

AI ASSIGNMENT-6.5

Name : B.Harshini

Hall No. : 2303A52242

Batch : 36

Task Description #1 (AI-Based Code Completion for Conditional Eligibility Check)

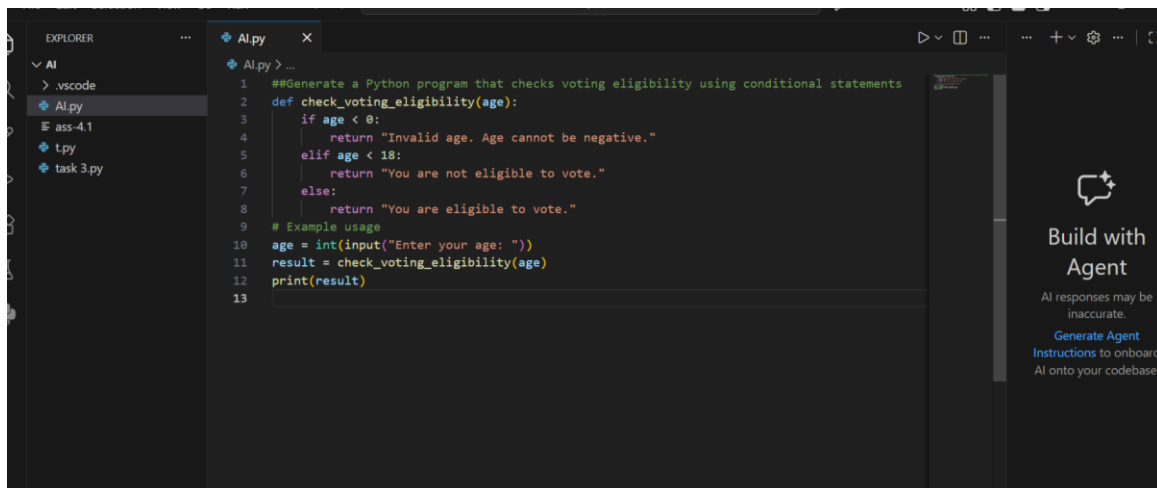
Task: Use an AI tool to generate eligibility logic.

Prompt:

“Generate Python code to check voting eligibility based on age and citizenship.”

Expected Output:

- AI-generated conditional logic.
- Correct eligibility decisions.
- Explanation of conditions.

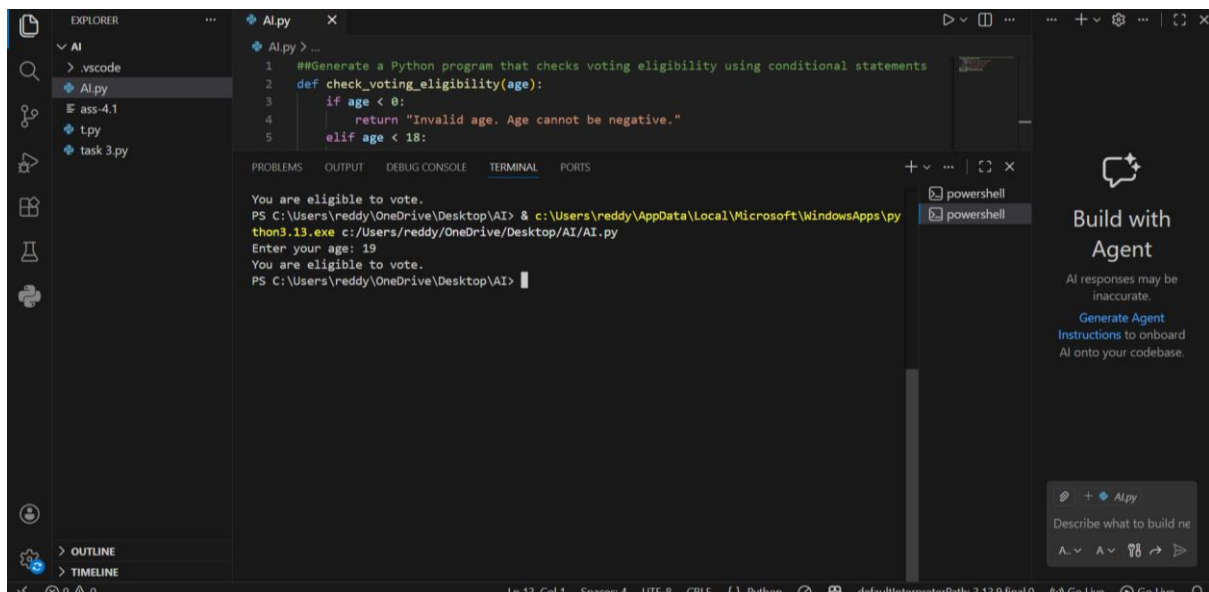


The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'AI' with files 'ass-4.1', 't.py', and 'task3.py'. The code editor shows a Python file named 'Al.py' with the following code:

```
1  ##Generate a Python program that checks voting eligibility using conditional statements
2  def check_voting_eligibility(age):
3      if age < 0:
4          return "Invalid age. Age cannot be negative."
5      elif age < 18:
6          return "You are not eligible to vote."
7      else:
8          return "You are eligible to vote."
9  # Example usage
10 age = int(input("Enter your age: "))
11 result = check_voting_eligibility(age)
12 print(result)
13
```

On the right side of the code editor, there is a sidebar with the text "Build with Agent" and a small icon of a person with a speech bubble. Below this, it says "AI responses may be inaccurate." and "Generate Agent Instructions to onboard AI onto your codebase."

Output:



Task Description #2(AI-Based Code Completion for Loop-Based

String Processing)

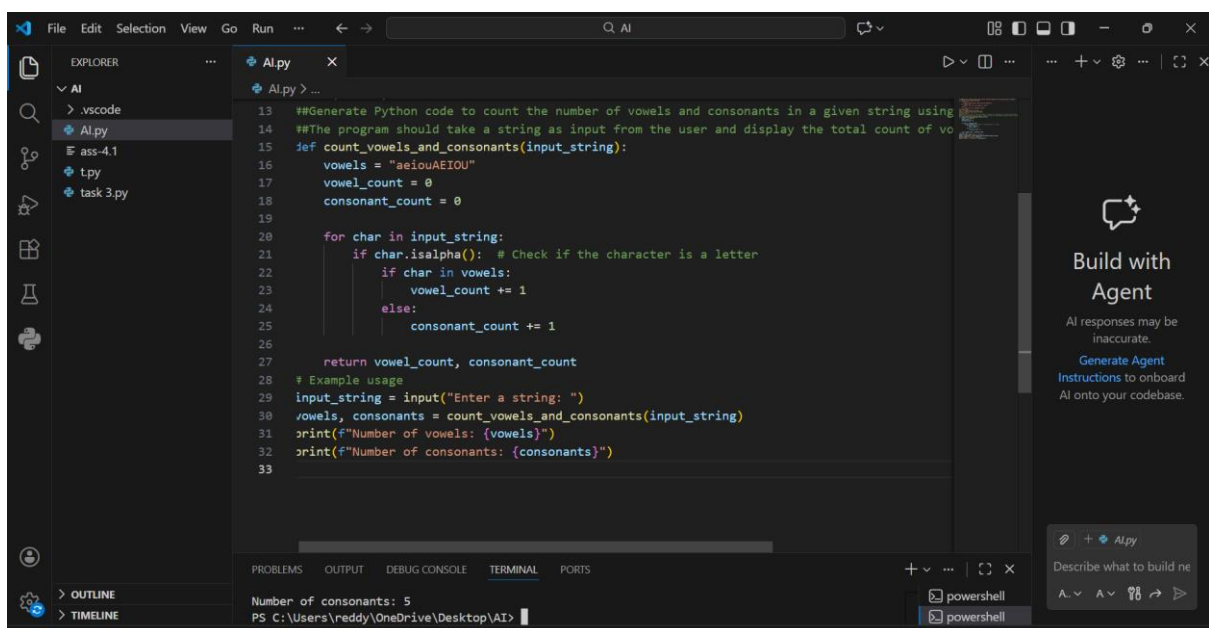
Task: Use an AI tool to process strings using loops.

Prompt:

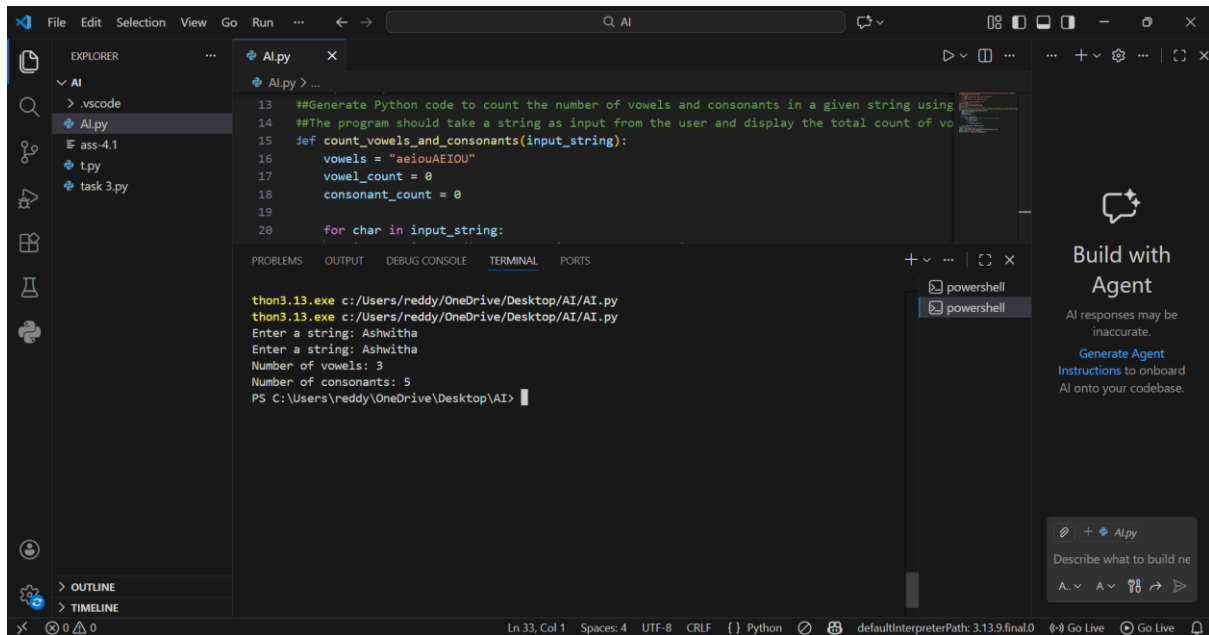
“Generate Python code to count vowels and consonants in a string using a loop.”

Expected Output:

- AI-generated string processing logic.
- Correct counts.
- Output verification.



Output:



The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left shows a project named 'AI' with files 'ass-4.1', 'tpy', and 'task 3.py'. The main editor window displays a Python file 'AI.py' with the following code:

```
13  ##Generate Python code to count the number of vowels and consonants in a given string using
14  ##The program should take a string as input from the user and display the total count of vo
15  def count_vowels_and_consonants(input_string):
16      vowels = "aeiouAEIOU"
17      vowel_count = 0
18      consonant_count = 0
19
20      for char in input_string:
```

Below the code editor is a terminal window showing the execution of the script:

```
thon3.13.exe c:/Users/reddy/OneDrive/Desktop/AI/AI.py
thon3.13.exe c:/Users/reddy/OneDrive/Desktop/AI/AI.py
Enter a string: Ashwitha
Enter a string: Ashwitha
Number of vowels: 3
Number of consonants: 5
PS C:\Users\reddy\OneDrive\Desktop\AI>
```

On the right side of the editor, there is a 'Build with Agent' panel. It contains the text 'AI responses may be inaccurate.' and a button 'Generate Agent'. Below this, it says 'Instructions to onboard AI onto your codebase.' At the bottom of the panel, there is a section for 'Alpy' with the text 'Describe what to build ne' and some navigation icons.

Task Description #3 (AI-Assisted Code Completion Reflection

Task)

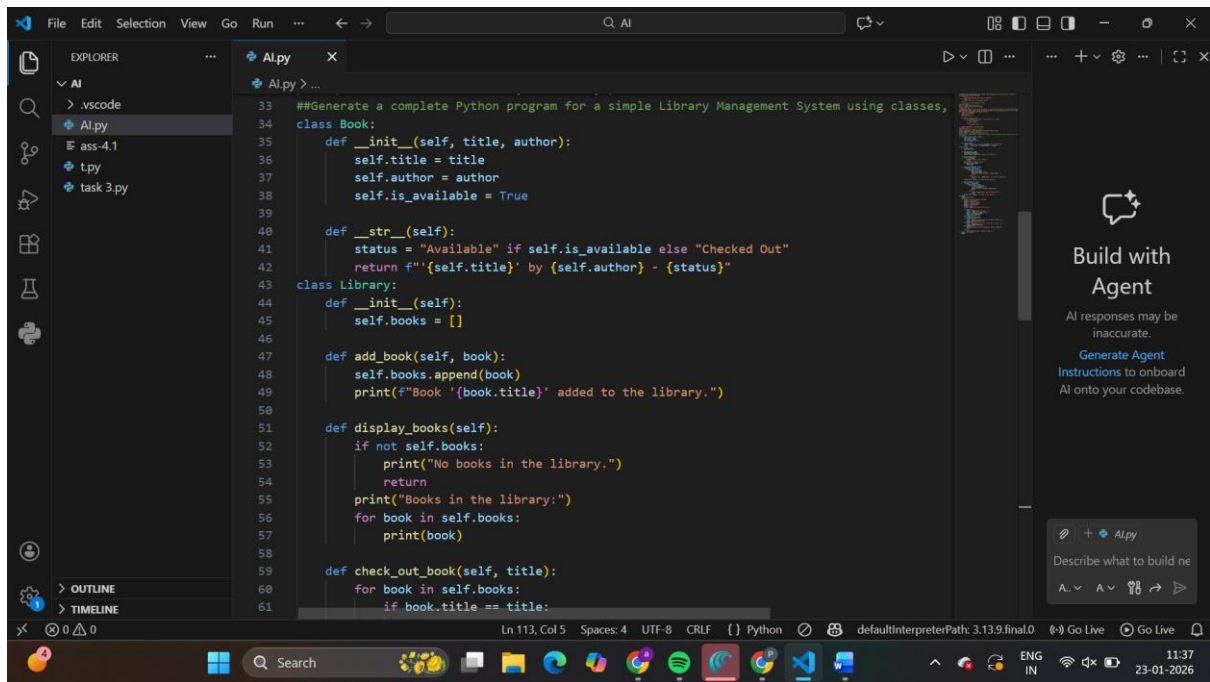
Task: Use an AI tool to generate a complete program using classes, loops, and conditionals.

Prompt:

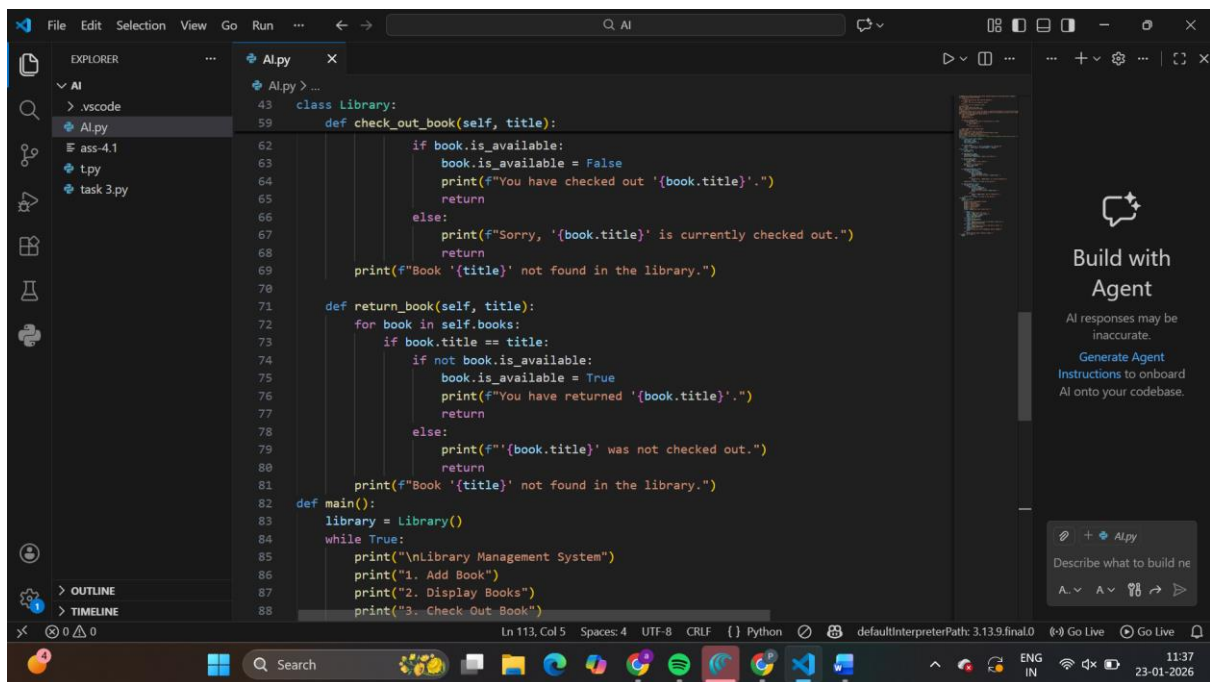
“Generate a Python program for a library management system using classes, loops, and conditional statements.”

Expected Output:

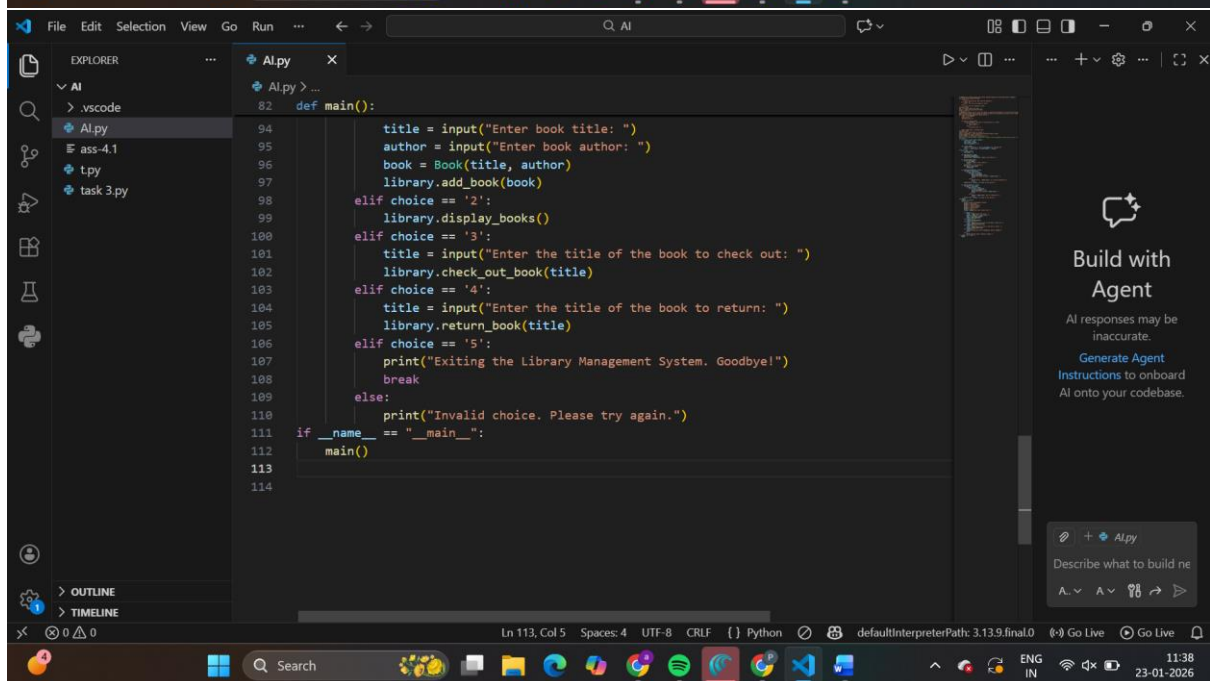
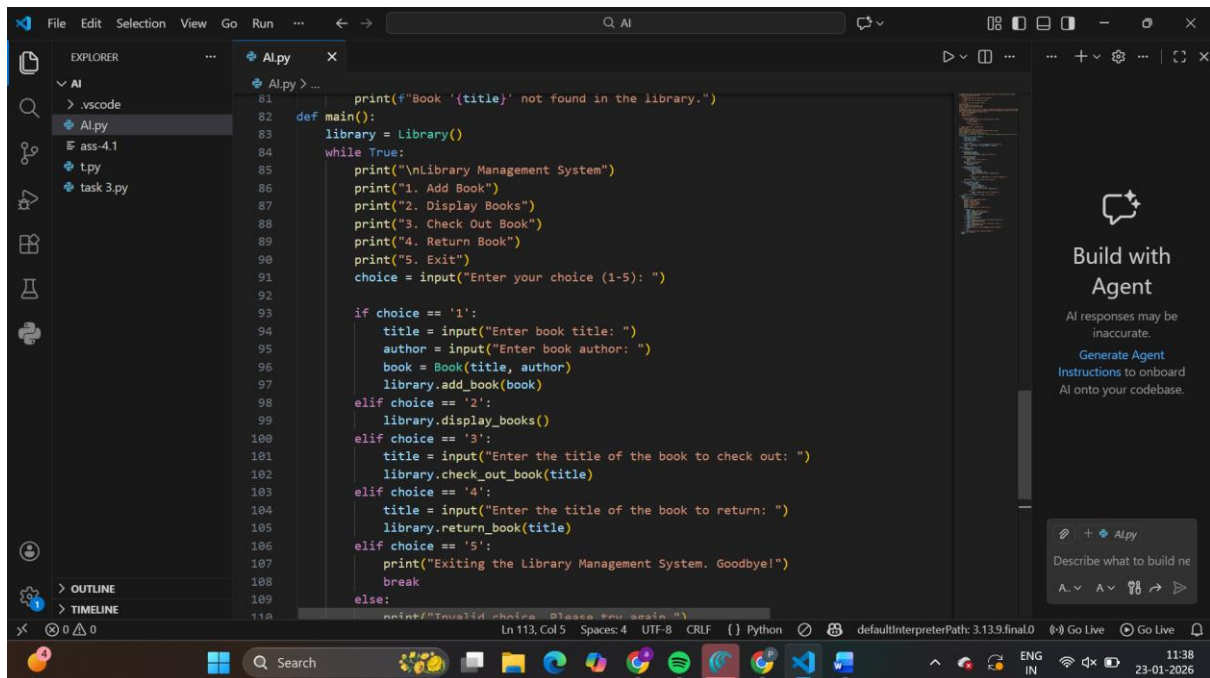
- Complete AI-generated program.
- Review of AI suggestions quality.
- Short reflection on AI-assisted coding experience.



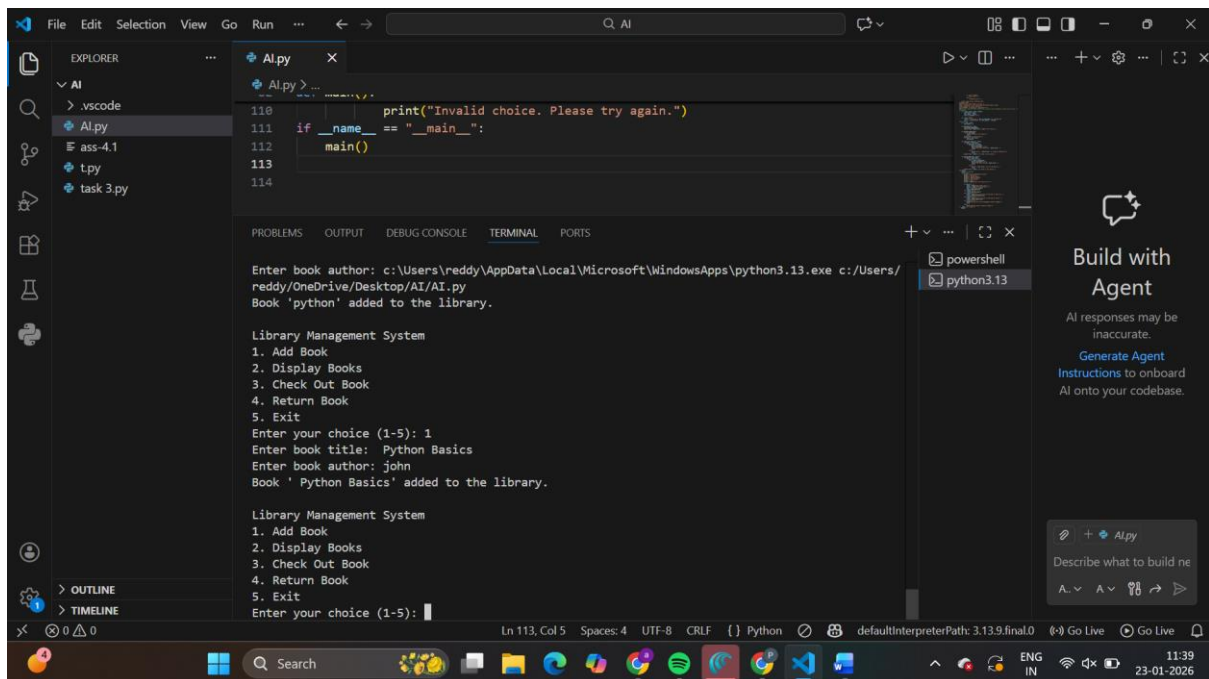
```
33 ##Generate a complete Python program for a simple Library Management System using classes,
34 class Book:
35     def __init__(self, title, author):
36         self.title = title
37         self.author = author
38         self.is_available = True
39
40     def __str__(self):
41         status = "Available" if self.is_available else "Checked Out"
42         return f'{self.title}' by {self.author} - {status}"
43 class Library:
44     def __init__(self):
45         self.books = []
46
47     def add_book(self, book):
48         self.books.append(book)
49         print(f"Book '{book.title}' added to the library.")
50
51     def display_books(self):
52         if not self.books:
53             print("No books in the library.")
54             return
55         print("Books in the library:")
56         for book in self.books:
57             print(book)
58
59     def check_out_book(self, title):
60         for book in self.books:
61             if book.title == title:
```



```
43 class Library:
59     def check_out_book(self, title):
62         if book.is_available:
63             book.is_available = False
64             print(f"You have checked out '{book.title}'")
65             return
66         else:
67             print(f"Sorry, '{book.title}' is currently checked out.")
68             return
69         print(f"Book '{title}' not found in the library.")
70
71     def return_book(self, title):
72         for book in self.books:
73             if book.title == title:
74                 if not book.is_available:
75                     book.is_available = True
76                     print(f"You have returned '{book.title}'")
77                     return
78                 else:
79                     print(f"'{book.title}' was not checked out.")
80                     return
81             print(f"Book '{title}' not found in the library.")
82 def main():
83     library = Library()
84     while True:
85         print("\nLibrary Management System")
86         print("1. Add Book")
87         print("2. Display Books")
88         print("3. Check Out Book")
```



Output:



The screenshot shows a Visual Studio Code editor window with a Python file named `Alpy` open. The code in the editor is as follows:

```
110         print("Invalid choice. Please try again.")
111     if __name__ == "__main__":
112         main()
113
114
```

The terminal output shows the execution of the script. It prompts the user to enter a book author, then displays a menu for the Library Management System. The user enters '1' for 'Add Book', then enters 'Python Basics' as the book title and 'john' as the author. The output confirms that the book has been added to the library.

```
Enter book author: c:\Users\reddy\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/
reddy/OneDrive/Desktop/AI/AI.py
Book 'python' added to the library.

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5): 1
Enter book title: Python Basics
Enter book author: john
Book ' Python Basics' added to the library.

Library Management System
1. Add Book
2. Display Books
3. Check Out Book
4. Return Book
5. Exit
Enter your choice (1-5):
```

The status bar at the bottom indicates the current line and column (Ln 113, Col 5), the file encoding (UTF-8), the line ending (CRLF), the file type (Python), and the default interpreter path (defaultInterpreterPath: 3.13.9.final.0).

Task Description #4 (AI-Assisted Code Completion for Class-Based Attendance System)

Task: Use an AI tool to generate an attendance management class.

Prompt: “Generate a Python class to mark and display student attendance using loops.”

Expected Output:

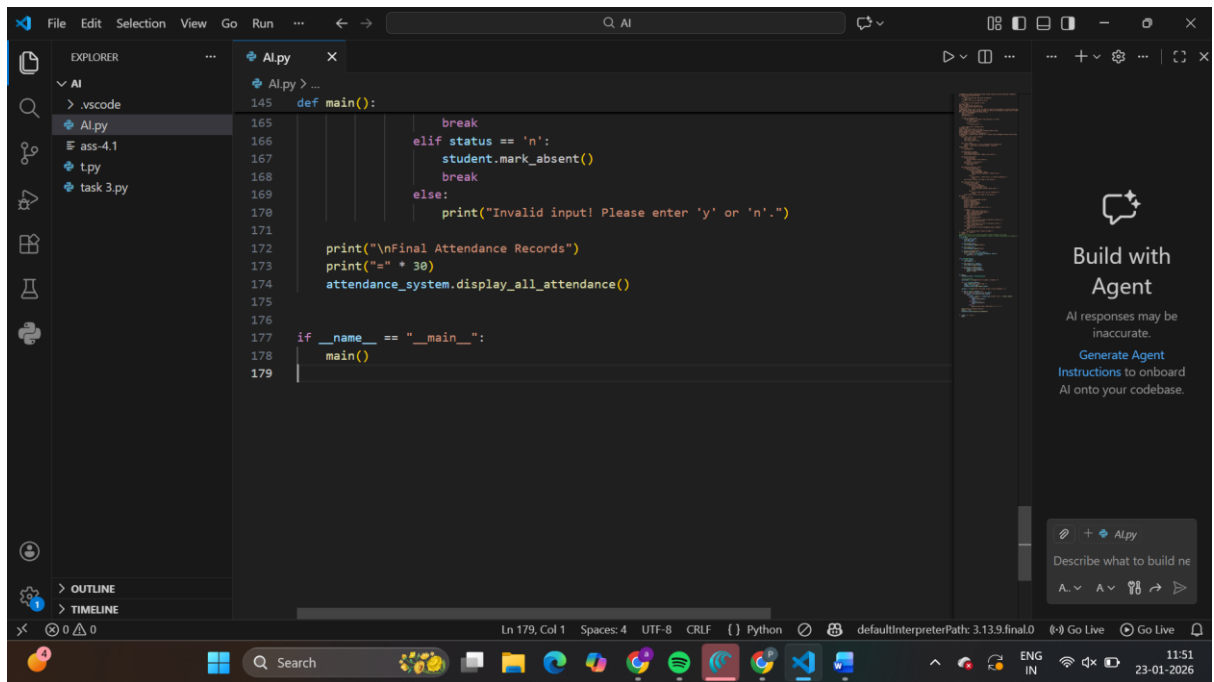
- AI-generated attendance logic.
- Correct display of attendance.

This screenshot shows the first part of a Python program in VS Code. The Explorer sidebar on the left shows a project named 'AI' with files '.vscode', 'Al.py', 'ass-4.1', 'tpy', and 'task 3.py'. The main editor window displays the code for the 'Student' and 'AttendanceSystem' classes. The 'Student' class has methods for initialization, marking attendance, and displaying attendance. The 'AttendanceSystem' class has methods for adding students and displaying all attendance. The status bar at the bottom indicates the cursor is at line 179, column 1, in a Python file.

```
113 ##Generate a Python class to mark and display student attendance using loops.
114 ##The program should allow marking students as present or absent and display the attendance
115 class Student:
116     def __init__(self, name):
117         self.name = name
118         self.attendance = []
119
120     def mark_present(self):
121         self.attendance.append("Present")
122
123     def mark_absent(self):
124         self.attendance.append("Absent")
125
126     def display_attendance(self):
127         print(f"Attendance for {self.name}:")
128         for i, status in enumerate(self.attendance, start=1):
129             print(f"Day {i}: {status}")
130
131
132 class AttendanceSystem:
133     def __init__(self):
134         self.students = []
135
136     def add_student(self, student):
137         self.students.append(student)
138
139     def display_all_attendance(self):
140         for student in self.students:
141             student.display_attendance()
```

This screenshot shows the second part of the Python program in VS Code, specifically the 'main' function. The Explorer sidebar remains the same. The main editor window displays the code for the 'main' function, which handles user input for the number of students, their names, and the number of days for attendance tracking. It uses loops and conditional statements to mark students as present or absent. The status bar at the bottom indicates the cursor is at line 179, column 1, in a Python file.

```
143
144
145 def main():
146     attendance_system = AttendanceSystem()
147
148     # Input validation
149     num_students = int(input("Enter the number of students: "))
150
151     for _ in range(num_students):
152         name = input("Enter student name: ")
153         student = Student(name)
154         attendance_system.add_student(student)
155
156     num_days = int(input("Enter the number of days to mark attendance: "))
157
158     for day in range(1, num_days + 1):
159         print(f"\nMarking attendance for Day {day}")
160         for student in attendance_system.students:
161             while True:
162                 status = input(f"Is {student.name} present? (y/n): ").strip().lower()
163                 if status == 'y':
164                     student.mark_present()
165                     break
166                 elif status == 'n':
167                     student.mark_absent()
168                     break
169                 else:
170                     print("Invalid input! Please enter 'y' or 'n'.")
171
```



Output:

File Edit Selection View Go Run ... AI

EXPLORER

AI
 .vscode
 Al.py
 ass-4.1
 tpy
 task 3.py

Al.py

145 def main():
PS C:\Users\reddy\OneDrive\Desktop\AI> & c:\Users\reddy\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/Users/reddy/OneDrive/Desktop/AI/AI.py
Enter the number of students: 4
Enter student name: Ashwutha
Enter student name: Harshini
Enter student name: Akshitha
Enter student name: varshitha
Enter the number of days to mark attendance: 4

Marking attendance for Day 1
Is Ashwutha present? (y/n): y
Is Harshini present? (y/n): y
Is Akshitha present? (y/n): y
Is varshitha present? (y/n): y

Marking attendance for Day 2
Is Ashwutha present? (y/n): n
Is Harshini present? (y/n): y
Is Akshitha present? (y/n): y
Is varshitha present? (y/n): y

Marking attendance for Day 3
Is Ashwutha present? (y/n): y
Is Harshini present? (y/n): n
Is Akshitha present? (y/n): n
Is varshitha present? (y/n): y

Marking attendance for Day 4
Is Ashwutha present? (y/n): y

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

powerShell

Build with Agent

AI responses may be inaccurate.
Generate Agent
Instructions to onboard AI onto your codebase.

+ Al.py
Describe what to build ne
A... A... ? ? →

Ln 179, Col 1 Spaces: 4 UTF-8 CRLF {} Python defaultInterpreterPath: 3.13.9.final.0 Go Live Go Live 11:51 23-01-2026

File Edit Selection View Go Run ... AI

EXPLORER

AI
 .vscode
 Al.py
 ass-4.1
 tpy
 task 3.py

Al.py

145 def main():
Is Akshitha present? (y/n): n
Is varshitha present? (y/n): y

Marking attendance for Day 4
Is Ashwutha present? (y/n): y
Is Harshini present? (y/n): y
Is Akshitha present? (y/n): y
Is varshitha present? (y/n): y

Final Attendance Records
=====

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

powerShell

Build with Agent

AI responses may be inaccurate.
Generate Agent
Instructions to onboard AI onto your codebase.

+ Al.py
Describe what to build ne
A... A... ? ? →

Attendance for Ashwutha:

Day 1: Present
Day 2: Absent
Day 3: Present
Day 4: Present

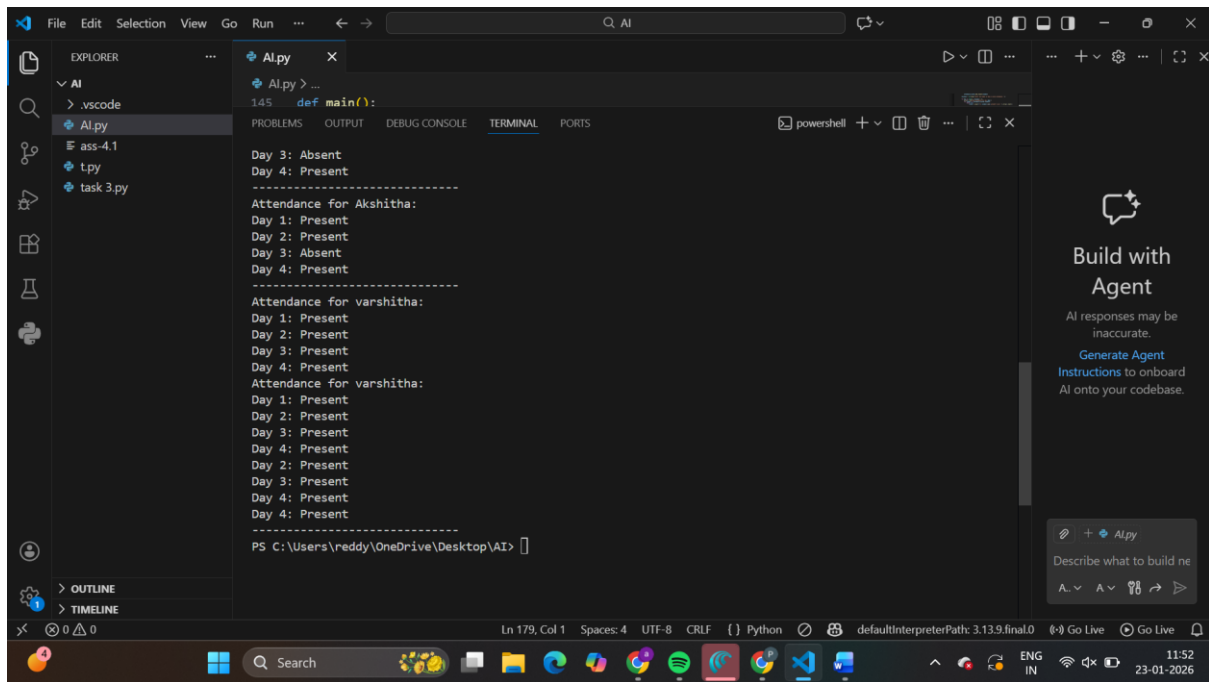
Attendance for Harshini:

Day 1: Present
Day 2: Present
Day 3: Absent
Day 4: Present

Attendance for Akshitha:

Day 1: Present
Day 2: Present
Day 3: Absent
Day 4: Present

Ln 179, Col 1 Spaces: 4 UTF-8 CRLF {} Python defaultInterpreterPath: 3.13.9.final.0 Go Live Go Live 11:51 23-01-2026



Task Description #5 (AI-Based Code Completion for Conditional

Menu Navigation)

Task: Use an AI tool to complete a navigation menu.

Prompt: “Generate a Python program using loops and conditionals to simulate an ATM menu.”

Expected Output:

- AI-generated menu logic.
- Correct option handling.
- Output verification.

