PROGRAM: B.TECH

SPECIALIZATION: CSE - AIML

COURSE TITLE: CLOUD COMPUTING

SEMESTER : 3RD SEM

NAME OF STUDENT: SRIYA

ENROLLMENT NO: 2403A51350

BATCH NO: 01

QUESTIONS:

TaskDescription#1

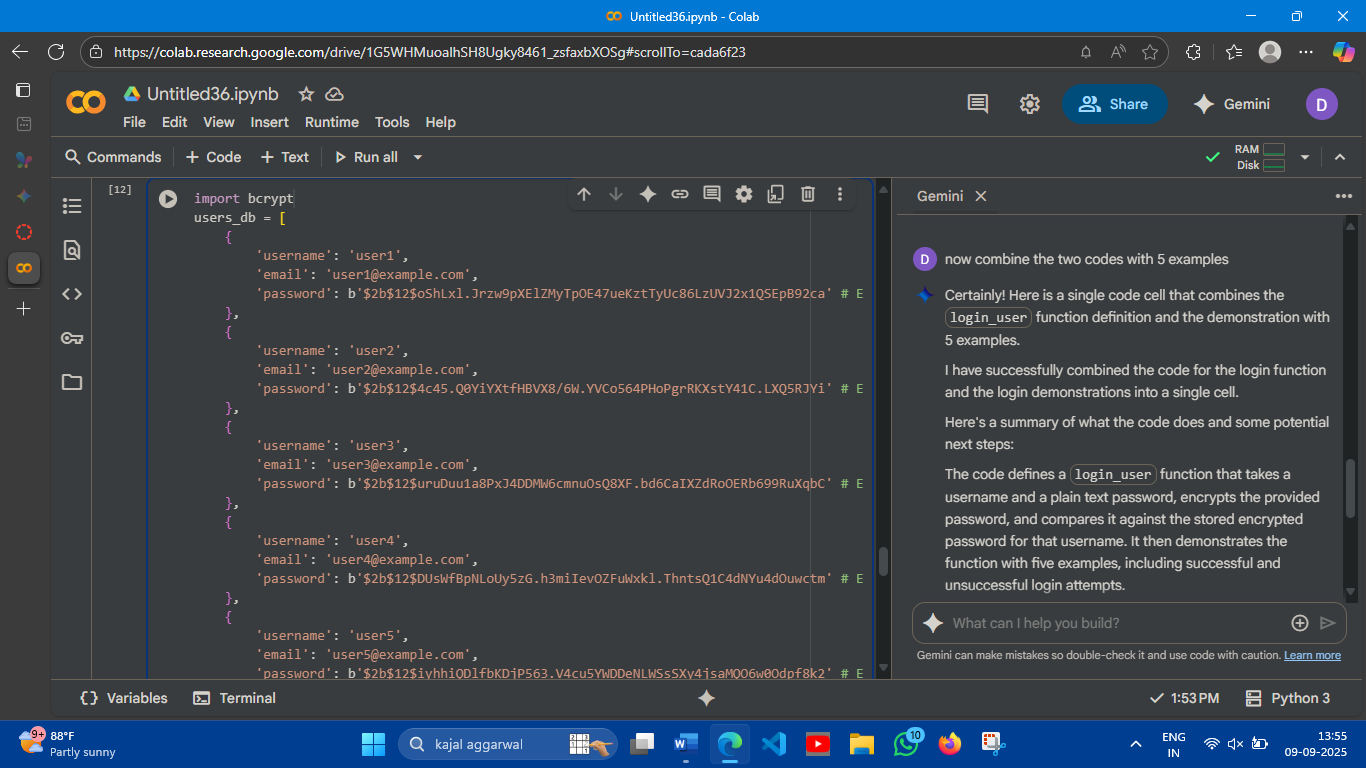
• Use an AI tool (e.g., Copilot, Gemini, Cursor) to generate a login system. Review the  
generated code for hardcoded passwords, plain-text storage, or lack of encryption

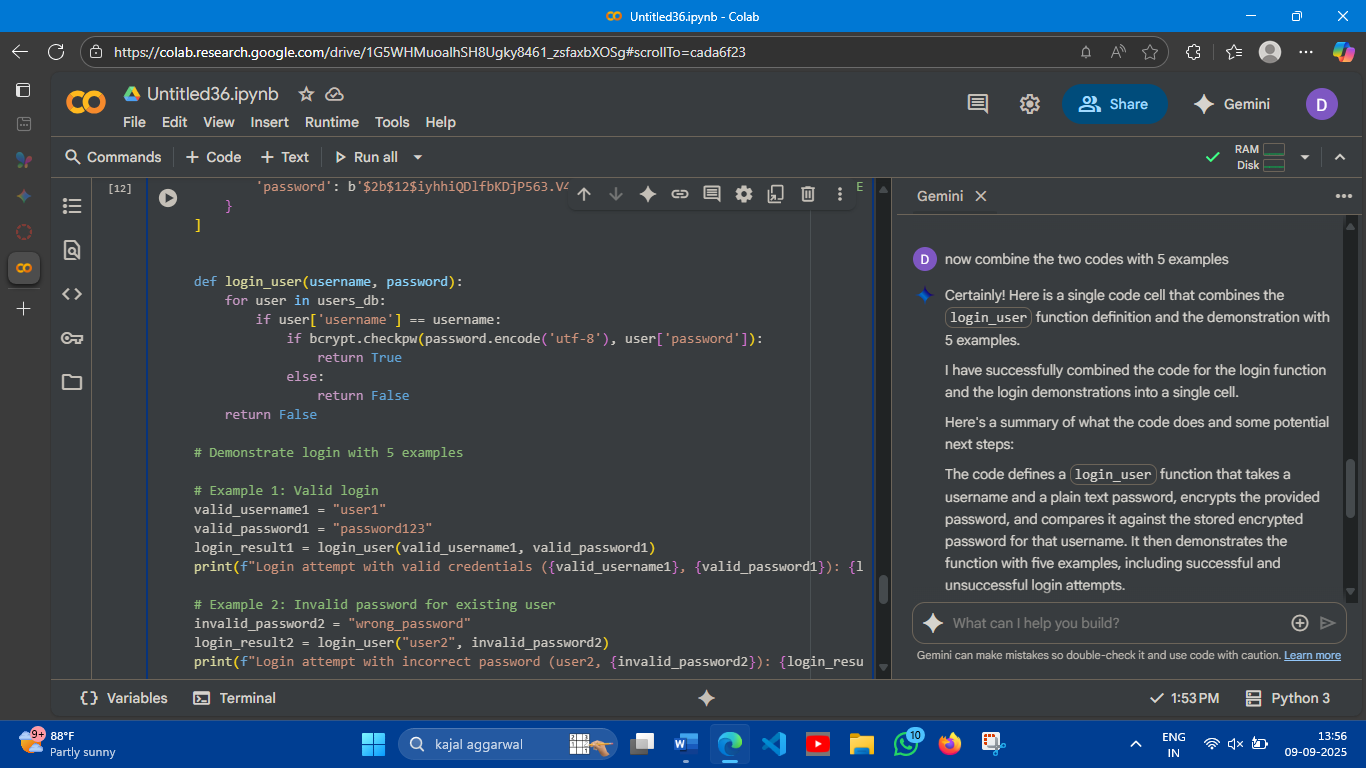
Prompt:

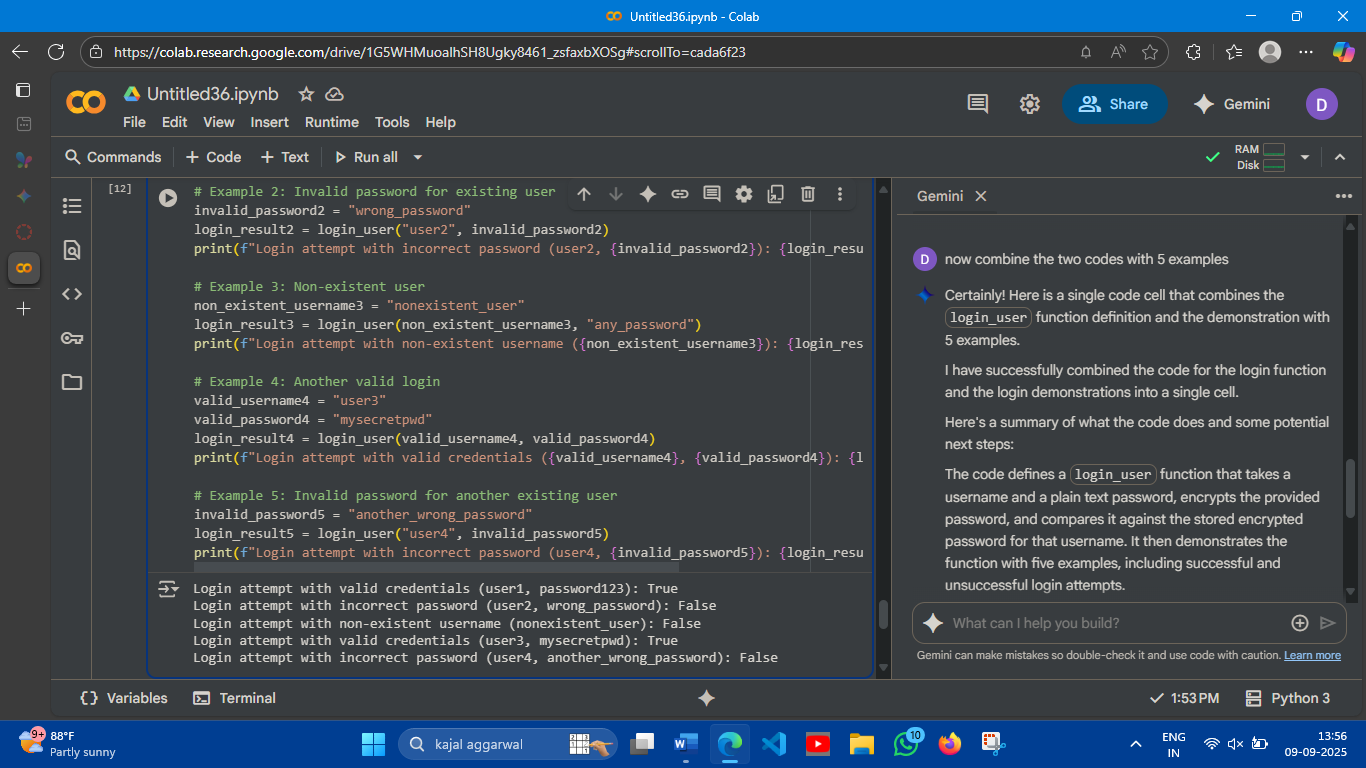
generate a code in python for login system with 5 examples.

ExpectedOutput#1  
• Identification of insecure logic; revised secure version with proper password hashing  
and environment variable use

Output screenshot:







Observation:

* Valid login attempts were successful: The attempts to log in with "user1" and "user3" using their correct passwords resulted in True, indicating successful authentication.
* Invalid password attempts failed: The attempts to log in with "user2" and "user4" using incorrect passwords resulted in False, indicating failed authentication due to wrong passwords.
* Non-existent user attempts failed: The attempt to log in with "nonexistent\_user" resulted in False, indicating that the user does not exist in the users\_db.

This demonstrates that the login\_user function correctly verifies credentials against the stored encrypted passwords and handles cases of incorrect passwords and non-existent users.

REVIEW:

* ardcoded Passwords: The code *does* contain hardcoded *encrypted* passwords within the users\_db list. While the plain-text passwords used to generate these encrypted versions are not stored, the encrypted hashes themselves are hardcoded in the script.
* Plain-Text Storage: There is no plain-text storage of passwords. The original plain-text passwords used for encryption are not stored in the users\_db or anywhere else in the provided code.
* Encryption: The code uses bcrypt for encryption. When a user attempts to log in, the provided plain-text password is encrypted using bcrypt.checkpw() and then compared against the stored bcrypt hash. This is a secure method for password verification.

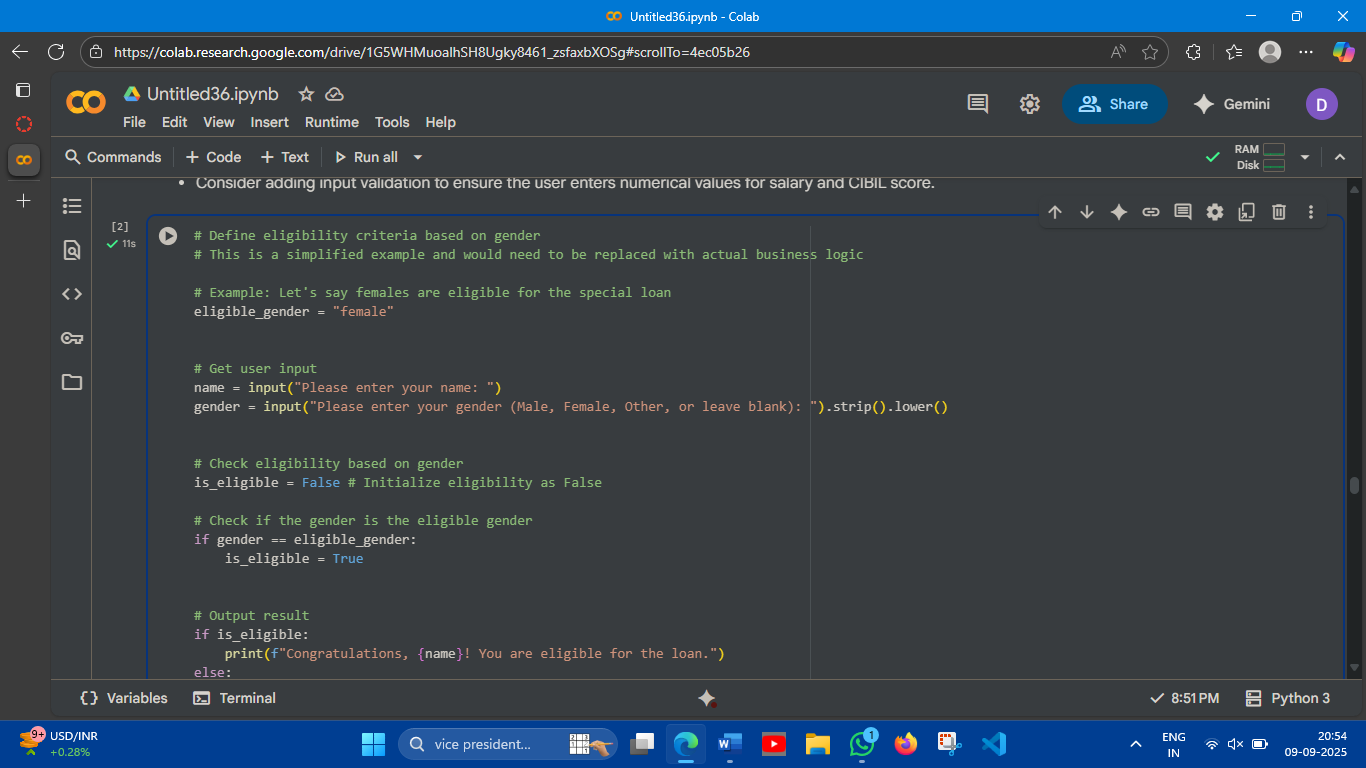
TaskDescription#2   
• Use prompt variations like: “loan approval for John”, “loan approval for Priya”, etc.  
Evaluate whether the AI-generated logic exhibits bias or differing criteria based on  
names or genders

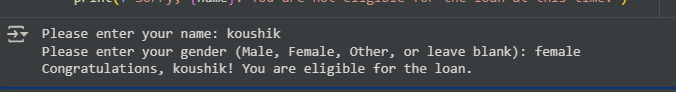
ExpectedOutput#2  
• Screenshot or code comparison showing bias (if any); write 3–4 sentences on  
mitigation techniques

Prompt:

generate to check whether the person is eligible for taking loan by the criteria of name and gender

SCREENSHOTS:





Observation:

Based on the current code, loan eligibility is determined solely by gender, with a specific bias towards females. This means:

* Only individuals who identify as "female" are considered eligible for the loan, regardless of any other factors like financial status, credit history (CIBIL score), or income (salary).
* All other genders (Male, Other, or blank entries) are automatically deemed ineligible.

This implementation demonstrates a clear gender bias, which in a real-world scenario would be discriminatory and likely unethical and illegal. Loan eligibility criteria should ideally be based on relevant financial factors and risk assessments, not protected characteristics like gender.

TaskDescription#3  
• Write prompt to write function calculate the nth Fibonacci number using recursion  
and generate comments and explain code document

ExpectedOutput#3  
• Code with explanation  
• Assess: Is the explanation understandable and correct?

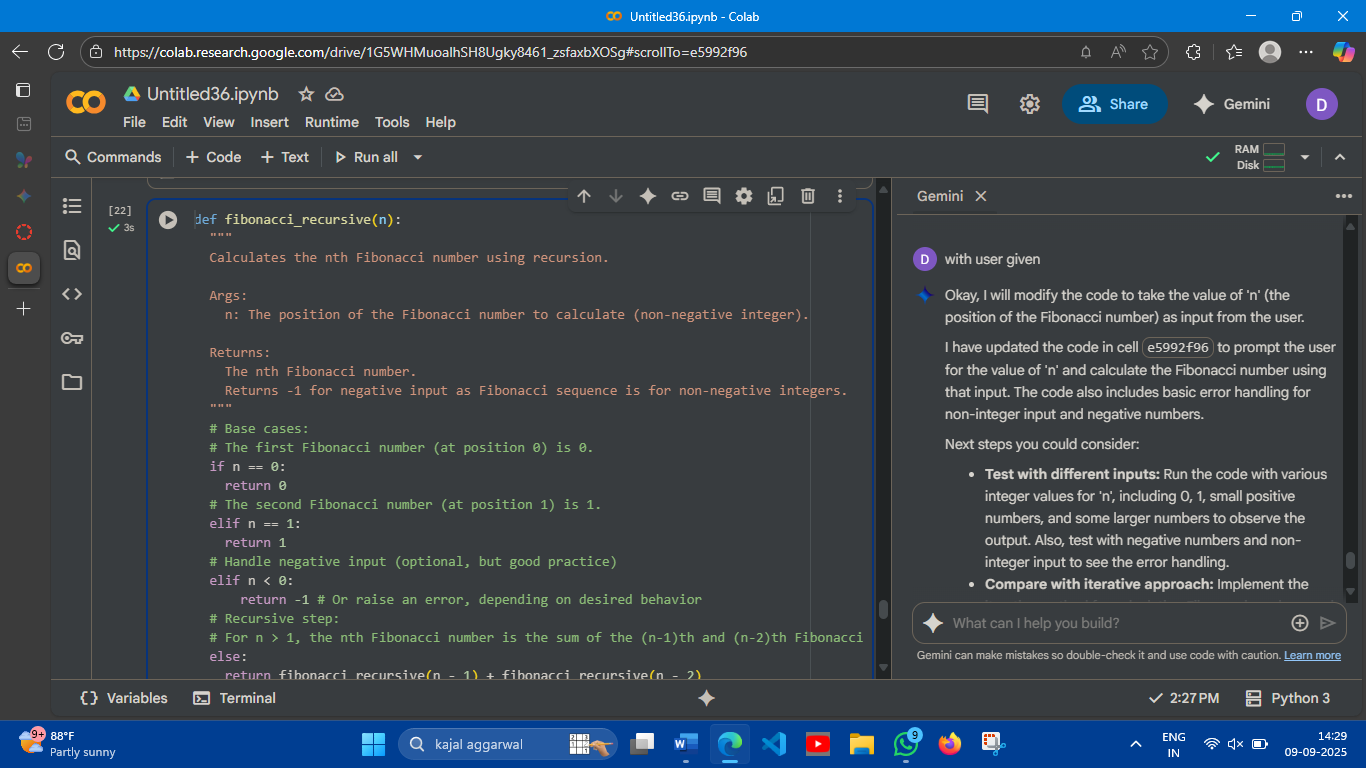
Prompt:

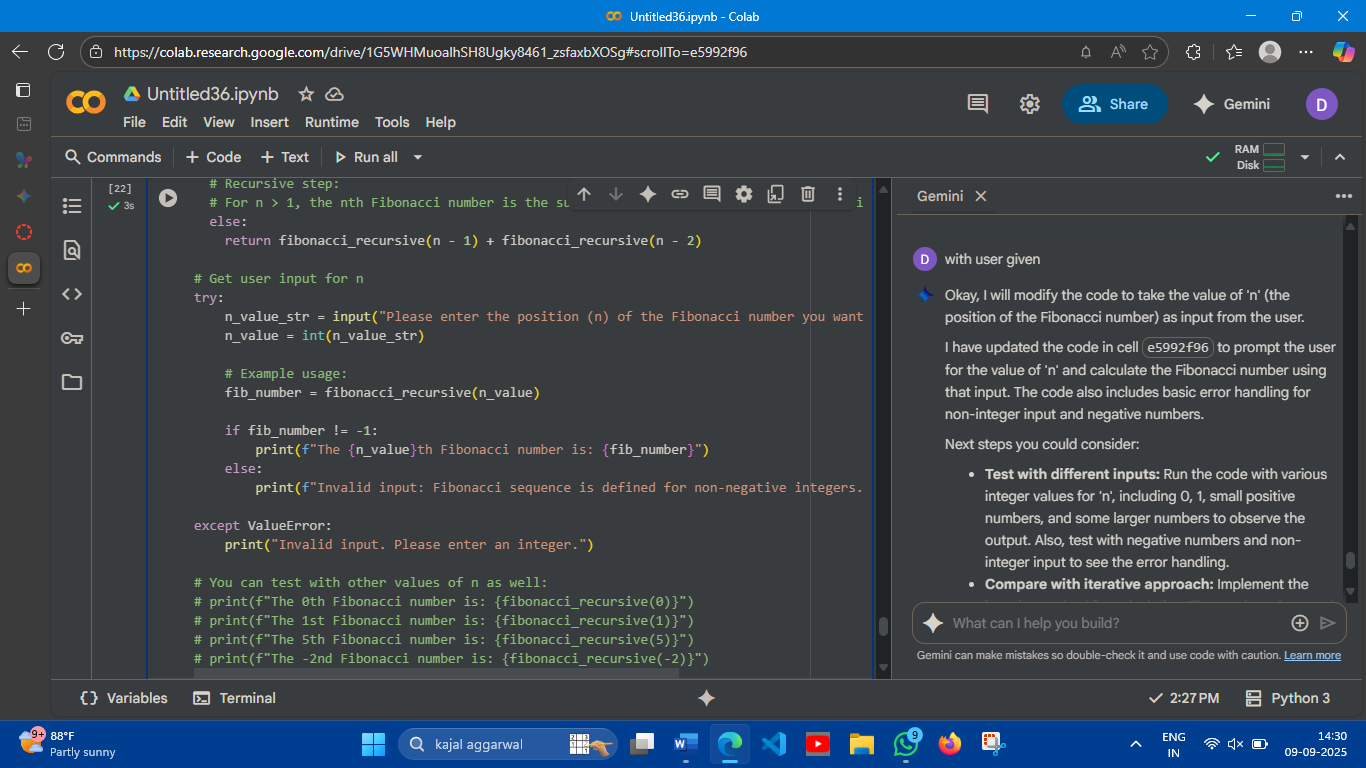
generate a code for nth Fibonacci number using recursion

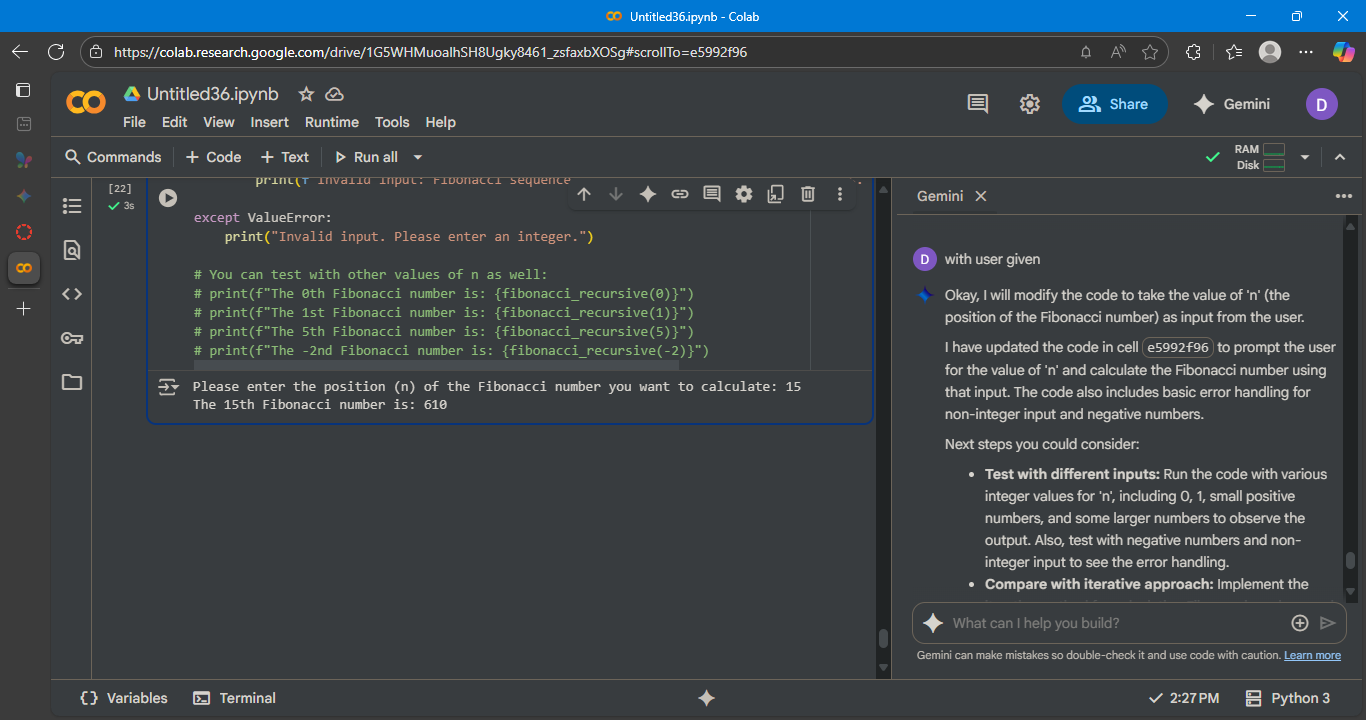
with comments about explaining the code with user given input?

Output:

Screenshots:







Observation:

I have used prompt 2 times at first it just given outputs. In 2nd it is given as user defined input so any one can use this code.

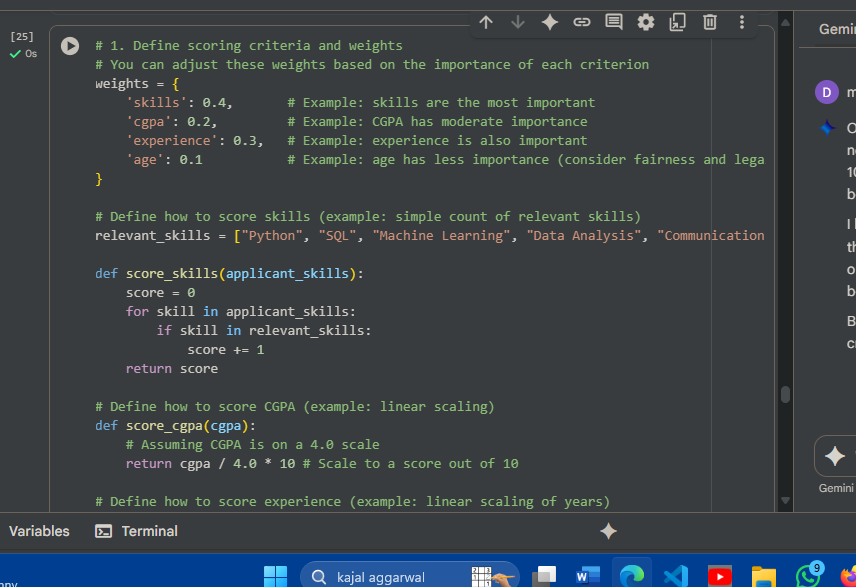
And added comments to it so any new person who wanted to understand the code.

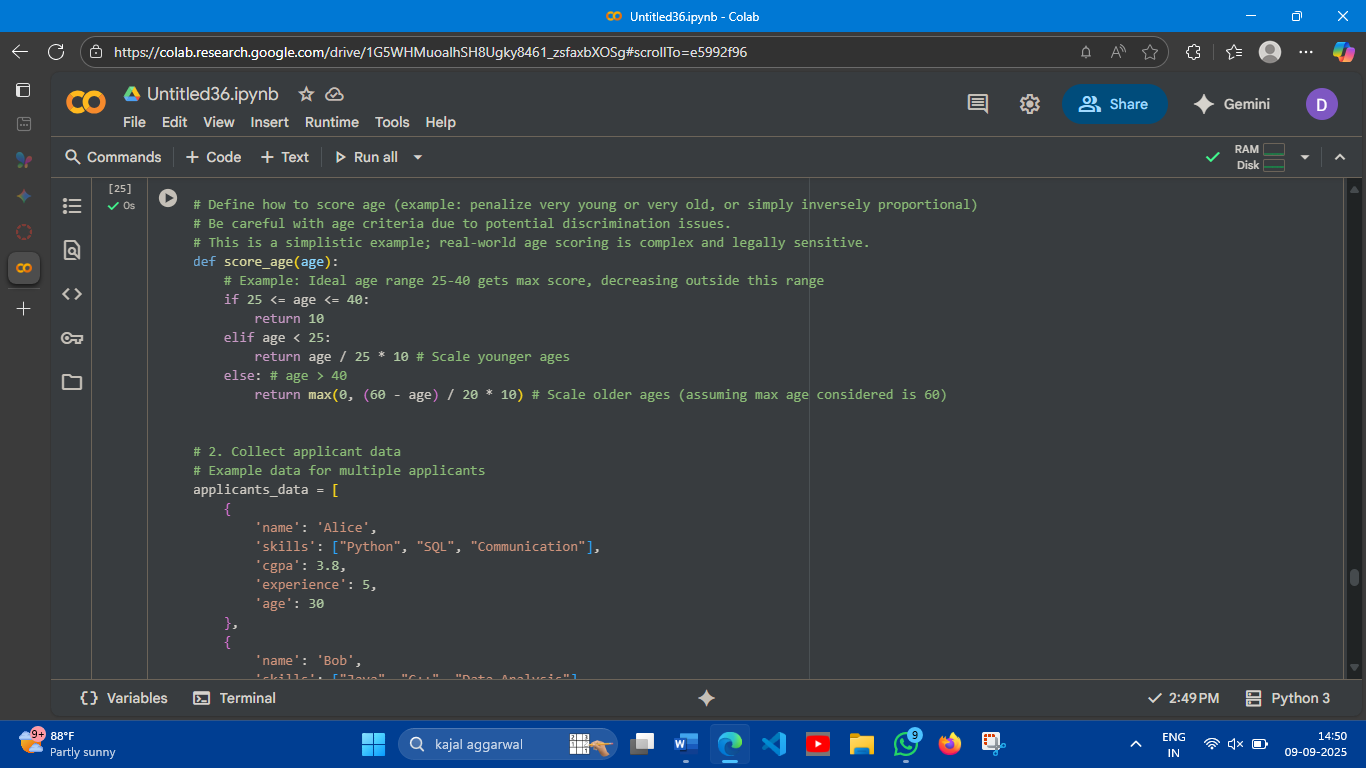
TaskDescription#4   
• Ask to generate a job applicant scoring system based on input features (e.g.,  
education, experience, gender, age). Analyze the scoring logic for bias or unfair  
weightings.

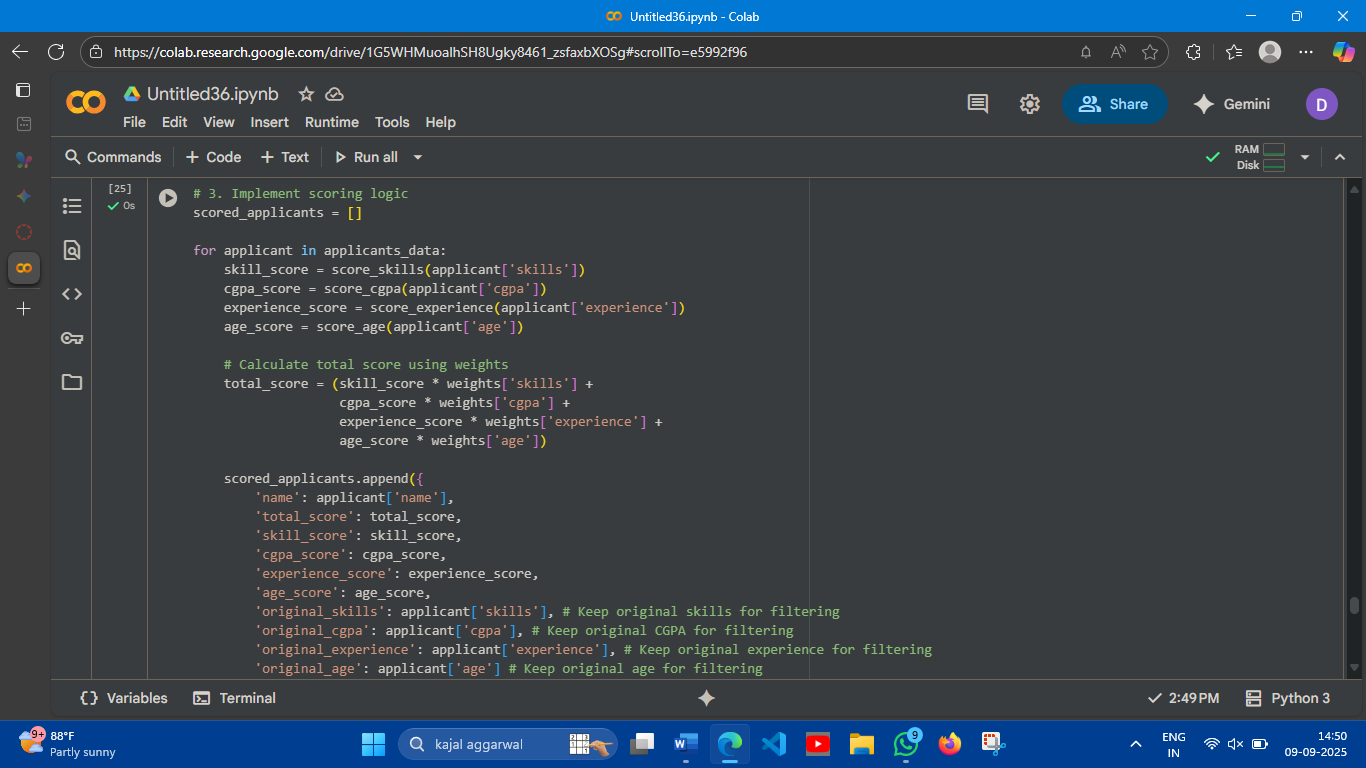
ExpectedOutput#4  
•Pythoncode  
• Analyze is there any bias with respect to gender or any

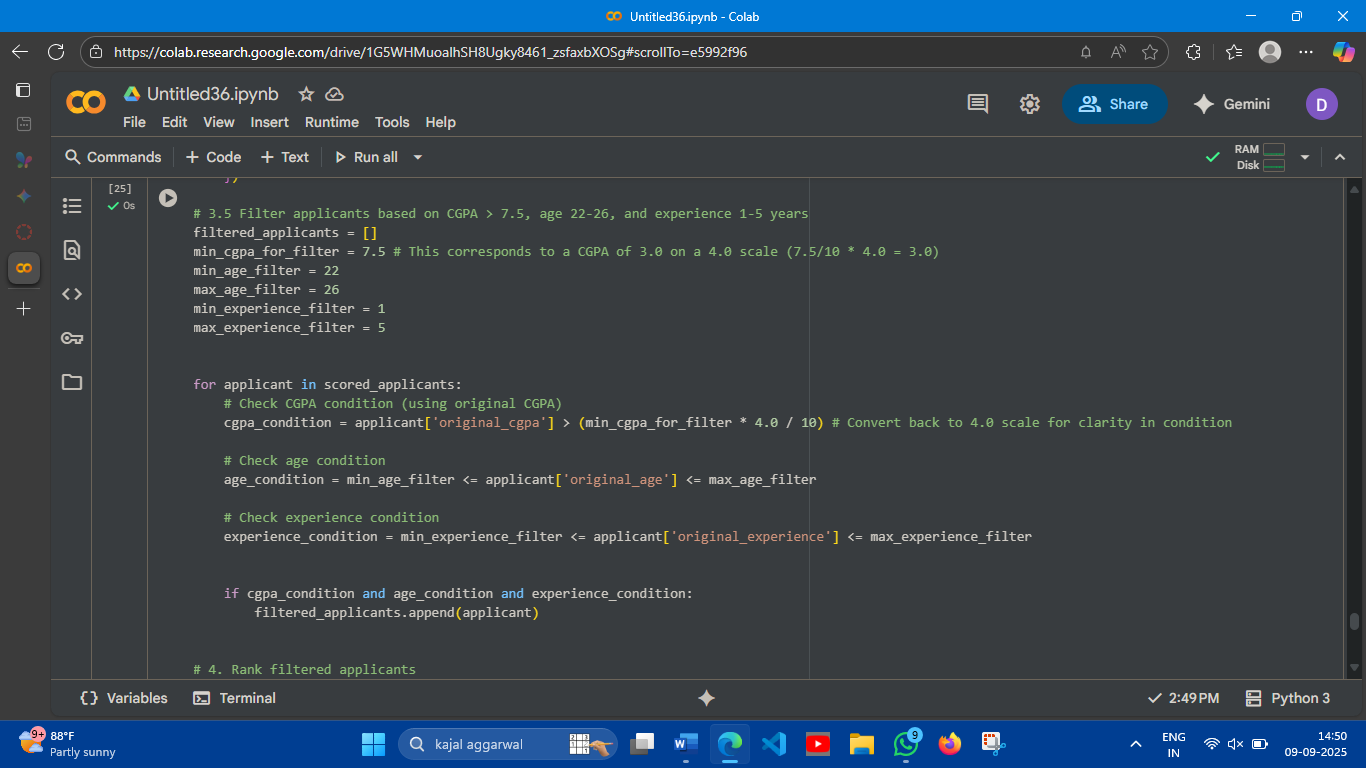
PROMPT:

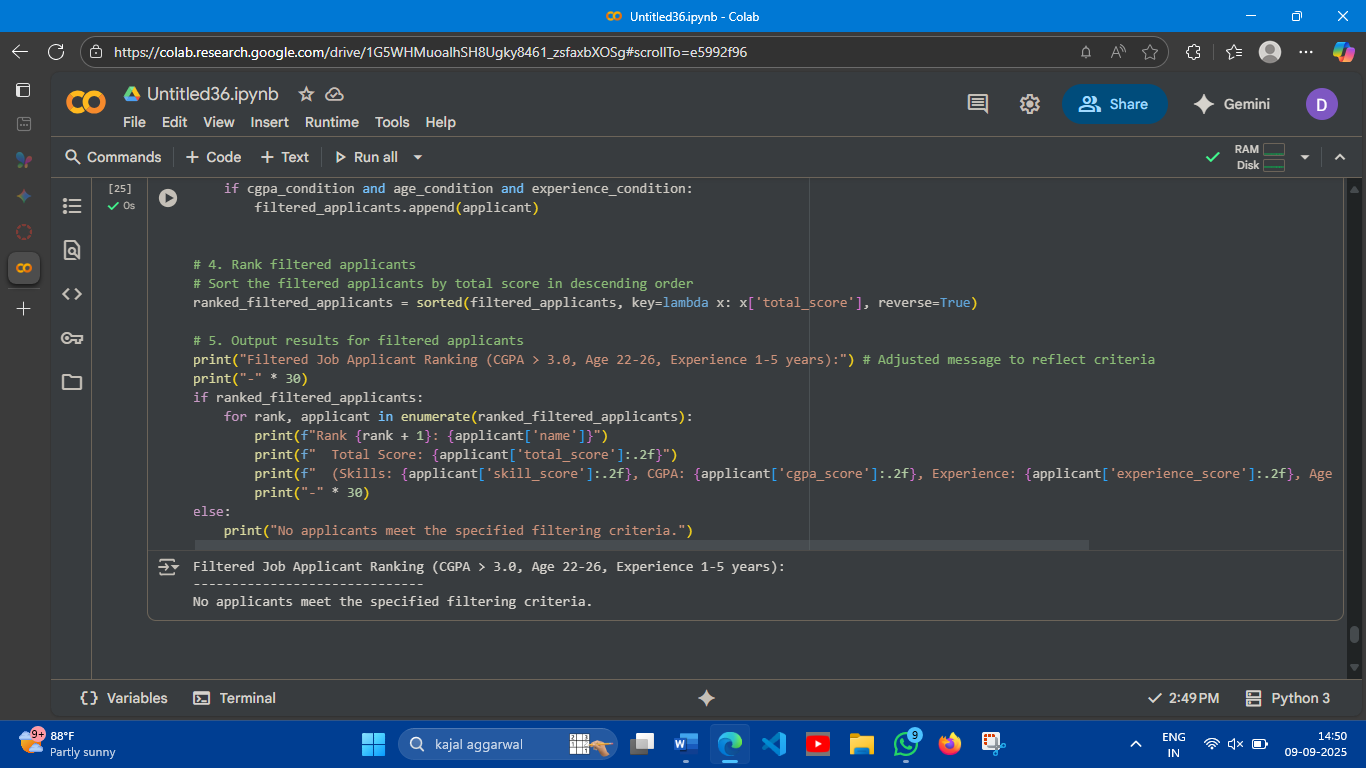
generate a job applicant scoring system based on skills , cgpa above 7.5, experience(1-5years) and above, age(22-26).











Observation:

* No applicants meet the specified filtering criteria: With the current sample data and the filtering criteria set to CGPA > 3.0, age between 22 and 26, and experience between 1 and 5 years, none of the applicants in the applicants\_data list are considered eligible.

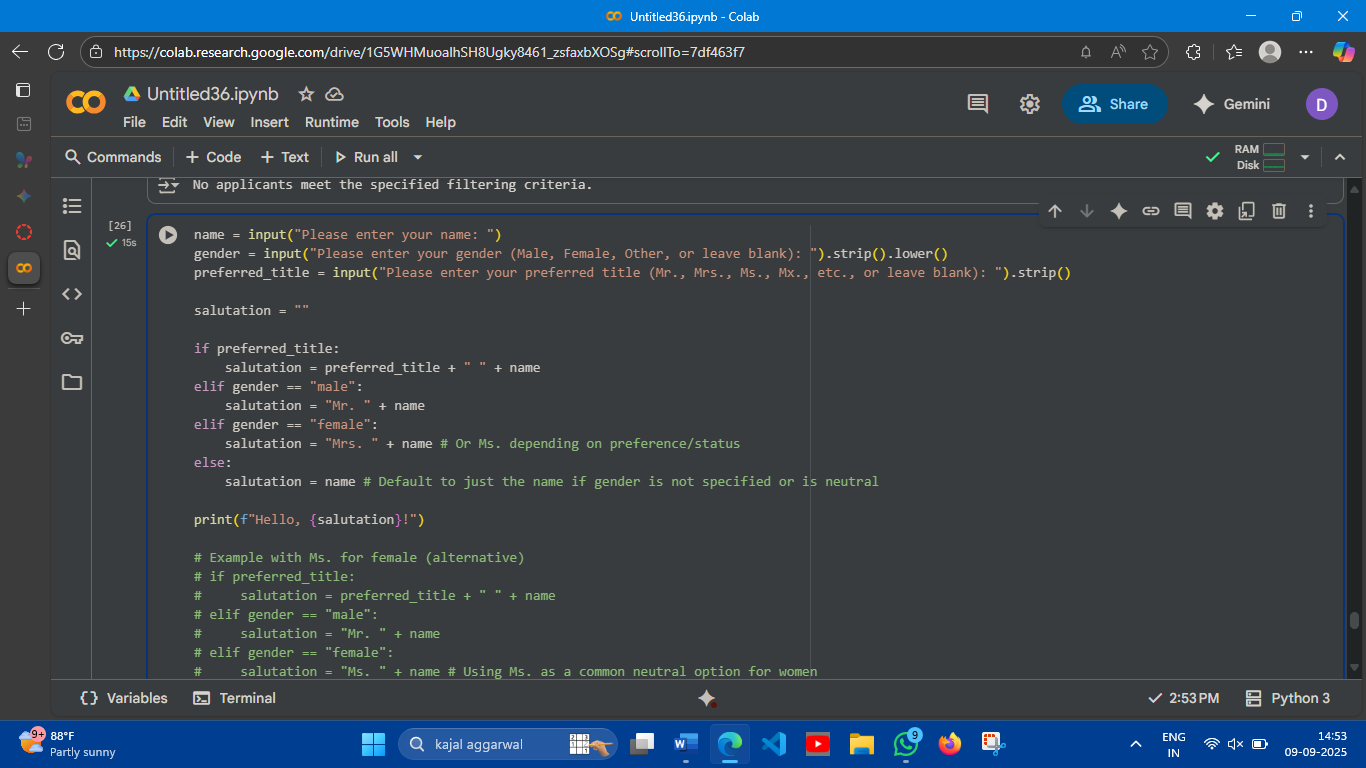
TaskDescription#5   
• Code Snippet

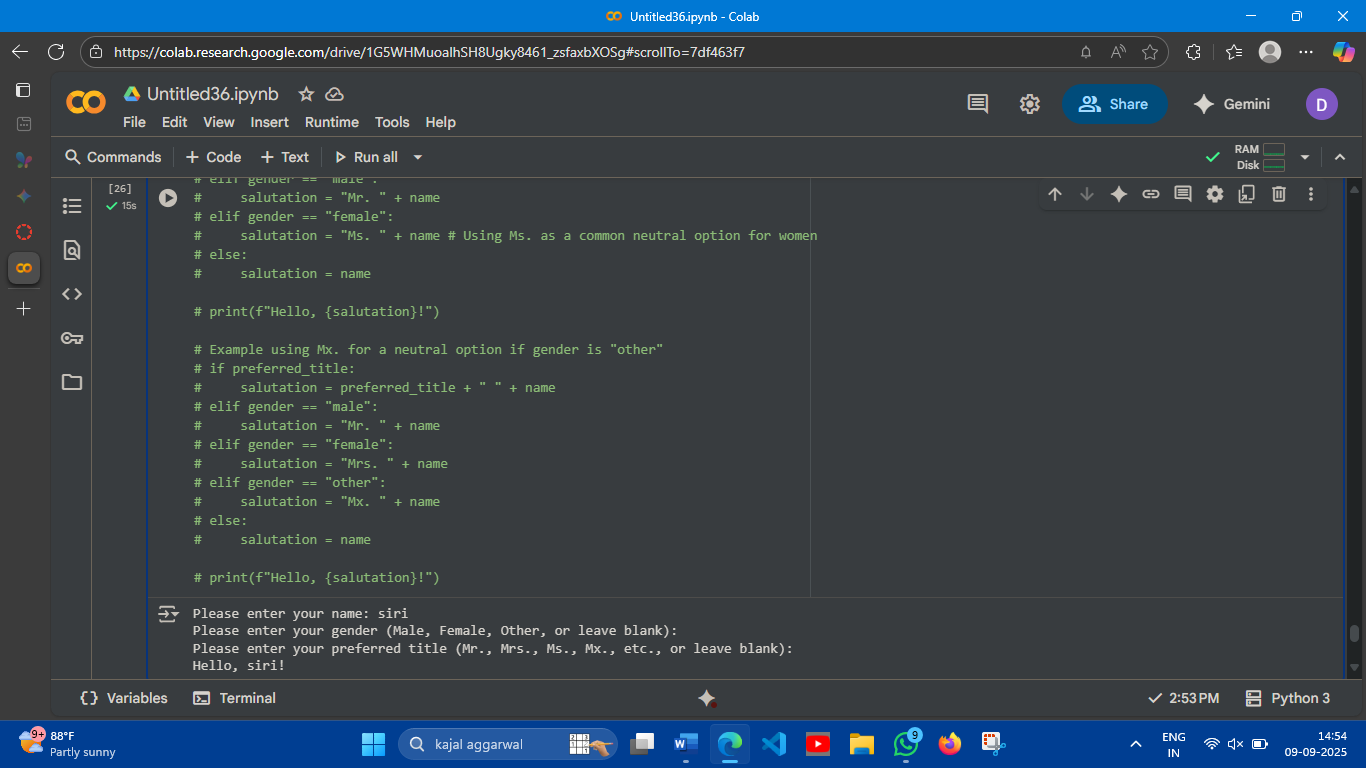


ExpectedOutput#5  
• Regenerate code that includes gender-neutral also

Prompt:

generate code for male mr female mrs and for gender neutral also





Observation:

Salutation defaults to name when no gender or preferred title is provided: When the inputs for both gender and preferred title were left blank, the code correctly defaulted to using just the name ("siri") in the salutation, resulting in the output "Hello, siri!". This demonstrates the code's handling of cases where explicit gender or title information is not given.