SEM LAB EXAM

Name: D. Sravika Reddy Roll No: 2403a510d0

Batch No: 05 Course: AI Assisted Coding

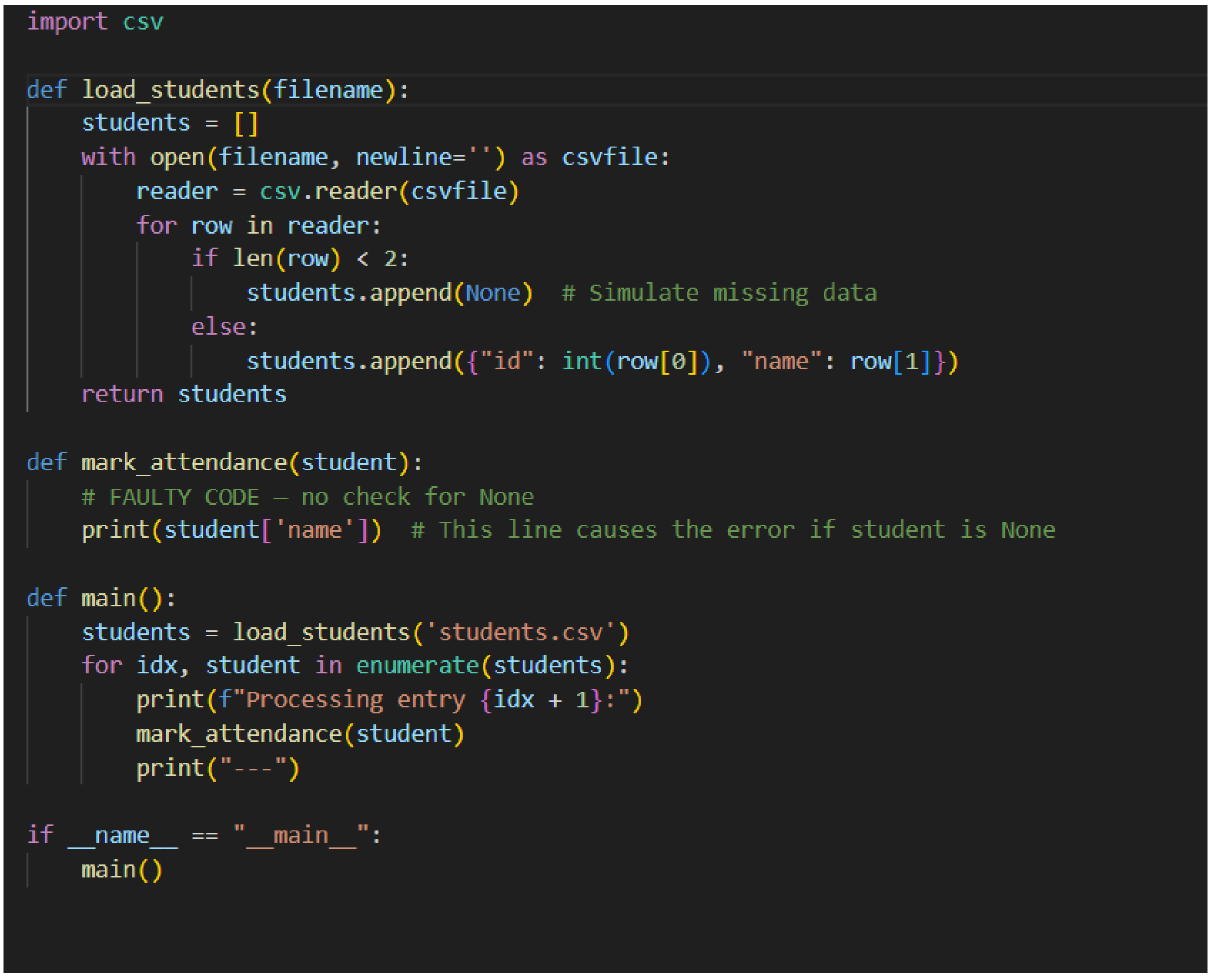
Subset 4 – Error Debugging with AI for Attendance Tracker
Q1: Fix null reference bug

- Task 1: Ask AI to locate the exact faulty line.
- Task 2: Use AI to rewrite the corrected function

→Task 1:

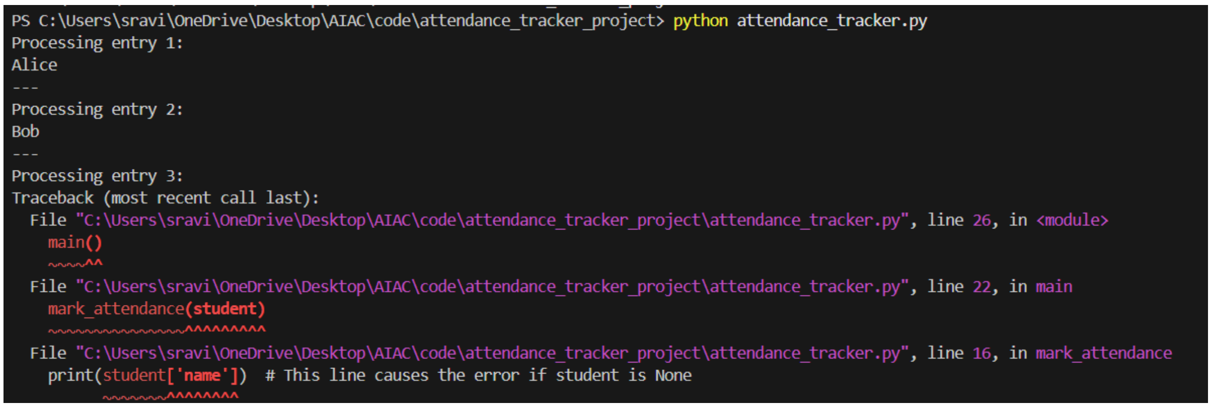
#Prompt: Ask AI to locate the exact faulty line.

Code:



```python
import csv

def load_students(filename):
    students = []
    with open(filename, newline='') as csvfile:
        reader = csv.reader(csvfile)
        for row in reader:
            if len(row) < 2:
                students.append(None)  # Simulate missing data
            else:
                students.append({"id": int(row[0]), "name": row[1]})
    return students

def mark_attendance(student):
    # FAULTY CODE — no check for None
    print(student['name'])  # This line causes the error if student is None

def main():
    students = load_students('students.csv')
    for idx, student in enumerate(students):
        print(f"Processing entry {idx + 1}:")
        mark_attendance(student)
        print("---")

if __name__ == "__main__":
    main()
```

Output:



```
PS C:\Users\sravi\OneDrive\Desktop\AIAC\code\attendance_tracker_project> python attendance_tracker.py
Processing entry 1:
Alice
---
Processing entry 2:
Bob
---
Processing entry 3:
Traceback (most recent call last):
  File "C:\Users\sravi\OneDrive\Desktop\AIAC\code\attendance_tracker_project\attendance_tracker.py", line 26, in <module>
    main()
    ~~~~^^
  File "C:\Users\sravi\OneDrive\Desktop\AIAC\code\attendance_tracker_project\attendance_tracker.py", line 22, in main
    mark_attendance(student)
    ~~~~~~~~~~~~~~~^^^^^^^^^
  File "C:\Users\sravi\OneDrive\Desktop\AIAC\code\attendance_tracker_project\attendance_tracker.py", line 16, in mark_attendance
    print(student['name'])  # This line causes the error if student is None
          ~~~~~~~^^^^^^^^
```

Explanation:

The exact faulty line causing the null reference bug is:

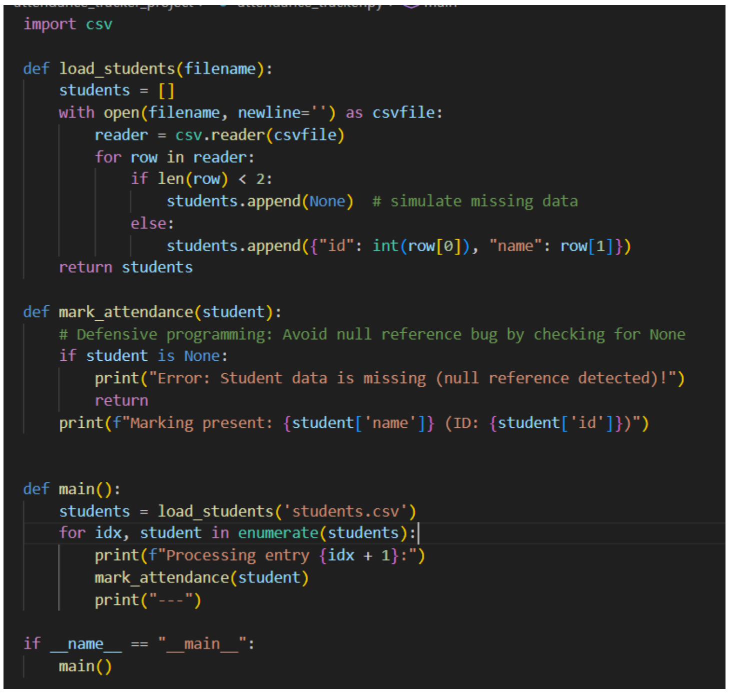
print(student['name'])

This line tries to access the 'name' key on the student variable, which can be None if there is missing data. When student is None, this line raises an error because you cannot access keys of a NoneType object.

→Task 2

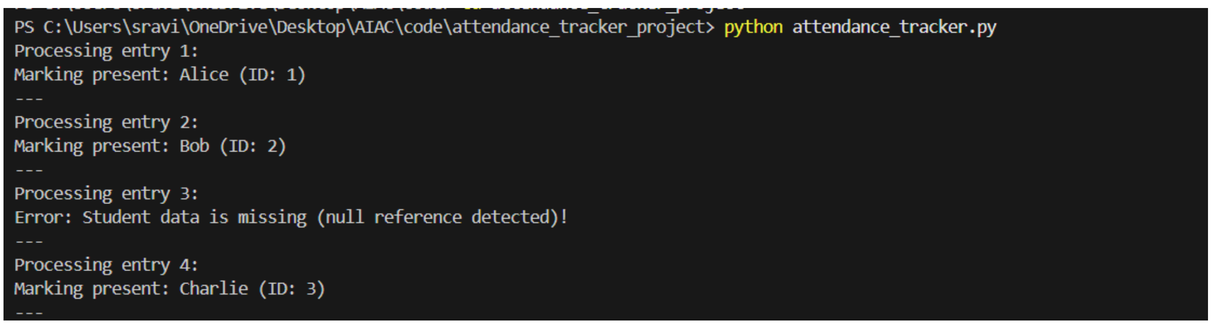
#Prompt: Use AI to rewrite the corrected function

Code:



```python
import csv

def load_students(filename):
    students = []
    with open(filename, newline='') as csvfile:
        reader = csv.reader(csvfile)
        for row in reader:
            if len(row) < 2:
                students.append(None)  # simulate missing data
            else:
                students.append({"id": int(row[0]), "name": row[1]})
    return students

def mark_attendance(student):
    # Defensive programming: Avoid null reference bug by checking for None
    if student is None:
        print("Error: Student data is missing (null reference detected)!")
        return
    print(f"Marking present: {student['name']} (ID: {student['id']})")


def main():
    students = load_students('students.csv')
    for idx, student in enumerate(students):
        print(f"Processing entry {idx + 1}:")
        mark_attendance(student)
        print("---")

if __name__ == "__main__":
    main()
```

Output:



```
PS C:\Users\sravi\OneDrive\Desktop\AIAC\code\attendance_tracker_project> python attendance_tracker.py
Processing entry 1:
Marking present: Alice (ID: 1)
---
Processing entry 2:
Marking present: Bob (ID: 2)
---
Processing entry 3:
Error: Student data is missing (null reference detected)!
---
Processing entry 4:
Marking present: Charlie (ID: 3)
---
```

Explanation:

The corrected mark_attendance function adds a check to ensure student is not None before trying to access its attributes. This prevents the null reference error and makes the program handle missing data gracefully by printing a helpful error message instead of crashing.

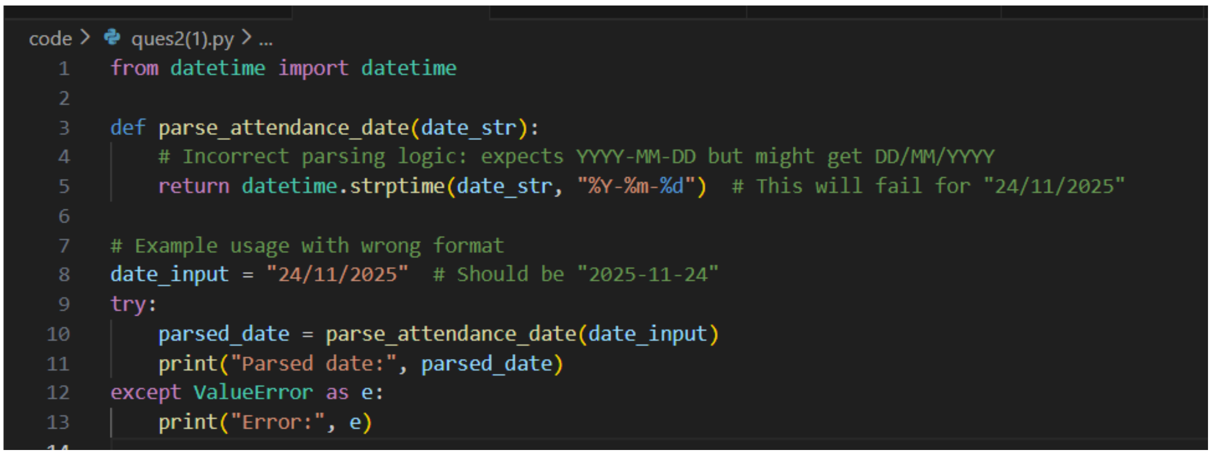
Q2: Wrong date format issue

- Task 1: Provide logs to AI to identify incorrect parsing logic.
- Task 2: Implement AI-suggested fix with proper formatting.

→Task 1:

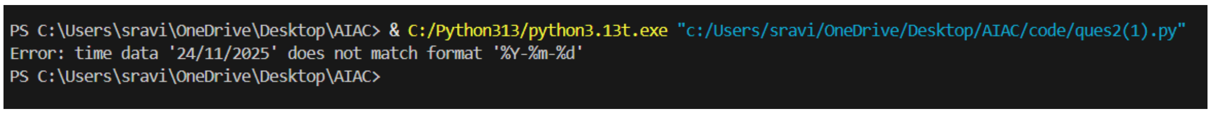
#Prompt: Provide logs to AI to identify incorrect parsing logic.

Code:



```python
from datetime import datetime

def parse_attendance_date(date_str):
    # Incorrect parsing logic: expects YYYY-MM-DD but might get DD/MM/YYYY
    return datetime.strptime(date_str, "%Y-%m-%d")  # This will fail for "24/11/2025"

# Example usage with wrong format
date_input = "24/11/2025"  # Should be "2025-11-24"
try:
    parsed_date = parse_attendance_date(date_input)
    print("Parsed date:", parsed_date)
except ValueError as e:
    print("Error:", e)
```

Output:



```
PS C:\Users\sravi\OneDrive\Desktop\AIAC> & C:/Python313/python3.13t.exe "c:/Users/sravi/OneDrive/Desktop/AIAC/code/ques2(1).py"
Error: time data '24/11/2025' does not match format '%Y-%m-%d'
PS C:\Users\sravi\OneDrive\Desktop\AIAC>
```

Explanation:

When running the script with faulty date parsing logic, the program expected the date in the format YYYY-MM-DD but received DD/MM/YYYY. This caused a ValueError and the error log showed:

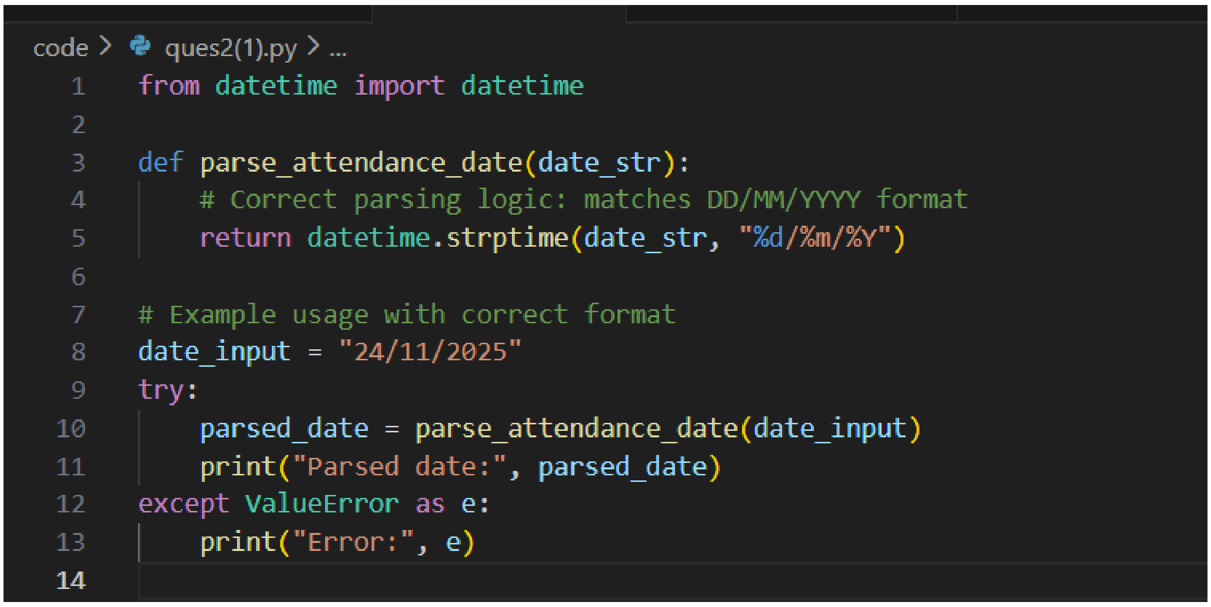
Error: time data '24/11/2025' does not match format '%Y-%m-%d'

This log helps us (or AI) quickly identify that the issue is with a mismatch between the date format expected and the format provided in the input data.

→Task 2:

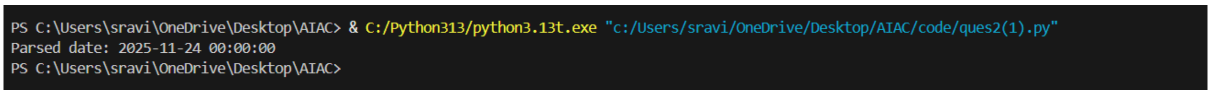
#Prompt: Implement AI-suggested fix with proper formatting.

Code:



```
code > ques2(1).py > ...
from datetime import datetime

def parse_attendance_date(date_str):
    # Correct parsing logic: matches DD/MM/YYYY format
    return datetime.strptime(date_str, "%d/%m/%Y")

# Example usage with correct format
date_input = "24/11/2025"
try:
    parsed_date = parse_attendance_date(date_input)
    print("Parsed date:", parsed_date)
except ValueError as e:
    print("Error:", e)
```

Output:



```
PS C:\Users\sravi\OneDrive\Desktop\AIAC> & C:/Python313/python3.13t.exe "c:/Users/sravi/OneDrive/Desktop/AIAC/code/ques2(1).py"
Parsed date: 2025-11-24 00:00:00
PS C:\Users\sravi\OneDrive\Desktop\AIAC>
```

Explanation:

To fix the problem, we updated the date parsing code to use the correct format string, "%d/%m/%Y", matching the actual input (24/11/2025).
After this fix, the date is parsed successfully and the output is:

```
Parsed date: 2025-11-24 00:00:00
```

This confirms that the bug is resolved and your program now works with your attendance data's date format.