jsonify
5  import re
6  app = Flask(__name__)
7  API_KEY = "your_secure_api_key"
8  registered_users = []
9  def is_valid_email(email):
10     # Simple regex for email validation
11     return re.match(r"[\w\.-]+@[^\.\w\.-]+\.[\w\.-]+", email)
12 @app.route('/register', methods=['POST'])
13 def register():
14     api_key = request.headers.get('x-api-key')
15     if api_key != API_KEY:
16         return jsonify({"error": "Unauthorized"}), 401
17     data = request.json
18     username = data.get('username')
19     email = data.get('email')
20     password = data.get('password')
21     if not username or not email or not password:
22         return jsonify({"error": "Missing required fields"}), 400
23     if not is_valid_email(email):
```

The bottom part of the editor shows a terminal window with the following output:

```
/usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
(base) akash@AKASH-MacBook-Air ai_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
* Serving Flask app 'lab_5.2'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 397-141-148
127.0.0.1 - - [20/Jan/2026 22:28:24] "GET / HTTP/1.1" 404 -
127.0.0.1 - - [20/Jan/2026 22:28:24] "GET /favicon.ico HTTP/1.1" 404 -
```

Task Description-2

- **Fair Decision Logic:** Prompt an AI tool to generate a scholarship eligibility checker based on academic score, family income, and location. Examine whether the logic unfairly favors or disadvantages any group.

```

32 # Task Description-2
33 # Fair Decision Logic: Prompt an AI tool to generate a scholarship eligibility checker based on academic score, family income, and location.
34 # Examine whether the logic unfairly favors or disadvantages any group.
35 def is_eligible_for_scholarship(academic_score, family_income, location):
36     # Define thresholds
37     score_threshold = 85
38     income_threshold = 50000 # Annual family income
39     disadvantaged_locations = ['rural', 'underserved']
40
41     # Check academic score
42     if academic_score < score_threshold:
43         return False
44
45     # Check family income
46     if family_income > income_threshold:
47         return False
48
49     # Check location
50     if location not in disadvantaged_locations:
51         return False
52
53     return True
54
55 # Example usage
56 applicants = [
57     {"name": "Alice", "academic_score": 90, "family_income": 40000, "location": "rural"},
58     {"name": "Bob", "academic_score": 80, "family_income": 30000, "location": "urban"},
59     {"name": "Charlie", "academic_score": 88, "family_income": 60000, "location": "underserved"},
60     {"name": "David", "academic_score": 92, "family_income": 45000, "location": "urban"},
61 ]
62 for applicant in applicants:
63     eligible = is_eligible_for_scholarship(applicant['academic_score'], applicant['family_income'], applicant['location'])
64     print(f'{applicant["name"]} eligibility: {"Eligible" if eligible else "Not Eligible"}')

```

OUTPUT:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + ×
/usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
(base) akash@AKASHs-MacBook-Air ai_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
Alice eligibility: Eligible
Bob eligibility: Not Eligible
Charlie eligibility: Not Eligible
David eligibility: Not Eligible
(base) akash@AKASHs-MacBook-Air ai_assis %

```

Task Description-3

Explainability: Ask the AI tool to generate a function that checks whether a number is prime and include inline comments and a brief textual explanation of the algorithm.

```

lab_5.2.py > ...
65
66
67 # Task Description-3
68 # Explainability: Ask the AI tool to generate a function that checks whether a number is prime and
69 # include inline comments and a brief textual explanation of the algorithm.
70 def is_prime(n):
71     """Check if a number is prime."""
72     if n <= 1:
73         return False
74     for i in range(2, int(n**0.5) + 1):
75         if n % i == 0:
76             return False
77     return True
78
79 # Example usage
80 number = 29
81 if is_prime(number):
82     print(f'{number} is a prime number.')
83 else:
84     print(f'{number} is not a prime number.')
# Explanation:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + ×
/usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
(base) akash@AKASHs-MacBook-Air ai_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
29 is a prime number.
(base) akash@AKASHs-MacBook-Air ai_assis %

```

Task Description-4

- **Ethical Scoring System:** Generate an employee performance evaluation system using inputs such as project completion rate, teamwork score, and attendance. Analyze the scoring logic for any unethical weighting or hidden bias.

```
85
86 # Task Description-4
87 # ☐Ethical Scoring System: Generate an employee performance evaluation system using inputs such as project completion rate, teamwork score,
88 # and attendance. Analyze the scoring logic for any unethical weighting or hidden bias.
89
90 def evaluate_employee_performance(project_completion, teamwork, attendance):
91     # Define weights for each criterion
92     weights = {
93         "project_completion": 0.5,
94         "teamwork": 0.3,
95         "attendance": 0.2
96     }
97     # Calculate weighted score
98     score = (project_completion * weights["project_completion"] +
99               teamwork * weights["teamwork"] +
100              attendance * weights["attendance"])
101    return score
102 # Example usage
103 employees = [
104     {"name": "Eve", "project_completion": 90, "teamwork": 85, "attendance": 95},
105     {"name": "Frank", "project_completion": 70, "teamwork": 80, "attendance": 90},
106     {"name": "Grace", "project_completion": 95, "teamwork": 90, "attendance": 85},
107 ]
108 for employee in employees:
109     performance_score = evaluate_employee_performance(employee['project_completion'], employee['teamwork'], employee['attendance'])
110     print(f"{employee['name']}'s performance score: {performance_score}")
```

OUTPUT:



The screenshot shows a dark-themed terminal window in VS Code. The title bar includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS, with TERMINAL being the active tab. The status bar shows icons for Python, file operations, and zoom. The terminal content displays the execution of a Python script named lab_5.2.py, which calculates performance scores for Eve, Frank, and Grace. The output is as follows:

```
/usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
(base) akash@AKASHs-MacBook-Air ai_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
Eve's performance score: 89.5
Frank's performance score: 77.0
Grace's performance score: 91.5
(base) akash@AKASHs-MacBook-Air ai_assis %
```

Task Description-5

- **Accessibility and Inclusiveness:** Use an AI tool to generate a user feedback form application. Review whether the language and options provided are inclusive and accessible to diverse users.

```
111
112 # Task Description-5
113 # Accessibility and Inclusiveness: Use an AI tool to generate a user feedback form application. Review whether the language
114 # and options provided are inclusive and accessible to diverse users
115 from flask import Flask, request, jsonify
116
117 app = Flask(__name__)
118
119 @app.route('/feedback', methods=['POST'])
120 def feedback():
121     data = request.json
122     # Process feedback data
123     return jsonify({"message": "Feedback received", "data": data}), 201
124
125 if __name__ == '__main__':
126     app.run(debug=True)
127 # Example feedback data structure
128 [{"name": "Alex",
129  "email": "alex@example.com",
130  "feedback": "Great service!",
131  "rating": 5
132 }]
133
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
/usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
(base) akash@AKASHs-MacBook-Air ai_assis % /usr/local/bin/python3 /Users/akash/Desktop/ai_assis/lab_5.2.py
* Serving Flask app 'lab_5.2'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 397-141-148
```