School of Computer Science and Artificial Intelligence

Lab Assignment-04

Course Title : AI Assistant Coding
Name of Student : Shaik Naved Ahmed
Enrollment No. : 2303A54053
Batch No. : 47-B

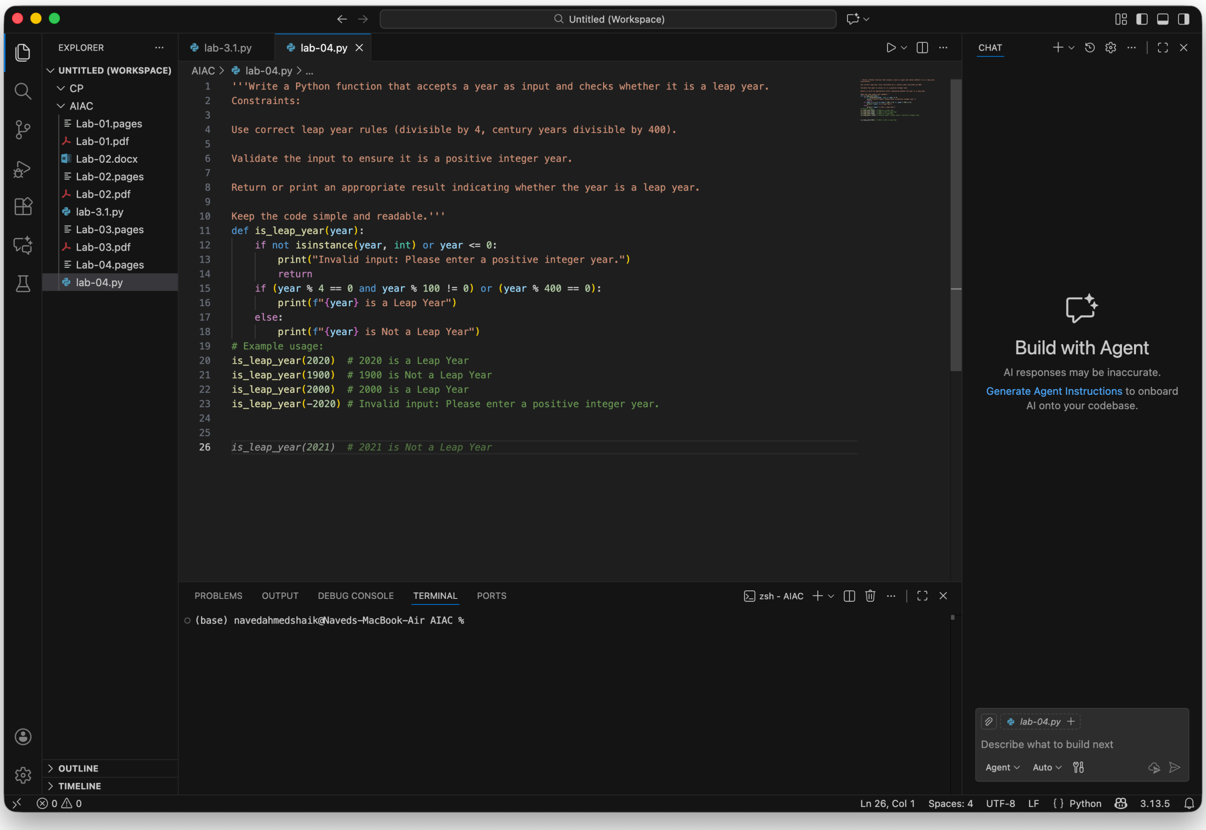
QUESTION 1: Zero-Shot Prompting – Leap Year Check

PROMPT:

Write a Python function that accepts a year as input and checks whether it is a **leap year**.
Constraints:

- Use correct leap year rules (divisible by 4, century years divisible by 400).
- Validate the input to ensure it is a positive integer year.
- Return or print an appropriate result indicating whether the year is a leap year.
- Keep the code simple and readable.

SCREENSHOT OF GENERATED CODE:



```python
'''Write a Python function that accepts a year as input and checks whether it is a leap year.
Constraints:

Use correct leap year rules (divisible by 4, century years divisible by 400).

Validate the input to ensure it is a positive integer year.

Return or print an appropriate result indicating whether the year is a leap year.

Keep the code simple and readable.'''
def is_leap_year(year):
    if not isinstance(year, int) or year <= 0:
        print("Invalid input: Please enter a positive integer year.")
        return
    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
        print(f"{year} is a Leap Year")
    else:
        print(f"{year} is Not a Leap Year")
# Example usage:
is_leap_year(2020)  # 2020 is a Leap Year
is_leap_year(1900)  # 1900 is Not a Leap Year
is_leap_year(2000)  # 2000 is a Leap Year
is_leap_year(-2020) # Invalid input: Please enter a positive integer year.


is_leap_year(2021)  # 2021 is Not a Leap Year
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

Untitled (Workspace)
lab-3.1.py
lab-04.py
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Describe what to build next
Ln 26, Col 1 Spaces: 4 UTF-8 LF Python 3.13.5

INPUT & OUTPUT:

Input	Output	Reason
2020	2020 is a Leap Year	Divisible by 4 and not divisible by 100.
1900	1900 is Not a Leap Year	Divisible by 100 but not divisible by 400.
2000	2000 is a Leap Year	Divisible by 400, so it is a leap year.
-2020	Invalid input: Please enter a positive integer year.	Year must be a positive integer.

EXPLANATION:

The function first validates the input to ensure it is a positive integer; if not, it prints an error message and exits. It then applies the standard leap year rules: a year is a leap year if it is divisible by 4 but not divisible by 100, or if it is divisible by 400. Based on these conditions, the function prints whether the given year is a Leap Year or Not a Leap Year.

Question 2: One-Shot Prompting – Centimeters to Inches Conversion

PROMPT:

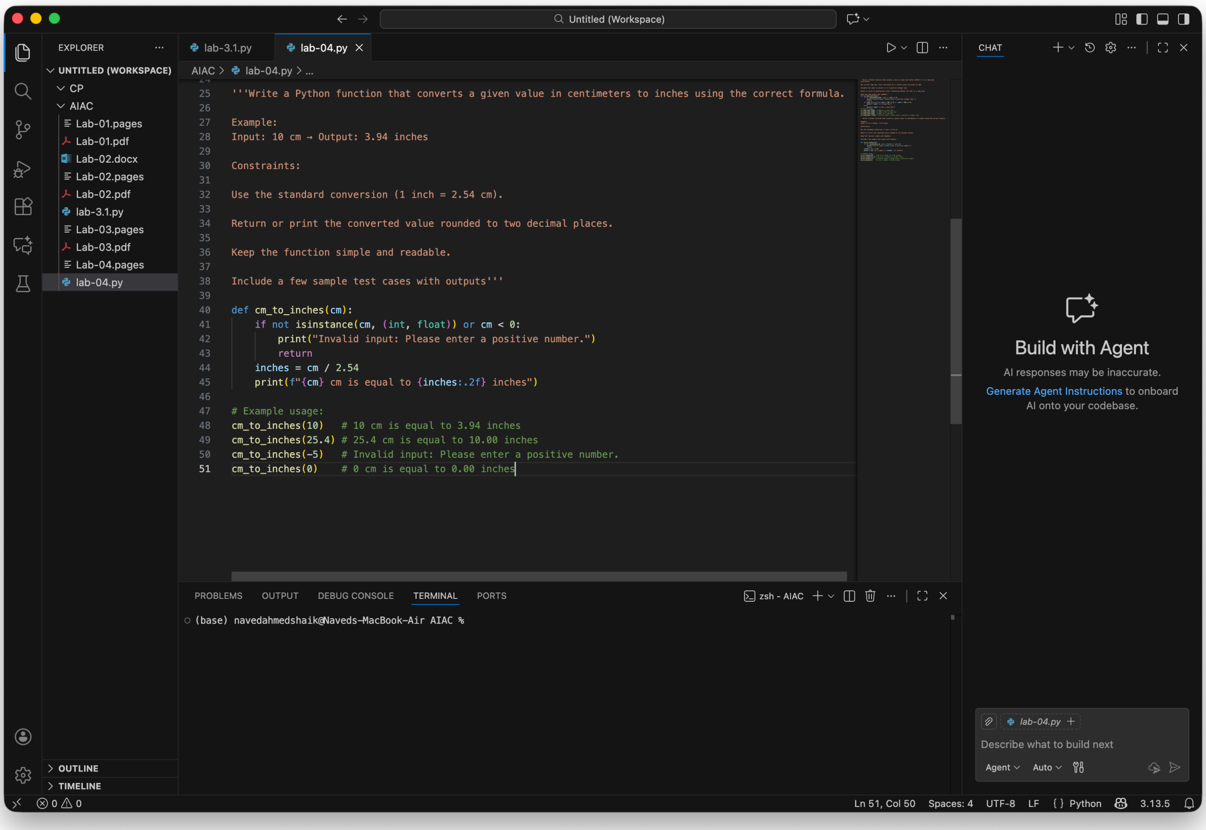
Write a Python function that converts a given value in **centimeters** to **inches** using the correct formula.

Example:
Input: 10 cm → Output: 3.94 inches

Constraints:

- Use the standard conversion (1 inch = 2.54 cm).
- Return or print the converted value rounded to two decimal places.
- Keep the function simple and readable.
- Include a few sample test cases with outputs

SCREENSHOT OF GENERATED CODE:



```python
'''Write a Python function that converts a given value in centimeters to inches using the correct formula.

Example:
Input: 10 cm → Output: 3.94 inches

Constraints:

Use the standard conversion (1 inch = 2.54 cm).

Return or print the converted value rounded to two decimal places.

Keep the function simple and readable.

Include a few sample test cases with outputs'''

def cm_to_inches(cm):
    if not isinstance(cm, (int, float)) or cm < 0:
        print("Invalid input: Please enter a positive number.")
        return
    inches = cm / 2.54
    print(f"{cm} cm is equal to {inches:.2f} inches")

# Example usage:
cm_to_inches(10)   # 10 cm is equal to 3.94 inches
cm_to_inches(25.4) # 25.4 cm is equal to 10.00 inches
cm_to_inches(-5)   # Invalid input: Please enter a positive number.
cm_to_inches(0)    # 0 cm is equal to 0.00 inches
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

INPUT & OUTPUT:

Input	Output	Reason
10	10 cm is equal to 3.94 inches	Converted using 10 ÷ 2.54 ≈ 3.94.
25.4	25.4 cm is equal to 10.00 inches	25.4 cm equals exactly 10 inches.
-5	Invalid input: Please enter a positive number.	Negative values are not valid for length.
0	0 cm is equal to 0.00 inches	Zero is a valid input and converts to zero inches.

EXPLANATION:

The function converts a length from centimeters to inches. It first validates the input to ensure it is a non-negative number (integer or float). If the input is invalid or negative, it prints an error message and stops execution. For valid inputs, it converts centimeters to inches using the standard conversion factor (1 inch = 2.54 cm) and prints the result rounded to two decimal places.

Question 3: Few-Shot Prompting – Name Formatting

PROMPT:

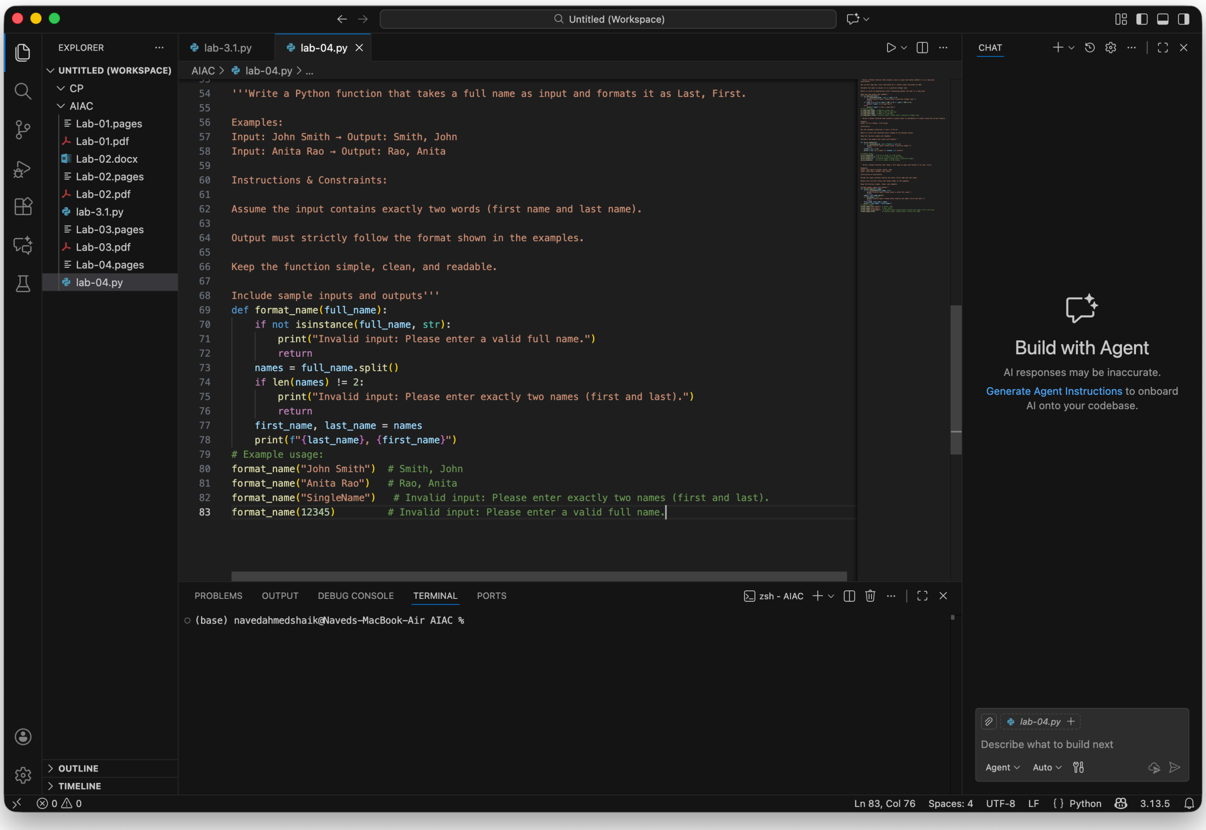
Write a Python function that takes a full name as input and formats it as **Last, First**.

Examples:
Input: John Smith → Output: Smith, John
Input: Anita Rao → Output: Rao, Anita

Instructions & Constraints:

- Assume the input contains exactly two words (first name and last name).
- Output must strictly follow the format shown in the examples.
- Keep the function simple, clean, and readable.
- Include sample inputs and outputs

SCREENSHOT OF GENERATED CODE:



```python
'''Write a Python function that takes a full name as input and formats it as Last, First.

Examples:
Input: John Smith → Output: Smith, John
Input: Anita Rao → Output: Rao, Anita

Instructions & Constraints:

Assume the input contains exactly two words (first name and last name).

Output must strictly follow the format shown in the examples.

Keep the function simple, clean, and readable.

Include sample inputs and outputs'''
def format_name(full_name):
    if not isinstance(full_name, str):
        print("Invalid input: Please enter a valid full name.")
        return
    names = full_name.split()
    if len(names) != 2:
        print("Invalid input: Please enter exactly two names (first and last).")
        return
    first_name, last_name = names
    print(f"{last_name}, {first_name}")
# Example usage:
format_name("John Smith")  # Smith, John
format_name("Anita Rao")   # Rao, Anita
format_name("SingleName")   # Invalid input: Please enter exactly two names (first and last).
format_name(12345)         # Invalid input: Please enter a valid full name.
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

INPUT & OUTPUT:

Input	Output	Reason
John Smith	Smith, John	Input contains exactly two names and is formatted correctly.
Anita	Rao, Anita	Valid first and last name, order is reversed
SingleName	Invalid input: Please enter exactly two names (first and last).	Only one name is provided instead of two.
12345	Invalid input: Please enter a valid full	Input is not a string.

EXPLANATION:

The function checks whether the input is a valid string representing a full name. It then splits the string into parts based on spaces. If the input does not contain exactly two words (a first name and a last name), the function prints an error message. When the input is valid, it rearranges the names and prints them in the format “LastName, FirstName”

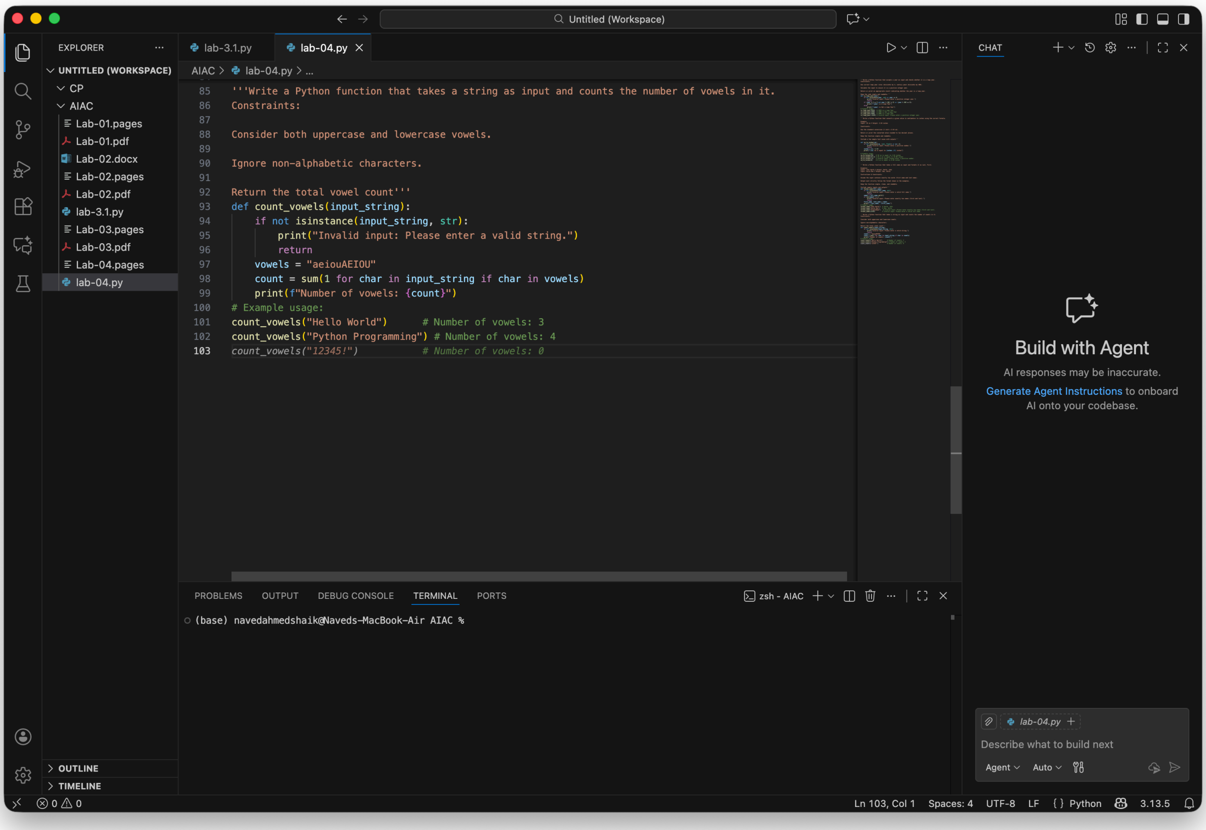
Question 4: Comparative Analysis – Zero-Shot vs Few-Shot

Zero-Shot PROMPT:

Write a Python function that takes a string as input and counts the number of vowels in it.
Constraints:

- Consider both uppercase and lowercase vowels.
- Ignore non-alphabetic characters.
- Return the total vowel count.

Zero-Shot SCREENSHOT OF GENERATED CODE:



```python
'''Write a Python function that takes a string as input and counts the number of vowels in it.
Constraints:

Consider both uppercase and lowercase vowels.

Ignore non-alphabetic characters.

Return the total vowel count'''
def count_vowels(input_string):
    if not isinstance(input_string, str):
        print("Invalid input: Please enter a valid string.")
        return
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    print(f"Number of vowels: {count}")
# Example usage:
count_vowels("Hello World")      # Number of vowels: 3
count_vowels("Python Programming") # Number of vowels: 4
count_vowels("12345!")           # Number of vowels: 0
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

Zero-Shot INPUT & OUTPUT:

Input	Output	Reason
"Hello World"	Number of vowels: 3	Vowels are **e, o, o**.
"Python Programming"	Number of vowels: 4	Vowels are **o, o, a, i**.

Zero-Shot EXPLANATION:

The function first checks whether the input is a valid string; if not, it prints an error message and stops execution. It defines a string containing all lowercase and uppercase vowels. Using a generator expression with the `sum()` function, it iterates through each character in the input string and counts how many characters are vowels. Finally, it prints the total number of vowels found in the string.

Few-Shot PROMPT:

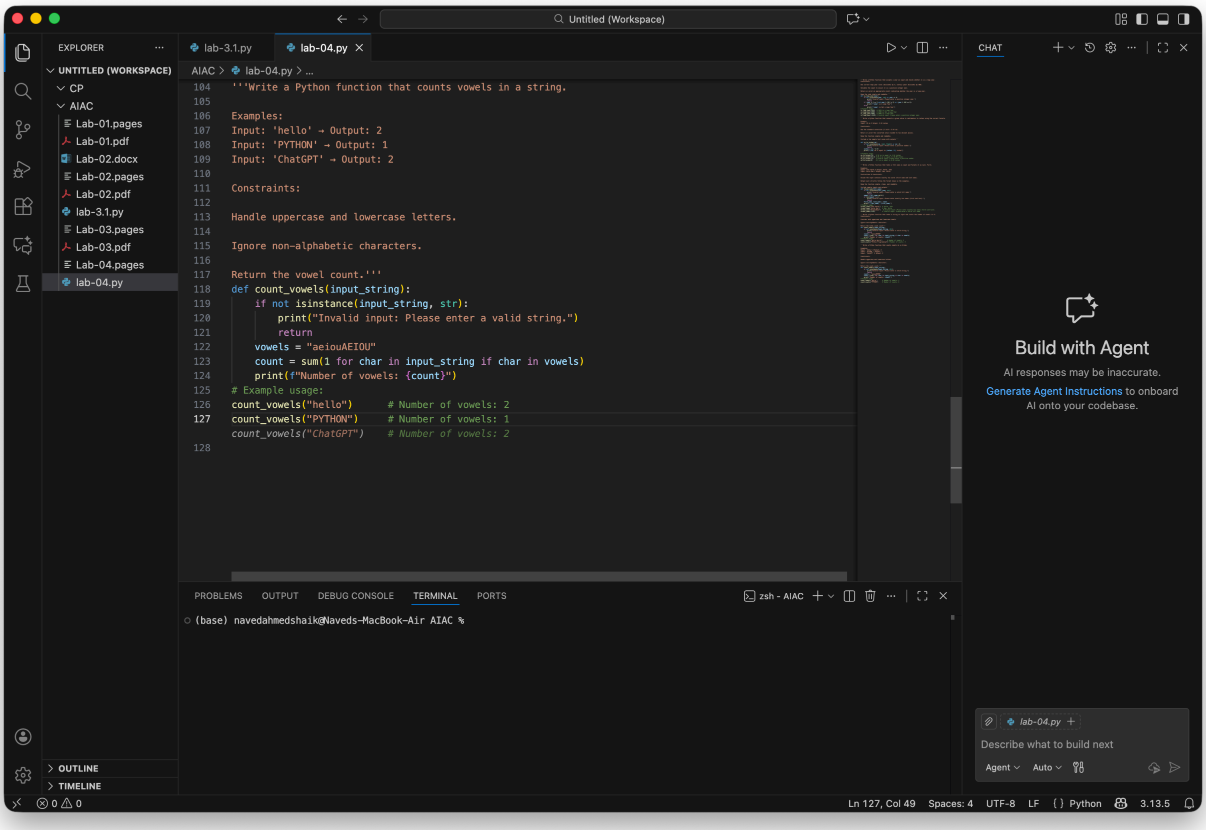
Write a Python function that counts vowels in a string.

Examples:
Input: 'hello' → Output: 2
Input: 'PYTHON' → Output: 1
Input: 'ChatGPT' → Output: 2

Constraints:

- Handle uppercase and lowercase letters.
- Ignore non-alphabetic characters.
- Return the vowel count.

Few-Shot SCREENSHOT OF GENERATED CODE:



```python
'''Write a Python function that counts vowels in a string.

Examples:
Input: 'hello' → Output: 2
Input: 'PYTHON' → Output: 1
Input: 'ChatGPT' → Output: 2

Constraints:

Handle uppercase and lowercase letters.

Ignore non-alphabetic characters.

Return the vowel count.'''
def count_vowels(input_string):
    if not isinstance(input_string, str):
        print("Invalid input: Please enter a valid string.")
        return
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    print(f"Number of vowels: {count}")
# Example usage:
count_vowels("hello")      # Number of vowels: 2
count_vowels("PYTHON")     # Number of vowels: 1
count_vowels("ChatGPT")    # Number of vowels: 2
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

Few-Shot INPUT & OUTPUT:

Input	Output	Reason
"hello"	Number of vowels: 2	Vowels **e** and **o** are present in the string.
"PYTHON"	Number of vowels: 1	Only **O** is a vowel; other letters are consonants.

Few-Shot EXPLANATION:

The function first verifies that the input is a string; if not, it prints an error message and exits. It then defines a list of vowels containing both lowercase and uppercase letters. Using a generator expression inside the sum() function, it iterates through each character in the string and counts how many of them are vowels. Finally, it prints the total number of vowels found in the given string

Zero-Shot vs Few-Shot Comparison:

Criteria	Zero-Shot	Few-Shot
Accuracy	Correct	Correct
Readability	Good	Better
Logical Clarity	Simple	Clearer due to examples
Handling Case Sensitivity	Explicit	Simplified using .lower()

Question 5: Few-Shot Prompting – File Handling

PROMPT:

Write a Python function that reads a .txt file and returns the number of lines.

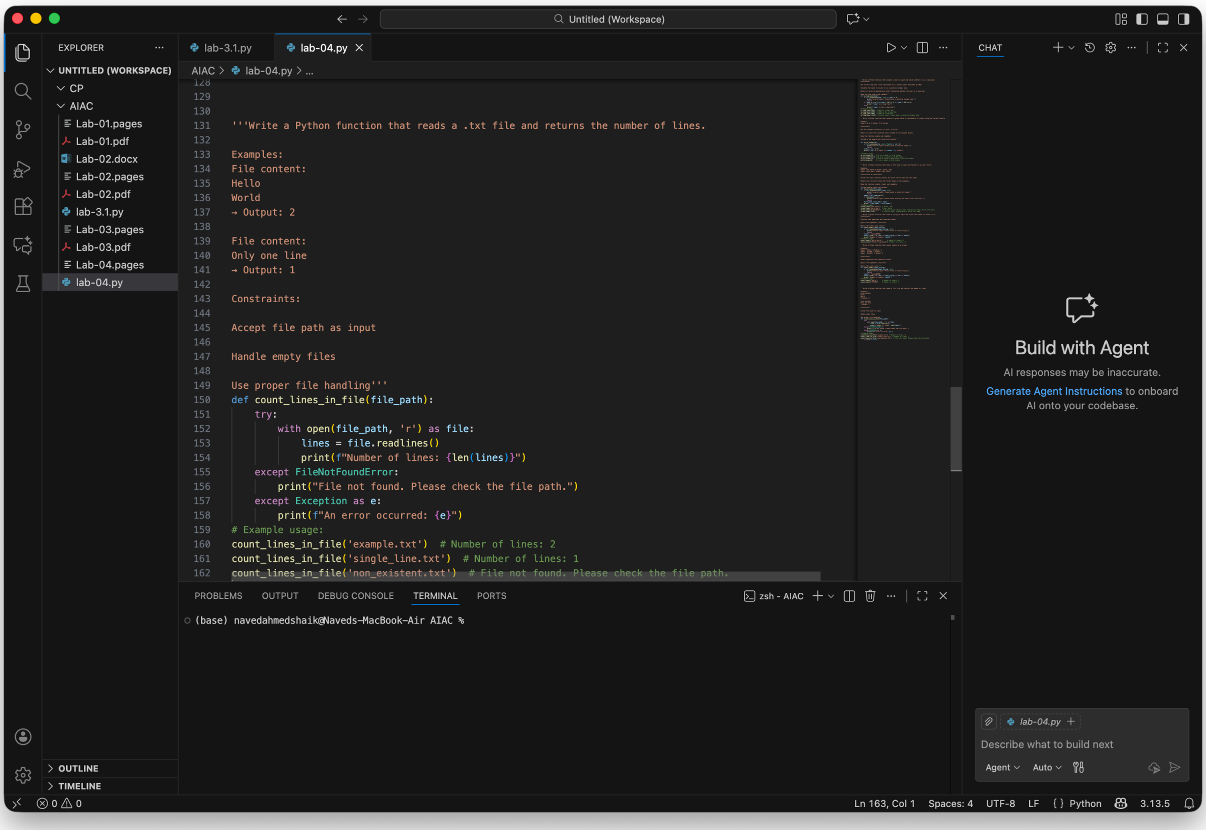
Examples:
File content:
Hello
World
→ Output: 2

File content:
Only one line
→ Output: 1

Constraints:

- Accept file path as input
- Handle empty files
- Use proper file handling

SCREENSHOT OF GENERATED CODE:



```python



'''Write a Python function that reads a .txt file and returns the number of lines.

Examples:
File content:
Hello
World
→ Output: 2

File content:
Only one line
→ Output: 1

Constraints:

Accept file path as input

Handle empty files

Use proper file handling'''
def count_lines_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            lines = file.readlines()
            print(f"Number of lines: {len(lines)}")
    except FileNotFoundError:
        print("File not found. Please check the file path.")
    except Exception as e:
        print(f"An error occurred: {e}")
# Example usage:
count_lines_in_file('example.txt')  # Number of lines: 2
count_lines_in_file('single_line.txt')  # Number of lines: 1
count_lines_in_file('non_existent.txt')  # File not found. Please check the file path.
```

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

INPUT & OUTPUT:

Input (File Path)	Output	Reason
example.txt	Number of lines: 2	File exists and contains two lines.
single_line.txt	Number of lines: 1	File exists with only one line.
non_existent.txt	File not found. Please check the file path.	File does not exist at the given path.

EXPLANATION:

The function takes a file path as input and attempts to open the file in read mode. If the file exists, it reads all the lines into a list and prints the total number of lines using the length of that list. If the specified file does not exist, the function catches a `FileNotFoundError` and prints a clear error message. Any other unexpected errors are also handled using a general exception block to prevent the program from crashing.