School of Computer Science and Artificial Intelligence

Lab Assignment -6.5

Program	**:B. Tech (CSE)**
Specialization	**:AIML**
Course Title	**: AI Assisted Coding**
Course Code	**: 23CS002PC304**
Semester	**: VI**
Academic Session	**: 2025-2026**
Name of Student	**: P.Karthisha**
Enrollment No.	**: 2303A52099**
Batch No.	**: 33**

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.”

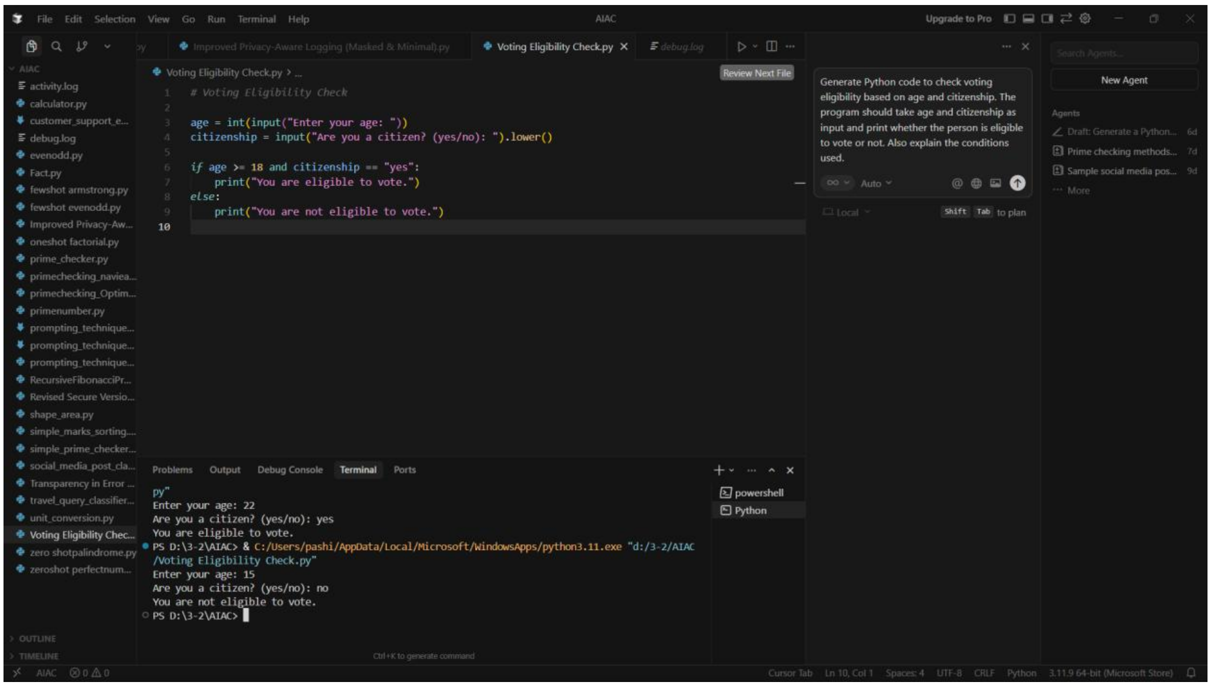
Expected Output:

• AI-generated conditional logic.

give me code in python

• Correct eligibility decisions.

• Explanation of conditions.


Generate Python code to check voting eligibility based on age and citizenship. The program should take age and citizenship as input and print whether the person is eligible to vote or not. Also explain the conditions used.

```python
# Voting Eligibility Check

age = int(input("Enter your age: "))
citizenship = input("Are you a citizen? (yes/no): ").lower()

if age >= 18 and citizenship == "yes":
    print("You are eligible to vote.")
else:
    print("You are not eligible to vote.")
```

```
py"
Enter your age: 22
Are you a citizen? (yes/no): yes
You are eligible to vote.
PS D:\3-2\AIAC> & C:/Users/pashi/AppData/Local/Microsoft/WindowsApps/python3.11.exe "d:/3-2/AIAC
/Voting Eligibility Check.py"
Enter your age: 15
Are you a citizen? (yes/no): no
You are not eligible to vote.
PS D:\3-2\AIAC>
```

Step-by-Step Explanation of Conditions

Step 1: Take input

- age → user's age (integer)
- citizenship → whether the user is a citizen (yes or no)

Step 2: Apply conditions

The condition used is:

age >= 18 and citizenship == "yes"

- age >= 18 → Voting is allowed only if the person is 18 years or older
- citizenship == "yes" → Person must be a citizen of the country

Step 3: Decision

- If both conditions are true, the person can vote
- If any condition is false, the person cannot vote

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

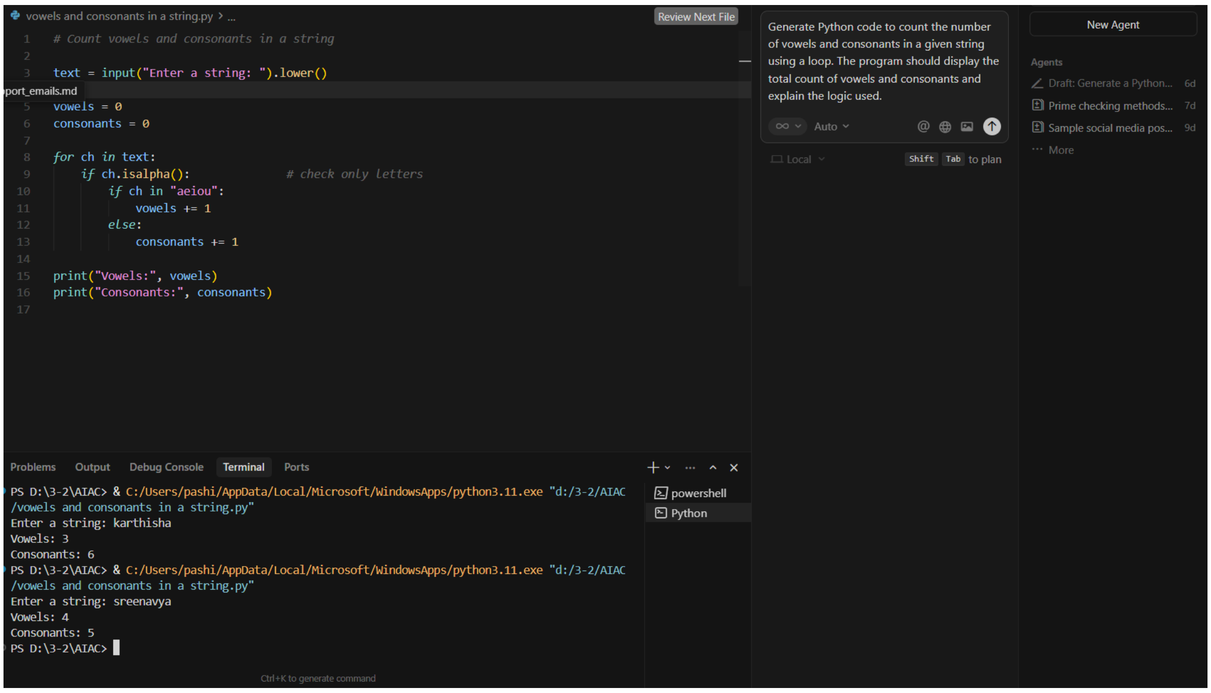
Task: Use an AI tool to process strings using loops.

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
# Count vowels and consonants in a string

text = input("Enter a string: ").lower()

vowels = 0
consonants = 0

for ch in text:
    if ch.isalpha():            # check only letters
        if ch in "aeiou":
            vowels += 1
        else:
            consonants += 1

print("Vowels:", vowels)
print("Consonants:", consonants)
```

```
PS D:\3-2\AIAC> & C:/Users/pashi/AppData/Local/Microsoft/WindowsApps/python3.11.exe "d:/3-2/AIAC
/vowels and consonants in a string.py"
Enter a string: karthisha
Vowels: 3
Consonants: 6
PS D:\3-2\AIAC> & C:/Users/pashi/AppData/Local/Microsoft/WindowsApps/python3.11.exe "d:/3-2/AIAC
/vowels and consonants in a string.py"
Enter a string: sreenavya
Vowels: 4
Consonants: 5
PS D:\3-2\AIAC>
```

Generate Python code to count the number of vowels and consonants in a given string using a loop. The program should display the total count of vowels and consonants and explain the logic used.

vowels and consonants in a string.py > ...
Review Next File
port_emails.md
Problems
Output
Debug Console
Terminal
Ports
powershell
Python
Ctrl+K to generate command
Auto
Local
Shift Tab to plan
New Agent
Agents
Draft: Generate a Python... 6d
Prime checking methods... 7d
Sample social media pos... 9d
More

Step-by-Step Explanation (String Processing Logic)

Step 1: Take input

- Read a string from the user
- Convert it to lowercase to avoid case issues (A and a)

Step 2: Initialize counters

- vowels = 0
- consonants = 0

Step 3: Loop through the string

for ch in text:

- Each character is checked one by one

Step 4: Apply conditions

- ch.isalpha() → ensures only letters are counted
- If character is in "aeiou" → vowel
- Otherwise → consonant

Step 5: Print results

- Display total number of vowels
- Display total number of consonants

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

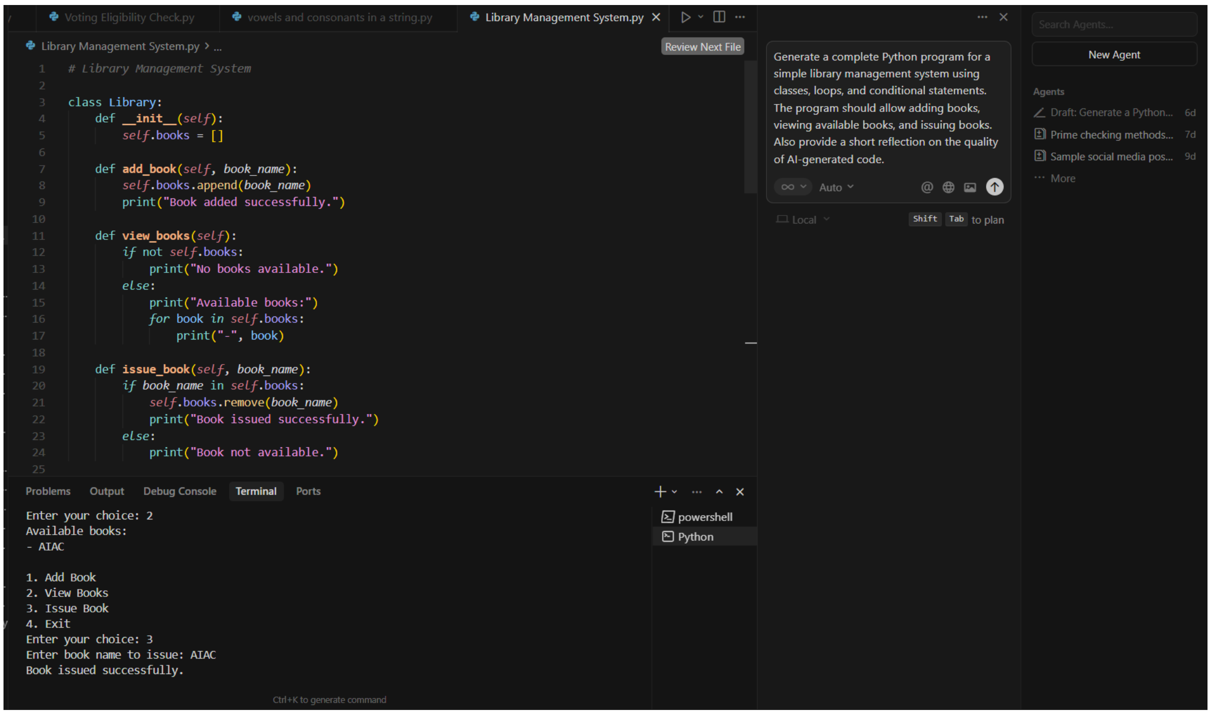
using classes, loops, and conditional statements.”

Expected Output:

• Complete AI-generated program.

• Review of AI suggestions quality.

• Short reflection on AI-assisted coding experience.


Voting Eligibility Check.py
vowels and consonants in a string.py
Library Management System.py
Library Management System.py > ...
Review Next File
Problems
Output
Debug Console
Terminal
Ports
powershell
Python
Ctrl+K to generate command
Auto
Local
Shift
Tab
to plan
Search Agents...
New Agent
Agents
Draft: Generate a Python... 6d
Prime checking methods... 7d
Sample social media pos... 9d
More

```python
# Library Management System

class Library:
    def __init__(self):
        self.books = []

    def add_book(self, book_name):
        self.books.append(book_name)
        print("Book added successfully.")

    def view_books(self):
        if not self.books:
            print("No books available.")
        else:
            print("Available books:")
            for book in self.books:
                print("-", book)

    def issue_book(self, book_name):
        if book_name in self.books:
            self.books.remove(book_name)
            print("Book issued successfully.")
        else:
            print("Book not available.")

```

```
Enter your choice: 2
Available books:
- AIAC

1. Add Book
2. View Books
3. Issue Book
4. Exit
Enter your choice: 3
Enter book name to issue: AIAC
Book issued successfully.
```

Generate a complete Python program for a simple library management system using classes, loops, and conditional statements. The program should allow adding books, viewing available books, and issuing books. Also provide a short reflection on the quality of AI-generated code.

Explanation of Program Components

Classes

- Library class stores and manages book data
- Uses a list books to store book names

Loops

- while True loop keeps the menu running until the user exits
- for loop displays available books

Conditionals

- if–elif–else used to:
    - Handle user menu choices
    - Check book availability before issuing

Review of AI Suggestions Quality

- The AI-generated code is structured and readable
- Proper use of object-oriented programming
- Logical flow using loops and conditionals
- Easy to understand and modify

- Suitable for beginner-level implementation

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

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

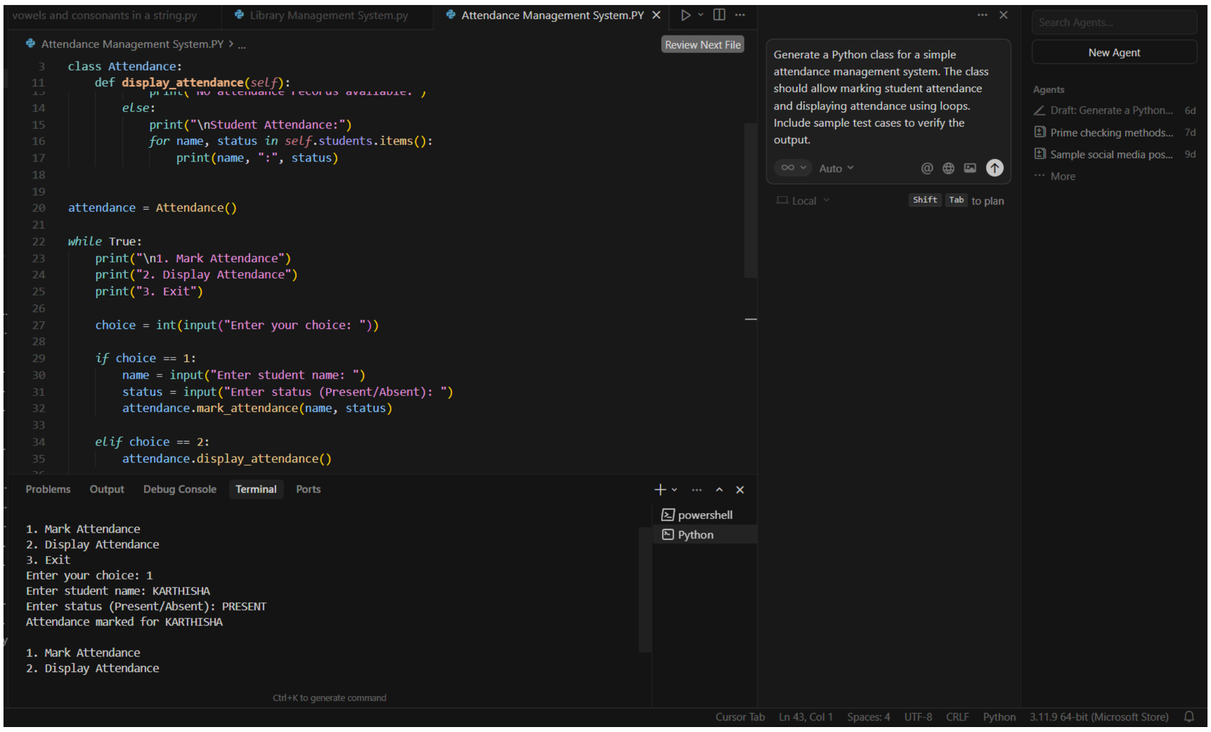
Prompt: “Generate a Python class to mark and display student attendance using loops.”

Expected Output:

• AI-generated attendance logic.

• Correct display of attendance.

• Test cases.


vowels and consonants in a string.py
Library Management System.py
Attendance Management System.PY
Attendance Management System.PY > ...
Review Next File
Problems
Output
Debug Console
Terminal
Ports
powershell
Python
Ctrl+K to generate command
Cursor Tab
Ln 43, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.11.9 64-bit (Microsoft Store)
Auto
Local
Shift Tab to plan
Search Agents...
New Agent
Agents
Draft: Generate a Python... 6d
Prime checking methods... 7d
Sample social media pos... 9d
More

```python
class Attendance:
    def display_attendance(self):
            print("No attendance records available.")
        else:
            print("\nStudent Attendance:")
            for name, status in self.students.items():
                print(name, ":", status)


attendance = Attendance()

while True:
    print("\n1. Mark Attendance")
    print("2. Display Attendance")
    print("3. Exit")

    choice = int(input("Enter your choice: "))

    if choice == 1:
        name = input("Enter student name: ")
        status = input("Enter status (Present/Absent): ")
        attendance.mark_attendance(name, status)

    elif choice == 2:
        attendance.display_attendance()
```

```
1. Mark Attendance
2. Display Attendance
3. Exit
Enter your choice: 1
Enter student name: KARTHISHA
Enter status (Present/Absent): PRESENT
Attendance marked for KARTHISHA

1. Mark Attendance
2. Display Attendance
```

Generate a Python class for a simple attendance management system. The class should allow marking student attendance and displaying attendance using loops. Include sample test cases to verify the output.

Explanation of Attendance Logic

Step 1: Class Creation

- Attendance class is created
- Uses a dictionary students
    - Key → student name

  - Value → attendance status (Present/Absent)

Step 2: Mark Attendance

- mark_attendance() method
- Stores or updates student attendance

Step 3: Display Attendance

- display_attendance() method
- Uses a for loop to print all records

Step 4: Loop & Conditionals

- while loop keeps menu running
- if–elif–else handles user choices

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

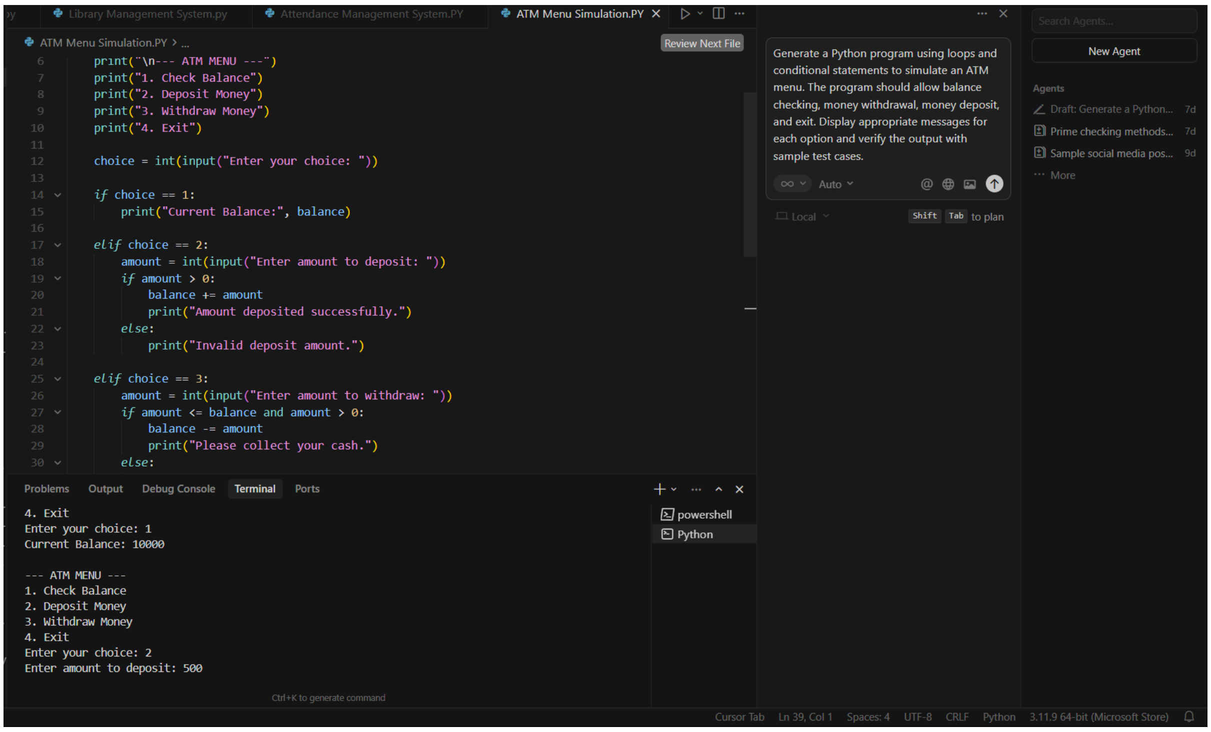
to simulate an ATM menu."

Expected Output:

• AI-generated menu logic.

• Correct option handling.

• Output verification.


Library Management System.py
Attendance Management System.PY
ATM Menu Simulation.PY
Review Next File
Problems
Output
Debug Console
Terminal
Ports
powershell
Python
Ctrl+K to generate command
New Agent
Agents
Draft: Generate a Python...
Prime checking methods...
Sample social media pos...
More
Cursor Tab
Ln 39, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.11.9 64-bit (Microsoft Store)

```python
    print("\n--- ATM MENU ---")
    print("1. Check Balance")
    print("2. Deposit Money")
    print("3. Withdraw Money")
    print("4. Exit")

    choice = int(input("Enter your choice: "))

    if choice == 1:
        print("Current Balance:", balance)

    elif choice == 2:
        amount = int(input("Enter amount to deposit: "))
        if amount > 0:
            balance += amount
            print("Amount deposited successfully.")
        else:
            print("Invalid deposit amount.")

    elif choice == 3:
        amount = int(input("Enter amount to withdraw: "))
        if amount <= balance and amount > 0:
            balance -= amount
            print("Please collect your cash.")
        else:
```

```
4. Exit
Enter your choice: 1
Current Balance: 10000

--- ATM MENU ---
1. Check Balance
2. Deposit Money
3. Withdraw Money
4. Exit
Enter your choice: 2
Enter amount to deposit: 500
```

Generate a Python program using loops and conditional statements to simulate an ATM menu. The program should allow balance checking, money withdrawal, money deposit, and exit. Display appropriate messages for each option and verify the output with sample test cases.

Explanation of Menu Logic

Step 1: Initialize balance

- balance = 10000 sets the starting amount

Step 2: Loop for menu

- while True keeps the ATM menu running
- Menu repeats until user chooses Exit

Step 3: Conditional handling

- if choice == 1 → Check balance
- elif choice == 2 → Deposit money
- elif choice == 3 → Withdraw money
- elif choice == 4 → Exit program
- else → Invalid option