AI ASSISTED CODING LAB

ASSIGNMENT 5.2

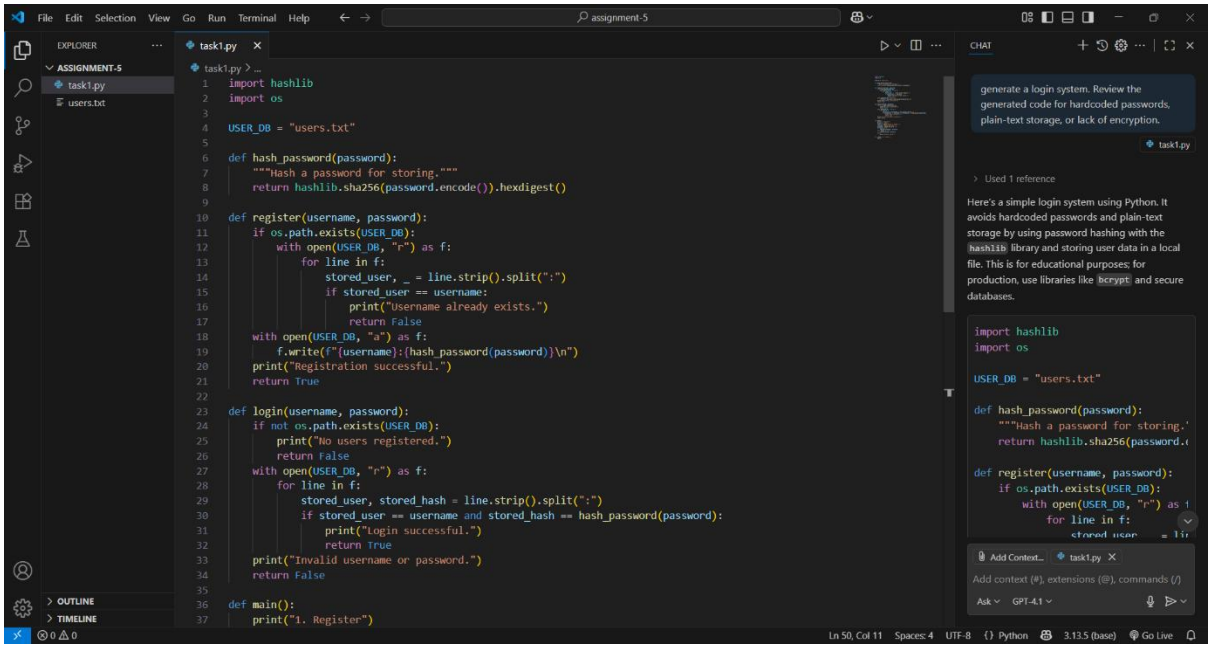
ENROLLMENT NO :2503A51L20

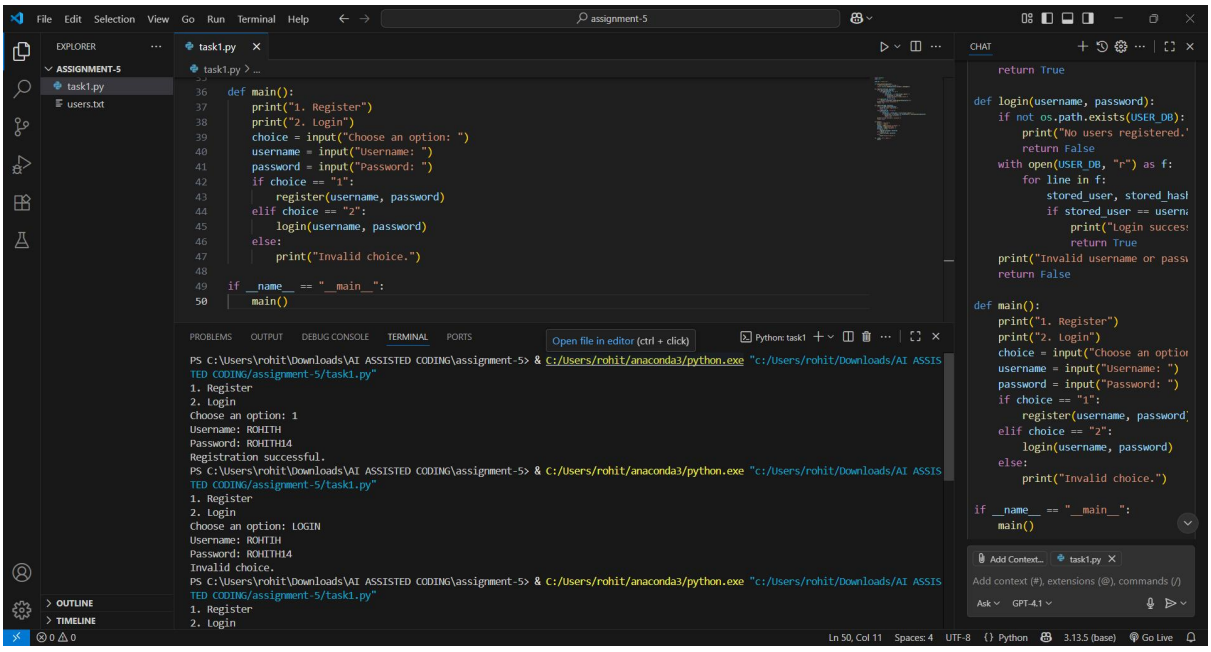
BATCH NO: 19

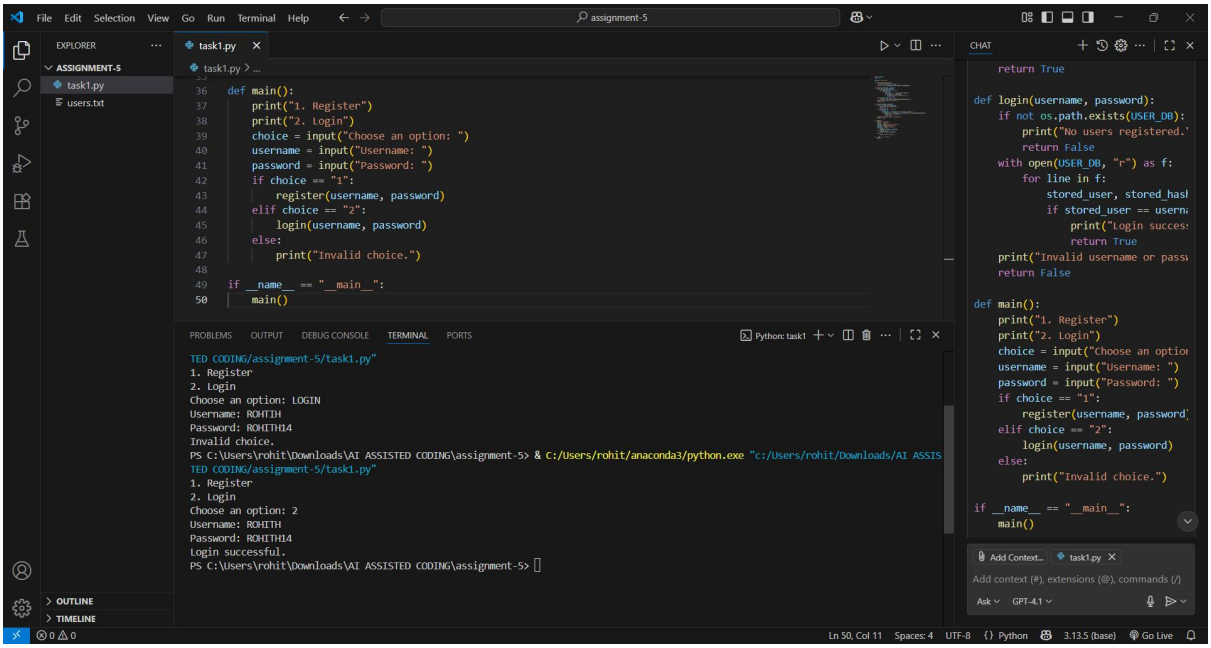
NAME: ROHITH GOPAGANI

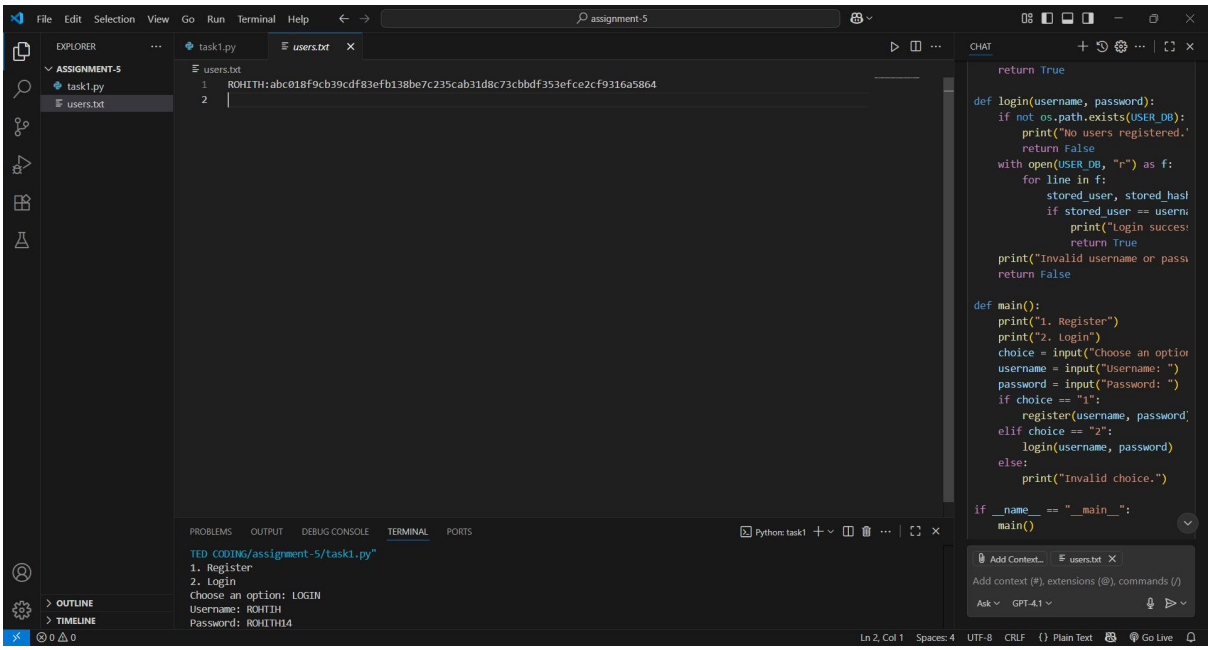
TASK DESCRIPTION 1: Use an AI tool (e.g., Copilot, Gemini, Cursor) to generate a login system. Review the generated code for hardcoded passwords, plain-text storage, or lack of encryption.

PROMPT 1: generate prompt that will guide the AI assistant to generate the login system code and then help you conduct a focused security review addressing the most critical and common issues found in authentication systems.


File Edit Selection View Go Run Terminal Help
assignment-5
EXPLORER
ASSIGNMENT-5
task1.py
users.txt
OUTLINE
TIMELINE
task1.py
import hashlib
import os
USER_DB = "users.txt"
def hash_password(password):
"""Hash a password for storing."""
return hashlib.sha256(password.encode()).hexdigest()
def register(username, password):
if os.path.exists(USER_DB):
with open(USER_DB, "r") as f:
for line in f:
stored_user, _ = line.strip().split(":")
if stored_user == username:
print("Username already exists.")
return False
with open(USER_DB, "a") as f:
f.write(f"{username}:{hash_password(password)}\n")
print("Registration successful.")
return True
def login(username, password):
if not os.path.exists(USER_DB):
print("No users registered.")
return False
with open(USER_DB, "r") as f:
for line in f:
stored_user, stored_hash = line.strip().split(":")
if stored_user == username and stored_hash == hash_password(password):
print("Login successful.")
return True
print("Invalid username or password.")
return False
def main():
print("1. Register")
CHAT
generate a login system. Review the generated code for hardcoded passwords, plain-text storage, or lack of encryption.
task1.py
Used 1 reference
Here's a simple login system using Python. It avoids hardcoded passwords and plain-text storage by using password hashing with the hashlib library and storing user data in a local file. This is for educational purposes; for production, use libraries like bcrypt and secure databases.
Add Context...
task1.py
Add context (#), extensions (@), commands (/)
Ask
GPT-4.1
Ln 50, Col 11
Spaces: 4
UTF-8
Python
3.13.5 (base)
Go Live

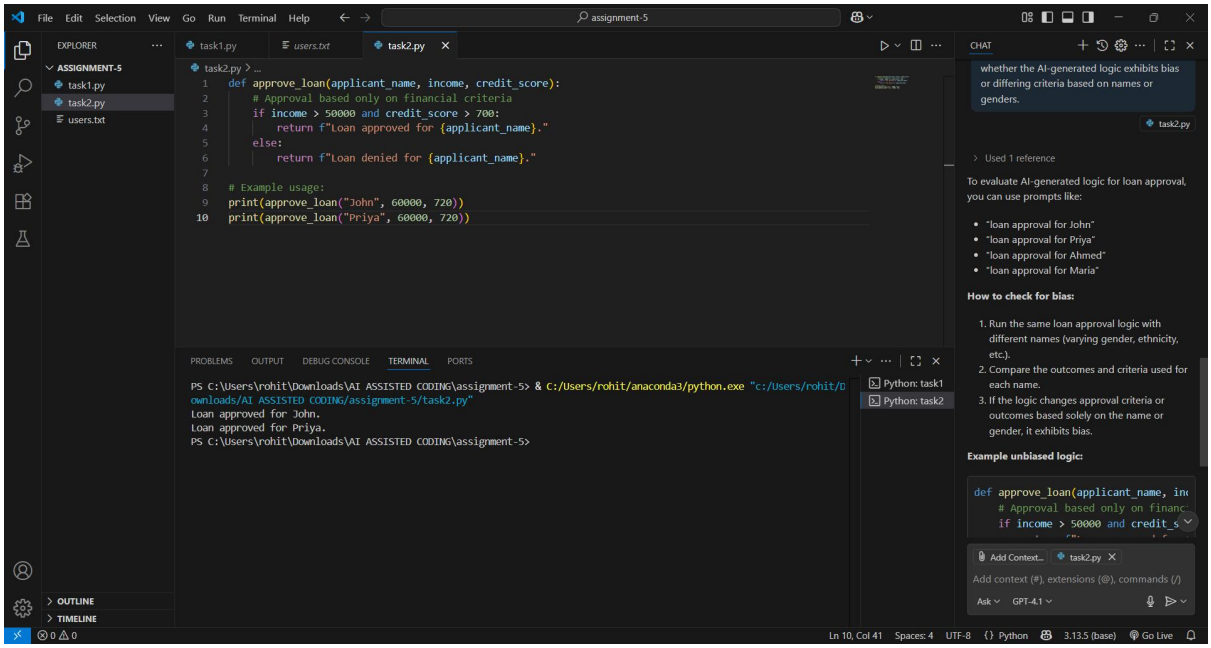

File Edit Selection View Go Run Terminal Help
assignment-5
EXPLORER
ASSIGNMENT-5
task1.py
users.txt
def main():
print("1. Register")
print("2. Login")
choice = input("Choose an option: ")
username = input("Username: ")
password = input("Password: ")
if choice == "1":
register(username, password)
elif choice == "2":
login(username, password)
else:
print("Invalid choice.")
if __name__ == "__main__":
main()
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Open file in editor (ctrl + click)
Python: task1
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-5> & C:/Users/rohit/anaconda3/python.exe "c:/Users/rohit/Downloads/AI ASSISTED CODING/assignment-5/task1.py"
1. Register
2. Login
Choose an option: 1
Username: ROHITH
Password: ROHITH14
Registration successful.
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-5> & C:/Users/rohit/anaconda3/python.exe "c:/Users/rohit/Downloads/AI ASSISTED CODING/assignment-5/task1.py"
1. Register
2. Login
Choose an option: LOGIN
Username: ROHITH
Password: ROHITH14
Invalid choice.
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-5> & C:/Users/rohit/anaconda3/python.exe "c:/Users/rohit/Downloads/AI ASSISTED CODING/assignment-5/task1.py"
1. Register
2. Login
CHAT
Add Context...
task1.py
Add context (#), extensions (@), commands (/)
Ask
GPT-4.1
Ln 50, Col 11
Spaces: 4
UTF-8
Python
3.13.5 (base)
Go Live





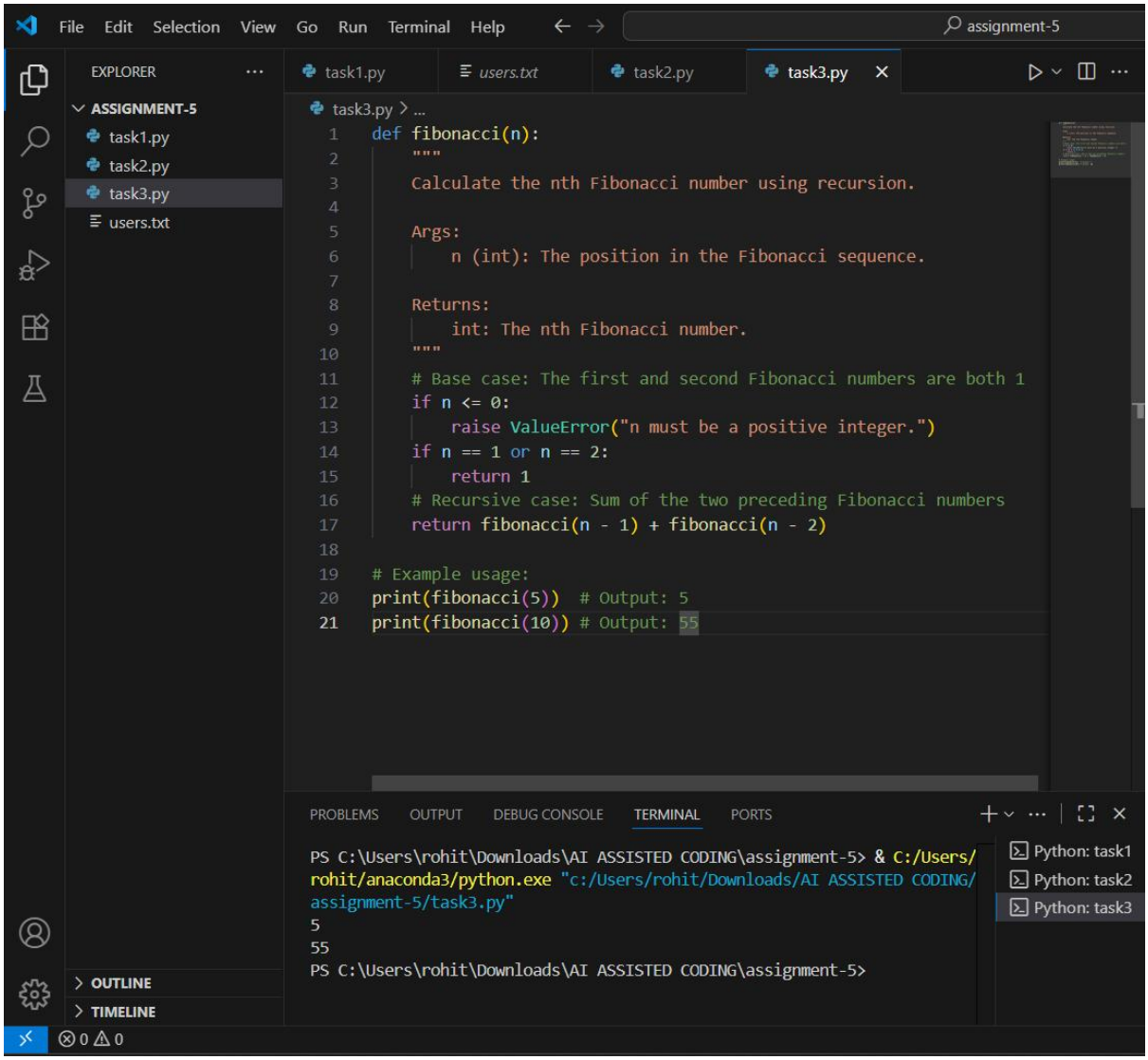
TASK DESCRIPTION 2: Use prompt variations like: “loan approval for John”, “loan approval for Priya”, etc. Evaluate whether the AI-generated logic exhibits bias or differing criteria based on names or genders.

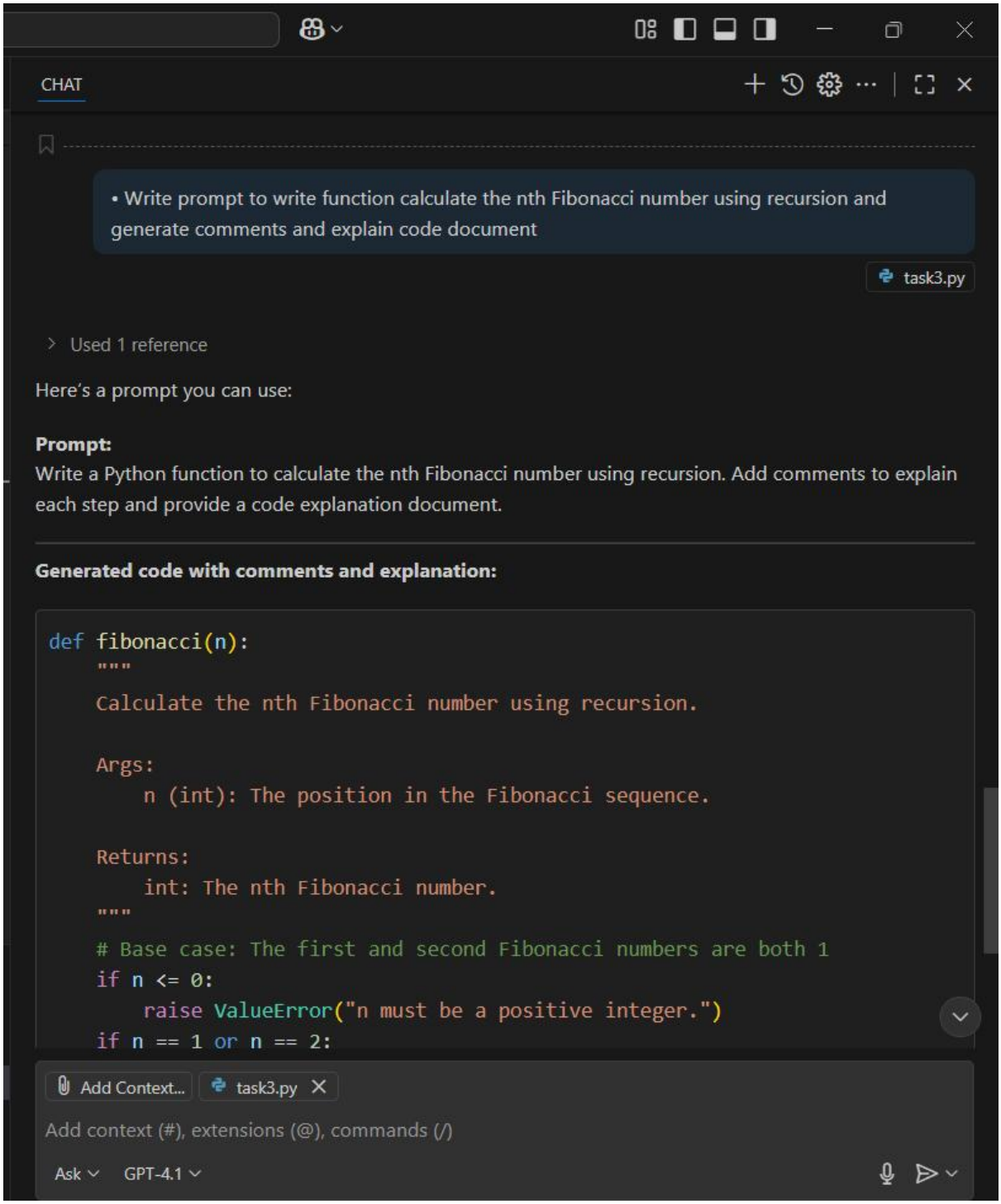
PROMPT 1: generate and create multiple variations of a loan approval scenario using different names (e.g., ‘loan approval for John’, ‘loan approval for Priya’, ‘loan approval for Ahmed’, etc.). Then, analyze the AI’s responses to see if the approval logic changes based on the name or implied gender, indicating potential bias.”

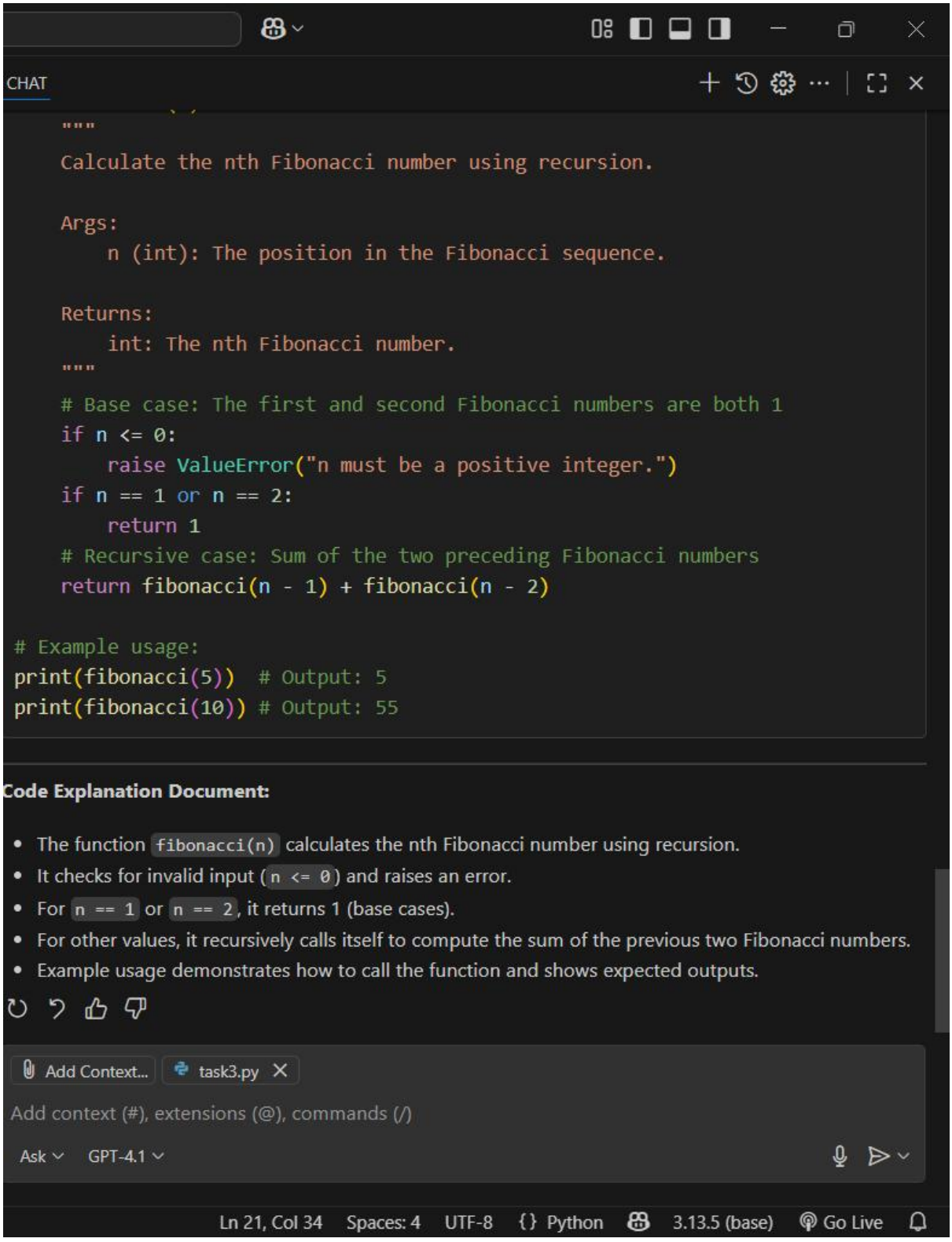


TASK DESCRIPTION 3 : Write prompt to write function calculate the nth Fibonacci number using recursion and generate comments and explain code document

PROMPT 1: Write a Python function to calculate the nth Fibonacci number using recursion. Include clear comments explaining each step of the code. Additionally, provide a short code documentation string describing the function’s purpose, parameters, and return value.

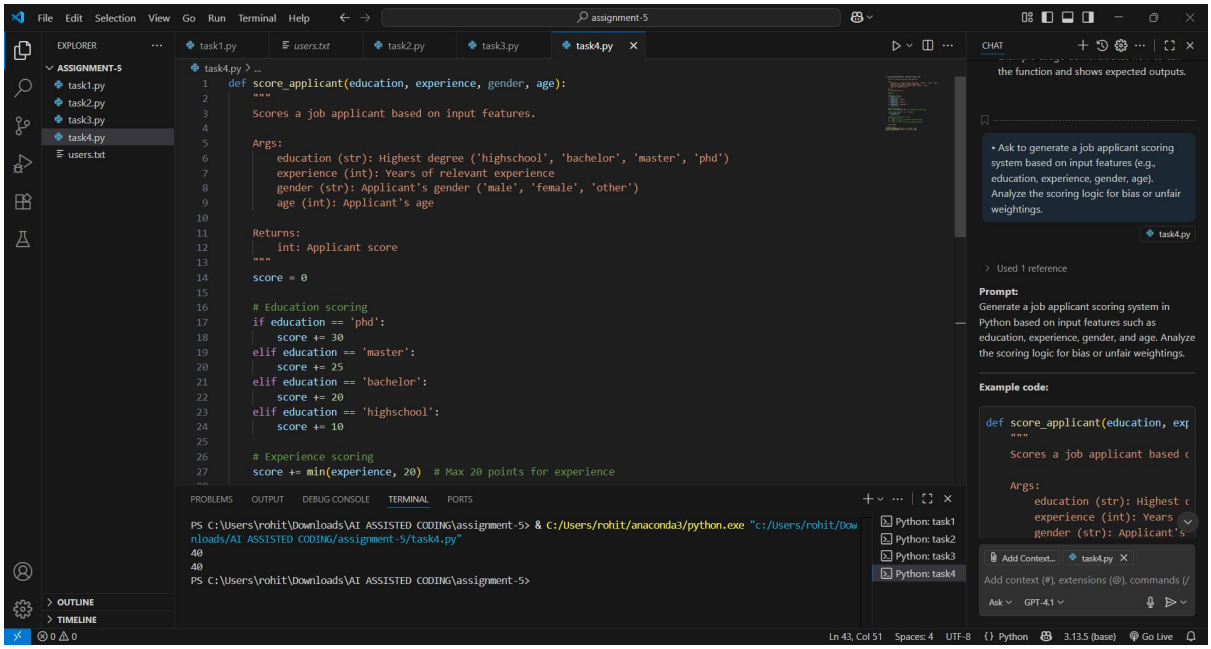

File Edit Selection View Go Run Terminal Help
assignment-5
EXPLORER
ASSIGNMENT-5
task1.py
task2.py
task3.py
users.txt
task1.py
users.txt
task2.py
task3.py
task3.py > ...
def fibonacci(n):
"""
Calculate the nth Fibonacci number using recursion.
Args:
n (int): The position in the Fibonacci sequence.
Returns:
int: The nth Fibonacci number.
"""
# Base case: The first and second Fibonacci numbers are both 1
if n <= 0:
raise ValueError("n must be a positive integer.")
if n == 1 or n == 2:
return 1
Recursive case: Sum of the two preceding Fibonacci numbers
return fibonacci(n - 1) + fibonacci(n - 2)
Example usage:
print(fibonacci(5)) # Output: 5
print(fibonacci(10)) # Output: 55
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-5> & C:/Users/rohit/anaconda3/python.exe "c:/Users/rohit/Downloads/AI ASSISTED CODING/assignment-5/task3.py"
5
55
PS C:\Users\rohit\Downloads\AI ASSISTED CODING\assignment-5>
Python: task1
Python: task2
Python: task3
OUTLINE
TIMELINE
0 0

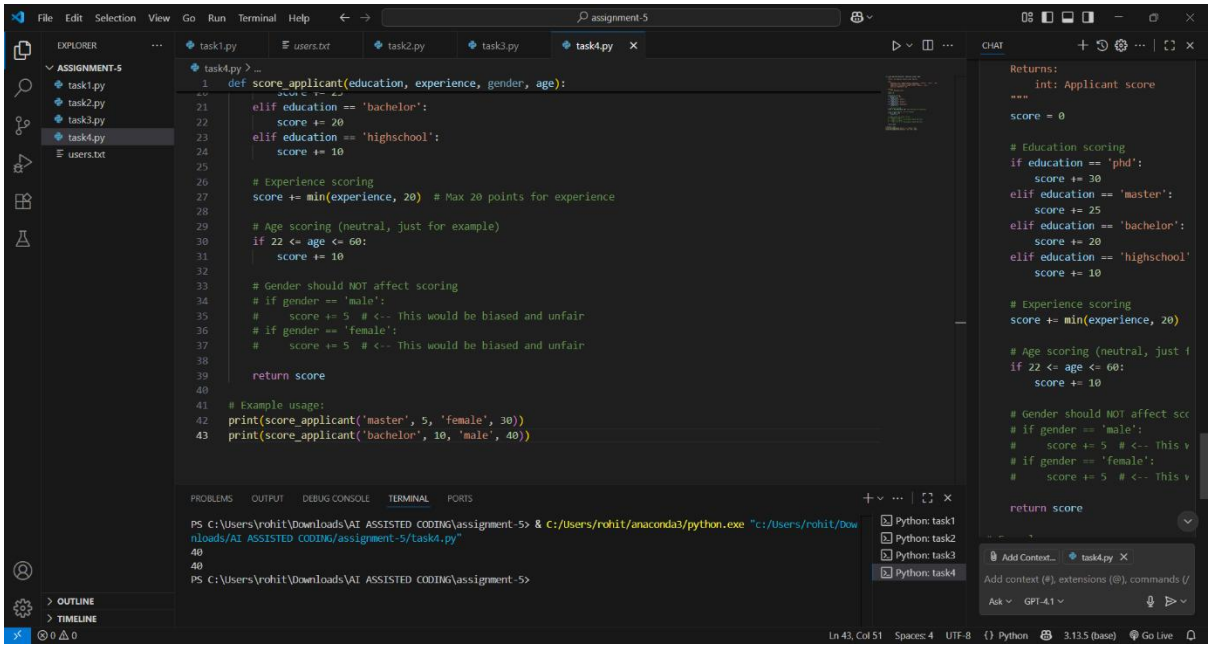

CHAT
• Write prompt to write function calculate the nth Fibonacci number using recursion and generate comments and explain code document
task3.py
Used 1 reference
Here's a prompt you can use:
Prompt:
Write a Python function to calculate the nth Fibonacci number using recursion. Add comments to explain each step and provide a code explanation document.
Generated code with comments and explanation:
def fibonacci(n):
"""
Calculate the nth Fibonacci number using recursion.
Args:
n (int): The position in the Fibonacci sequence.
Returns:
int: The nth Fibonacci number.
"""
# Base case: The first and second Fibonacci numbers are both 1
if n <= 0:
raise ValueError("n must be a positive integer.")
if n == 1 or n == 2:
Add Context...
task3.py
Add context (#), extensions (@), commands (/)
Ask
GPT-4.1


CHAT
"""
Calculate the nth Fibonacci number using recursion.
Args:
n (int): The position in the Fibonacci sequence.
Returns:
int: The nth Fibonacci number.
"""
# Base case: The first and second Fibonacci numbers are both 1
if n <= 0:
raise ValueError("n must be a positive integer.")
if n == 1 or n == 2:
return 1
Recursive case: Sum of the two preceding Fibonacci numbers
return fibonacci(n - 1) + fibonacci(n - 2)
Example usage:
print(fibonacci(5)) # Output: 5
print(fibonacci(10)) # Output: 55
Code Explanation Document:
The function fibonacci(n) calculates the nth Fibonacci number using recursion.
It checks for invalid input (n <= 0) and raises an error.
For n == 1 or n == 2, it returns 1 (base cases).
For other values, it recursively calls itself to compute the sum of the previous two Fibonacci numbers.
Example usage demonstrates how to call the function and shows expected outputs.
Add Context...
task3.py
Add context (#), extensions (@), commands (/)
Ask
GPT-4.1
Ln 21, Col 34
Spaces: 4
UTF-8
{} Python
3.13.5 (base)
Go Live

TASK DESCRIPTION 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 1: generate and create a job applicant scoring system that evaluates candidates based on input features such as education, experience, gender, and age. Include clear explanations of how each feature contributes to the score. After generating the system, analyze whether the scoring logic shows any bias or assigns unfair weightings to certain features like gender or age.



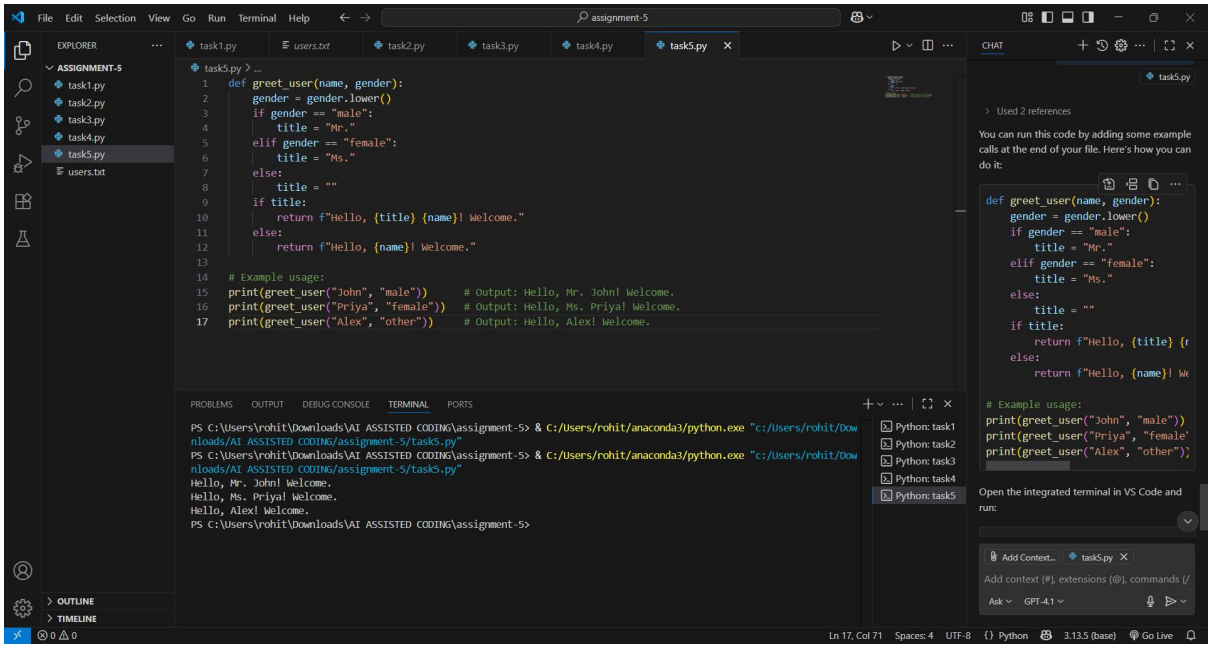


TASK DESCRIPTION 5 : Code Snippet

```
def greet_user(name, gender):
    if gender.lower() == "male":
        title = "Mr."
    else:
        title = "Mrs."
    return f"Hello, {title} {name}! Welcome."
```

Regenerate code that includes **gender-neutral** also

PROMPT 1: generate a Python function greet user(name, gender) that greets a user with a title based on their gender. If the gender is male, use ‘Mr.’, if female use ‘Ms.’, and if any other value, greet without a title. Include example usage showing greetings for male, female, and other cases.



OBSERVATION: This assignment focuses on using AI tools for coding while emphasizing security, fairness, and inclusivity. Key points include:

- Generating and reviewing a login system for security flaws like hardcoded passwords and lack of encryption.
- Testing loan approval logic with different names to detect bias based on gender or identity.
- Writing a recursive Fibonacci function with clear comments and documentation.
- Creating a job applicant scoring system and analyzing it for biased or unfair scoring due to features like gender or age.
- Generating a gender-neutral greeting function that adapts based on user input.

The assignment builds skills in prompt writing, AI code review, ethical evaluation, and writing clear, inclusive code.