SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
NAME: SRIDHAR PAIDAKULA **ENROLL NO.:2403A53013** **BATCH NO.:24BTCAICYB01**		**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	Week4 - Wednesday	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	
AssignmentNumber:7.3(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*ExpectedTime to complete*
1	Lab 7: Error Debugging with AI: Systematic approaches to finding and fixing bugs **Lab Objectives:**	Week4 – Wednesday

- To identify and correct syntax, logic, and runtime errors in Python programs using AI tools.
- To understand common programming bugs and AI-assisted debugging suggestions.
- To evaluate how AI explains, detects, and fixes different types of coding errors.
- To build confidence in using AI to perform structured debugging practices.

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

- Use AI tools to detect and correct syntax, logic, and runtime errors.
- Interpret AI-suggested bug fixes and explanations.
- Apply systematic debugging strategies supported by AI-generated insights.
- Refactor buggy code using responsible and reliable programming patterns.

Task Description#1

- Paste a function with a missing colon (add(a, b)), and let AI fix the syntax error.

```
python

def add(a, b)
    return a + b
```

CODE &OUTPUT:

```
def add(a,b):
  return a + b
print (add(2,3))
```

```
5
```

EXPLANATIONABOUTSYNTAXERROR:

In the above program, a colon (:) was missing at the end of the function definition line (def add(a, b):). This caused a syntax error. After adding the missing colon, the syntax error was resolved and the program ran correctly.

Task Description#2 (Loops)

- Identify and fix a logic error in a loop that causes infinite iteration.

```
python

def count_down(n):
    while n >= 0:
        print(n)
        n += 1  # Should be n -= 1
```

Expected Output#2

- AI fixes increment/decrement error

CODE &OUTPUT:

```
def count_down(n):
  while n >= 0:
    print(n)
    n -=1
count_down(5)
```

```
5
4
3
2
1
0
```

EXPLANATION:

The error is in this line: n += 1. It increases the number, so the loop never stops because n stays above 0 forever. To fix it, change it to n-= 1 to decrease n each time, letting the loop count down and end properly.

Task Description#3

- Debug a runtime error caused by division by zero. Let AI insert try-except.

```
# Debug the following code
def divide(a, b):
    return a / b

print(divide(10, 0))
```

Expected Output#3

- Corrected function with safe error handling

CODE &OUTPUT:

```
def divide(a, b):
    try:
      return a / b
    except ZeroDivisionError:
      return "Error:Division by zero is not allowed."
print(divide(10, 0))
```

```
Error:Division by zero is not allowed.
```

EXPLANATION:

The code tries to divide a by b inside a try block. If b is zero, it causes a ZeroDivisionError, which is caught by the except block, and instead of crashing, it returns a friendly error message. This way, the program handles division by zero gracefully.

Task Description#4

- Provide a faulty class definition (missing self in parameters). Let AI fix it

```python
class Rectangle:
    def __init__(length, width):
        self.length = length
        self.width = width
```

Expected Output#4

- Correct __init__() method and explanation

CODE &OUTPUT:

```
class Rectangle:
  def __init__(self, length, width):
    self.length = length
    self.width = width
rect = Rectangle(10, 5)
print("Length:", rect.length)
print("Width:", rect.width)
```

```
Length: 10
Width: 5
```

Explanation:

The class Rectangle has a constructor _init_ that initializes the object's length and width. The self parameter refers to the instance being created, allowing the method to set attributes for that specific object. Without self, the code won't work correctly.

Task Description#5

- Access an invalid list index and use AI to resolve the Index Error.

```python
numbers = [1, 2, 3]
print(numbers[5])
```

Expected Output#5

- AI suggests checking length or using safe access logic

CODE &OUTPUT:

```
numbers = [1, 2, 3]
try:
  print(numbers[5])
except IndexError:
    print("Error: Index out of range.")
```

```
Error: Index out of range.
```

EXPLANATION:

The code tries to access an index that doesn't exist in the list, causing an IndexError. The try-except block catches this error and prints a friendly message instead of crashing the program.

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
Identification of bugs	0.5
Application of AI-suggested fixes	0.5
Explanation and understanding of errors	0.5
Corrected code functionality	0.5
Report structure and reflection	0.5
Total	**2.5 Marks**

: