SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

ProgramName:B. Tech	Assignment Type: Lab	AcademicYear:2025-2026

CourseCoordinatorName	Venkataramana Veeramsetty
Instructor(s)Name	Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)

CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week2 - Wednesday	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	

AssignmentNumber:4.3(Present assignment number)/24(Total number of assignments)

Q.No.	Question	*ExpectedTime to complete*
1	Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques **Lab Objectives:** ● To explore and apply different levels of prompt examples in AI-assisted code	Week2 - Wednesday

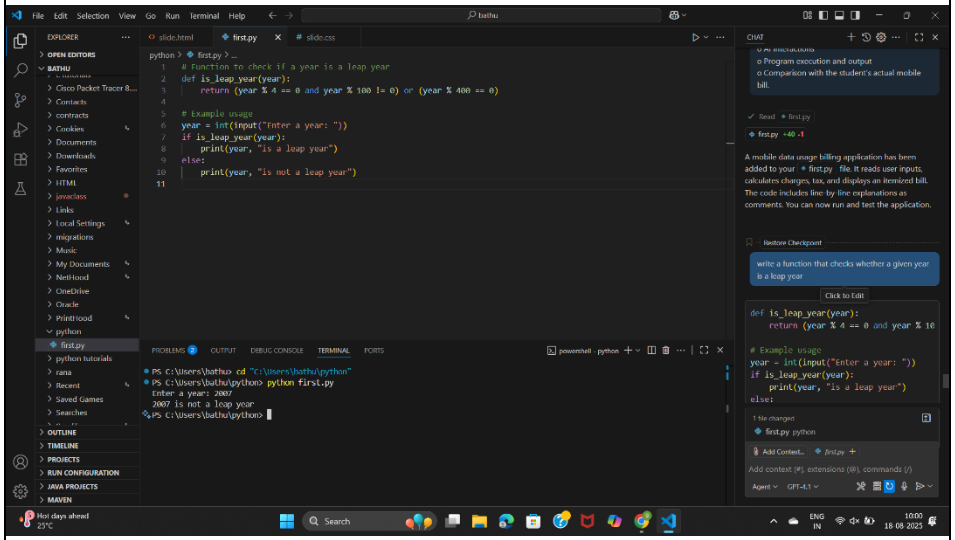
generation.
- To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality.
- To evaluate the impact of context richness and example quantity on AI performance.
- To build awareness of prompt strategy effectiveness for different problem types.

Lab Outcomes (LOs):
After completing this lab, students will be able to:

- Use zero-shot prompting to instruct AI with minimal context.
- Use one-shot prompting with a single example to guide AI code generation.
- Apply few-shot prompting using multiple examples to improve AI responses.
- Compare AI outputs across the three prompting strategies.

Task Description#1
- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.

Expected Output#1
- AI-generated function with no examples provided



```python
# Function to check if a year is a leap year
def is_leap_year(year):
    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)

# Example usage
year = int(input("Enter a year: "))
if is_leap_year(year):
    print(year, "is a leap year")
else:
    print(year, "is not a leap year")
```

```
PS C:\Users\bathu> cd "C:\Users\bathu\python"
PS C:\Users\bathu\python> python first.py
Enter a year: 2007
2007 is not a leap year
PS C:\Users\bathu\python>
```

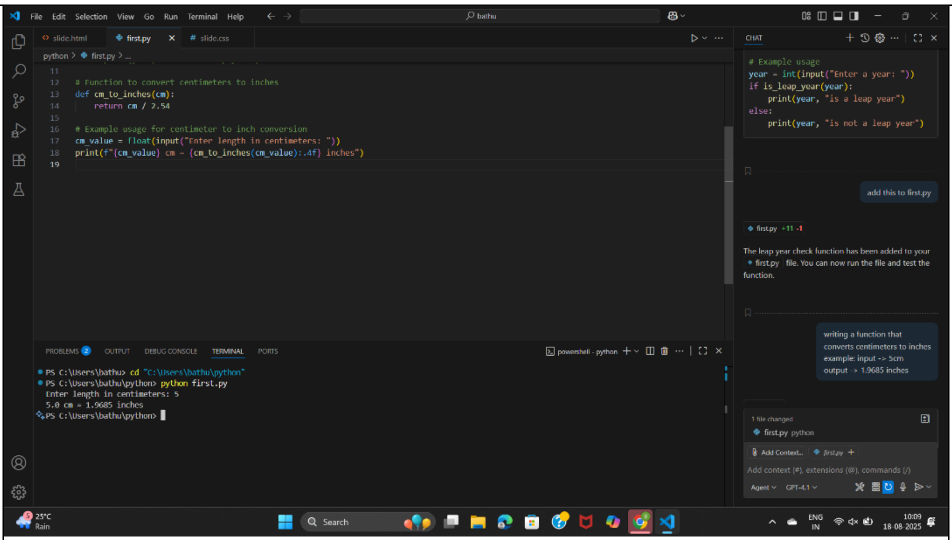
o Program execution and output
o Comparison with the student's actual mobile bill.

A mobile data usage billing application has been added to your first.py file. It reads user inputs, calculates charges, tax, and displays an itemized bill. The code includes line-by-line explanations as comments. You can now run and test the application.

write a function that checks whether a given year is a leap year

Task Description#2
- One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.

Expected Output#2
- Function with correct conversion logic



```python
# Function to convert centimeters to inches
def cm_to_inches(cm):
    return cm / 2.54

# Example usage for centimeter to inch conversion
cm_value = float(input("Enter length in centimeters: "))
print(f"{cm_value} cm = {cm_to_inches(cm_value):.4f} inches")
```

```
PS C:\Users\bathu> cd "C:\Users\bathu\python"
PS C:\Users\bathu\python> python first.py
Enter length in centimeters: 5
5.0 cm = 1.9685 inches
PS C:\Users\bathu\python>
```

```python
# Example usage
year = int(input("Enter a year: "))
if is_leap_year(year):
    print(year, "is a leap year")
else:
    print(year, "is not a leap year")
```

add this to first.py

first.py +11 -1

The leap year check function has been added to your first.py file. You can now run the file and test the function.

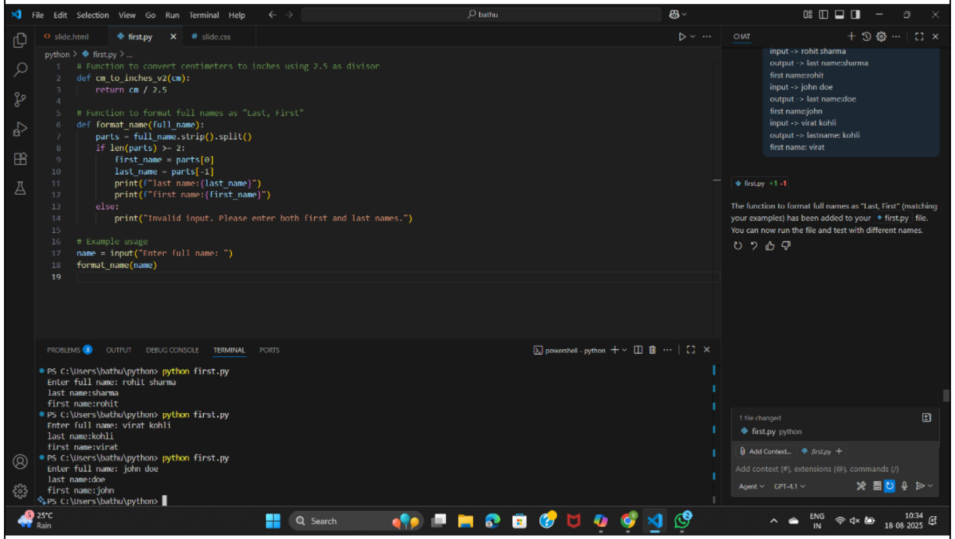
writing a function that converts centimeters to inches
example: input -> 5cm
output -> 1.9685 inches

Task Description#3

- Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First”.

Expected Output#3

- Well-structured function respecting the examples



```python
# Function to convert centimeters to inches using 2.5 as divisor
def cm_to_inches_v2(cm):
    return cm / 2.5

# Function to format full names as "Last, First"
def format_name(full_name):
    parts = full_name.strip().split()
    if len(parts) >= 2:
        first_name = parts[0]
        last_name = parts[-1]
        print(f"last name:{last_name}")
        print(f"first name:{first_name}")
    else:
        print("Invalid input. Please enter both first and last names.")

# Example usage
name = input("Enter full name: ")
format_name(name)
```

```
PS C:\Users\bathu\python> python first.py
Enter full name: rohit sharma
last name:sharma
first name:rohit
PS C:\Users\bathu\python> python first.py
Enter full name: virat kohli
last name:kohli
first name:virat
PS C:\Users\bathu\python> python first.py
Enter full name: john doe
last name:doe
first name:john
PS C:\Users\bathu\python>
```

input -> rohit sharma
output -> last name:sharma
first name:rohit
input -> john doe
output -> last name:doe
first name:john
input -> virat kohli
output -> lastname: kohli
first name: virat

first.py +1 -1

The function to format full names as "Last, First" (matching your examples) has been added to your first.py file. You can now run the file and test with different names.

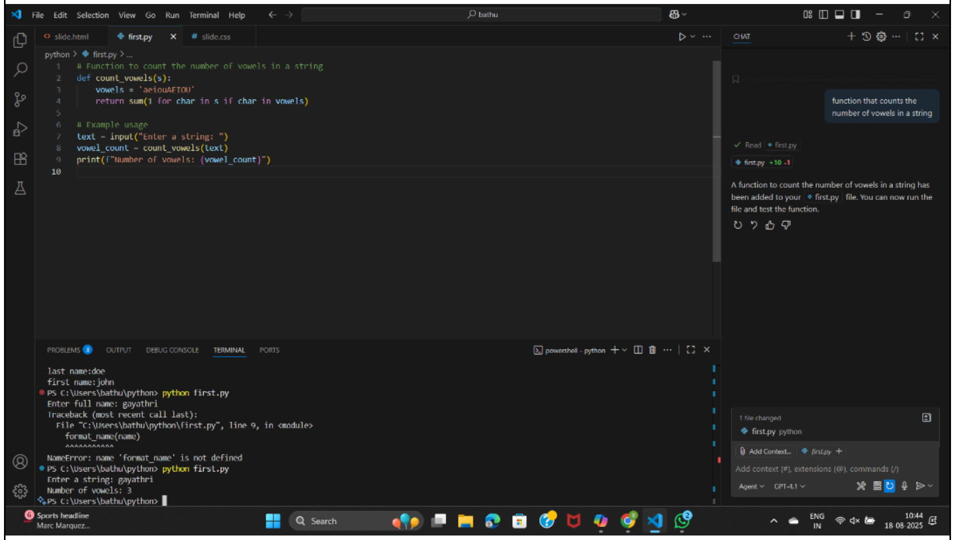
Task Description#4

- Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string.

Expected Output#4

- Functional output and comparative reflection

Zero shot



```python
# Function to count the number of vowels in a string
def count_vowels(s):
    vowels = 'aeiouAEIOU'
    return sum(1 for char in s if char in vowels)

# Example usage
text = input("Enter a string: ")
vowel_count = count_vowels(text)
print(f"Number of vowels: {vowel_count}")
```

```
last name:doe
first name:john
PS C:\Users\bathu\python> python first.py
Enter full name: gayathri
Traceback (most recent call last):
  File "C:\Users\bathu\python\first.py", line 9, in <module>
    format_name(name)
    ^^^^^^^^^^^
NameError: name 'format_name' is not defined
PS C:\Users\bathu\python> python first.py
Enter a string: gayathri
Number of vowels: 3
PS C:\Users\bathu\python>
```

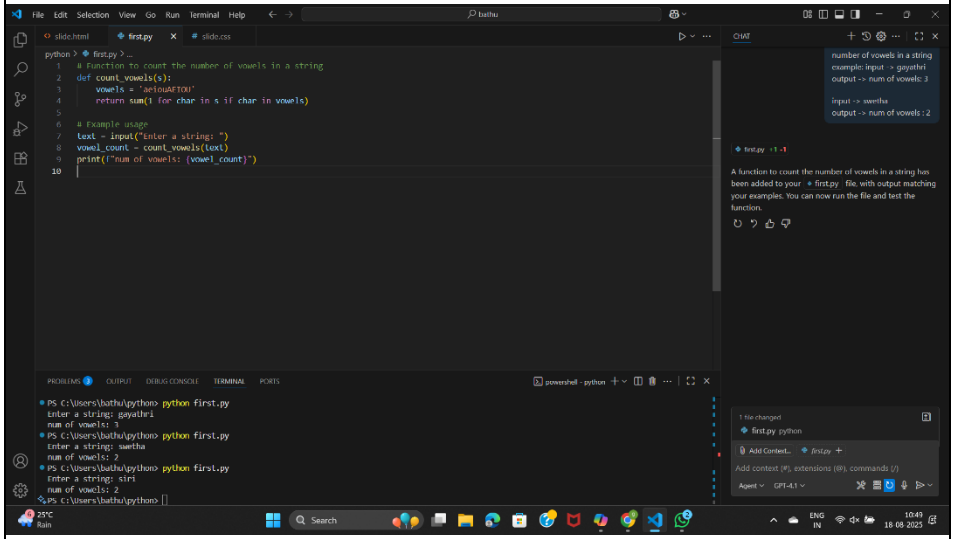
function that counts the number of vowels in a string

Read first.py

first.py +10 -1

A function to count the number of vowels in a string has been added to your first.py file. You can now run the file and test the function.

Few shot



```python
# Function to count the number of vowels in a string
def count_vowels(s):
    vowels = 'aeiouAEIOU'
    return sum(1 for char in s if char in vowels)

# Example usage
text = input("Enter a string: ")
vowel_count = count_vowels(text)
print(f"num of vowels: {vowel_count}")
```

```
PS C:\Users\bathu\python> python first.py
Enter a string: gayathri
num of vowels: 3
PS C:\Users\bathu\python> python first.py
Enter a string: swetha
num of vowels: 2
PS C:\Users\bathu\python> python first.py
Enter a string: siri
num of vowels: 2
PS C:\Users\bathu\python>
```

number of vowels in a string
example: input -> gayathri
output -> num of vowels: 3

input -> swetha
output -> num of vowels : 2

first.py +1 -1

A function to count the number of vowels in a string has been added to your first.py file, with output matching your examples. You can now run the file and test the function.

COMPARISON:

ZERO-SHOT:

- The AI was given no examples—just a task description.
- It produced a clean function named count_vowels() using a list comprehension with a simple vowel check.
- You tested the function with input "gayathri" and received correct output: Number of vowels: 3.

FEW-SHOT:

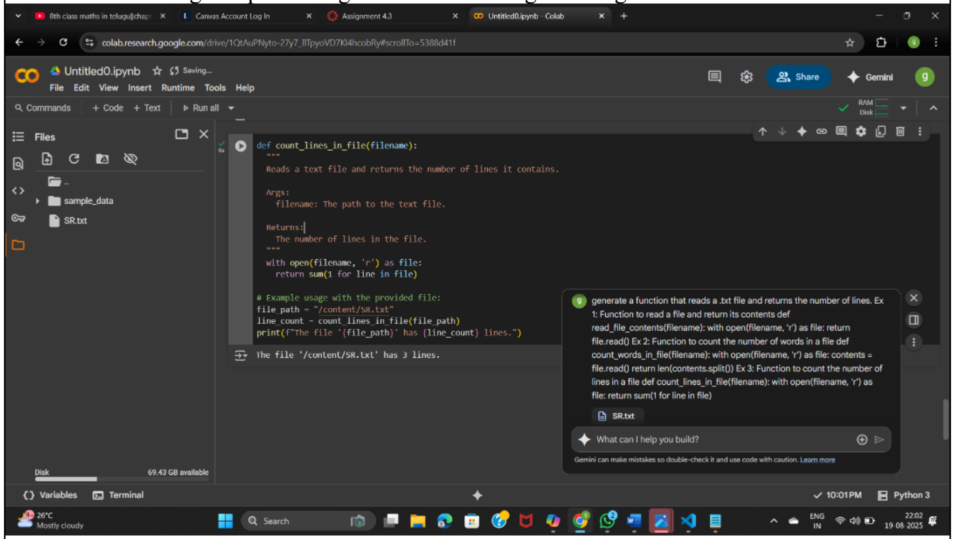
- The AI received examples of inputs and their expected outputs.
- It implemented the same logic, but this time aligned the print message format (num of vowels:) exactly as per the example**.**

Task Description#5

- Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines.

Expected Output#5

- Working file-processing function with AI-guided logic



```
def count_lines_in_file(filename):
  """
  Reads a text file and returns the number of lines it contains.

  Args:
    filename: The path to the text file.

  Returns:
    The number of lines in the file.
  """
  with open(filename, 'r') as file:
    return sum(1 for line in file)

# Example usage with the provided file:
file_path = "/content/SR.txt"
line_count = count_lines_in_file(file_path)
print(f"The file '{file_path}' has {line_count} lines.")
```

```
The file '/content/SR.txt' has 3 lines.
```

generate a function that reads a .txt file and returns the number of lines. Ex 1: Function to read a file and return its contents def read_file_contents(filename): with open(filename, 'r') as file: return file.read() Ex 2: Function to count the number of words in a file def count_words_in_file(filename): with open(filename, 'r') as file: contents = file.read() return len(contents.split()) Ex 3: Function to count the number of lines in a file def count_lines_in_file(filename): with open(filename, 'r') as file: return sum(1 for line in file)

SR.txt
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	**Max Marks**
Zero Shot (Task #1)	0.5
One Shot (Task#2)	0.5
Few Shot (Task#3 & Task #5)	1.0
Comparison (Task#4)	0.5
Total	**2.5 Marks**