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		1. Dr. Mohammed Ali Shaik 2. Dr. T Sampath Kumar 3. Mr. S Naresh Kumar 4. Dr. V. Rajesh 5. Dr. Brij Kishore 6. Dr Pramoda Patro 7. Dr. Venkataramana 8. Dr. Ravi Chander 9. Dr. Jagjeeth Singh	
CourseCode	24CS002PC215	**CourseTitle**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment		**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	
AssignmentNumber:3.3(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*
1	Lab 3: Prompt Engineering – Improving Prompts and Context Management **Lab Objectives:** • To understand how prompt structure and wording influence AI-generated code. • To explore how context (like comments and function names) helps AI generate relevant output. • To evaluate the quality and accuracy of code based on prompt clarity. • To develop effective prompting strategies for AI-assisted programming. **Lab Outcomes (LOs):** After completing this lab, students will be able to: • Generate Python code using Google Gemini in Google Colab. • Analyze the effectiveness of code explanations and suggestions by Gemini. • Set up and use Cursor AI for AI-powered coding assistance. • Evaluate and refactor code using Cursor AI features. • Compare AI tool behavior and code quality across different platforms.	03.08.2025 EOD

Task Description#1

- Try 3 different prompts to generate a factorial function.

Expected Output#1

- Comparison of AI-generated code styles

Task Description#2

- Provide a clear example input-output prompt to generate a sorting function.

Expected Output#2

- Functional sorting code from AI

Task Description#3

- Start with the vague prompt "Generate python code to calculate power bill" and improve it step-by-step

Expected Output#3

- Enhanced AI output with clearer prompts

Task Description#4

- Write structured comments to help AI generate two linked functions (e.g., login_user() and register_user()).

Expected Output#4

- Consistent functions with shared logic

Task Description#5

- Analyzing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions

Expected Output#5

- Code quality difference analysis for various prompts

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
Factorial Function (Task#1)	0.5
Sorting Function (Task#2)	0.5
Vogue Vs. Specific Prompting (Task #3)	0.5
Linked Functions (Task #4)	0.5
Temperature Conversion Function (Task #5)	0.5
Total	**2.5 Marks**

TASK-1:

PROMPT-1: write a python function to generate factorial of a number with one example.

```python
def factorial(n):
    """Compute the factorial of a non-negative integer n."""
    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Example usage:
print(factorial(5))  # Output: 120
```

Output:

```
120
PS C:\Users\Namitha\Desktop\AIAC\LAB-3> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c
:/Users/Namitha/Desktop/AIAC/LAB-3/Task-1.py
120
PS C:\Users\Namitha\Desktop\AIAC\LAB-3>
```

Ctrl+K to generate a command

PROMPT-2: write a python funtion to give factorial of given number in which input is taken from user.

```python
def factorial(n):
    """Compute the factorial of a non-negative integer n."""
    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

try:
    num = int(input("Enter a non-negative integer: "))
    print("Factorial of", num, "is", factorial(num))
except ValueError as e:
    print("Invalid input:", e)
```

Ctrl+L to chat, Ctrl+K to generate

Output:

```
s/Namitha/Desktop/AIAC/Task-1_2.py
Enter a non-negative integer: 5
Factorial of 5 is 120
PS C:\Users\Namitha\Desktop\AIAC> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
s/Namitha/Desktop/AIAC/Task-1_2.py
Enter a non-negative integer: 6
Factorial of 6 is 720
PS C:\Users\Namitha\Desktop\AIAC>
```

PROMPT-3: General python code to find factorial of given number.

```python
def factorial(n):
    """Calculate the factorial of a given non-negative integer n."""
    if n < 0:
        raise ValueError("Factorial is not defined for negative numbers.")
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Example usage:
try:
    num = int(input("Enter a non-negative integer: "))
    print(f"Factorial of {num} is {factorial(num)}")
except ValueError as e:
    print("Invalid input:", e)
```

Ctrl+I to chat, Ctrl+K to generate

Output:

```
PS C:\Users\Namitha\Desktop\AIAC> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
s/Namitha/Desktop/AIAC/Task-1_3.py
Enter a non-negative integer: 8
Factorial of 8 is 40320
PS C:\Users\Namitha\Desktop\AIAC>
```

REPORT:In promt-1 input is not taken from user whereas a random number factorial is given as output. In prompt-2,3 input is taken from user and that number factorial is given as output.

TASK-2:

PROMPT-1: write a python function to sort a list with one example.

```python
def sort_list(lst):
    """Sorts a list in ascending order and returns the sorted list."""
    return sorted(lst)

# Example usage:
example_list = [5, 2, 9, 1, 5, 6]
sorted_list = sort_list(example_list)
print("Original list:", example_list)
print("Sorted list:", sorted_list)
```
Ctrl+L to chat, Ctrl+K to generate

OUTPUT:

```
PS C:\Users\Namitha\Desktop\AIAC> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
s/Namitha/Desktop/AIAC/Task-2.py
Original list: [5, 2, 9, 1, 5, 6]
Sorted list: [1, 2, 5, 5, 6, 9]
PS C:\Users\Namitha\Desktop\AIAC>
```

PROMPT-2: write a python function to sort a list in which input is taken from user.

```python
def sort_user_list():
    """Takes a list of numbers as input from the user, sorts it, and prints the sorted list."""
    user_input = input("Enter numbers separated by spaces: ")
    try:
        num_list = [int(x) for x in user_input.strip().split()]
        sorted_list = sorted(num_list)
        print("Sorted list:", sorted_list)
    except ValueError:
        print("Please enter only integers separated by spaces.")

# Example usage:
sort_user_list()
```
Ctrl+L to chat, Ctrl+K to generate

Output:

```
s/Namitha/Desktop/AIAC/Task-2_2.py
Enter numbers separated by spaces: 1 5 3 76 23
Sorted list: [1, 3, 5, 23, 76]
PS C:\Users\Namitha\Desktop\AIAC>
```

PROMPT-3: write a python function to sort a list in which it must take strings as inputs from user and sort them .

```
def sort_user_strings():
    (variable) user_input: str  input from the user, sorts it, and prints the sorted list."""
    user_input = input("Enter strings separated by spaces: ")
    str_list = user_input.strip().split()
    sorted_list = sorted(str_list)
    print("Sorted list:", sorted_list)

# Example usage:
sort_user_strings()
```

Ctrl+L to chat, Ctrl+K to generate

Output:

```
PS C:\Users\Namitha\Desktop\AIAC> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
s/Namitha/Desktop/AIAC/Task-2_3.py
Enter strings separated by spaces: namitha vyshu bhavana hansika
Sorted list: ['bhavana', 'hansika', 'namitha', 'vyshu']
PS C:\Users\Namitha\Desktop\AIAC>
```

REPORT:In prompt-1 input is not taken from user .A list is taken and sorted randomly.
In prompt-2 input is taken from user and the given list is sorted.
In prompt-3 input is taken from user but in list strings are given as input and sorted in alphabetical wise.

TASK-3:

PROMPT-1:Write a python code to generate a power bill.

```
def calculate_power_bill(units):
    elif units <= 200:
        bill = (100 * 5) + ((units - 100) * 7)
    else:
        bill = (100 * 5) + (100 * 7) + ((units - 200) * 10)
    return bill

try:
    units = int(input("Enter the number of units consumed: "))
    total_bill = calculate_power_bill(units)
    print(f"Total power bill for {units} units is: Rs. {total_bill}")
except ValueError as e:
    print("Invalid input:", e)
```

Ctrl+L to chat, Ctrl+K to generate

Output:

```
PS C:\Users\Namitha\Desktop\AIAC> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
s/Namitha/Desktop/AIAC/Task-3.py
Enter the number of units consumed: 320
Total power bill for 320 units is: Rs. 2400
PS C:\Users\Namitha\Desktop\AIAC>
```

PROMPT-2:write a python function to generate power bill. It must take number of units and type of user from user.

```python
def generate_power_bill():
    try:
        units = float(input("Enter the number of units consumed: "))
        if units < 0:
            print("Units consumed cannot be negative.")
            return

        user_type = input("Enter user type (domestic/commercial): ").strip().lower()
        if user_type not in ['domestic', 'commercial']:
            print("Invalid user type. Please enter 'domestic' or 'commercial'.")
            return

        if user_type == 'domestic':
            # Example slab rates for domestic users
            if units <= 100:
                bill = units * 1.5
            elif units <= 300:
                bill = 100 * 1.5 + (units - 100) * 2.5
            else:
                bill = 100 * 1.5 + 200 * 2.5 + (units - 300) * 4.0
            fixed_charge = 50
        else:  # commercial
            # Example slab rates for commercial users
            if units <= 100:
                bill = units * 2.5
            elif units <= 300:
                bill = 100 * 2.5 + (units - 100) * 4.0
            else:
                bill = 100 * 2.5 + 200 * 4.0 + (units - 300) * 6.0
            fixed_charge = 100

        total_bill = bill + fixed_charge

        print(f"User Type: {user_type.capitalize()}")
        print(f"Units Consumed: {units}")
        print(f"Energy Charge: ₹{bill:.2f}")
        print(f"Fixed Charge: ₹{fixed_charge:.2f}")
        print(f"Total Power Bill: ₹{total_bill:.2f}")
    except ValueError:
        print("Invalid input. Please enter a valid number for units consumed.")

generate_power_bill()
```

Outputs:

```
Enter the number of units consumed: 240
Enter user type (domestic/commercial): domestic
User Type: Domestic
Units Consumed: 240.0
Energy Charge: ₹500.00
Fixed Charge: ₹50.00
Total Power Bill: ₹550.00
PS C:\Users\Namitha\Desktop\AIAC\LAB-3> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/Nam
```

```
itha/Desktop/AIAC/LAB-3/Task-3_2.py
Enter the number of units consumed: 1500
Enter user type (domestic/commercial): commercial
User Type: Commercial
Units Consumed: 1500.0
Energy Charge: ₹8250.00
Fixed Charge: ₹100.00
Total Power Bill: ₹8350.00
PS C:\Users\Namitha\Desktop\AIAC\LAB-3>
```

PROMPT-3: write a python function to generate powerbill. it must take number of units,charge per unit and type of user from user.

```python
def generate_power_bill():
    try:
        units = float(input("Enter the number of units consumed: "))
        if units < 0:
            print("Units consumed cannot be negative.")
            return
        charge_per_unit = float(input("Enter the charge per unit: "))
        if charge_per_unit < 0:
            print("Charge per unit cannot be negative.")
            return
        user_type = input("Enter user type (domestic/commercial): ").strip().lower()
        if user_type not in ['domestic', 'commercial']:
            print("Invalid user type. Please enter 'domestic' or 'commercial'.")
            return

        if user_type == 'domestic':
            fixed_charge = 50
        else:  # commercial
            fixed_charge = 100

        bill = units * charge_per_unit
        total_bill = bill + fixed_charge
```

```python
        print(f"User Type: {user_type.capitalize()}")
        print(f"Units Consumed: {units}")
        print(f"Charge per Unit: ₹{charge_per_unit:.2f}")
        print(f"Energy Charge: ₹{bill:.2f}")
        print(f"Fixed Charge: ₹{fixed_charge:.2f}")
        print(f"Total Power Bill: ₹{total_bill:.2f}")
    except ValueError:
        print("Invalid input. Please enter valid numbers for units and charge per unit.")

generate_power_bill()
```

Output:

```
itha/Desktop/AIAC/LAB-3/Task-3_3.py
Enter the number of units consumed: 300
Enter the charge per unit: 150
Enter user type (domestic/commercial): domestic
User Type: Domestic
Units Consumed: 300.0
Charge per Unit: ₹150.00
Energy Charge: ₹45000.00
Fixed Charge: ₹50.00
Total Power Bill: ₹45050.00
PS C:\Users\Namitha\Desktop\AIAC\LAB-3>
```

REPORT:

In prompt-1 user will give number of units only ,then bill is calculated accordingly. In prompt-2 user gives number of units along with type of user to enhance the code.According to type of user bill is calculated.

In prompt-3 user gives number of units,type of user and charge per unit also .so that bill is calculated more accurately.

TASK-4:

- PROMPT-1: Write a python code with structured comments to help AI generate two linked functions (e.g., login_user() and register_user()).

```python
# User Authentication System: Registration and Login Functions

# Dictionary to store user credentials in-memory (username: password)
user_db = {}

def register_user():
    """
    Register a new user by asking for a username and password.
    - Checks if the username already exists.
    - If not, adds the username and password to the user_db.
    """
    username = input("Enter a new username: ").strip()
    if username in user_db:
        print("Username already exists. Please choose a different username.")
        return
    password = input("Enter a new password: ").strip()
    user_db[username] = password
    print(f"User '{username}' registered successfully.")

def login_user():
    """
    Log in an existing user by verifying username and password.
    - Checks if the username exists.
```

```
- Verifies the password.
"""
username = input("Enter your username: ").strip()
if username not in user_db:
    print("Username does not exist. Please register first.")
    return
password = input("Enter your password: ").strip()
if user_db[username] == password:
    print(f"Login successful. Welcome, {username}!")
else:
    print("Incorrect password. Please try again.")

xample usage:
ncomment the following lines to test the functions interactively.
le True:
 action = input("Do you want to register or login? (register/login/exit): ").strip().lower()
 if action == "register":
     register_user()
 elif action == "login":
     login_user()
 elif action == "exit":
```

```
     break
 else:
     print("Invalid option. Please enter 'register', 'login', or 'exit'.")
```

Output:

```
S C:\Users\Namitha\Desktop\AIAC\LAB-3> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/Nam
itha/Desktop/AIAC/LAB-3/Task-4.py
o you want to register or login? (register/login/exit): register
nter a new username: 2403A52255
nter a new password: student
ser '2403A52255' registered successfully.
o you want to register or login? (register/login/exit):
```

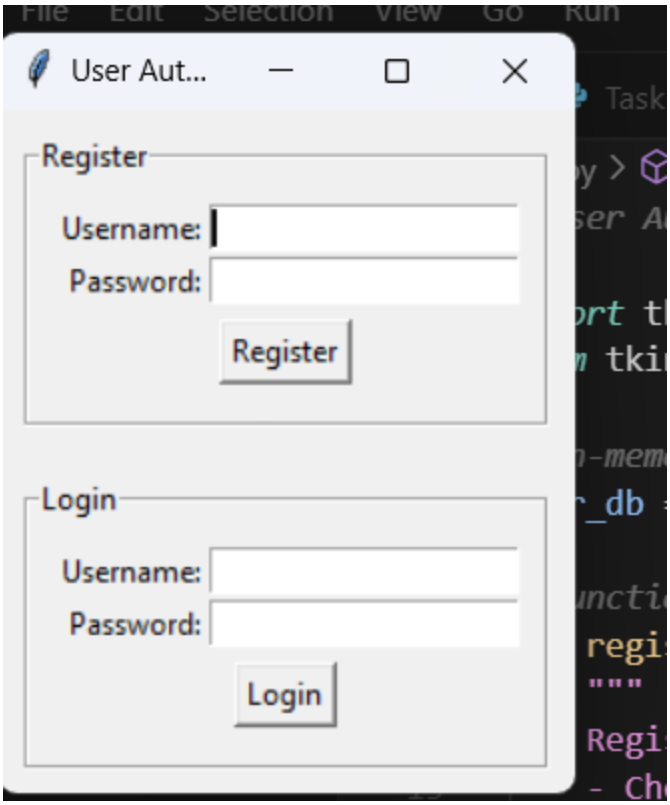
PROMPT-2: Write a python code with structured comments to help AI generate two linked functions (e.g., login_user() and register_user()). Using GUI.

```python
# User Authentication System with GUI using tkinter

import tkinter as tk
from tkinter import messagebox

# In-memory user database (username: password)
user_db = {}

# Function to register a new user
def register_user_gui():
    """
    Register a new user via GUI.
    - Checks if the username already exists.
    - If not, adds the username and password to the user_db.
    """
    username = reg_username_entry.get().strip()
    password = reg_password_entry.get().strip()
    if not username or not password:
        messagebox.showwarning("Input Error", "Please enter both username and password.")
        return
    if username in user_db:
        messagebox.showerror("Registration Failed", "Username already exists. Please choose a different username.")
    else:
```

```python
    messagebox.showerror("Registration Failed", "Username already exists. Please choose a different username.")
else:
    user_db[username] = password
    messagebox.showinfo("Registration Successful", f"User '{username}' registered successfully.")

# Function to log in an existing user
def login_user_gui():
    """
    Log in an existing user via GUI.
    - Checks if the username exists.
    - Verifies the password.
    """
    username = login_username_entry.get().strip()
    password = login_password_entry.get().strip()
    if not username or not password:
        messagebox.showwarning("Input Error", "Please enter both username and password.")
        return
    if username not in user_db:
        messagebox.showerror("Login Failed", "Username does not exist. Please register first.")
    elif user_db[username] == password:
        messagebox.showinfo("Login Successful", f"Welcome, {username}!")
    else:
```

```
    else:
        messagebox.showerror("Login Failed", "Incorrect password. Please try again.")

# Create main window
root = tk.Tk()
root.title("User Authentication System")

# Registration Frame
reg_frame = tk.LabelFrame(root, text="Register", padx=10, pady=10)
reg_frame.grid(row=0, column=0, padx=10, pady=10, sticky="nsew")

tk.Label(reg_frame, text="Username:").grid(row=0, column=0, sticky="e")
reg_username_entry = tk.Entry(reg_frame)
reg_username_entry.grid(row=0, column=1)

tk.Label(reg_frame, text="Password:").grid(row=1, column=0, sticky="e")
reg_password_entry = tk.Entry(reg_frame, show="*")
reg_password_entry.grid(row=1, column=1)

reg_button = tk.Button(reg_frame, text="Register", command=register_user_gui)
reg_button.grid(row=2, column=0, columnspan=2, pady=5)

# Login Frame
login_frame = tk.LabelFrame(root, text="Login", padx=10, pady=10)
login_frame.grid(row=1, column=0, padx=10, pady=10, sticky="nsew")

tk.Label(login_frame, text="Username:").grid(row=0, column=0, sticky="e")
login_username_entry = tk.Entry(login_frame)
login_username_entry.grid(row=0, column=1)

tk.Label(login_frame, text="Password:").grid(row=1, column=0, sticky="e")
login_password_entry = tk.Entry(login_frame, show="*")
login_password_entry.grid(row=1, column=1)

login_button = tk.Button(login_frame, text="Login", command=login_user_gui)
login_button.grid(row=2, column=0, columnspan=2, pady=5)

# Start the GUI event loop
root.mainloop()
```

Output:



REPORT:
In prompt-1 both the functions login user() and register user() are generated and all details are taken from user.In prompt-2 same functions are created using tkinter library in GUI(graphical user interface).

TASK-5:

PROMPT-1:write a python function for temperature conversions .inputs must be taken from user.

```python
def temperature_conversion():
    """
    Converts temperature between Celsius, Fahrenheit, and Kelvin.
    Takes user input for the temperature value and the conversion type.
    """
    print("Temperature Conversion Tool")
    print("Choose the conversion type:")
    print("1. Celsius to Fahrenheit")
    print("2. Fahrenheit to Celsius")
    print("3. Celsius to Kelvin")
    print("4. Kelvin to Celsius")
    print("5. Fahrenheit to Kelvin")
    print("6. Kelvin to Fahrenheit")
    choice = input("Enter your choice (1-6): ").strip()

    if choice not in {'1', '2', '3', '4', '5', '6'}:
        print("Invalid choice. Please select a number from 1 to 6.")
        return

    try:
        temp = float(input("Enter the temperature value: ").strip())
    except ValueError:
        print("Invalid temperature value. Please enter a numeric value.")
        return

    if choice == '1':
        # Celsius to Fahrenheit
        result = (temp * 9/5) + 32
        print(f"{temp}°C = {result:.2f}°F")
    elif choice == '2':
        # Fahrenheit to Celsius
        result = (temp - 32) * 5/9
        print(f"{temp}°F = {result:.2f}°C")
    elif choice == '3':
        # Celsius to Kelvin
        result = temp + 273.15
        print(f"{temp}°C = {result:.2f}K")
    elif choice == '4':
        # Kelvin to Celsius
        result = temp - 273.15
        print(f"{temp}K = {result:.2f}°C")
    elif choice == '5':
        # Fahrenheit to Kelvin
        result = (temp - 32) * 5/9 + 273.15
```

```
        result = (temp - 32) * 5/9 + 273.15
        print(f"{temp}°F = {result:.2f}K")
    elif choice == '6':
        # Kelvin to Fahrenheit
        result = (temp - 273.15) * 9/5 + 32
        print(f"{temp}K = {result:.2f}°F")
1
# Example usage:
temperature_conversion()
```

Output:

```
PS C:\Users\Namitha\Desktop\AIAC\LAB-3> & C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
itha/Desktop/AIAC/LAB-3/Task-5.py
Temperature Conversion Tool
Choose the conversion type:
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
3. Celsius to Kelvin
4. Kelvin to Celsius
5. Fahrenheit to Kelvin
6. Kelvin to Fahrenheit
Enter your choice (1-6): 1

Enter your choice (1-6): 1
Enter the temperature value: 35
35.0°C = 95.00°F
PS C:\Users\Namitha\Desktop\AIAC\LAB-3> 11111& C:/Users/Namitha/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/User
s/Namitha/Desktop/AIAC/LAB-3/Task-5.py
```