AI Assisted coding

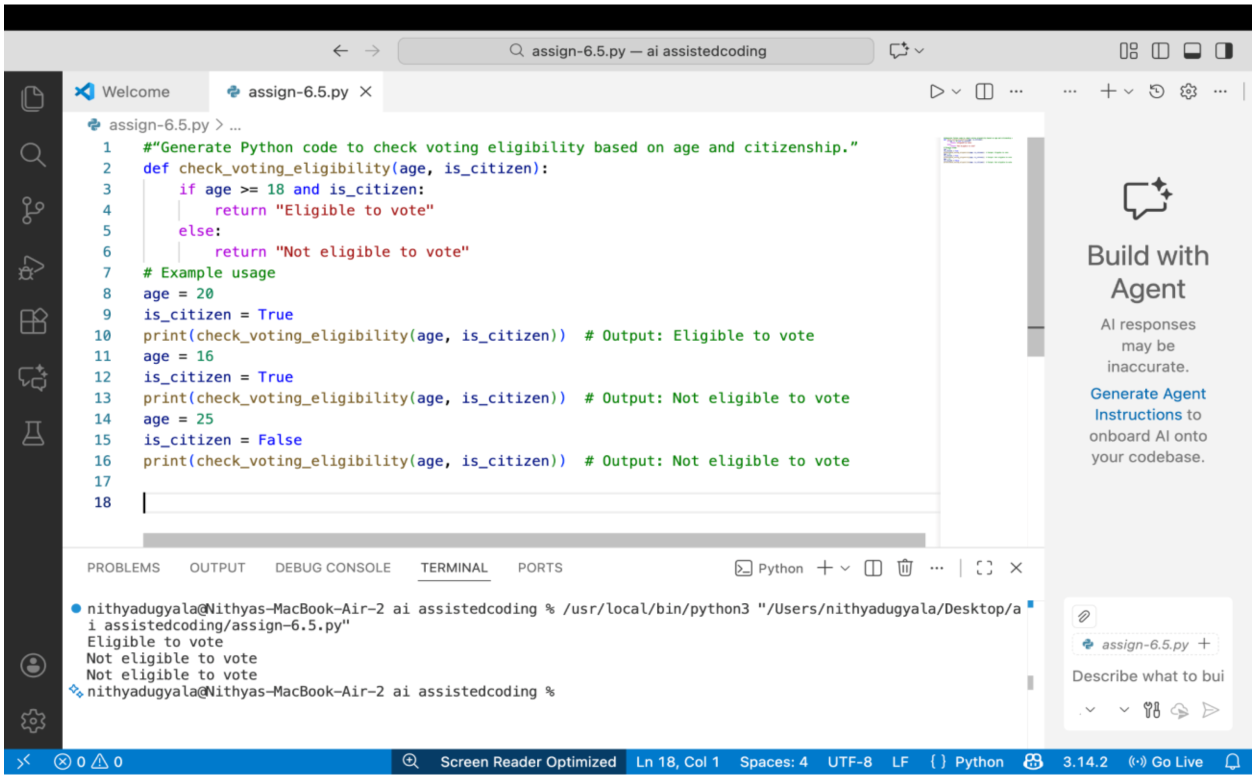
Assignment-6.5

S.Rajesh

2303a52301

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.



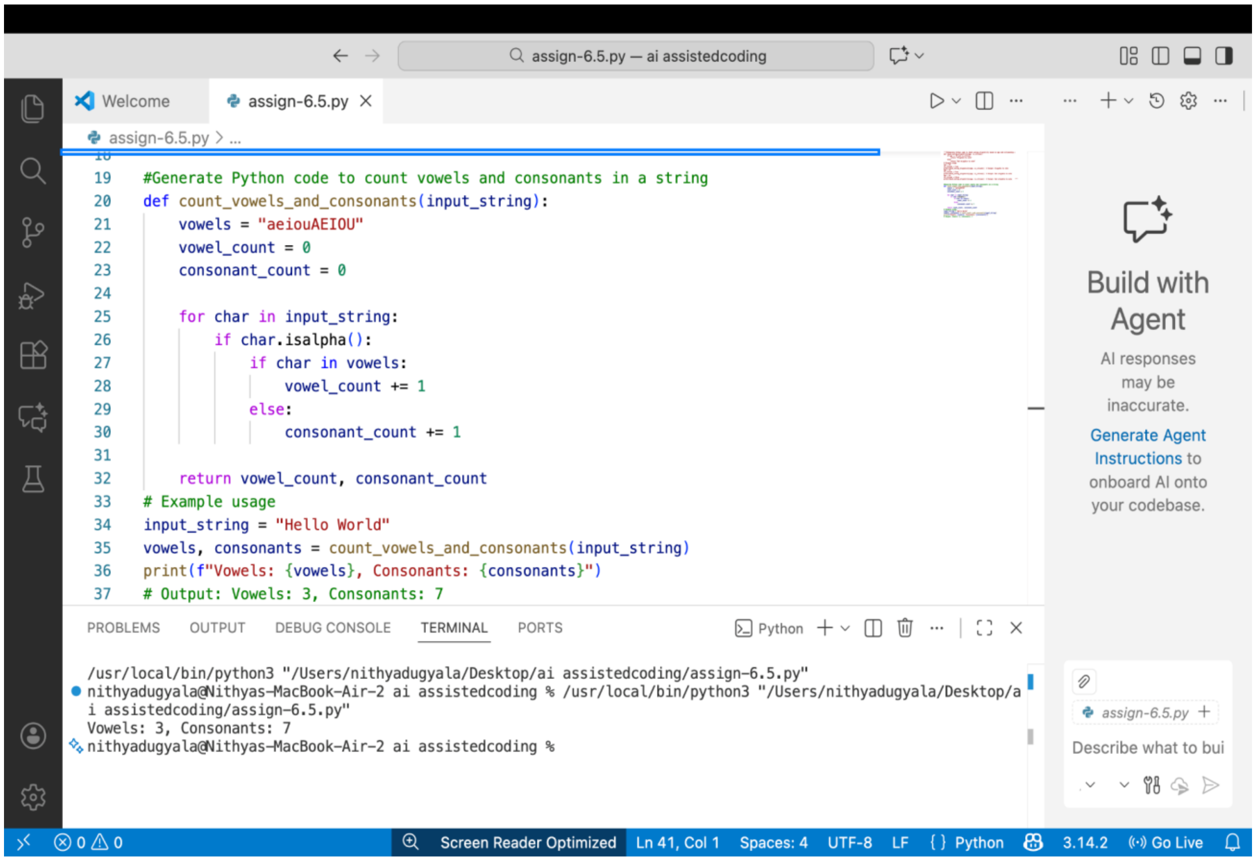
```python
#"Generate Python code to check voting eligibility based on age and citizenship."
def check_voting_eligibility(age, is_citizen):
    if age >= 18 and is_citizen:
        return "Eligible to vote"
    else:
        return "Not eligible to vote"
# Example usage
age = 20
is_citizen = True
print(check_voting_eligibility(age, is_citizen))  # Output: Eligible to vote
age = 16
is_citizen = True
print(check_voting_eligibility(age, is_citizen))  # Output: Not eligible to vote
age = 25
is_citizen = False
print(check_voting_eligibility(age, is_citizen))  # Output: Not eligible to vote

```

```
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/a
i assistedcoding/assign-6.5.py"
Eligible to vote
Not eligible to vote
Not eligible to vote
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

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.



```python
#Generate Python code to count vowels and consonants in a string
def count_vowels_and_consonants(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0

    for char in input_string:
        if char.isalpha():
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1

    return vowel_count, consonant_count
# Example usage
input_string = "Hello World"
vowels, consonants = count_vowels_and_consonants(input_string)
print(f"Vowels: {vowels}, Consonants: {consonants}")
# Output: Vowels: 3, Consonants: 7
```

```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/a
i assistedcoding/assign-6.5.py"
Vowels: 3, Consonants: 7
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

Task Description #3 (AI-Assisted Code Completion Reflection Task)

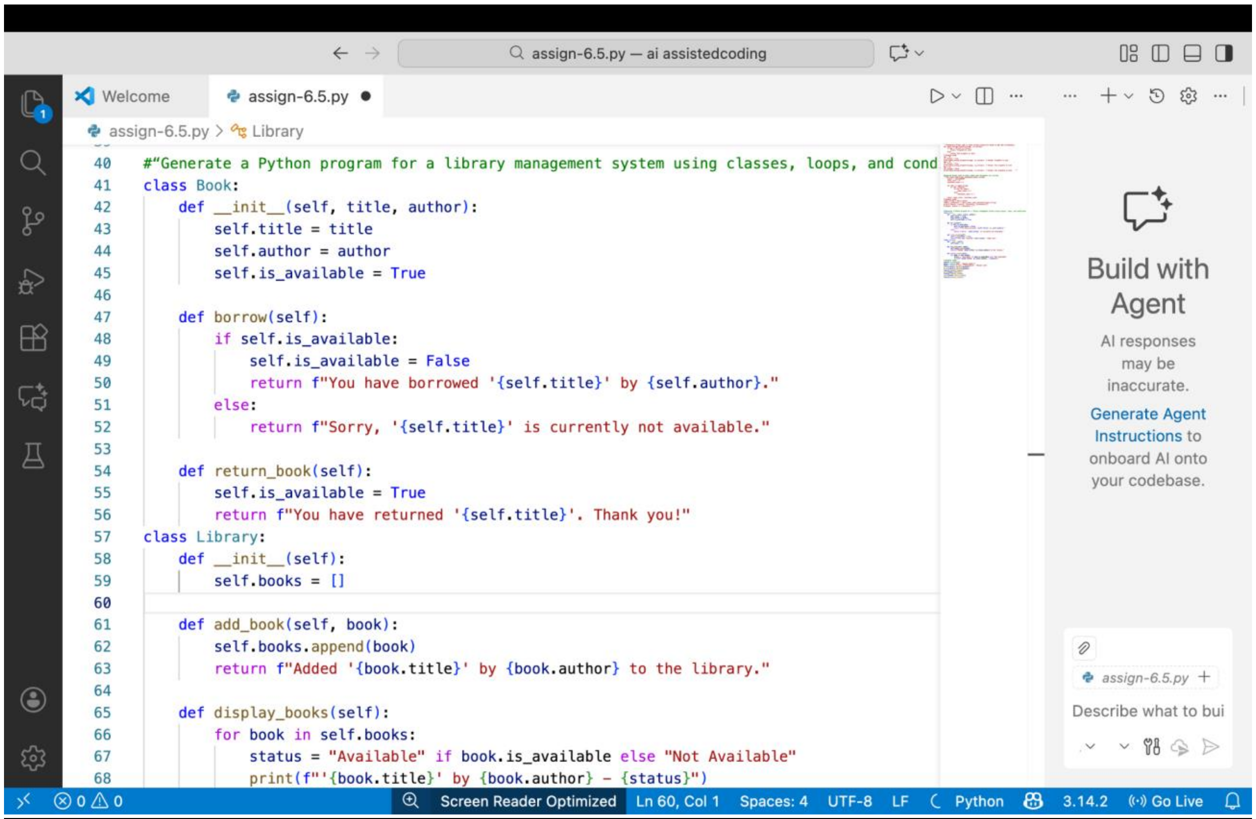
Task: Use an AI tool to generate a complete program using classes, loops, and conditionals.

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.



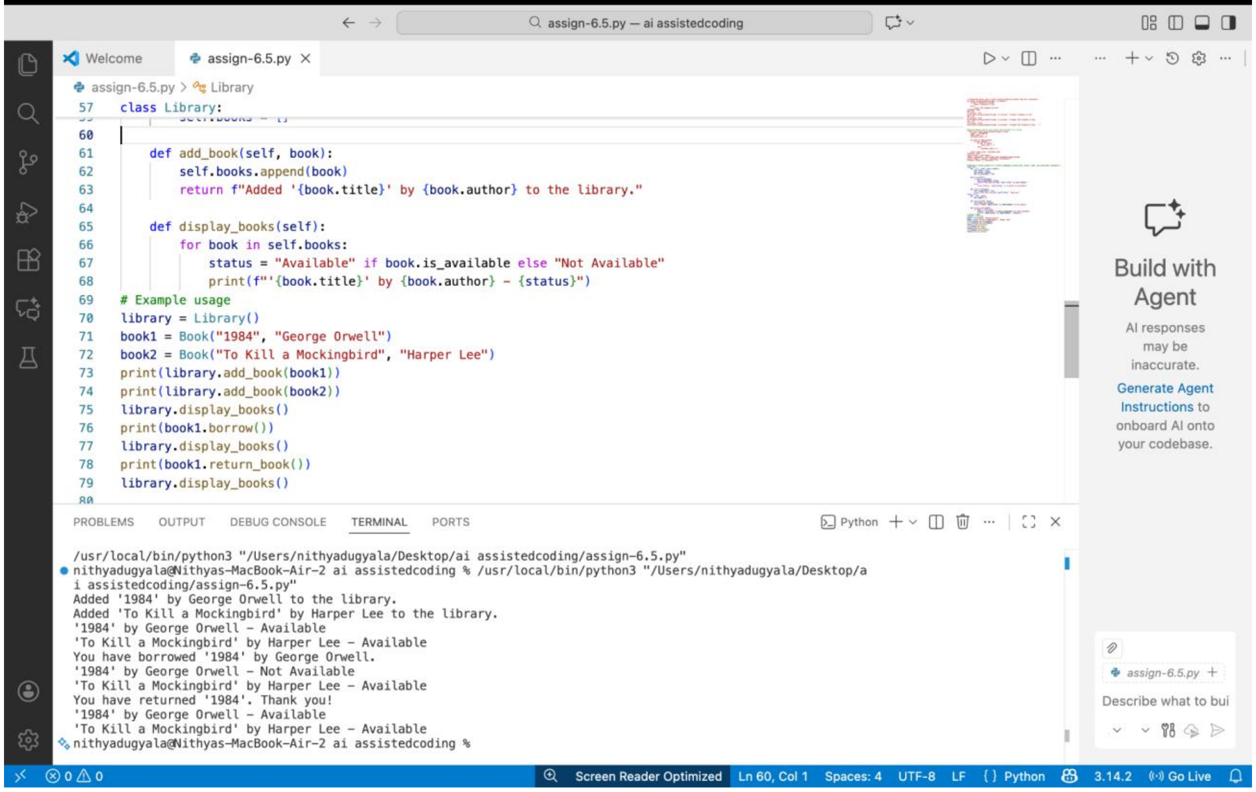
```python
#"Generate a Python program for a library management system using classes, loops, and cond
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.is_available = True

    def borrow(self):
        if self.is_available:
            self.is_available = False
            return f"You have borrowed '{self.title}' by {self.author}."
        else:
            return f"Sorry, '{self.title}' is currently not available."

    def return_book(self):
        self.is_available = True
        return f"You have returned '{self.title}'. Thank you!"
class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book):
        self.books.append(book)
        return f"Added '{book.title}' by {book.author} to the library."

    def display_books(self):
        for book in self.books:
            status = "Available" if book.is_available else "Not Available"
            print(f"'{book.title}' by {book.author} - {status}")
```



```python
class Library:

    def add_book(self, book):
        self.books.append(book)
        return f"Added '{book.title}' by {book.author} to the library."

    def display_books(self):
        for book in self.books:
            status = "Available" if book.is_available else "Not Available"
            print(f"'{book.title}' by {book.author} - {status}")
# Example usage
library = Library()
book1 = Book("1984", "George Orwell")
book2 = Book("To Kill a Mockingbird", "Harper Lee")
print(library.add_book(book1))
print(library.add_book(book2))
library.display_books()
print(book1.borrow())
library.display_books()
print(book1.return_book())
library.display_books()
```

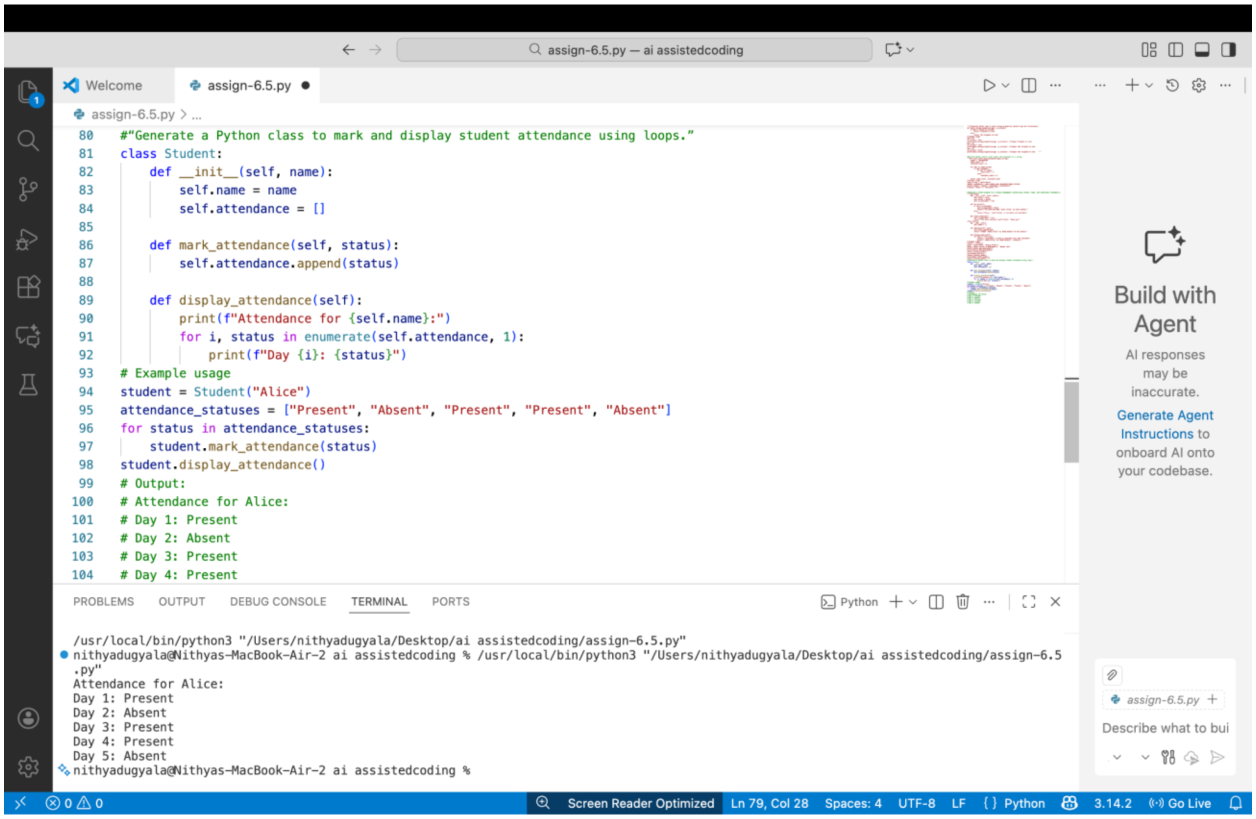
```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/a
i assistedcoding/assign-6.5.py"
Added '1984' by George Orwell to the library.
Added 'To Kill a Mockingbird' by Harper Lee to the library.
'1984' by George Orwell - Available
'To Kill a Mockingbird' by Harper Lee - Available
You have borrowed '1984' by George Orwell.
'1984' by George Orwell - Not Available
'To Kill a Mockingbird' by Harper Lee - Available
You have returned '1984'. Thank you!
'1984' by George Orwell - Available
'To Kill a Mockingbird' by Harper Lee - Available
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

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.
• Test cases.



assign-6.5.py — ai assistedcoding
Welcome
assign-6.5.py
assign-6.5.py > ...
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Describe what to bui
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Screen Reader Optimized Ln 79, Col 28 Spaces: 4 UTF-8 LF { } Python 3.14.2 Go Live

```python
#“Generate a Python class to mark and display student attendance using loops.”
class Student:
    def __init__(self, name):
        self.name = name
        self.attendance = []

    def mark_attendance(self, status):
        self.attendance.append(status)

    def display_attendance(self):
        print(f"Attendance for {self.name}:")
        for i, status in enumerate(self.attendance, 1):
            print(f"Day {i}: {status}")
# Example usage
student = Student("Alice")
attendance_statuses = ["Present", "Absent", "Present", "Present", "Absent"]
for status in attendance_statuses:
    student.mark_attendance(status)
student.display_attendance()
# Output:
# Attendance for Alice:
# Day 1: Present
# Day 2: Absent
# Day 3: Present
# Day 4: Present
```

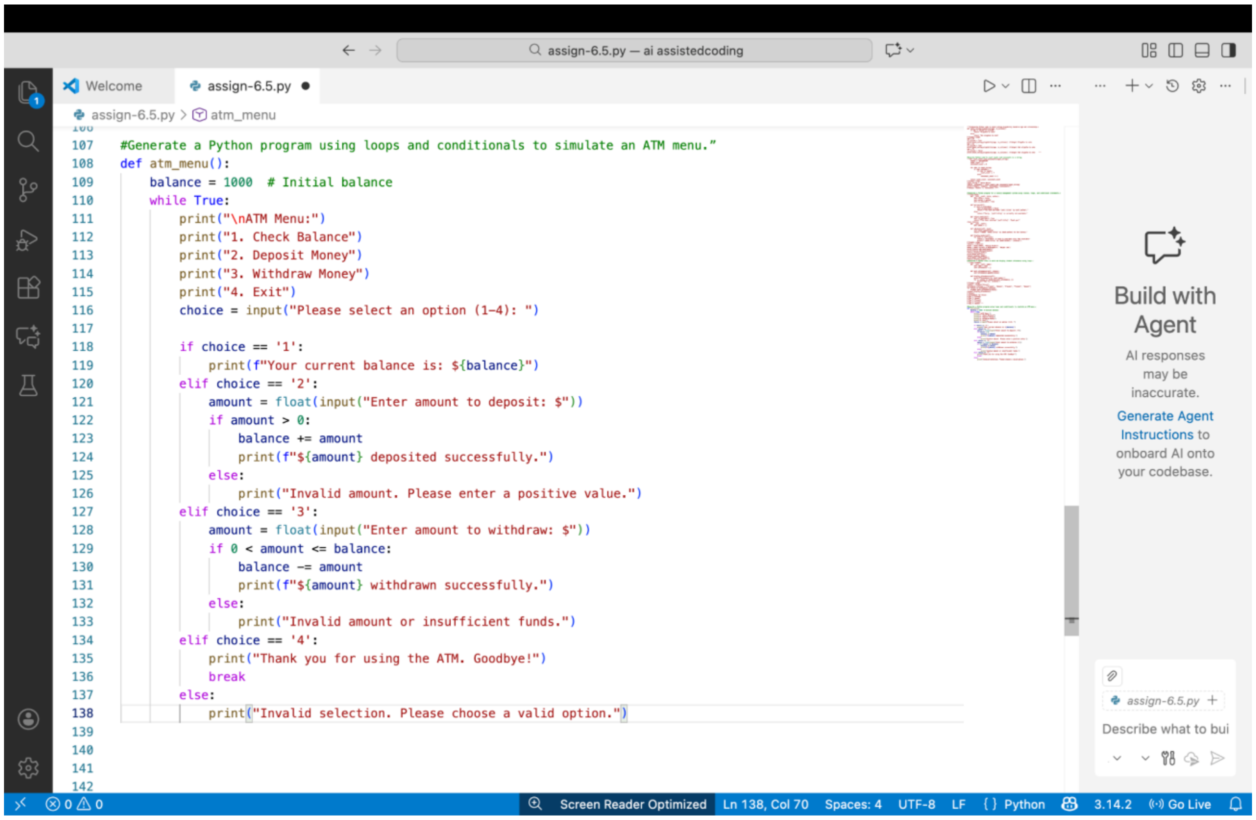
```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5
.py"
Attendance for Alice:
Day 1: Present
Day 2: Absent
Day 3: Present
Day 4: Present
Day 5: Absent
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding %
```

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

Menu Navigation)

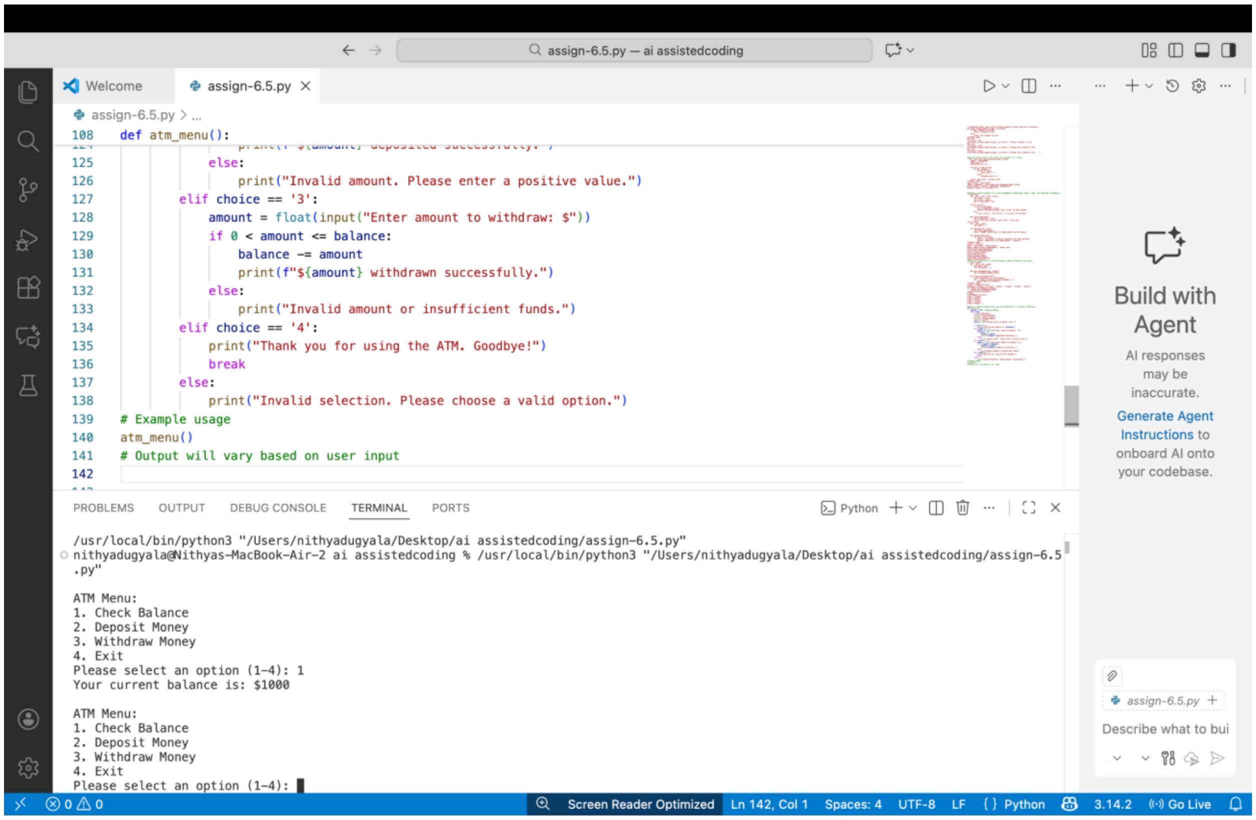
Task: Use an AI tool to complete a navigation menu.
Prompt: “Generate a Python program using loops and conditionals
to simulate an ATM menu.”
Expected Output:

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



```
#Generate a Python program using loops and conditionals to simulate an ATM menu."
def atm_menu():
    balance = 1000  # Initial balance
    while True:
        print("\nATM Menu:")
        print("1. Check Balance")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Exit")
        choice = input("Please select an option (1-4): ")

        if choice == '1':
            print(f"Your current balance is: ${balance}")
        elif choice == '2':
            amount = float(input("Enter amount to deposit: $"))
            if amount > 0:
                balance += amount
                print(f"${amount} 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} withdrawn successfully.")
            else:
                print("Invalid amount or insufficient funds.")
        elif choice == '4':
            print("Thank you for using the ATM. Goodbye!")
            break
        else:
            print("Invalid selection. Please choose a valid option.")
```



```
        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} withdrawn successfully.")
        else:
            print("Invalid amount or insufficient funds.")
    elif choice == '4':
        print("Thank you for using the ATM. Goodbye!")
        break
    else:
        print("Invalid selection. Please choose a valid option.")
# Example usage
atm_menu()
# Output will vary based on user input
```

```
/usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5.py"
nithyadugyala@Nithyas-MacBook-Air-2 ai assistedcoding % /usr/local/bin/python3 "/Users/nithyadugyala/Desktop/ai assistedcoding/assign-6.5
.py"

ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4): 1
Your current balance is: $1000

ATM Menu:
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Please select an option (1-4):
```