AI ASSIGNMENT-6.5

Name : B.Harshini

Hall No. : 2303A52242

Batch : 36

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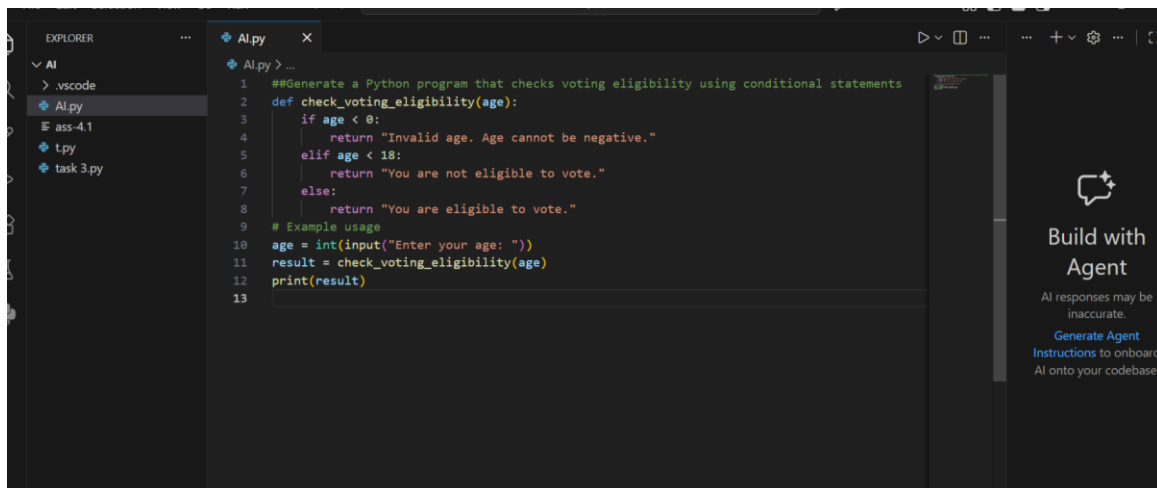
Eligibility Check)

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.

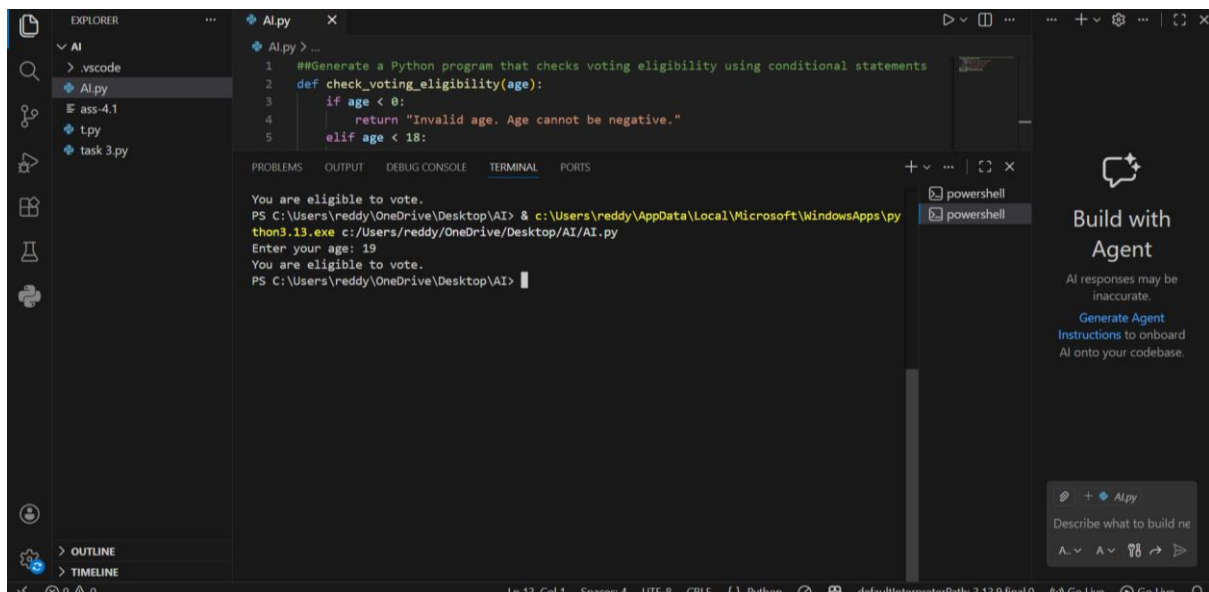• Correct eligibility decisions.

• Explanation of conditions.

Output:

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

String Processing)

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

Prompt:
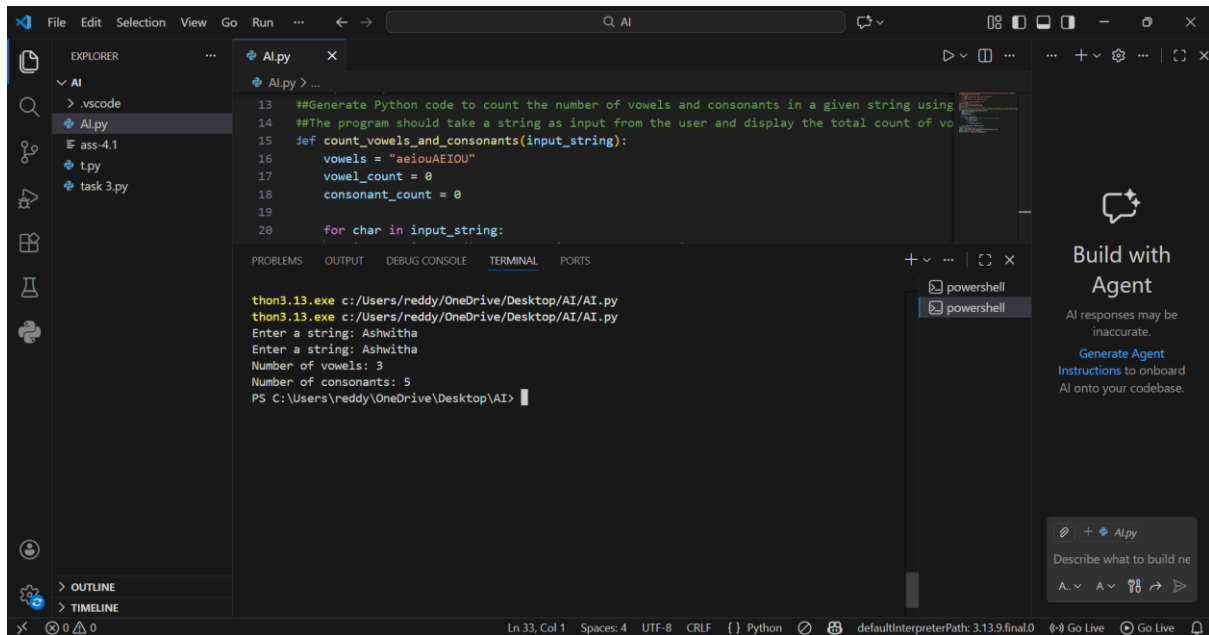
"Generate Python code to count vowels and consonants in a string

using a loop."

Expected Output:

• AI-generated string processing logic.

• Correct counts.

• Output verification.

Output:

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.
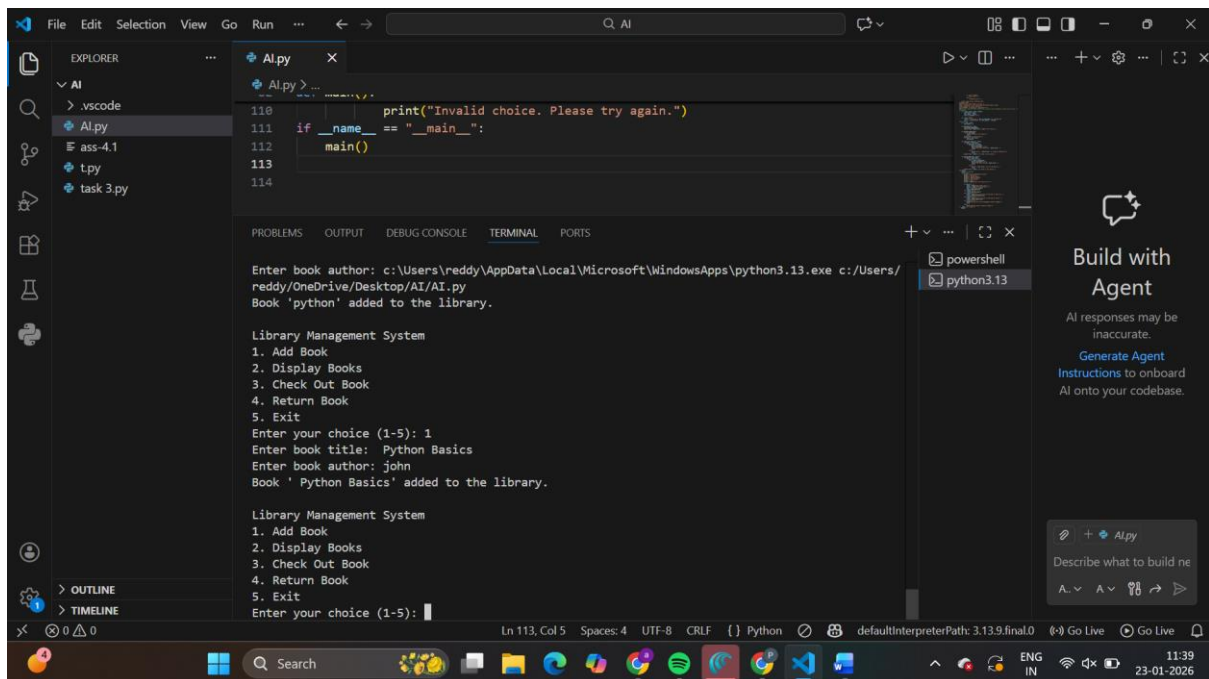
```python
##Generate a complete Python program for a simple Library Management System using classes,
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_available = True

    def __str__(self):
        status = "Available" if self.is_available else "Checked Out"
        return f"'{self.title}' by {self.author} - {status}"
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)
        print(f"Book '{book.title}' added to the library.")

    def display_books(self):
        if not self.books:
            print("No books in the library.")
            return
        print("Books in the library:")
        for book in self.books:
            print(book)

    def check_out_book(self, title):
        for book in self.books:
            if book.title == title:
```
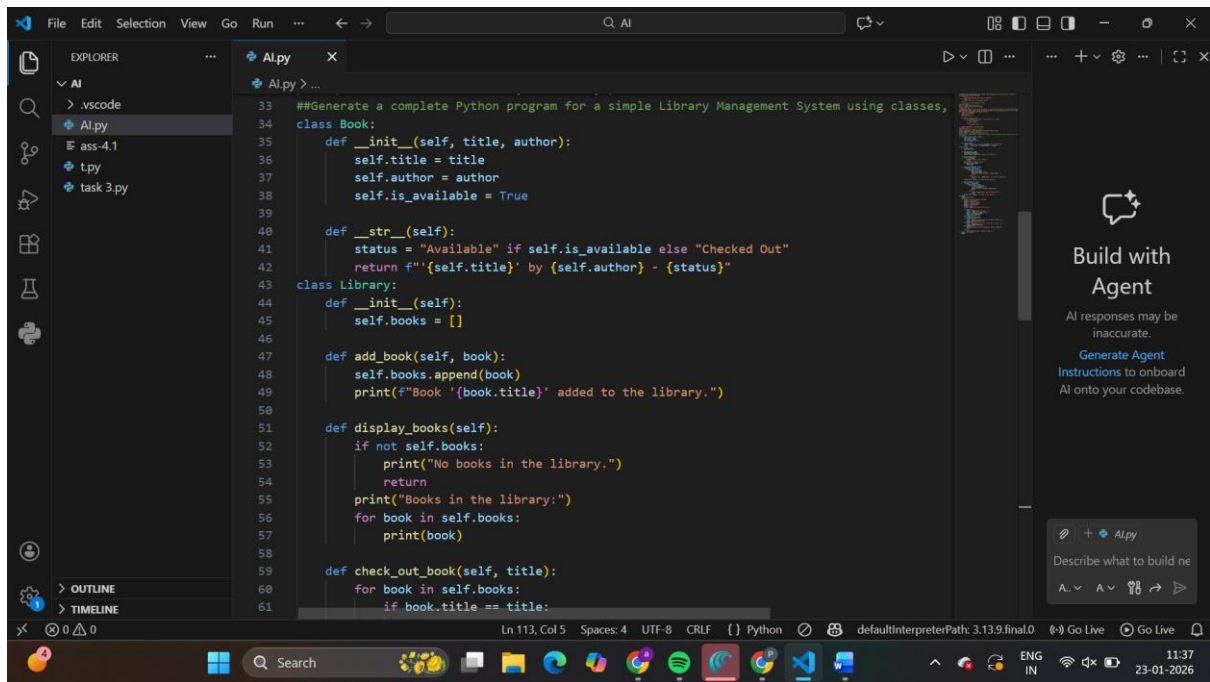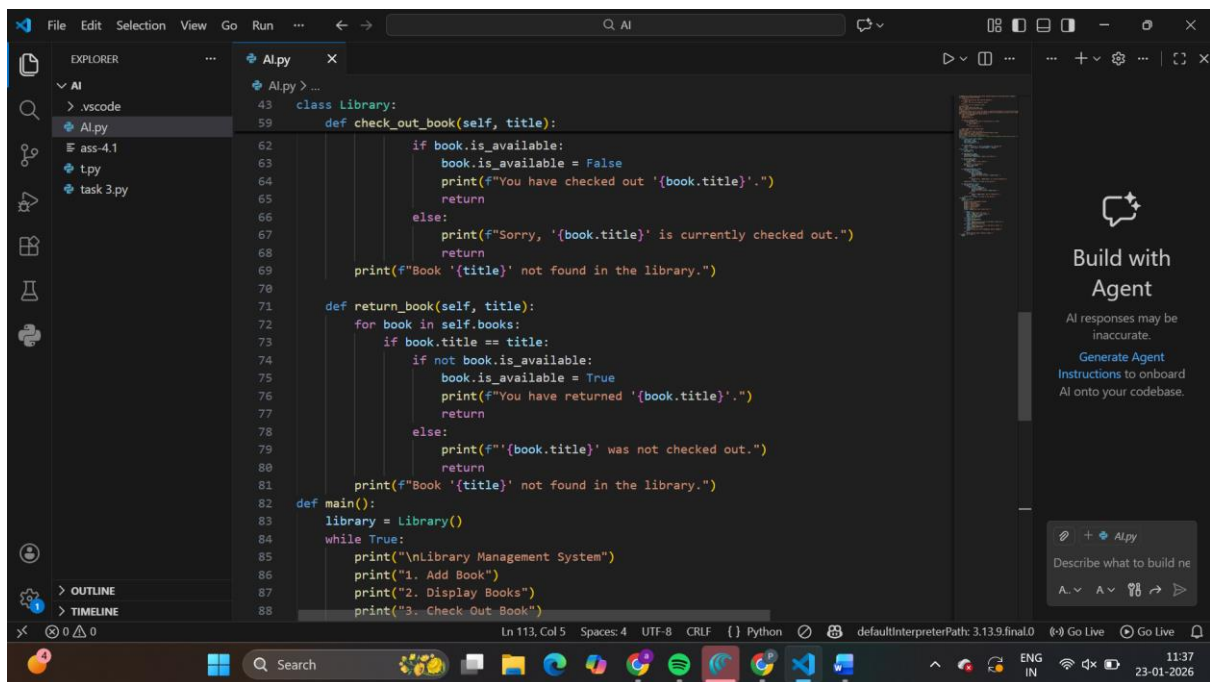
```python
class Library:
    def check_out_book(self, title):
            if book.is_available:
                book.is_available = False
                print(f"You have checked out '{book.title}'.")
                return
            else:
                print(f"Sorry, '{book.title}' is currently checked out.")
                return
        print(f"Book '{title}' not found in the library.")

    def return_book(self, title):
        for book in self.books:
            if book.title == title:
                if not book.is_available:
                    book.is_available = True
                    print(f"You have returned '{book.title}'.")
                    return
                else:
                    print(f"'{book.title}' was not checked out.")
                    return
        print(f"Book '{title}' not found in the library.")
def main():
    library = Library()
    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. Display Books")
        print("3. Check Out Book")
```
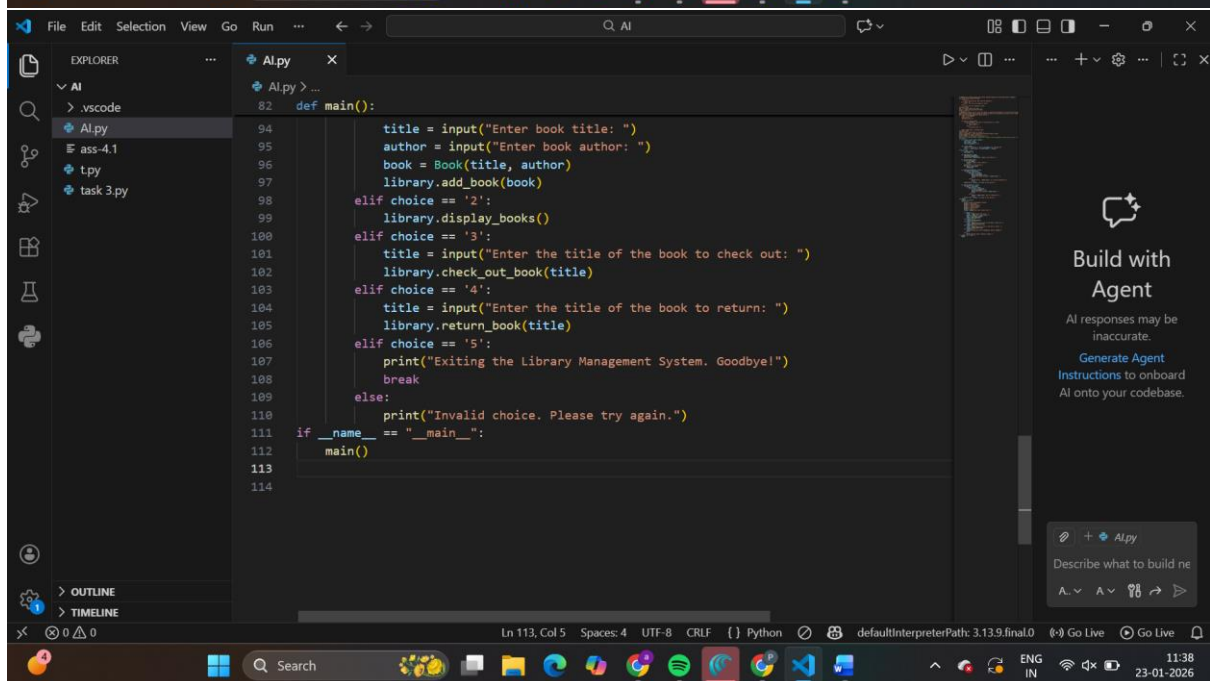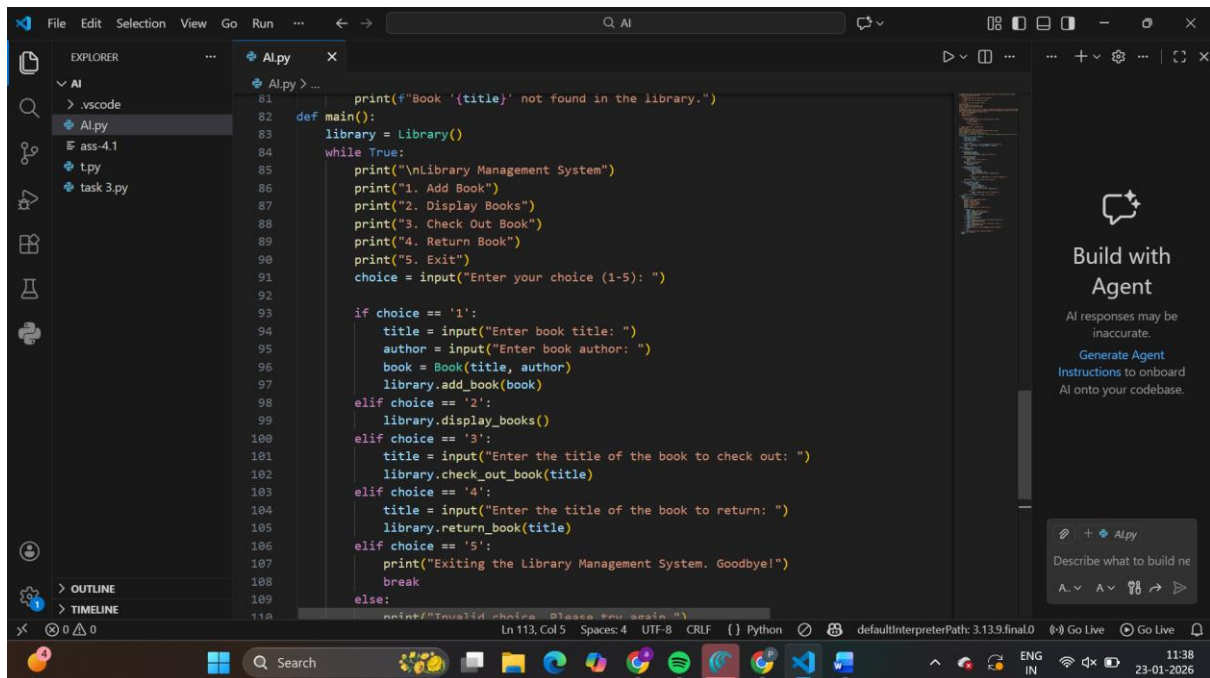
```python
        print(f"Book '{title}' not found in the library.")
def main():
    library = Library()
    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. Display Books")
        print("3. Check Out Book")
        print("4. Return Book")
        print("5. Exit")
        choice = input("Enter your choice (1-5): ")

        if choice == '1':
            title = input("Enter book title: ")
            author = input("Enter book author: ")
            book = Book(title, author)
            library.add_book(book)
        elif choice == '2':
            library.display_books()
        elif choice == '3':
            title = input("Enter the title of the book to check out: ")
            library.check_out_book(title)
        elif choice == '4':
            title = input("Enter the title of the book to return: ")
            library.return_book(title)
        elif choice == '5':
            print("Exiting the Library Management System. Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")
if __name__ == "__main__":
    main()
```

Output:

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.

```python
##Generate a Python class to mark and display student attendance using loops.
##The program should allow marking students as present or absent and display the attendance
class Student:
    def __init__(self, name):
        self.name = name
        self.attendance = []

    def mark_present(self):
        self.attendance.append("Present")

    def mark_absent(self):
        self.attendance.append("Absent")

    def display_attendance(self):
        print(f"Attendance for {self.name}:")
        for i, status in enumerate(self.attendance, start=1):
            print(f"Day {i}: {status}")


class AttendanceSystem:
    def __init__(self):
        self.students = []

    def add_student(self, student):
        self.students.append(student)

    def display_all_attendance(self):
        for student in self.students:
            student.display_attendance()
```

```python
def main():
    attendance_system = AttendanceSystem()

    # Input validation
    num_students = int(input("Enter the number of students: "))

    for _ in range(num_students):
        name = input("Enter student name: ")
        student = Student(name)
        attendance_system.add_student(student)

    num_days = int(input("Enter the number of days to mark attendance: "))

    for day in range(1, num_days + 1):
        print(f"\nMarking attendance for Day {day}")
        for student in attendance_system.students:
            while True:
                status = input(f"Is {student.name} present? (y/n): ").strip().lower()
                if status == 'y':
                    student.mark_present()
                    break
                elif status == 'n':
                    student.mark_absent()
                    break
                else:
                    print("Invalid input! Please enter 'y' or 'n'.")
```

```python
def main():
            break
                    elif status == 'n':
                        student.mark_absent()
                        break
                    else:
                        print("Invalid input! Please enter 'y' or 'n'.")

        print("\nFinal Attendance Records")
        print("=" * 30)
        attendance_system.display_all_attendance()


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

Output:

Screen 1 (Top):

```
AI.py > ...
145    def main():
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS
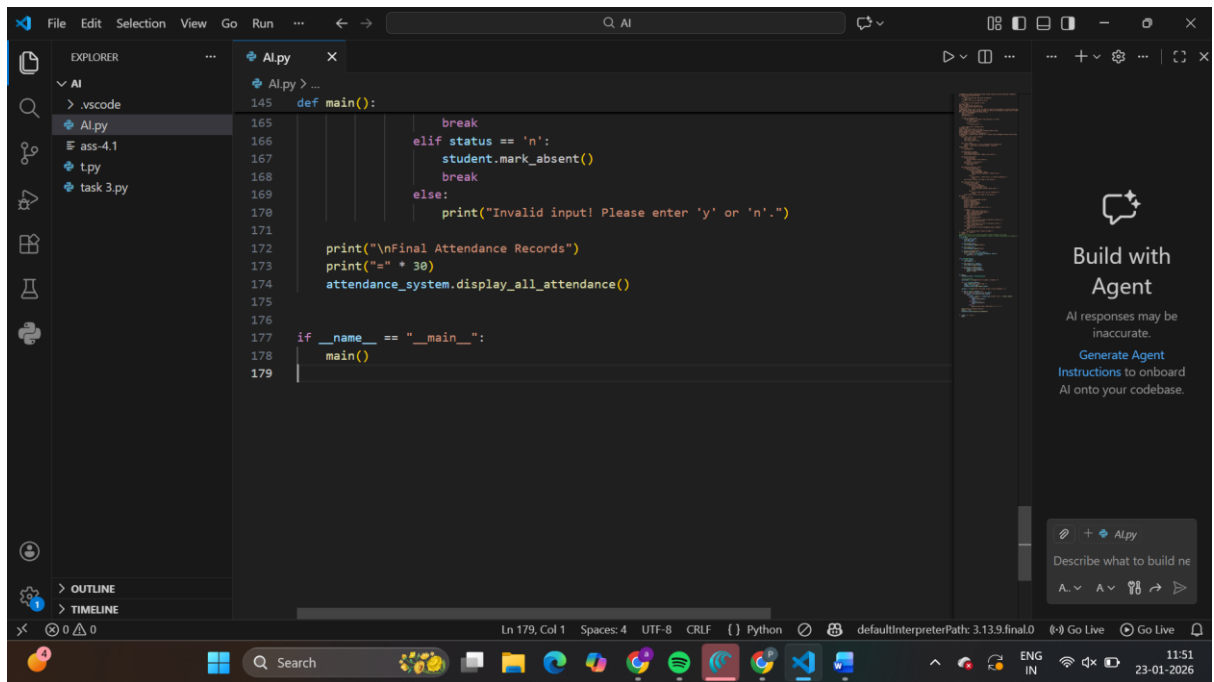
```
PS C:\Users\reddy\OneDrive\Desktop\AI> & c:\Users\reddy\AppData\Local\Microsoft\WindowsApps\python3.13.exe c:/U
sers/reddy/OneDrive/Desktop/AI/AI.py
Enter the number of students: 4
Enter student name: Ashwutha
Enter student name: Harshini
Enter student name: Akshitha
Enter student name: varshitha
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
```

Screen 2 (Bottom):

```
AI.py > ...
145    def main():
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

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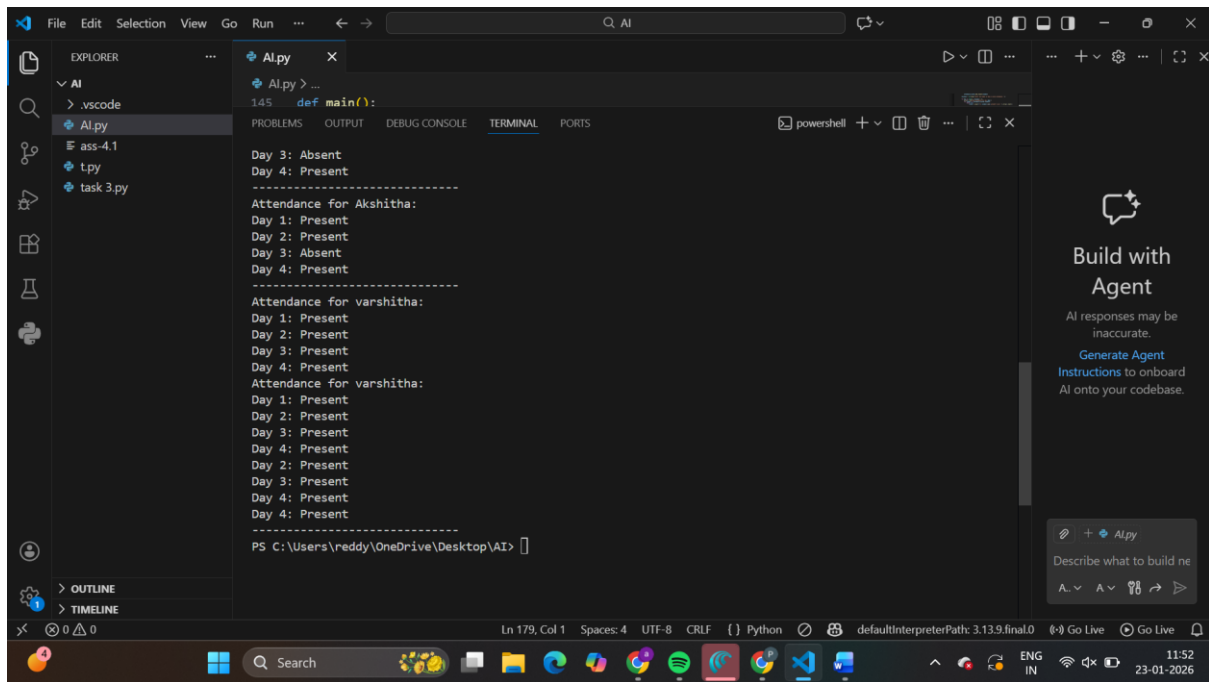
Task Description #5 (AI-Based Code Completion for Conditional

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.

```python
185    #Withdraw
186    #Exit
187    def atm_menu():
188        balance = 1000   # Initial balance
189
190        while True:
191            print("\nATM Menu:")
192            print("1. Check Balance")
193            print("2. Deposit")
194            print("3. Withdraw")
195            print("4. Exit")
196
197            choice = input("Please select an option (1-4): ")
198
199            if choice == '1':
200                print(f"Your current balance is: ${balance:.2f}")
201            elif choice == '2':
202                amount = float(input("Enter amount to deposit: $"))
203                if amount > 0:
204                    balance += amount
205                    print(f"${amount:.2f} deposited successfully.")
206                else:
207                    print("Invalid amount. Please enter a positive value.")
208            elif choice == '3':
209                amount = float(input("Enter amount to withdraw: $"))
210                if 0 < amount <= balance:
211                    balance -= amount
212                    print(f"${amount:.2f} withdrawn successfully.")
213                else:
```

```python
187    def atm_menu():
219                print("Invalid choice. Please select a valid option.")
220    if __name__ == "__main__":
221        atm_menu()
222
```

Terminal output:
```
Please select an option (1-4): 1
Your current balance is: $1000.00

ATM Menu:
1. Check Balance
2. Deposit
Your current balance is: $1000.00

ATM Menu:
1. Check Balance
2. Deposit
3. Withdraw
1. Check Balance
2. Deposit
3. Withdraw
3. Withdraw
4. Exit
Please select an option (1-4): 2
Enter amount to deposit: $1000
$1000.00 deposited successfully.
```

```python
def atm_menu():
            balance += amount
            print(f"${amount:.2f} deposited successfully.")
        else:
            print("Invalid amount. Please enter a positive value.")
    elif choice == '3':
        amount = float(input("Enter amount to withdraw: $"))
        if 0 < amount <= balance:
            balance -= amount
            print(f"${amount:.2f} withdrawn successfully.")
        else:
            print("Invalid amount. Please enter a positive value within your balance.")
    elif choice == '4':
        print("Thank you for using the ATM. Goodbye!")
        break
    else:
        print("Invalid choice. Please select a valid option.")
if __name__ == "__main__":
    atm_menu()
```