

AI ASSIGNMENT-6.5

Name : B.Harshini

Hall No. : 2303A52242

Batch : 36

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

Eligibility Check)

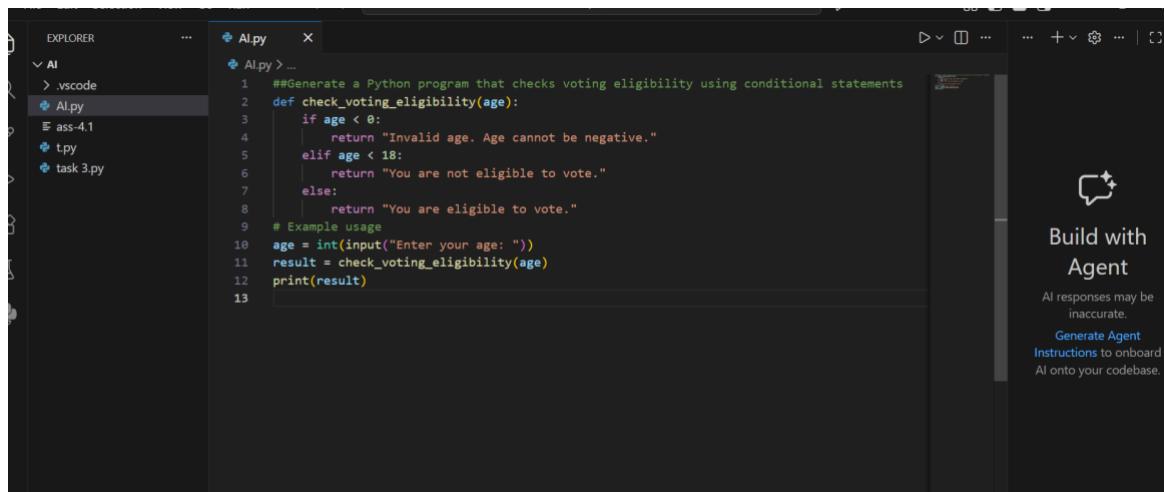
Task: Use an AI tool to generate eligibility logic.

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 the VS Code interface with the Explorer sidebar open, displaying files like AI, .vscode, Al.py, ass-4.1, tpy, and task 3.py. The Al.py file is selected in the Explorer. The main editor tab is titled 'Alpy' and contains the following Python code:

```
##Generate a Python program that checks voting eligibility using conditional statements
def check_voting_eligibility(age):
    if age < 0:
        return "Invalid age. Age cannot be negative."
    elif age < 18:
        return "You are not eligible to vote."
    else:
        return "You are eligible to vote."
# Example usage
age = int(input("Enter your age: "))
result = check_voting_eligibility(age)
print(result)
```

To the right of the editor, there is a sidebar with the heading 'Build with Agent'. It includes a note that 'AI responses may be inaccurate.', a 'Generate Agent' button, and a link to 'Instructions to onboard AI onto your codebase.'

Output:

The screenshot shows the VS Code interface with the AI extension active. The Explorer sidebar shows files like AI.py, ass-4.1, tpy, and task 3.py. The AI.py file is open in the editor, displaying the following Python code:

```
#Generate a Python program that checks voting eligibility using conditional statements
def check_voting_eligibility(age):
    if age < 0:
        return "Invalid age. Age cannot be negative."
    elif age < 18:
        You are eligible to vote.
```

The terminal window shows the output of running the code:

```
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 your age: 19
You are eligible to vote.
PS C:\Users\reddy\OneDrive\Desktop\AI>
```

A sidebar on the right titled "Build with Agent" contains instructions to onboard AI onto the codebase.

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.

The screenshot shows the VS Code interface with the AI extension active. The Explorer sidebar shows files like AI.py, ass-4.1, tpy, and task 3.py. The AI.py file is open in the editor, displaying the following Python code:

```
##Generate Python code to count the number of vowels and consonants in a given string using a loop.
def count_vowels_and_consonants(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0

    for char in input_string:
        if char.isalpha(): # Check if the character is a letter
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1

    return vowel_count, consonant_count

# Example usage
input_string = input("Enter a string: ")
vowels, consonants = count_vowels_and_consonants(input_string)
print(f"Number of vowels: {vowels}")
print(f"Number of consonants: {consonants}")
```

The terminal window shows the output of running the code:

```
Number of consonants: 5
PS C:\Users\reddy\OneDrive\Desktop\AI>
```

A sidebar on the right titled "Build with Agent" contains instructions to onboard AI onto the codebase.

Output:

The screenshot shows the Visual Studio Code interface with the AI extension active. The Explorer sidebar on the left has a tree view with items like '.vscode', 'AI', 'Alpy' (which is expanded), 'ass-4.1', 'tpy', and 'task 3.py'. The Alpy folder contains several files. The main editor area shows a Python script named 'Alpy.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:
21          if char in vowels:
22              vowel_count += 1
23          else:
24              consonant_count += 1
25
26      print("Number of vowels:", vowel_count)
27      print("Number of consonants:", consonant_count)
28
29  if __name__ == "__main__":
30      input_string = input("Enter a string: ")
31      count_vowels_and_consonants(input_string)
```

The terminal below the editor shows the output of running 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: Ashwatha
Enter a string: Ashwatha
Number of vowels: 3
Number of consonants: 5
PS C:\Users\reddy\OneDrive\Desktop\AI>
```

The right side of the interface features the 'Build with Agent' panel, which includes a message bubble icon, a 'Build with Agent' button, and a note about AI responses being inaccurate. It also has a 'Describe what to build next' input field and a 'Go Live' button.

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.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files in the workspace: `.vscode`, `ass-4.1`, `t.py`, and `task 3.py`.
- Code Editor:** Displays the `Al.py` file with the following code:

```
#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:
                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 main():
    library = Library()
    while True:
        print("\nLibrary Management System")
        print("1. Add Book")
        print("2. Display Books")
        print("3. Check-Out Book")

        choice = input("Enter your choice (1/2/3): ")
        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 book title to check out: ")
            library.check_out_book(title)
        else:
            print("Invalid choice. Please enter 1, 2, or 3.")
```

Right Panel: An AI assistant sidebar titled "Build with Agent". It includes a message "AI responses may be inaccurate.", a "Generate Agent" button, and an "Instructions to onboard AI onto your codebase" section.

Bottom Bar: Shows the current file is `Al.py`, the interpreter path is `3.13.9.final.0`, and the date is `23-01-2026`.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files in the workspace: `.vscode`, `ass-4.1`, `t.py`, and `task 3.py`.
- Code Editor:** Displays the `Al.py` file with the following code:

```
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")

        choice = input("Enter your choice (1/2/3): ")
        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 book title to check out: ")
            library.check_out_book(title)
        else:
            print("Invalid choice. Please enter 1, 2, or 3.")
```

Right Panel: An AI assistant sidebar titled "Build with Agent". It includes a message "AI responses may be inaccurate.", a "Generate Agent" button, and an "Instructions to onboard AI onto your codebase" section.

Bottom Bar: Shows the current file is `Al.py`, the interpreter path is `3.13.9.final.0`, and the date is `23-01-2026`.

The screenshot shows two instances of the Visual Studio Code (VS Code) interface, both displaying the same Python code for a Library Management System. The code is contained in a file named `Al.py`.

```
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")
89         print("4. Return Book")
90         print("5. Exit")
91         choice = input("Enter your choice (1-5): ")
92
93         if choice == '1':
94             title = input("Enter book title: ")
95             author = input("Enter book author: ")
96             book = Book(title, author)
97             library.add_book(book)
98         elif choice == '2':
99             library.display_books()
100        elif choice == '3':
101            title = input("Enter the title of the book to check out: ")
102            library.check_out_book(title)
103        elif choice == '4':
104            title = input("Enter the title of the book to return: ")
105            library.return_book(title)
106        elif choice == '5':
107            print("Exiting the Library Management System. Goodbye!")
108            break
109        else:
110            print("Invalid choice. Please try again.")
111 if __name__ == "__main__":
112     main()
```

The interface includes the following elements:

- File Bar:** File, Edit, Selection, View, Go, Run, ...
- Search Bar:** AI
- Explorer Bar:** Shows files in the project: `.vscode`, `Al.py`, `ass-4.1`, `tpy`, and `task 3.py`.
- Code Editor:** The main workspace where the Python code is written.
- Output Panel:** Shows the status message "Ln 113, Col 5 Spaces: 4 CRLF {} Python".
- Bottom Status Bar:** defaultInterpreterPath: 3.13.9.final.0, Go Live, Go Live, ENG IN, 23-01-2026.

A floating panel on the right side of the interface displays the text "Build with Agent" and provides instructions for AI integration.

Output:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** Shows files in the current workspace, including `.vscode`, `AI.py`, `ass-4.1`, `t.py`, and `task 3.py`.
- Code Editor:** The active file is `AI.py`, containing Python code for a library management system. It includes imports for `os` and `sqlite3`, and defines a `Book` class with methods for adding, displaying, checking out, returning, and deleting books. The `main()` function handles user input for these operations.
- Terminal:** The terminal window shows the execution of the script and its output. It prompts for a book author, adds a book titled "Python Basics" by John, and displays the updated library list.
- AI Assistant:** A sidebar on the right is titled "Build with Agent". It includes a speech bubble icon, a message "AI responses may be inaccurate.", a "Generate Agent" button, and instructions "Instructions to onboard AI onto your codebase."
- Taskbar:** The taskbar at the bottom shows the current interpreter path as "defaultInterpreterPath: 3.13.9.final.0", and icons for "Go Live" and "Go Live" (with a different icon).

Task Description #4 (AI-Assisted Code Completion for Class-Defining Code)

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.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files in the AI folder, including `.vscode`, `AI.py`, `ass-4.1`, `tpy`, and `task 3.py`.
- Code Editor:** Displays the `AI.py` file content. The code defines two classes: `Student` and `AttendanceSystem`. The `Student` class has methods for marking present or absent and displaying attendance. The `AttendanceSystem` class adds students and displays all attendance.
- Right Panel:** Features an "AI" sidebar with a speech bubble icon and the text "Build with Agent". It includes instructions: "AI responses may be inaccurate.", "Generate Agent", and "Instructions to onboard AI onto your codebase". A text input field says "Describe what to build next".
- Bottom Status Bar:** Shows the line number (Ln 179, Col 1), spaces (Spaces: 4), encoding (UTF-8), and interpreter path (defaultInterpretorPath: 3.13.9.final.0). It also has "Go Live" and "ENG IN" buttons.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files in the AI folder, including `.vscode`, `AI.py`, `ass-4.1`, `tpy`, and `task 3.py`.
- Code Editor:** Displays the `AI.py` file content. The code defines a `main` function that creates an `AttendanceSystem` object, takes input for student names, adds them to the system, and then loops through days to mark attendance. It uses `input` for status and prints messages for invalid input.
- Right Panel:** Features an "AI" sidebar with a speech bubble icon and the text "Build with Agent". It includes instructions: "AI responses may be inaccurate.", "Generate Agent", and "Instructions to onboard AI onto your codebase". A text input field says "Describe what to build next".
- Bottom Status Bar:** Shows the line number (Ln 179, Col 1), spaces (Spaces: 4), encoding (UTF-8), and interpreter path (defaultInterpretorPath: 3.13.9.final.0). It also has "Go Live" and "ENG IN" buttons.

The screenshot shows the Microsoft Visual Studio Code interface. The left sidebar displays the 'EXPLORER' view with files: 'AI', '.vscode', 'ass-4.1', 't.py', and 'task 3.py'. The main editor window shows a Python script named 'Al.py' with the following code:

```
145     def main():
146         ...
147         break
148     elif status == 'n':
149         student.mark_absent()
150         break
151     else:
152         print("Invalid input! Please enter 'y' or 'n'.")
153
154     print("\nFinal Attendance Records")
155     print("-" * 30)
156     attendance_system.display_all_attendance()
157
158 if __name__ == "__main__":
159     main()
```

The status bar at the bottom indicates: Ln 179, Col 1 | Spaces: 4 | UTF-8 | CRLF | Python | defaultInterpreterPath: 3.13.9.final.0 | Go Live | Go Live | ENG IN | 11:51 | 23-01-2026.

A floating panel on the right titled 'Build with Agent' contains the following text:
AI responses may be inaccurate.
Generate Agent
Instructions to onboard AI onto your codebase.

Output:

```
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
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
```

Build with Agent

AI responses may be inaccurate.

Generate Agent

Instructions to onboard AI onto your codebase.

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files like `AI`, `.vscode`, `Al.py`, `ass-4.1`, `t.py`, and `task 3.py`.
- Code Editor:** The `Al.py` file contains Python code for generating attendance reports. It includes sections for "Attendance for Akshitha" and "Attendance for varshitha", each listing "Day 1: Present", "Day 2: Present", "Day 3: Present", and "Day 4: Present".
- Terminal:** The terminal window shows the command `PS C:\Users\reddy\OneDrive\Desktop\AI> []`.
- Right Panel:** A sidebar titled "Build with Agent" provides instructions for generating an AI agent. It includes a note about AI responses being inaccurate, a "Generate Agent" button, and a "Instructions to onboard AI onto your codebase" section.
- Bottom Bar:** Includes standard system icons for search, file operations, and network status, along with the date and time (23-01-2026) and a language setting (ENG IN).

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.

The screenshot shows two instances of the Visual Studio Code (VS Code) interface, demonstrating the integration of AI features like "Build with Agent".

Top Window (VS Code):

- Explorer:** Shows files in the AI folder, including `Al.py`.
- Code Editor:** Displays the `Al.py` script for an ATM application. The code includes a menu loop, deposit, withdraw, and exit options.
- Right Panel:** Features a "Build with Agent" sidebar with a speech bubble icon, AI status message, and "Generate Agent" button.
- Bottom Status Bar:** Shows file path, line/col, spaces, encoding, and date (23-01-2026).

Bottom Window (VS Code):

- Explorer:** Shows files in the AI folder, including `Al.py`.
- Code Editor:** Displays the `Al.py` script. A cursor is at line 220, which contains an if statement for the main module.
- Terminal:** Shows the output of the application execution.
- Output:** Shows the terminal output, including the ATM menu and a successful deposit message.
- Right Panel:** Features a "Build with Agent" sidebar with a speech bubble icon, AI status message, and "Generate Agent" button.
- Bottom Status Bar:** Shows file path, line/col, spaces, encoding, and date (23-01-2026).

The screenshots illustrate how AI can be used to build and run Python code directly from the IDE, providing real-time feedback and assistance.

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with an AI integration. The main area displays Python code for an ATM menu system. A sidebar on the right provides AI-generated code completion and suggestions. The bottom status bar shows system information like battery level, network, and date.

Code Snippet:

```
187     def atm_menu():
188         balance += amount
189         print(f"${amount:.2f} deposited successfully.")
190     else:
191         print("Invalid amount. Please enter a positive value.")
192     elif choice == '3':
193         amount = float(input("Enter amount to withdraw: $"))
194         if 0 < amount <= balance:
195             balance -= amount
196             print(f"${amount:.2f} withdrawn successfully.")
197         else:
198             print("Invalid amount. Please enter a positive value within your balance.")
199     elif choice == '4':
200         print("Thank you for using the ATM. Goodbye!")
201         break
202     else:
203         print("Invalid choice. Please select a valid option.")
204 if __name__ == "__main__":
205     atm_menu()
```