

Vyshnavi Parisha

2403a51l34

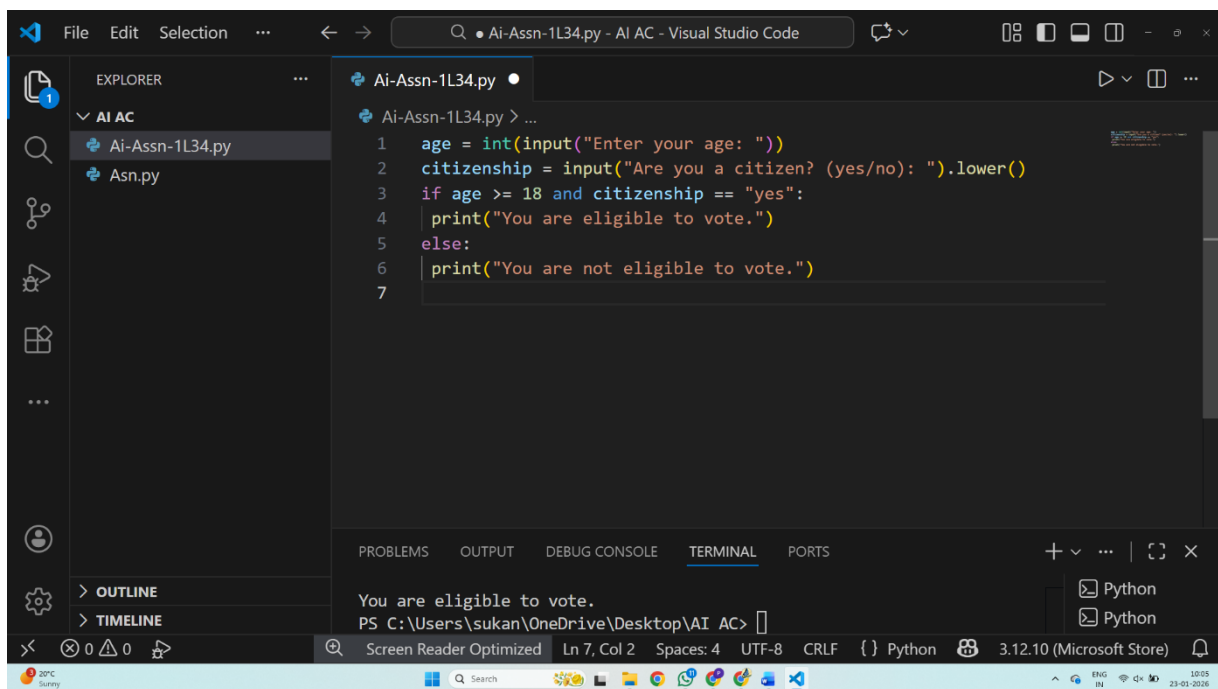
B-52

Lab 5: AI-Based Code Completion: Working with suggestions for classes, loops, conditionals

Task Description – 1: AI-Based Code Completion for Conditional Eligibility Check

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

Code

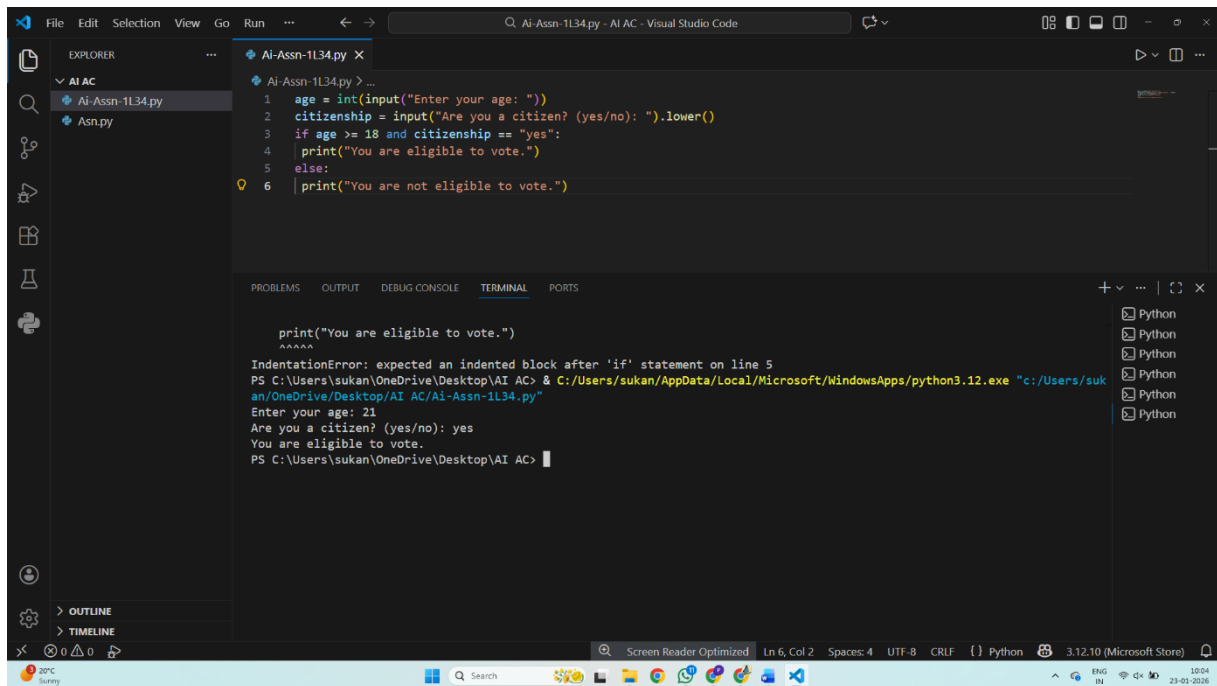


The screenshot shows the Visual Studio Code interface with a Python file named 'Ai-Assn-1L34.py'. The code is as follows:

```
1 age = int(input("Enter your age: "))
2 citizenship = input("Are you a citizen? (yes/no): ").lower()
3 if age >= 18 and citizenship == "yes":
4     print("You are eligible to vote.")
5 else:
6     print("You are not eligible to vote.")
7
```

The bottom panel shows the terminal output: "You are eligible to vote." and the command prompt "PS C:\Users\sukan\OneDrive\Desktop\AI AC>". The status bar at the bottom indicates the file is encoded in UTF-8 with CRLF line endings, using the Python 3.12.10 interpreter from the Microsoft Store.

OUTPUT:



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a file named 'AI-Assn-1L34.py'. The main editor displays the following Python code:

```
1 age = int(input("Enter your age: "))
2 citizenship = input("Are you a citizen? (yes/no): ").lower()
3 if age >= 18 and citizenship == "yes":
4     print("You are eligible to vote.")
5 else:
6     print("You are not eligible to vote.")
```

The TERMINAL pane at the bottom shows the execution output:

```
print("You are eligible to vote.")
^^^^^
IndentationError: expected an indented block after 'if' statement on line 5
PS C:\Users\sukan\OneDrive\Desktop\AI AC> & C:/Users/sukan/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/sukan/OneDrive/Desktop/AI AC/AI-Assn-1L34.py"
Enter your age: 21
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS C:\Users\sukan\OneDrive\Desktop\AI AC>
```

Explanation :

- The program accepts the user's age and citizenship status.
- The conditional statement checks if age is 18 or above and citizenship is "yes".
- If both conditions are satisfied, eligibility is confirmed; otherwise, the user is not eligible.

Verification:

Age = 20, Citizenship = yes → Eligible

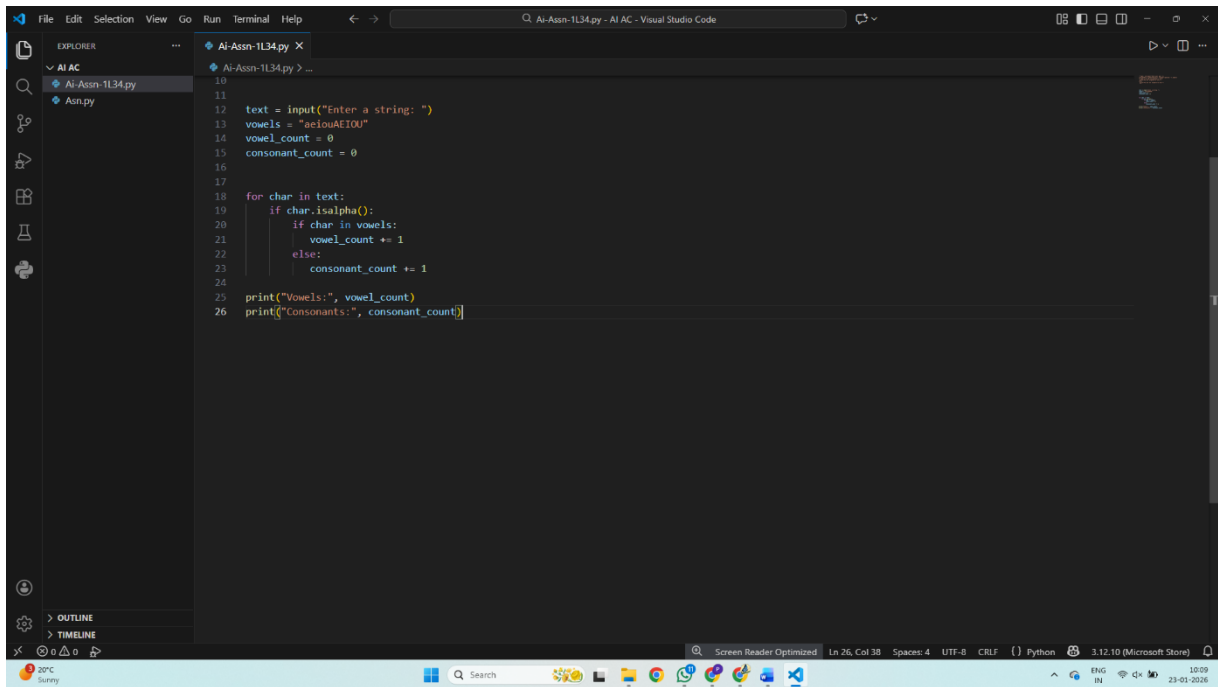
Age = 16, Citizenship = yes → Not eligible

Observations: The initial API code is insecure because it uses a hardcoded API key and does not protect user data. The corrected version improves security by validating inputs, hashing passwords, and using token-based authentication for safer access control.

Task Description – 2: AI-Based Code Completion for Loop-Based String Processing)

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

AI-Generated Code:

A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a file named 'Ai-Assn-1L34.py'. The main editor window displays the following Python code:

```
10
11
12 text = input("Enter a string: ")
13 vowels = "aeiouAEIOU"
14 vowel_count = 0
15 consonant_count = 0
16
17
18 for char in text:
19     if char.isalpha():
20         if char in vowels:
21             vowel_count += 1
22         else:
23             consonant_count += 1
24
25 print("Vowels:", vowel_count)
26 print("Consonants:", consonant_count)
```

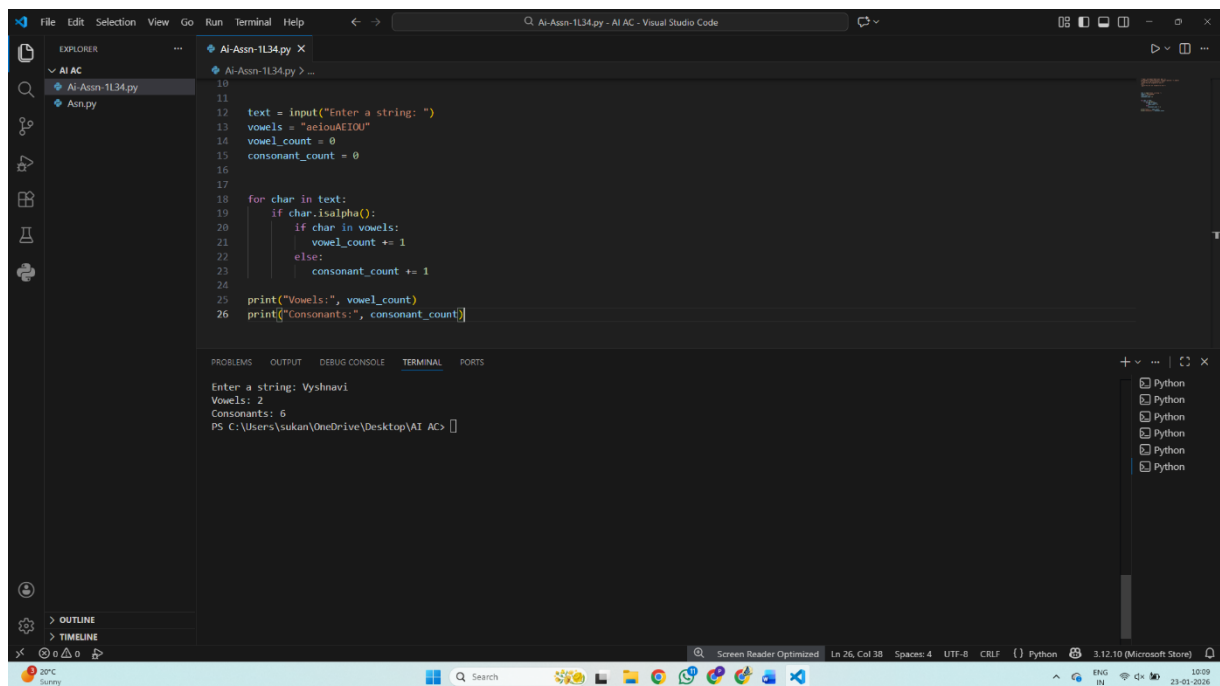
The status bar at the bottom indicates 'Ln 26, Col 38', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python', and '3.12.10 (Microsoft Store)'. The Windows taskbar is visible at the very bottom.

Observations:

1. The logic unfairly favors urban students
2. Rural or semi-urban students are excluded
3. No flexibility or weighted scoring approach

Improved Version:

OUTPUT:



The screenshot displays the Visual Studio Code interface. The Explorer panel on the left shows a project named 'AI AC' containing a file 'AI-Assn-1L34.py'. The main editor window shows the code for this file. The code is a Python script that takes a string input and counts the number of vowels and consonants. The script uses a for loop to iterate through each character, checks if it is an alphabetic character using `char.isalpha()`, and then checks if it is a vowel. Vowels are counted in the `vowel_count` variable, and consonants are counted in the `consonant_count` variable. The script prints the results at the end. The Terminal panel at the bottom shows the execution output: 'Enter a string: Vyshnavi', 'Vowels: 2', and 'Consonants: 6'. The status bar at the bottom indicates the file is 'Ln 26, Col 38' and the Python version is '3.12.10 (Microsoft Store)'.

```
10
11
12 text = input("Enter a string: ")
13 vowels = "aeiouAEIOU"
14 vowel_count = 0
15 consonant_count = 0
16
17
18 for char in text:
19     if char.isalpha():
20         if char in vowels:
21             vowel_count += 1
22         else:
23             consonant_count += 1
24
25 print("Vowels:", vowel_count)
26 print("Consonants:", consonant_count)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Enter a string: Vyshnavi
Vowels: 2
Consonants: 6
PS C:\Users\sukan\OneDrive\Desktop\AI AC>

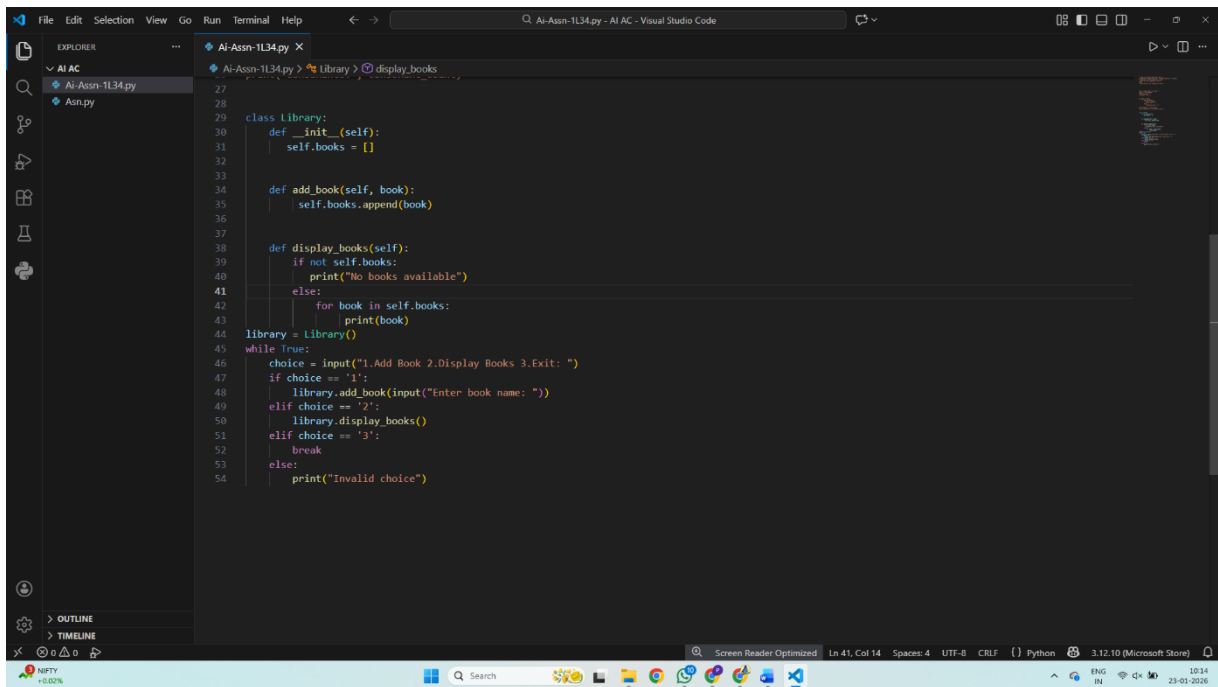
Explanation:

- A loop iterates through each character in the string.
- Alphabetic characters are checked.
- Vowels and consonants are counted separately

Task Description – 3: AI-Assisted Code Completion Reflection Task

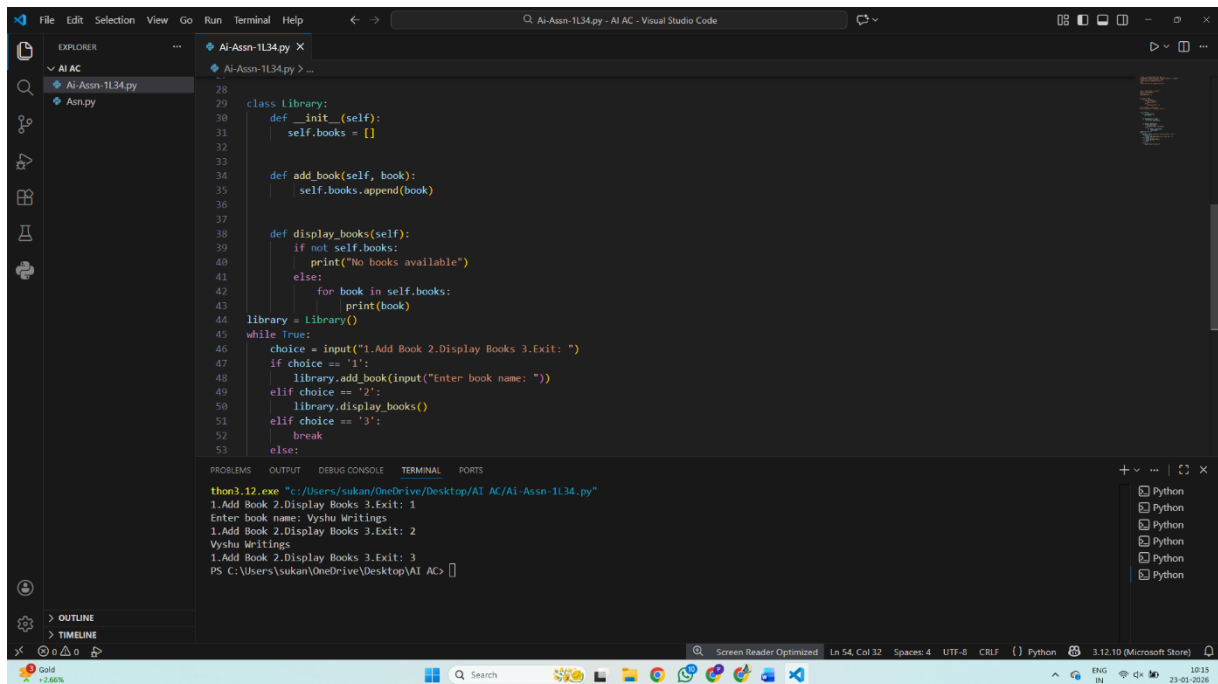
Prompt: “Generate a Python program for a library management system using classes, loops, and conditional statements.”

Code:



```
27
28
29 class Library:
30     def __init__(self):
31         self.books = []
32
33
34     def add_book(self, book):
35         self.books.append(book)
36
37
38     def display_books(self):
39         if not self.books:
40             print("No books available")
41         else:
42             for book in self.books:
43                 print(book)
44
45 library = Library()
46 while True:
47     choice = input("1.Add Book 2.Display Books 3.Exit: ")
48     if choice == '1':
49         library.add_book(input("Enter book name: "))
50     elif choice == '2':
51         library.display_books()
52     elif choice == '3':
53         break
54     else:
55         print("Invalid choice")
```

OUTPUT:



```
thon3.12.exe "c:/Users/sukan/OneDrive/Desktop/AI AC/AI-Assn-1L34.py"
1.Add Book 2.Display Books 3.Exit: 1
Enter book name: Vyshu Writings
1.Add Book 2.Display Books 3.Exit: 2
Vyshu Writings
1.Add Book 2.Display Books 3.Exit: 3
PS C:\Users\sukan\OneDrive\Desktop\AI AC>
```

Explanation:

- The AI generated a logically correct and readable program.
- It effectively used classes, loops, and conditionals.

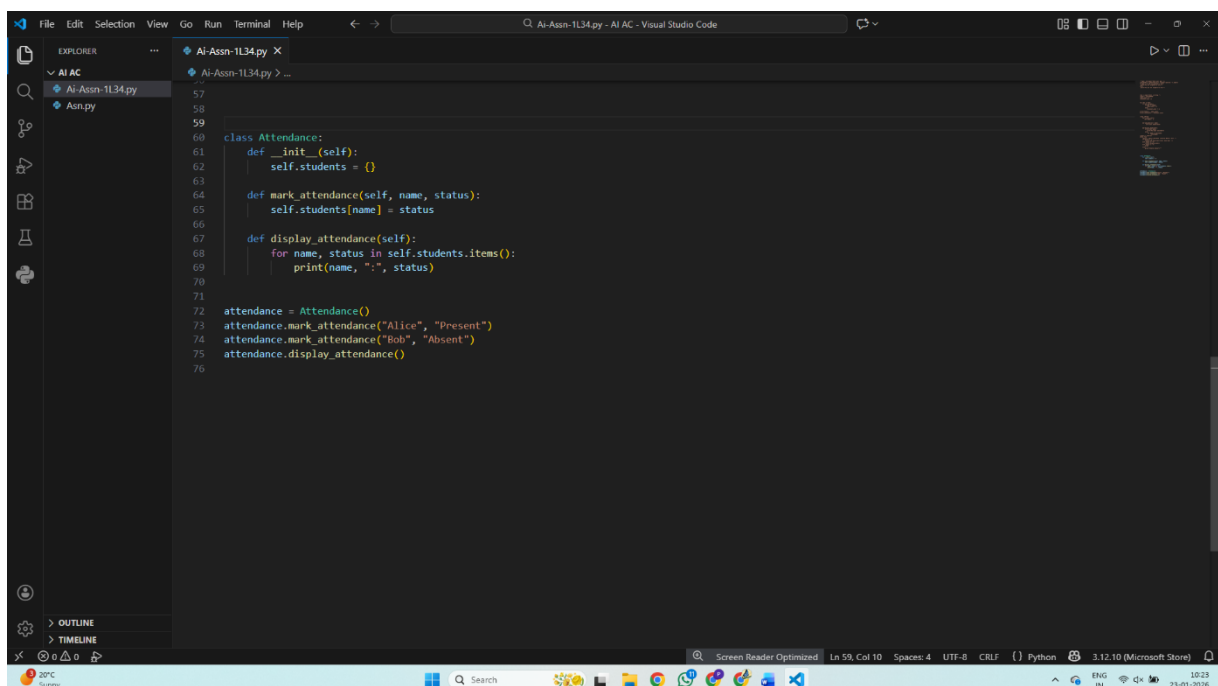
Observations:

AI-assisted coding helps speed up development and provides structured solutions. However, human review is essential to ensure efficiency, security, and correctness.

Task Description – 4: AI-Assisted Code Completion for Class-Based Attendance System

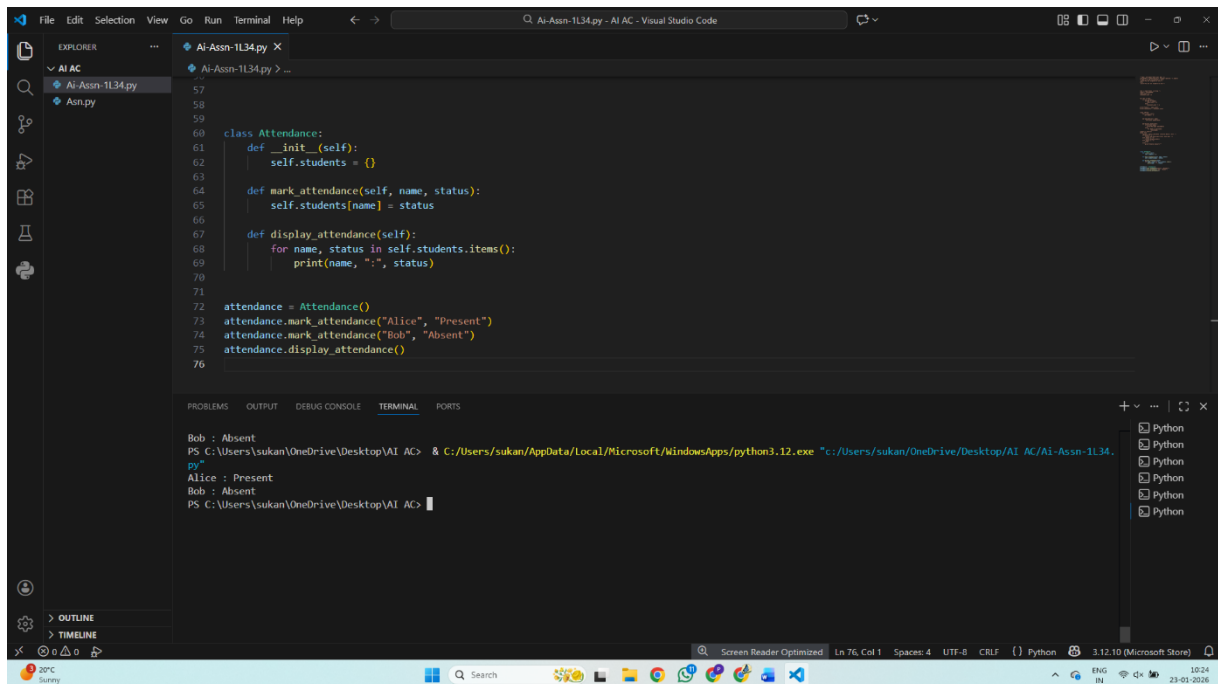
Prompt: “Generate a Python class to mark and display student attendance using loops.”

Code:



```
57
58
59
60 class Attendance:
61     def __init__(self):
62         self.students = {}
63
64     def mark_attendance(self, name, status):
65         self.students[name] = status
66
67     def display_attendance(self):
68         for name, status in self.students.items():
69             print(name, ":", status)
70
71
72 attendance = Attendance()
73 attendance.mark_attendance("Alice", "Present")
74 attendance.mark_attendance("Bob", "Absent")
75 attendance.display_attendance()
76
```

OUTPUT:



```
File Edit Selection View Go Run Terminal Help
AI-Assn-1L34.py X
AI-Assn-1L34.py > ...
57
58
59
60 class Attendance:
61     def __init__(self):
62         self.students = {}
63
64     def mark_attendance(self, name, status):
65         self.students[name] = status
66
67     def display_attendance(self):
68         for name, status in self.students.items():
69             print(name, ":", status)
70
71
72 attendance = Attendance()
73 attendance.mark_attendance("Alice", "Present")
74 attendance.mark_attendance("Bob", "Absent")
75 attendance.display_attendance()
76

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Bob : Absent
PS C:\Users\sukan\OneDrive\Desktop\AI AC> & C:/Users/sukan/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/sukan/OneDrive/Desktop/AI AC/AI-Assn-1L34.py"
Alice : Present
Bob : Absent
PS C:\Users\sukan\OneDrive\Desktop\AI AC>
```

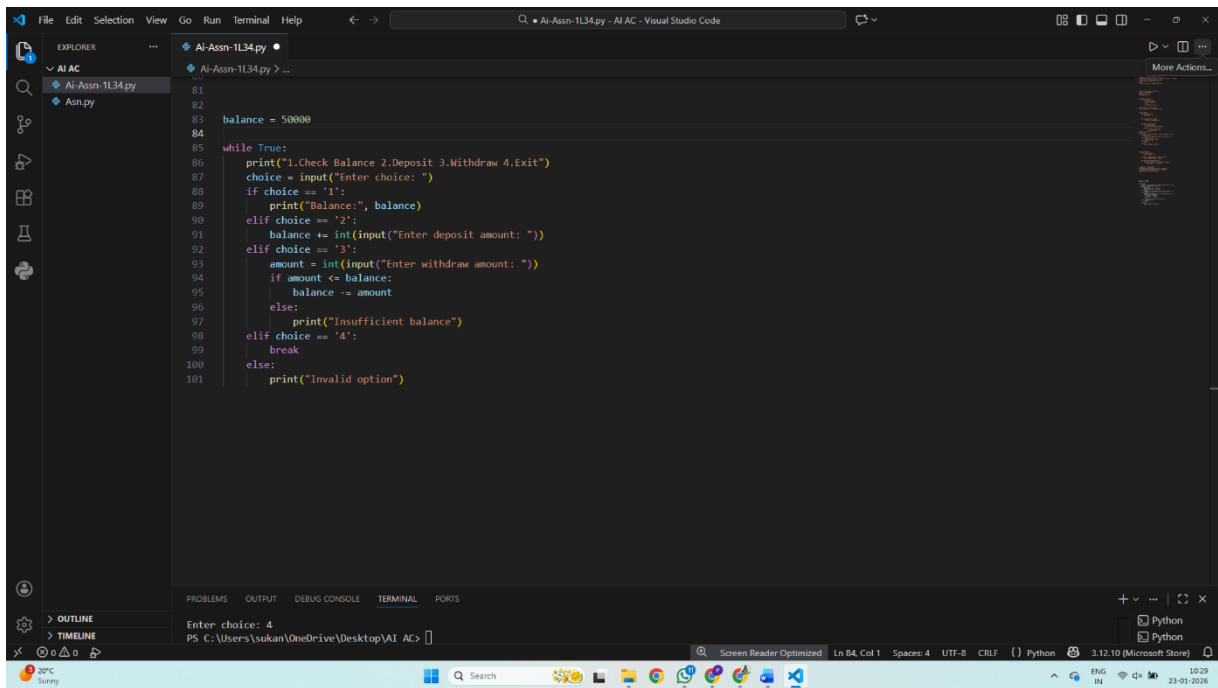
Observations:

- The class-based design improves code organization and readability.
- Dictionary usage ensures efficient storage and retrieval of attendance data.
- The program produces accurate output with minimal code.

Task Description – 5: AI-Based Code Completion for Conditional Menu Navigation

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

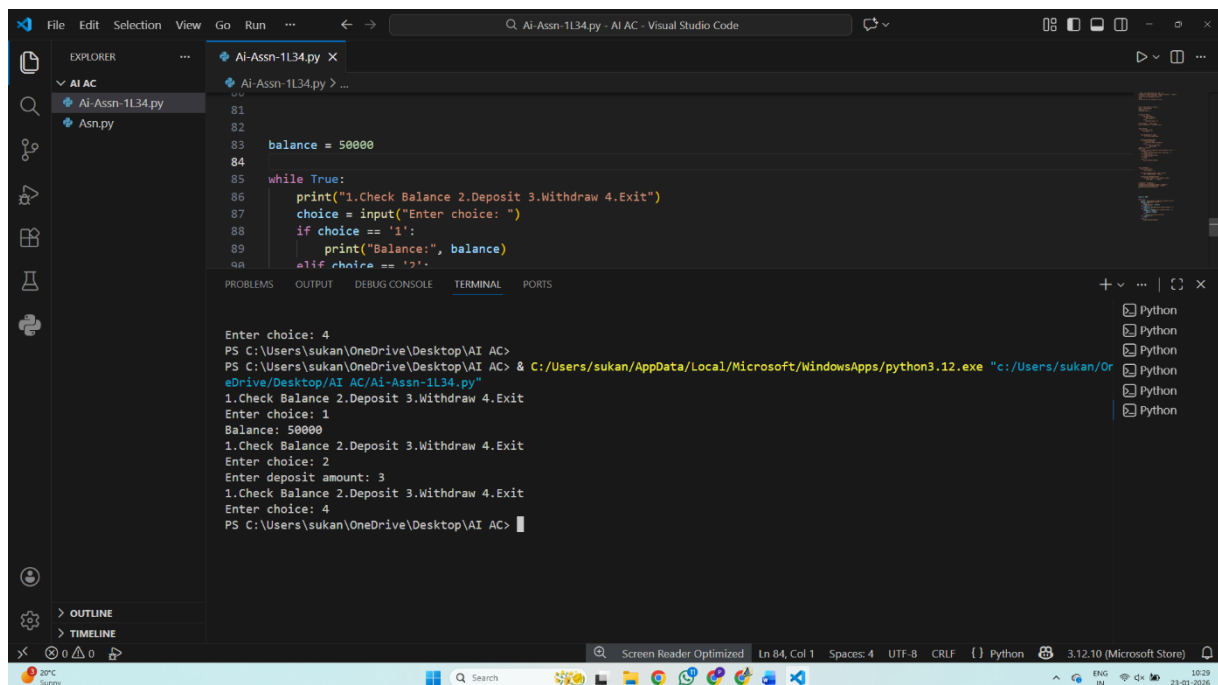
Code:



```
81
82
83 balance = 50000
84
85 while True:
86     print("1.Check Balance 2.Deposit 3.Withdraw 4.Exit")
87     choice = input("Enter choice: ")
88     if choice == '1':
89         print("Balance:", balance)
90     elif choice == '2':
91         balance += int(input("Enter deposit amount: "))
92     elif choice == '3':
93         amount = int(input("Enter withdraw amount: "))
94         if amount <= balance:
95             balance -= amount
96         else:
97             print("Insufficient balance")
98     elif choice == '4':
99         break
100    else:
101        print("Invalid option")
```

Enter choices: 4
PS C:\Users\sukan\OneDrive\Desktop\AI AC>

OUTPUT:



```
Enter choice: 4
PS C:\Users\sukan\OneDrive\Desktop\AI AC>
PS C:\Users\sukan\OneDrive\Desktop\AI AC> & C:/Users/sukan/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/sukan/OneDrive/Desktop/AI AC/AI-Assn-1L34.py"
1.Check Balance 2.Deposit 3.Withdraw 4.Exit
Enter choice: 1
Balance: 50000
1.Check Balance 2.Deposit 3.Withdraw 4.Exit
Enter choice: 2
Enter deposit amount: 3
1.Check Balance 2.Deposit 3.Withdraw 4.Exit
Enter choice: 4
PS C:\Users\sukan\OneDrive\Desktop\AI AC>
```


Observations:

- The program effectively simulates real-world ATM operations.
- Conditional statements and loops are used correctly.
- The system handles invalid inputs and balance conditions properly.