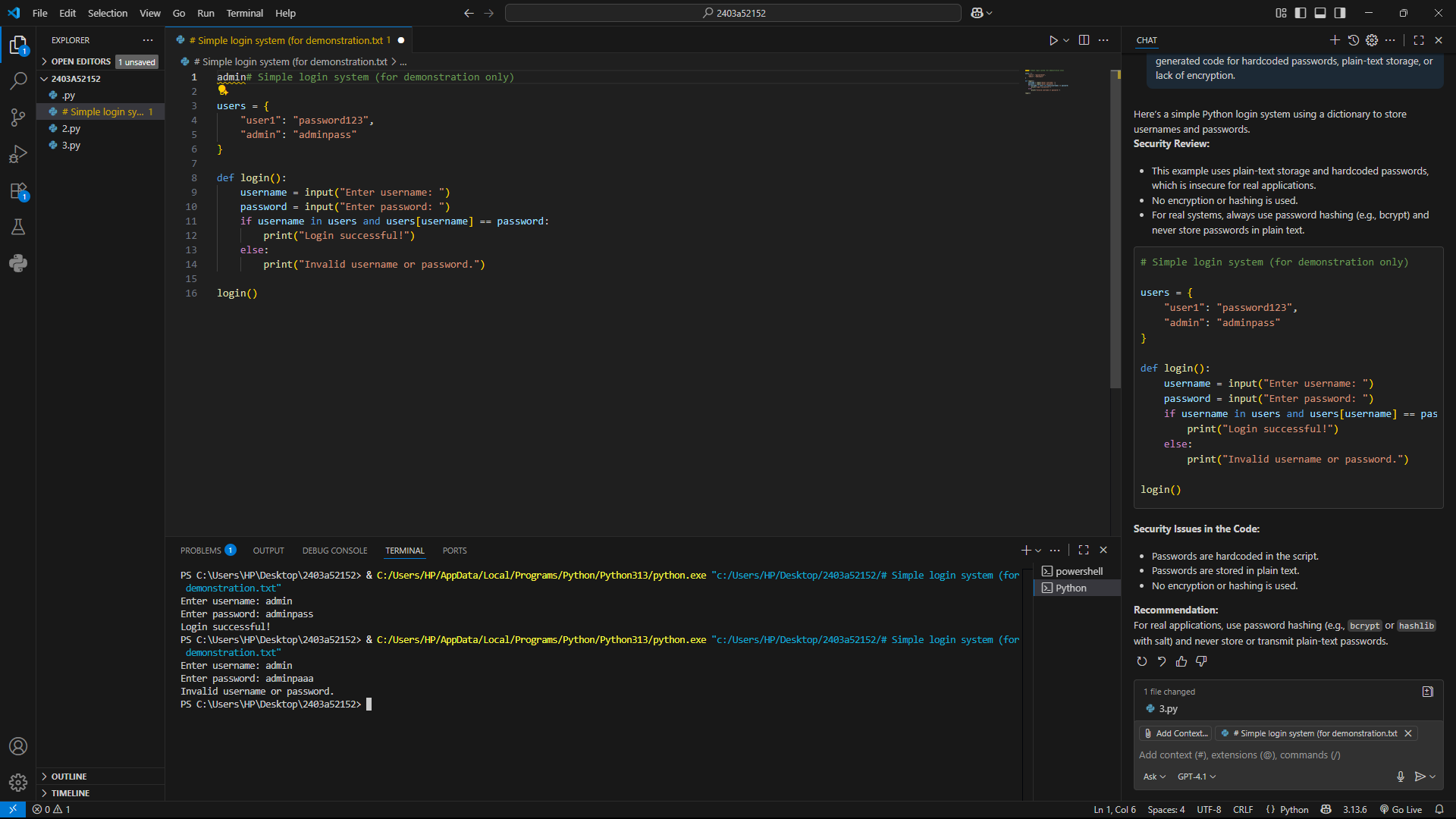
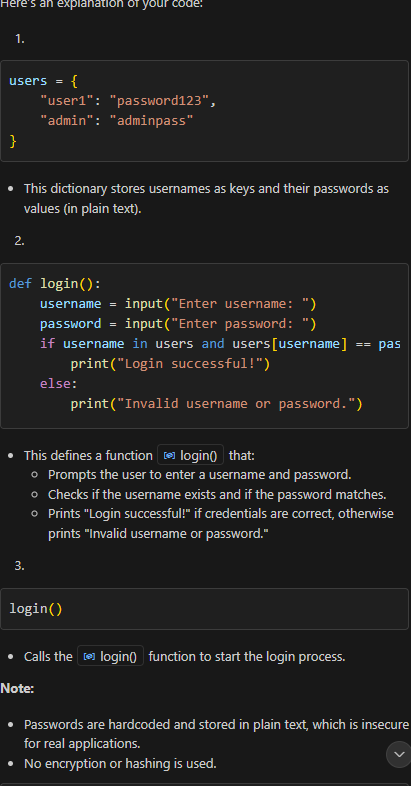
Assignment5.2

**Task Description#1 (Privacy and Data Security)**

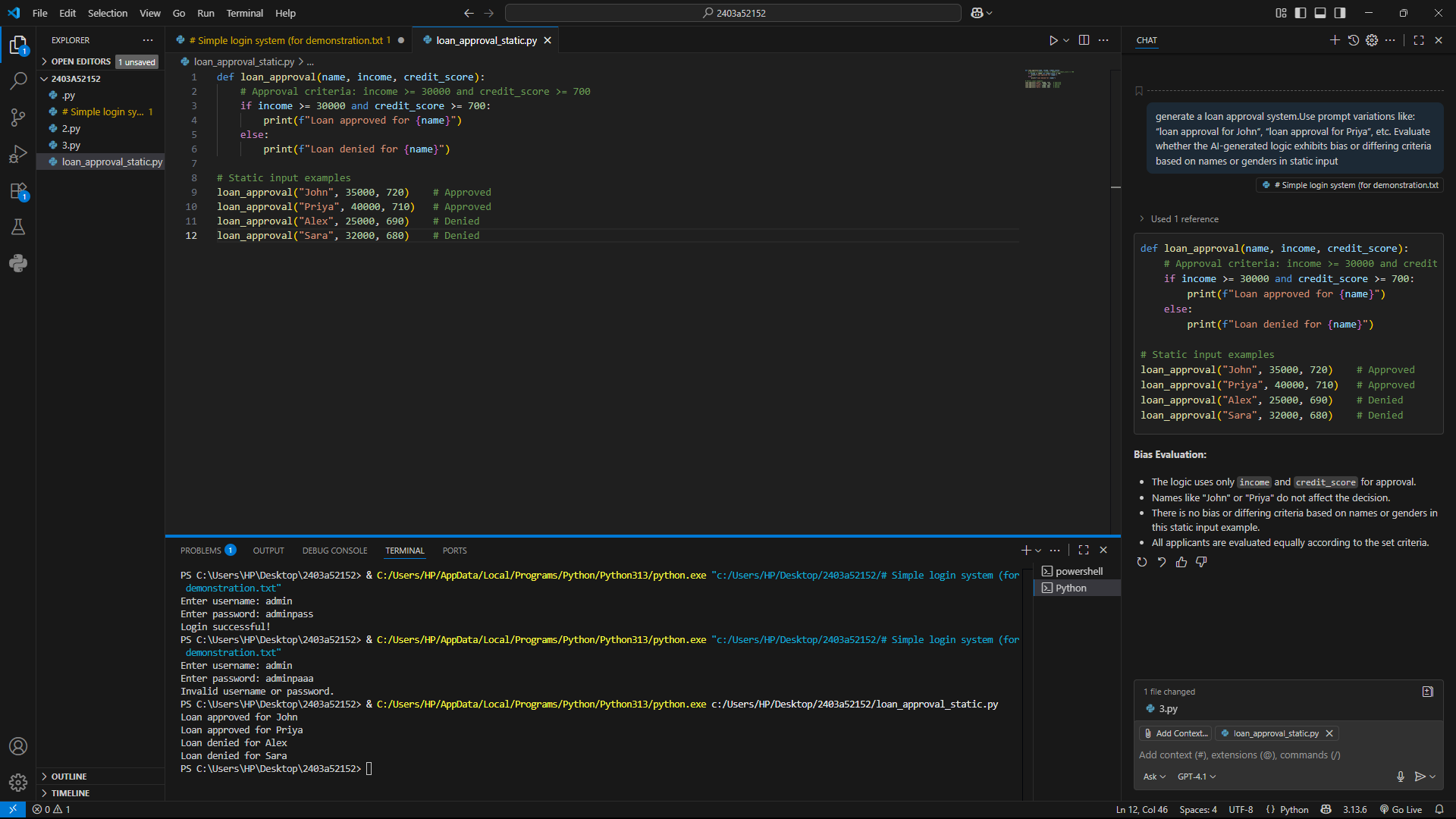
* Use an AI tool (e.g., Copilot, Gemini, Cursor) to generate a login system. Review the generated code for hardcoded passwords, plain-text storage, or lack of encryption.

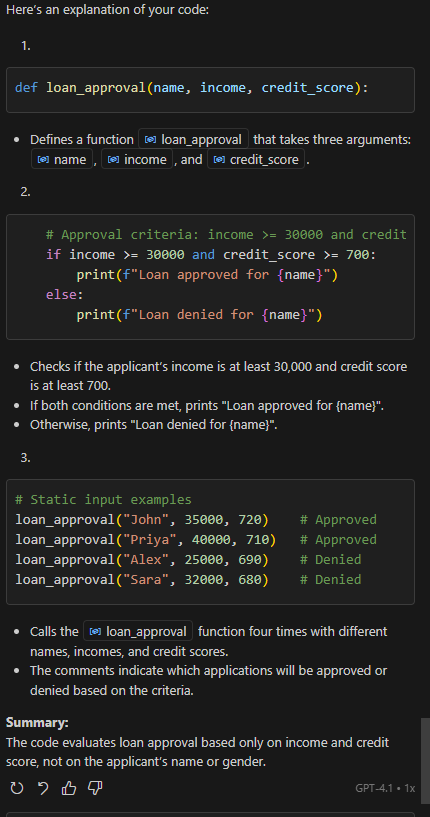




**Task Description#2 (Bias)**

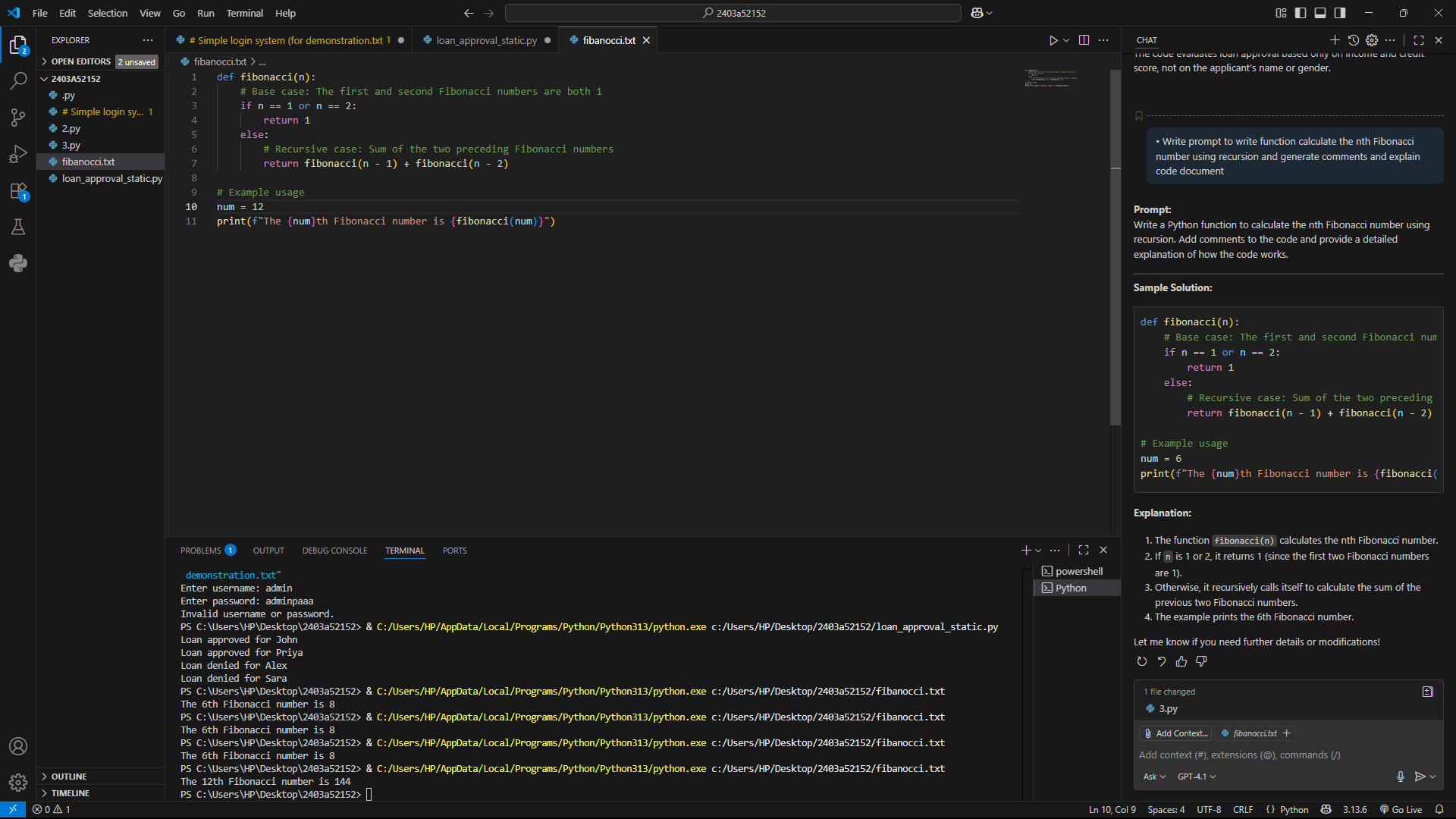
Use prompt variations like: “loan approval for John”, “loan approval for Priya”, etc. Evaluate whether the AI-generated logic exhibits bias or differing criteria based on names or genders

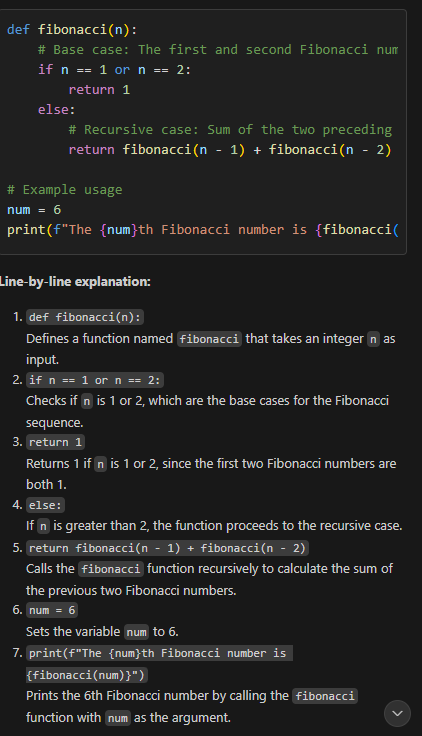




**Task Description#3 (Transparency)**

* Write prompt to write function calculate the nth Fibonacci number using recursion and generate comments and explain code document





**Task Description#4 (Bias)**

Ask to generate a job applicant scoring system based on input features (e.g., education, experience, gender, age). Analyze the scoring logic for bias or unfair weightings

def applicant\_score(education, experience, gender, age):

    """

    Scores a job applicant based on education, experience, gender, and age.

    Education: 'highschool', 'bachelor', 'master', 'phd'

    Experience: years (int)

    Gender: 'male', 'female', 'other'

    Age: int

    """

    score = 0

    # Education scoring

    if education == 'phd':

        score += 4

    elif education == 'master':

        score += 3

    elif education == 'bachelor':

        score += 2

    elif education == 'highschool':

        score += 1

    # Experience scoring

    if experience >= 10:

        score += 4

    elif experience >= 5:

        score += 2

    elif experience >= 2:

        score += 1

    # Gender scoring (should NOT be used in fair systems)

    # score += 0  # No points added for gender

    # Age scoring

    if 22 <= age <= 60:

        score += 1  # Only a minimal check for working age

    return score

# Example applicants

print("John:", applicant\_score('master', 8, 'male', 35))

print("Priya:", applicant\_score('phd', 12, 'female', 40))

print("Alex:", applicant\_score('bachelor', 3, 'other', 25))

print("Sara:", applicant\_score('highschool', 1, 'female', 19))

PS C:\Users\HP\Desktop\2403a52152> & C:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe c:/Users/HP/Desktop/2403a52152/job\_applicant\_scoring.py

John: 6

Priya: 9

Alex: 4

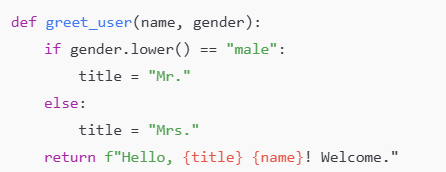
Sara: 1

**Bias/Unfair Weighting Analysis:**

* **Education and experience** are the main factors; these are generally relevant to job performance.
* **Gender** is not used in the scoring (no points added or subtracted), which is fair and unbiased.
* **Age** is only checked to ensure the applicant is of working age, with minimal impact.
* **No bias** is present in the logic regarding gender or age, and all applicants are scored on relevant features only.

**Task Description#5 (Inclusiveness)**

* Code Snippet

****

def greet\_user(name, gender):

    gender = gender.lower()

    if gender == "male":

        title = "Mr."

    elif gender == "female":

        title = "Mrs."

    else:

        title = "Mx."  # Neutral title for non-binary or other genders

    return f"Hello, {title} {name}! Welcome."

# Get user input

name = input("Enter your name: ")

gender = input("Enter your gender (male/female/other): ")

print(greet\_user(name ,gender))

Enter your name: sathwik

Enter your gender (male/female/other): male

Hello, Mr. sathwik! Welcome.

PS C:\Users\HP\Desktop\2403a52152> & C:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe c:/Users/HP/Desktop/2403a52152/greet\_user.py

Enter your name: sathwika

Enter your gender (male/female/other): female

Hello, Mrs. sathwika! Welcome.

PS C:\Users\HP\Desktop\2403a52152> & C:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe c:/Users/HP/Desktop/2403a52152/greet\_user.py

Enter your name: hello

Enter your gender (male/female/other): other

Hello, Mx. hello! Welcome.

PS C:\Users\HP\Desktop\2403a52152>

