Sreeja Atla

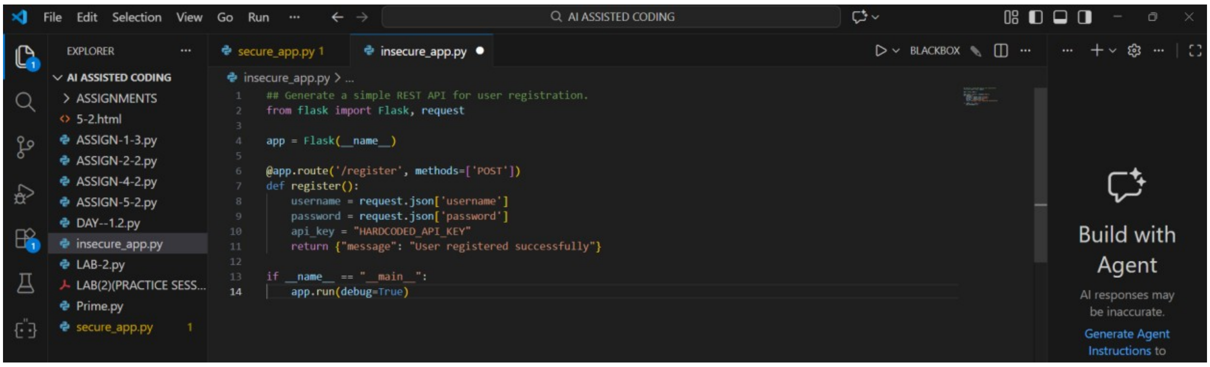
2403a51l02

B-51

Lab 5: Ethical Foundations – Responsible AI Coding Practices

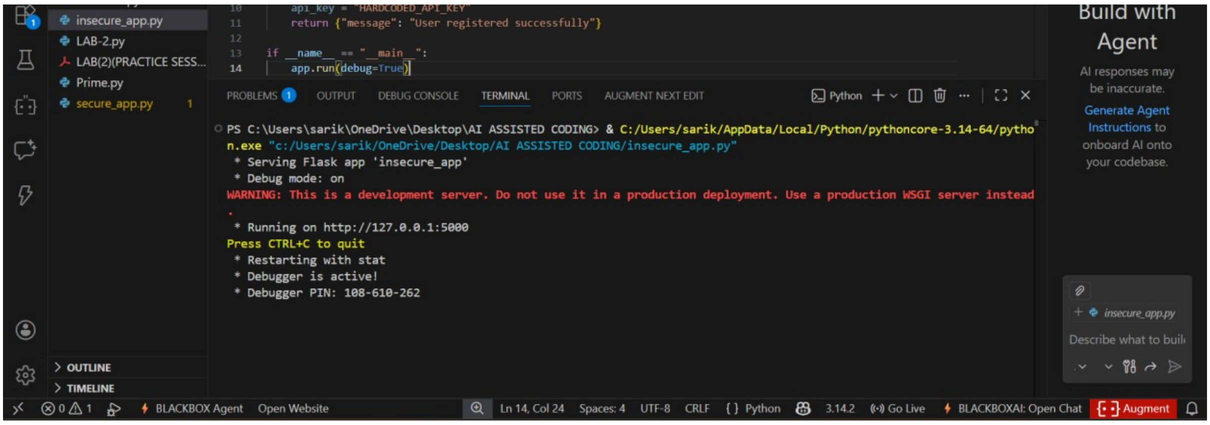
Task Description – 1: Secure API Usage

Prompt: Generate a simple REST API for user registration.

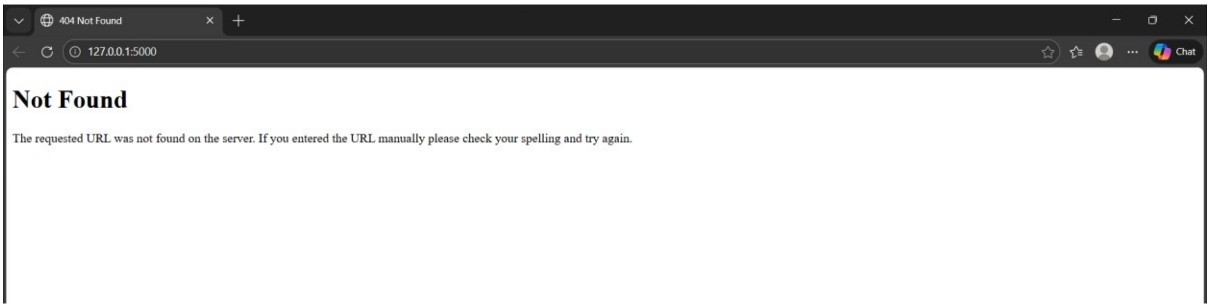


```python
## Generate a simple REST API for user registration.
from flask import Flask, request

app = Flask(__name__)

@app.route('/register', methods=['POST'])
def register():
    username = request.json['username']
    password = request.json['password']
    api_key = "HARDCODED_API_KEY"
    return {"message": "User registered successfully"}

if __name__ == "__main__":
    app.run(debug=True)
```

OUTPUT:

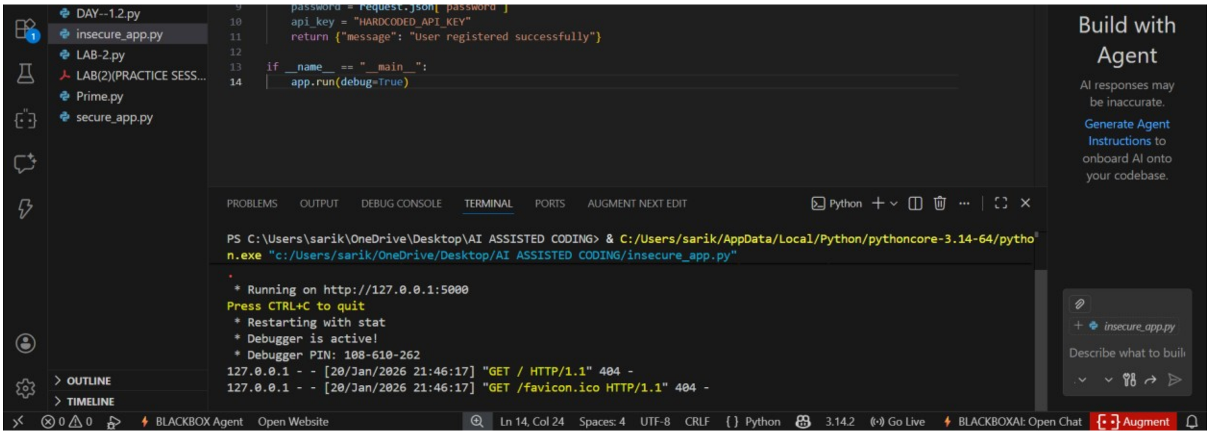


```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/pytho
n.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/insecure_app.py"
 * Serving Flask app 'insecure_app'
 * Debug mode: on
WARNING: This is a development server. Do not use 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: 108-610-262
```



Not Found

The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.



```
    password = request.json['password']
    api_key = "HARDCODED_API_KEY"
    return {"message": "User registered successfully"}

if __name__ == "__main__":
    app.run(debug=True)
```

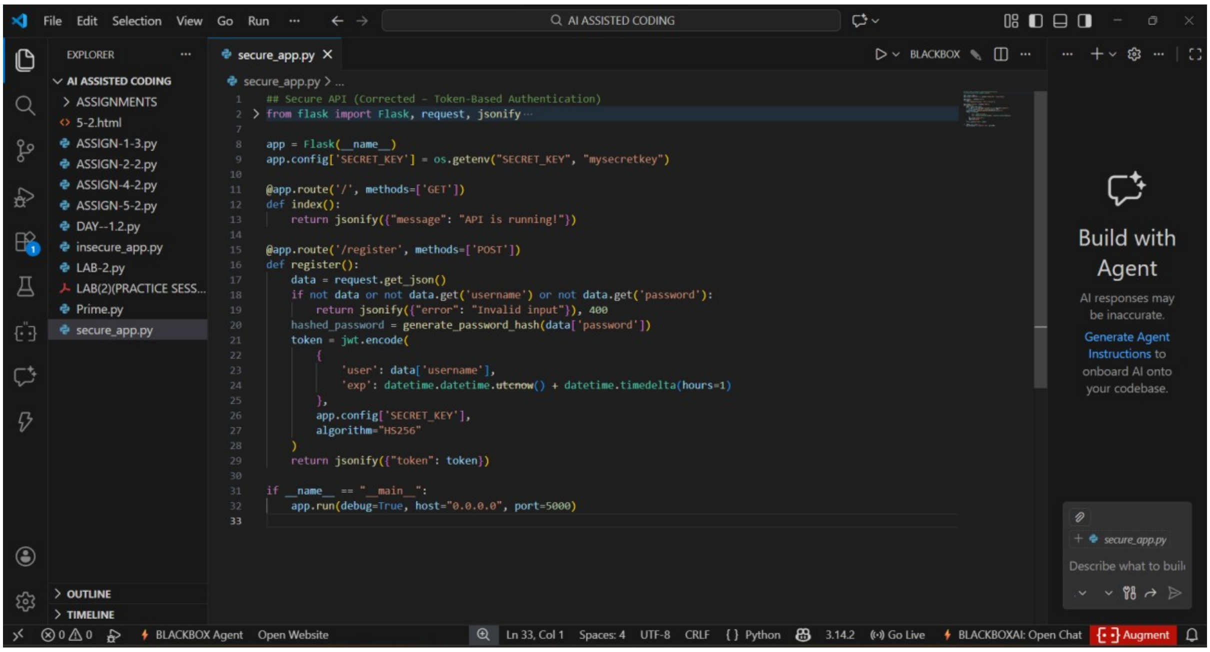
```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/pytho
n.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/insecure_app.py"
.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 108-610-262
127.0.0.1 - - [20/Jan/2026 21:46:17] "GET / HTTP/1.1" 404 -
127.0.0.1 - - [20/Jan/2026 21:46:17] "GET /favicon.ico HTTP/1.1" 404 -
```

Explanation: You got 404 error because your Flask app does not have a home (/) route, so the browser cannot find that page.

Identified Security Flaws:

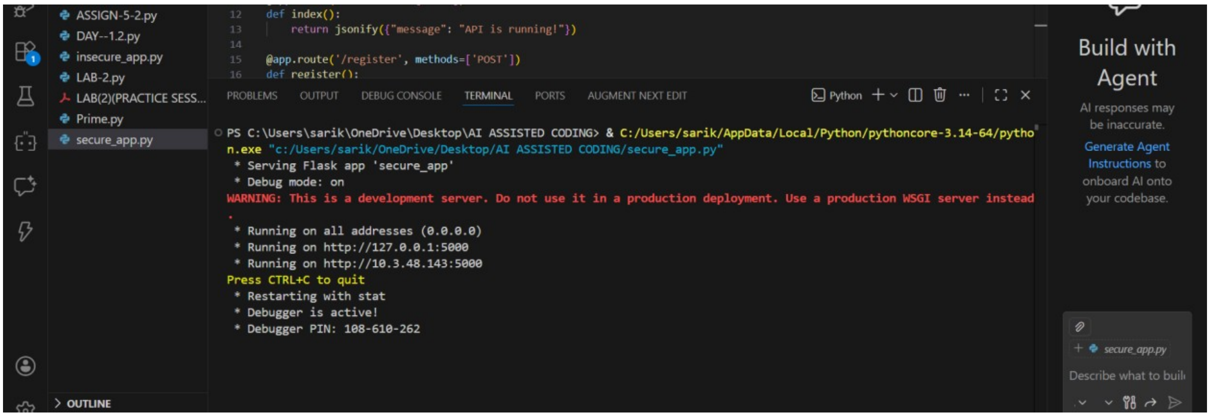
1. API key is **hardcoded**, exposing sensitive credentials
2. No authentication or authorization mechanism
3. No input validation (password strength, missing fields)
4. Password stored/used in **plain text**
5. No token-based access control

Corrected Secure Version (Token-Based Authentication):



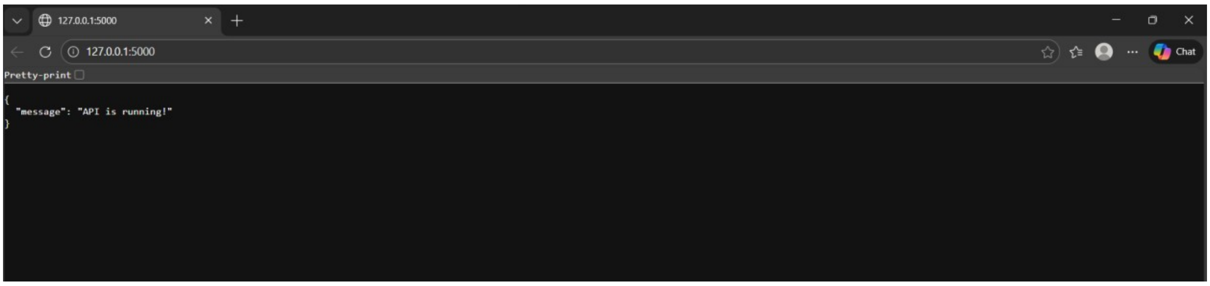
```
## Secure API (Corrected - Token-Based Authentication)
from flask import Flask, request, jsonify ···

app = Flask(__name__)
app.config['SECRET_KEY'] = os.getenv("SECRET_KEY", "mysecretkey")

@app.route('/', methods=['GET'])
def index():
    return jsonify({"message": "API is running!"})

@app.route('/register', methods=['POST'])
def register():
    data = request.get_json()
    if not data or not data.get('username') or not data.get('password'):
        return jsonify({"error": "Invalid input"}), 400
    hashed_password = generate_password_hash(data['password'])
    token = jwt.encode(
        {
            'user': data['username'],
            'exp': datetime.datetime.utcnow() + datetime.timedelta(hours=1)
        },
        app.config['SECRET_KEY'],
        algorithm="HS256"
    )
    return jsonify({"token": token})

if __name__ == "__main__":
    app.run(debug=True, host="0.0.0.0", port=5000)
```

OUTPUT:

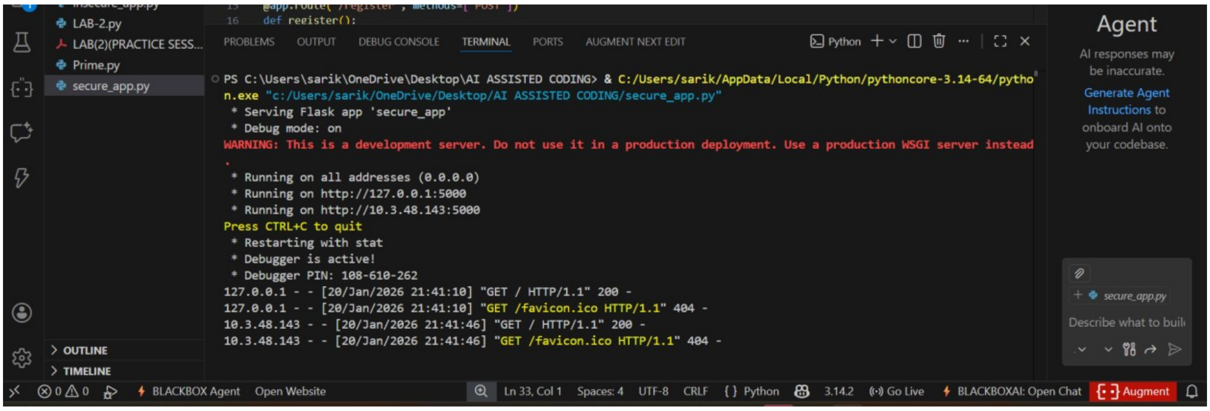


```
def index():
    return jsonify({"message": "API is running!"})

@app.route('/register', methods=['POST'])
def register():
```

```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/pytho
n.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/secure_app.py"
 * Serving Flask app 'secure_app'
 * 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 all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://10.3.48.143:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 108-610-262
```



```
{
  "message": "API is running!"
}
```



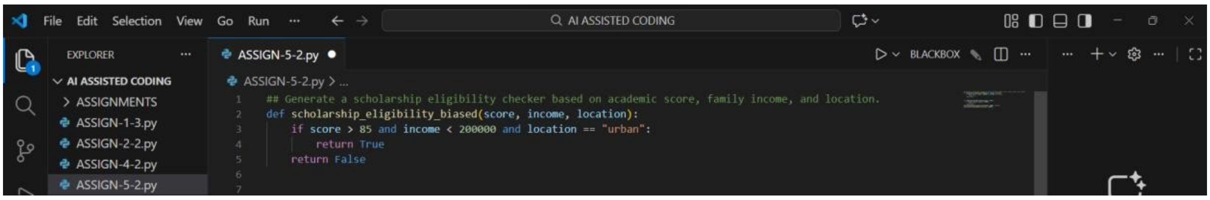
```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/pytho
n.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/secure_app.py"
 * Serving Flask app 'secure_app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead
.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://10.3.48.143:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 108-610-262
127.0.0.1 - - [20/Jan/2026 21:41:10] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [20/Jan/2026 21:41:10] "GET /favicon.ico HTTP/1.1" 404 -
10.3.48.143 - - [20/Jan/2026 21:41:46] "GET / HTTP/1.1" 200 -
10.3.48.143 - - [20/Jan/2026 21:41:46] "GET /favicon.ico HTTP/1.1" 404 -
```

Observations: The initial API code is insecure because it uses a hardcoded API key and does not protect user data. The corrected version improves security by validating inputs, hashing passwords, and using token-based authentication for safer access control.

Task Description – 2: Fair Decision Logic

Prompt: Generate a scholarship eligibility checker based on academic score, family income, and location.

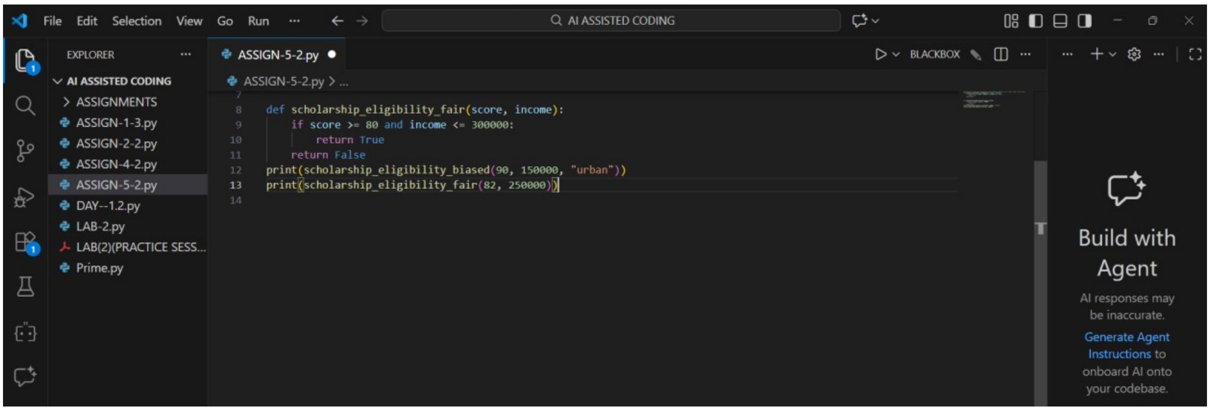
AI-Generated Code:



```python
## Generate a scholarship eligibility checker based on academic score, family income, and location.
def scholarship_eligibility_biased(score, income, location):
    if score > 85 and income < 200000 and location == "urban":
        return True
    return False
```

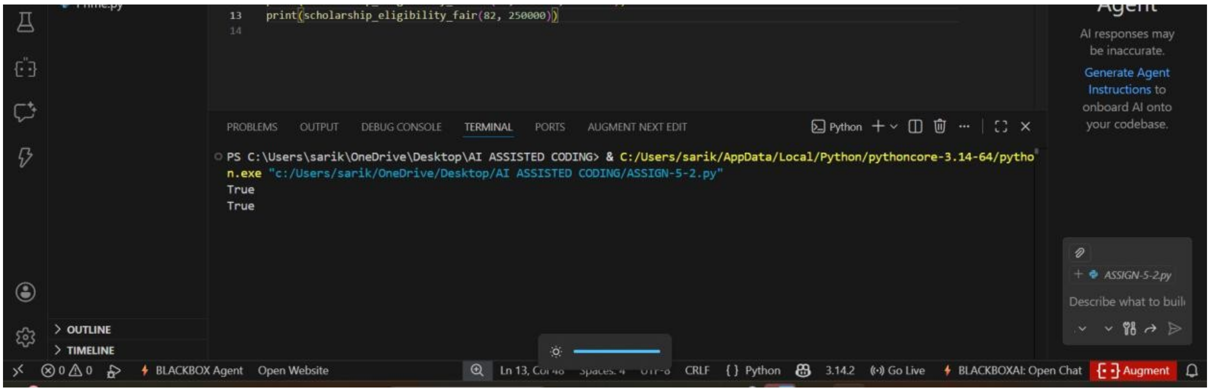
Observations:

1. The logic unfairly favors urban students
2. Rural or semi-urban students are excluded
3. No flexibility or weighted scoring approach **Improved Version:**



```python

def scholarship_eligibility_fair(score, income):
    if score >= 80 and income <= 300000:
        return True
    return False
print(scholarship_eligibility_biased(90, 150000, "urban"))
print(scholarship_eligibility_fair(82, 250000))
```

OUTPUT:



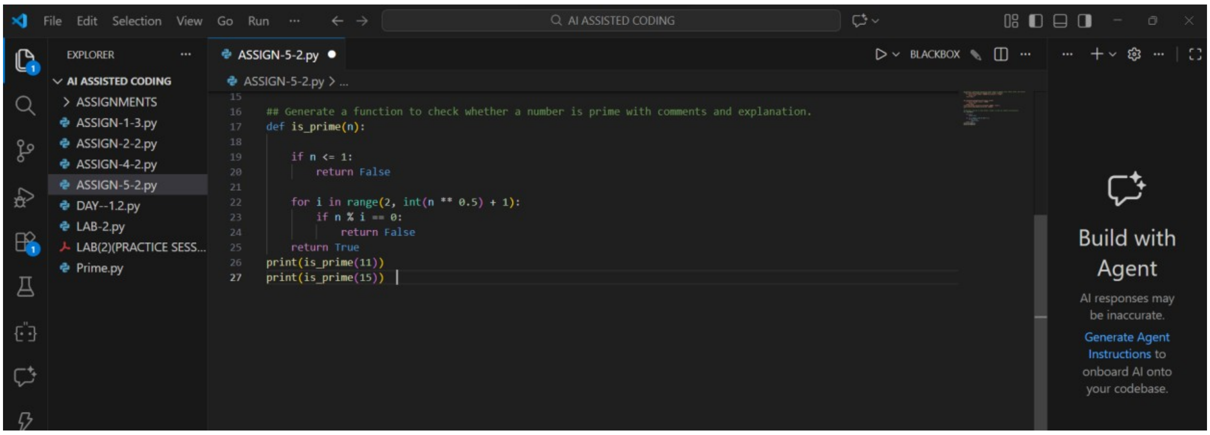
```python
print(scholarship_eligibility_fair(82, 250000))
```

```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/pytho
n.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-5-2.py"
True
True
```

Explanation: The original logic introduces geographic bias by favoring urban students. Location should not be a deciding factor unless justified by policy. A fair system focuses on merit and economic need. Weighted or threshold-based criteria help ensure equitable access.

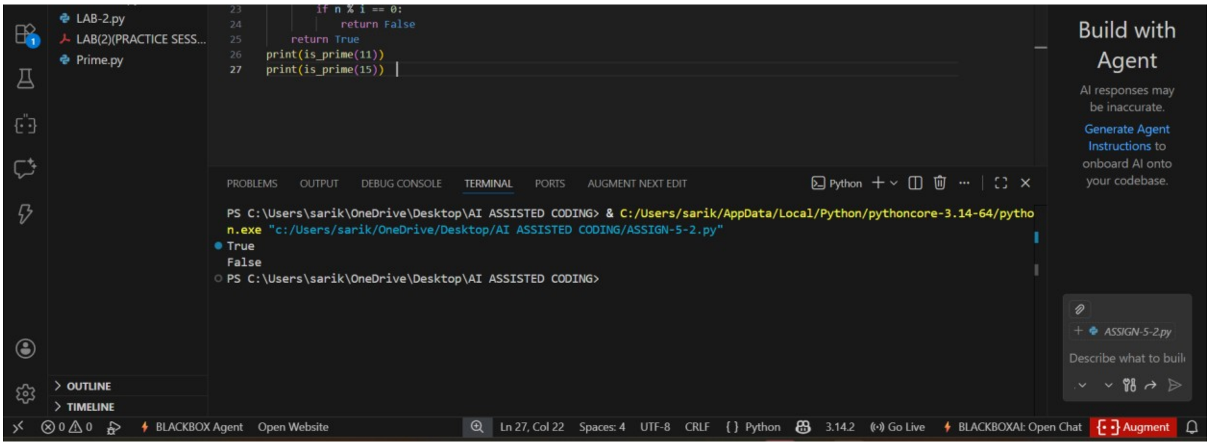
Task Description – 3: Explainability

Prompt: Generate a function to check whether a number is prime with comments and explanation.



```python
## Generate a function to check whether a number is prime with comments and explanation.
def is_prime(n):

    if n <= 1:
        return False

    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True
print(is_prime(11))
print(is_prime(15))
```

OUTPUT:



```python
    if n % i == 0:
        return False
    return True
print(is_prime(11))
print(is_prime(15))
```

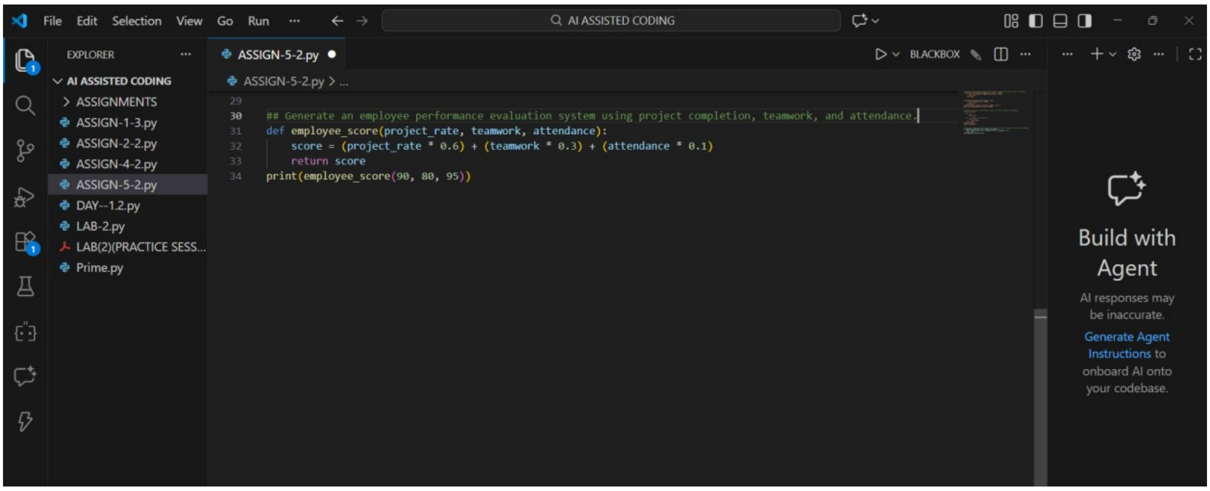
```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-5-2.py"
True
False
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING>
```

Explanation: The function first checks if the number is greater than 1. It then tests divisibility from 2 up to the square root of the number to reduce computation. If any divisor is found, the number is not prime; otherwise, it is prime.

The explanation is clear, correct, and efficient. Inline comments improve readability and help beginners understand the logic easily.

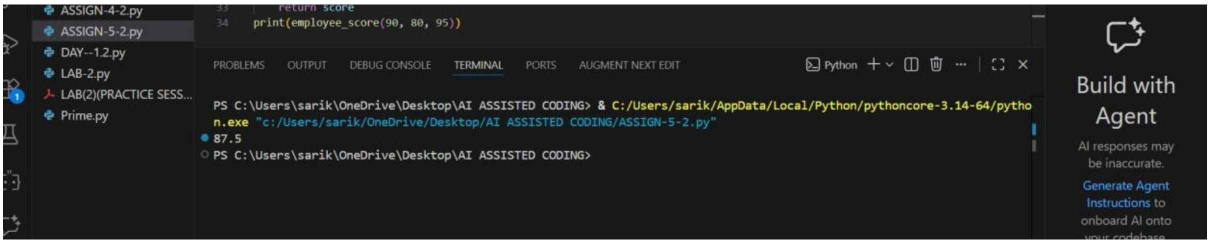
Task Description – 4: Ethical Scoring System

Prompt: Generate an employee performance evaluation system using project completion, teamwork, and attendance.



```
## Generate an employee performance evaluation system using project completion, teamwork, and attendance.
def employee_score(project_rate, teamwork, attendance):
    score = (project_rate * 0.6) + (teamwork * 0.3) + (attendance * 0.1)
    return score
print(employee_score(90, 80, 95))
```

OUTPUT:



```
    return score
print(employee_score(90, 80, 95))
```

```
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING> & C:/Users/sarik/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/sarik/OneDrive/Desktop/AI ASSISTED CODING/ASSIGN-5-2.py"
87.5
PS C:\Users\sarik\OneDrive\Desktop\AI ASSISTED CODING>
```

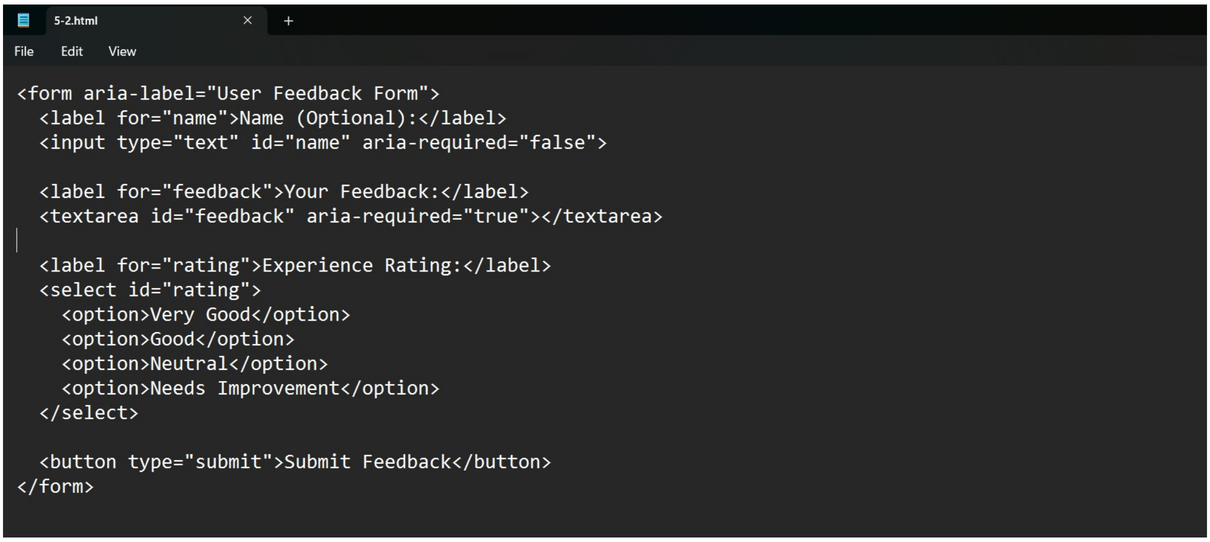
Observations:

1. Heavy weight on project completion may disadvantage collaborative roles
2. Attendance weighting may penalize employees with health or caregiving needs
3. Teamwork score depends on subjective evaluation

The criteria are reasonable but require transparency and flexibility. Ethical systems should allow contextual review and avoid over-reliance on single metrics.

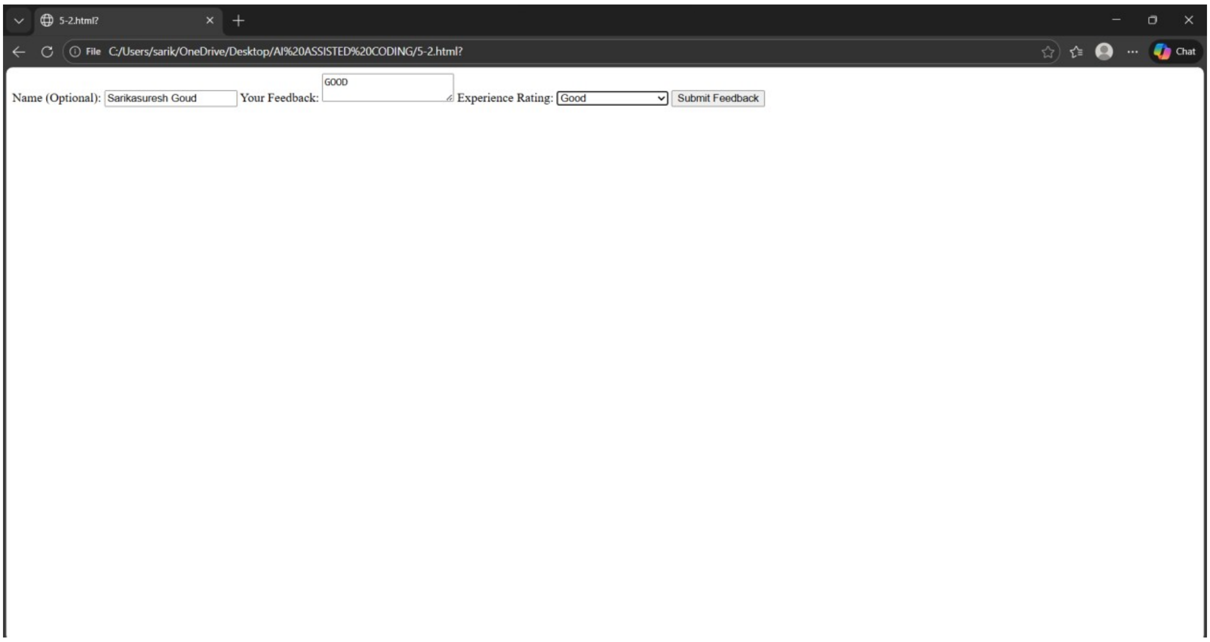
Task Description – 5: Accessibility and Inclusiveness Prompt:

Generate a user feedback form application.



```html
<form aria-label="User Feedback Form">
  <label for="name">Name (Optional):</label>
  <input type="text" id="name" aria-required="false">

  <label for="feedback">Your Feedback:</label>
  <textarea id="feedback" aria-required="true"></textarea>

  <label for="rating">Experience Rating:</label>
  <select id="rating">
    <option>Very Good</option>
    <option>Good</option>
    <option>Neutral</option>
    <option>Needs Improvement</option>
  </select>

  <button type="submit">Submit Feedback</button>
</form>
```

OUTPUT:



5-2.html?
File
C:/Users/sarik/OneDrive/Desktop/AI%20ASSISTED%20CODING/5-2.html?
Chat

Name (Optional): Sarikasuresh Goud Your Feedback: GOOD Experience Rating: Good Submit Feedback

Observations: The feedback form uses neutral and inclusive language to avoid exclusion of any user group. Accessibility is enhanced through ARIA labels, optional fields, and simple input options for diverse users.