ASSIGNMENT – 5

NAME : NALLALA SIRI

HALL TICKET NO : 2403A52037
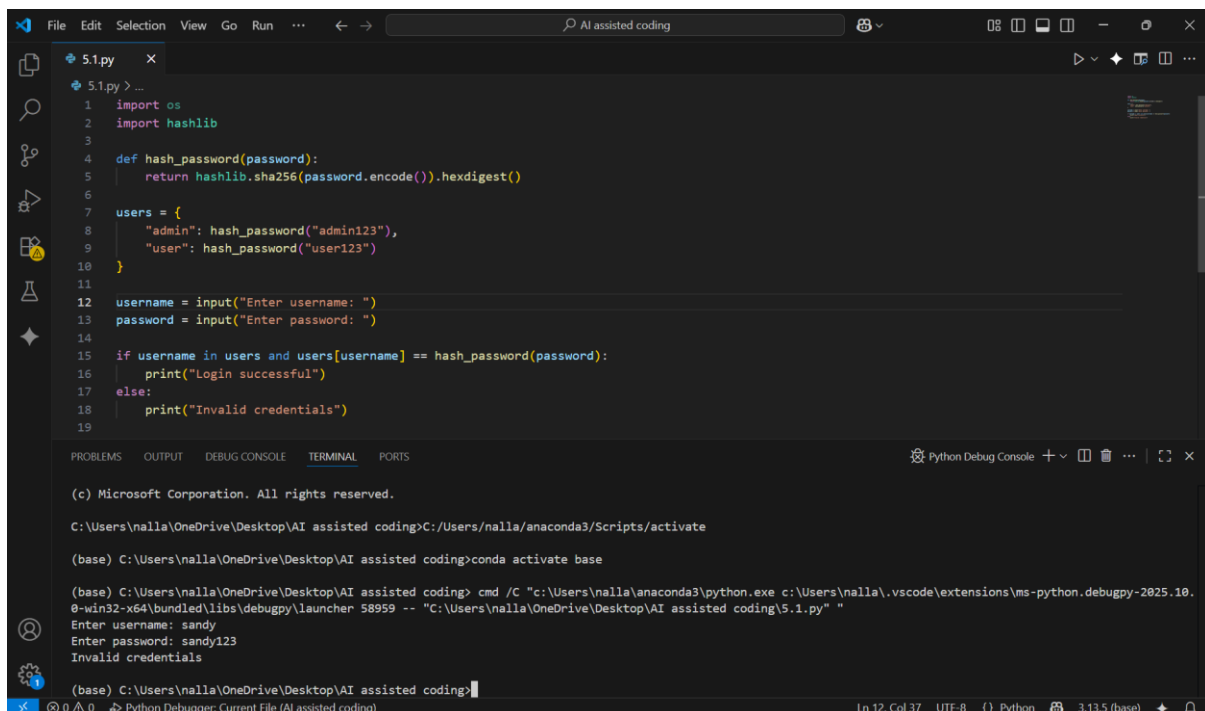
BATCH NO : AIB03

TASK 1:

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.

PROMPT:

generate a login system. Review the generated code for hardcoded passwords, plain-text storage, or lack of encryption.

CODE:


```python
import os
import hashlib

def hash_password(password):
    return hashlib.sha256(password.encode()).hexdigest()

users = {
    "admin": hash_password("admin123"),
    "user": hash_password("user123")
}

username = input("Enter username: ")
password = input("Enter password: ")

if username in users and users[username] == hash_password(password):
    print("Login successful")
else:
    print("Invalid credentials")
```

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\nalla\OneDrive\Desktop\AI assisted coding>C:/Users/nalla/anaconda3/Scripts/activate

(base) C:\Users\nalla\OneDrive\Desktop\AI assisted coding>conda activate base

(base) C:\Users\nalla\OneDrive\Desktop\AI assisted coding> cmd /C "c:\Users\nalla\anaconda3\python.exe c:\Users\nalla\.vscode\extensions\ms-python.debugpy-2025.10.
0-win32-x64\bundled\libs\debugpy\launcher 58959 -- "C:\Users\nalla\OneDrive\Desktop\AI assisted coding\5.1.py" "
Enter username: sandy
Enter password: sandy123
Invalid credentials

(base) C:\Users\nalla\OneDrive\Desktop\AI assisted coding>
```

OBSERVATION:

This Python code implements a simple login system with **hashed passwords**. It first imports the os and hashlib modules, then defines a

function hash_password that takes a password, encodes it, and returns its SHA-256 hash. A   dictionary -users stores usernames as keys and their hashed passwords as values. The program prompts the user to enter a username and password, then checks if the username exists in users and if the hashed input password matches the stored hash. If both conditions are true, it prints "Login successful"; otherwise, it prints "Invalid credentials".
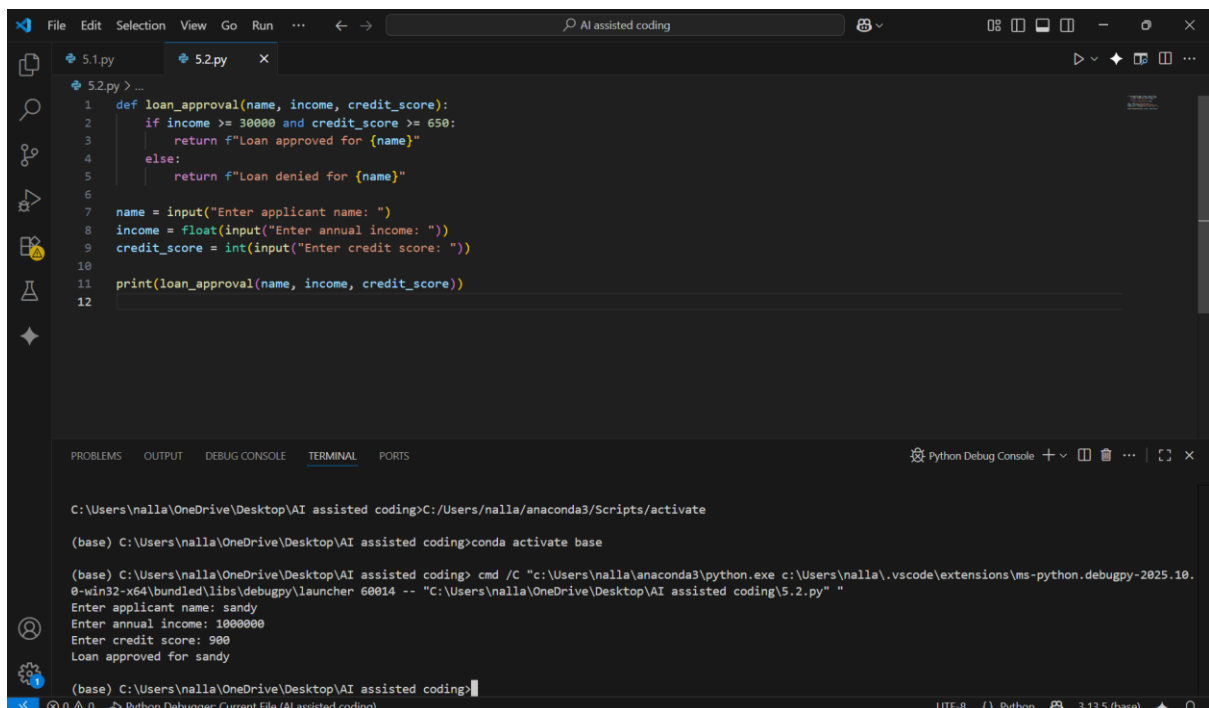
TASK 2:

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.

PROMPT :

Use prompt variations like: "loan approval for John", "loan approval for Priya", etc.Evaluate differing criteria based on names or genders.

CODE:

OBSERVATION:

This Python code defines a function loan_approval that decides whether a loan should be approved based on an applicant's income and credit score. The function takes name, income, and credit_score as inputs. If the income is at least 30,000 and the credit score is at least 650, it returns a message saying the loan is approved for that person; otherwise, it returns a message saying the loan is denied. The program prompts the user to enter their name, annual income, and credit score, then calls the loan_approval function with these inputs and prints the result.
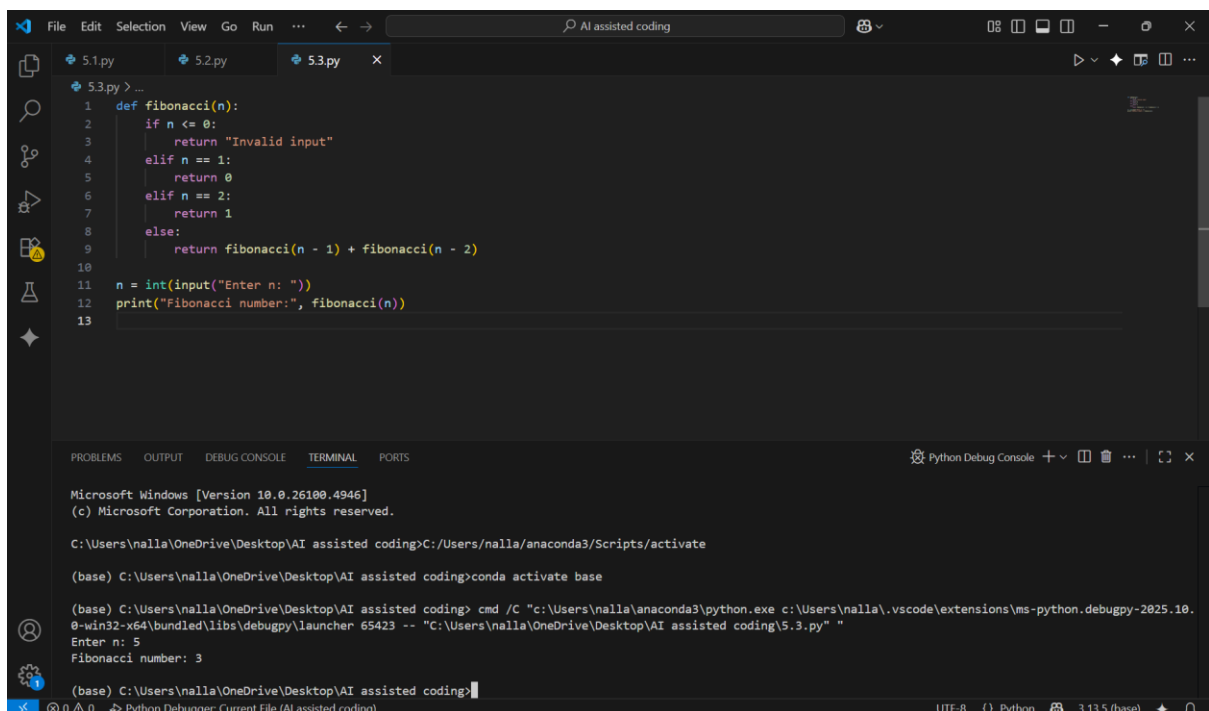
TASK 3:

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

PROMPT:

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

CODE:

OBSERVATION:

This Python code defines a recursive function fibonacci to calculate the nth Fibonacci number. The function takes an integer n as input. If n is less than or equal to 0, it returns "Invalid input". If n is 1, it returns 0, and if n is 2, it returns 1. For any larger value of n, the function calls itself recursively to calculate the sum of the two preceding Fibonacci numbers (fibonacci(n-1) + fibonacci(n-2)). The program prompts the user to enter a value for n and then prints the corresponding Fibonacci number.
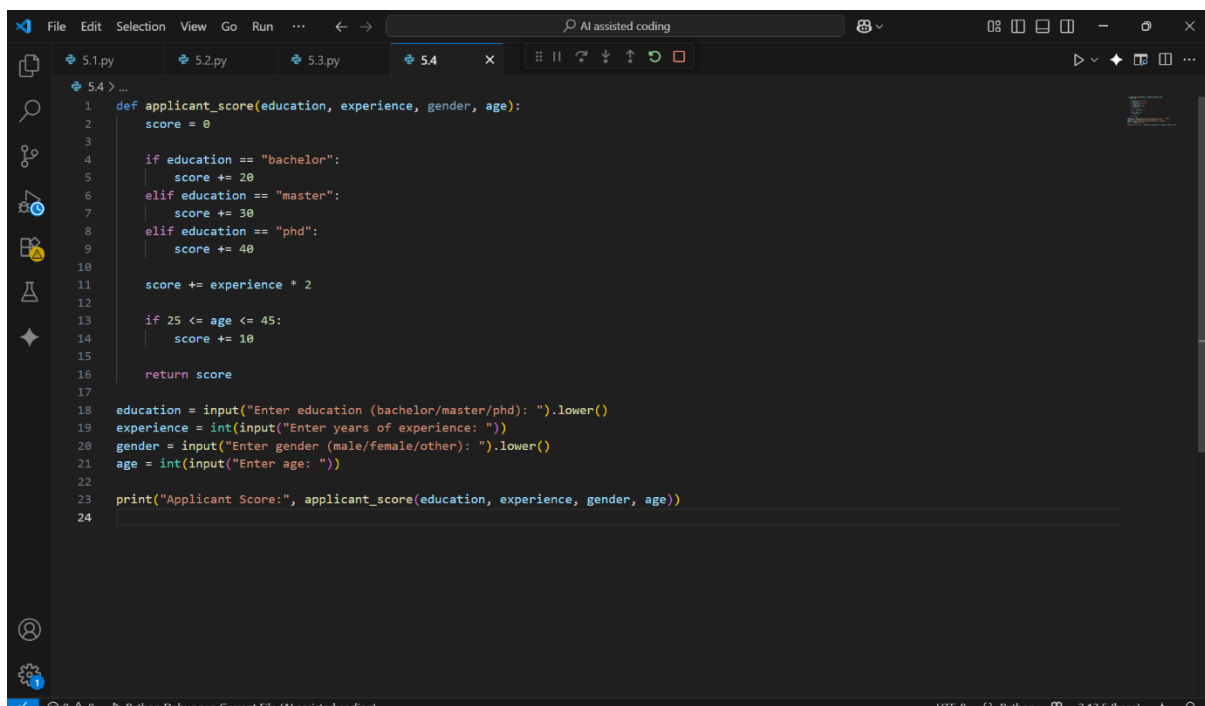
TASK 4:

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.
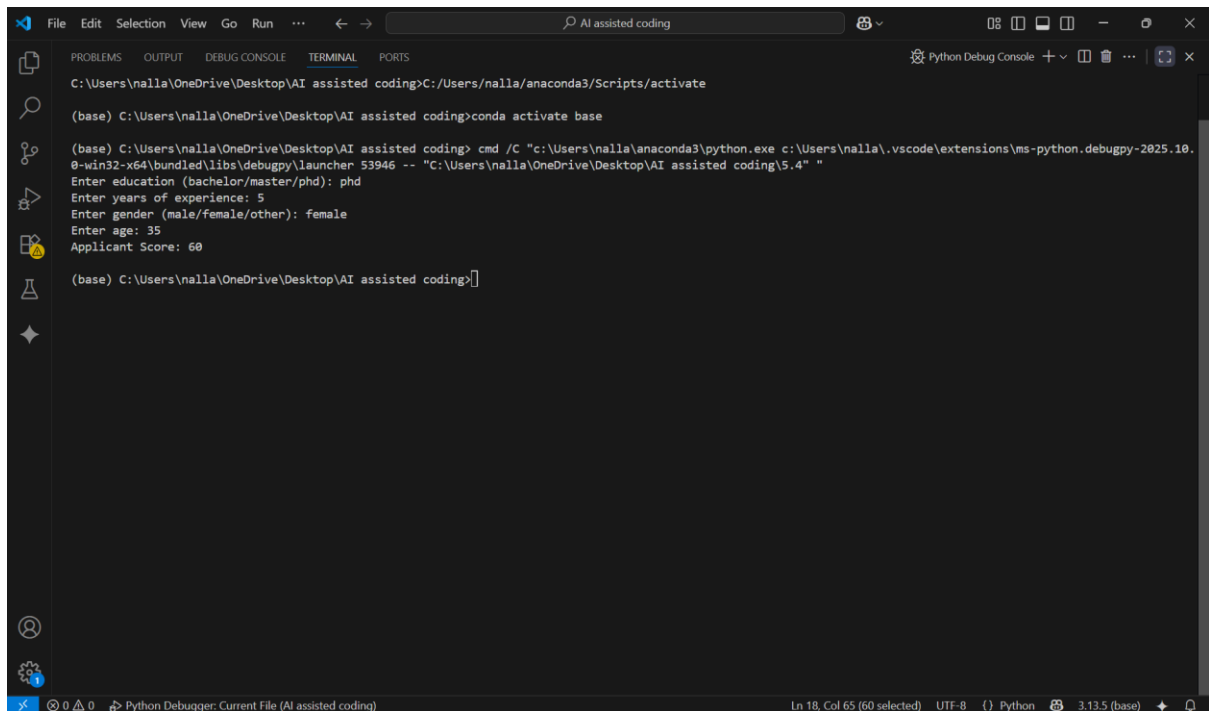
PROMPT:

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.

CODE:

```python
def applicant_score(education, experience, gender, age):
    score = 0

    if education == "bachelor":
        score += 20
    elif education == "master":
        score += 30
    elif education == "phd":
        score += 40

    score += experience * 2

    if 25 <= age <= 45:
        score += 10

    return score

education = input("Enter education (bachelor/master/phd): ").lower()
experience = int(input("Enter years of experience: "))
gender = input("Enter gender (male/female/other): ").lower()
age = int(input("Enter age: "))

print("Applicant Score:", applicant_score(education, experience, gender, age))
```

```
C:\Users\nalla\OneDrive\Desktop\AI assisted coding>C:/Users/nalla/anaconda3/Scripts/activate

(base) C:\Users\nalla\OneDrive\Desktop\AI assisted coding>conda activate base

(base) C:\Users\nalla\OneDrive\Desktop\AI assisted coding> cmd /C "c:\Users\nalla\anaconda3\python.exe c:\Users\nalla\.vscode\extensions\ms-python.debugpy-2025.10.
0-win32-x64\bundled\libs\debugpy\launcher 53946 -- "C:\Users\nalla\OneDrive\Desktop\AI assisted coding\5.4" "
Enter education (bachelor/master/phd): phd
Enter years of experience: 5
Enter gender (male/female/other): female
Enter age: 35
Applicant Score: 60

(base) C:\Users\nalla\OneDrive\Desktop\AI assisted coding>
```

OBSERVATION:

This Python code defines a function applicant_score to calculate a job applicant's score based on their education, work experience, and age. The function initializes a score variable at 0. It adds points depending on the education level: 20 for a bachelor's, 30 for a master's, and 40 for a PhD. It then adds 2 points for each year of work experience. Additionally, if the applicant's age is between 25 and 45, it adds 10 more points. The program prompts the user to enter their education, years of experience, gender, and age, then calls applicant_score with these inputs and prints the calculated score. Note that gender is collected but not used in the scoring.
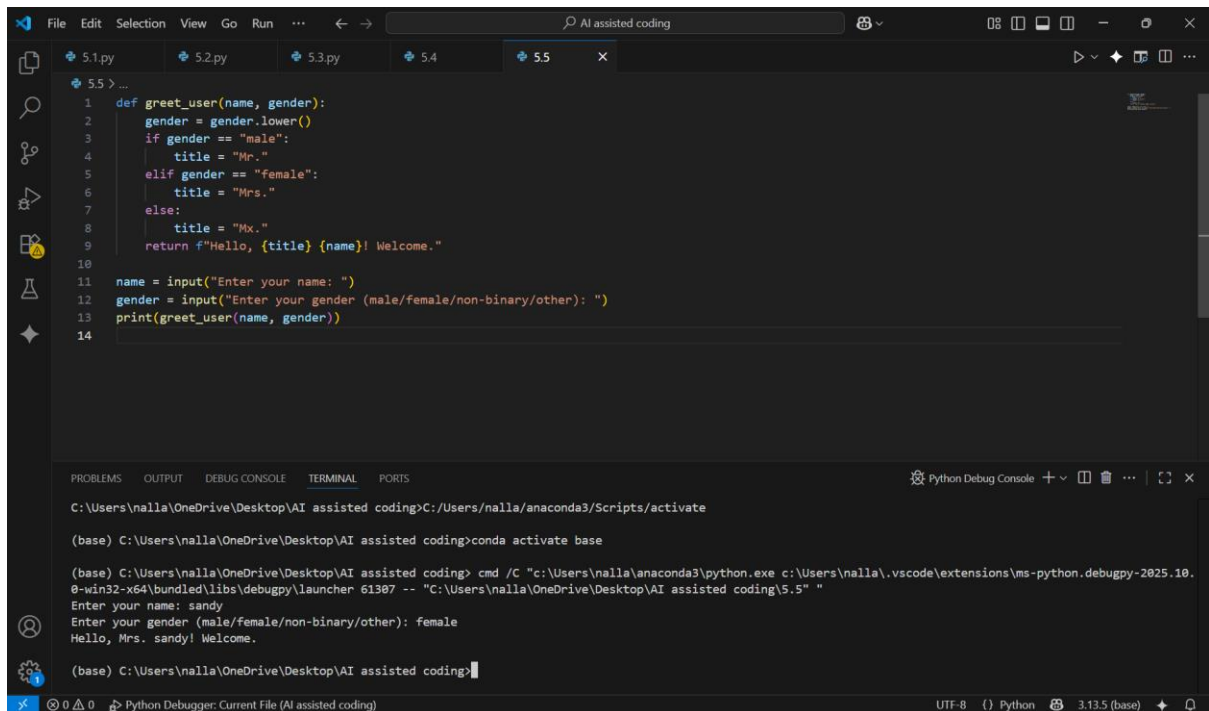
TASK 5:

Code Snippet, screenshot

PROMPT:

Given the screen shot as prompt.

CODE:

OBSERVATION:

This Python code defines a function greet_user that generates a personalized greeting based on the user's name and gender. The function first converts the gender input to lowercase, then assigns a title: "Mr." for male, "Mrs." for female, and "Mx." for any other input. It returns a greeting string that includes the title and the user's name. The program prompts the user to enter their name and gender, calls the greet_user function with these inputs, and prints the resulting personalized greeting.