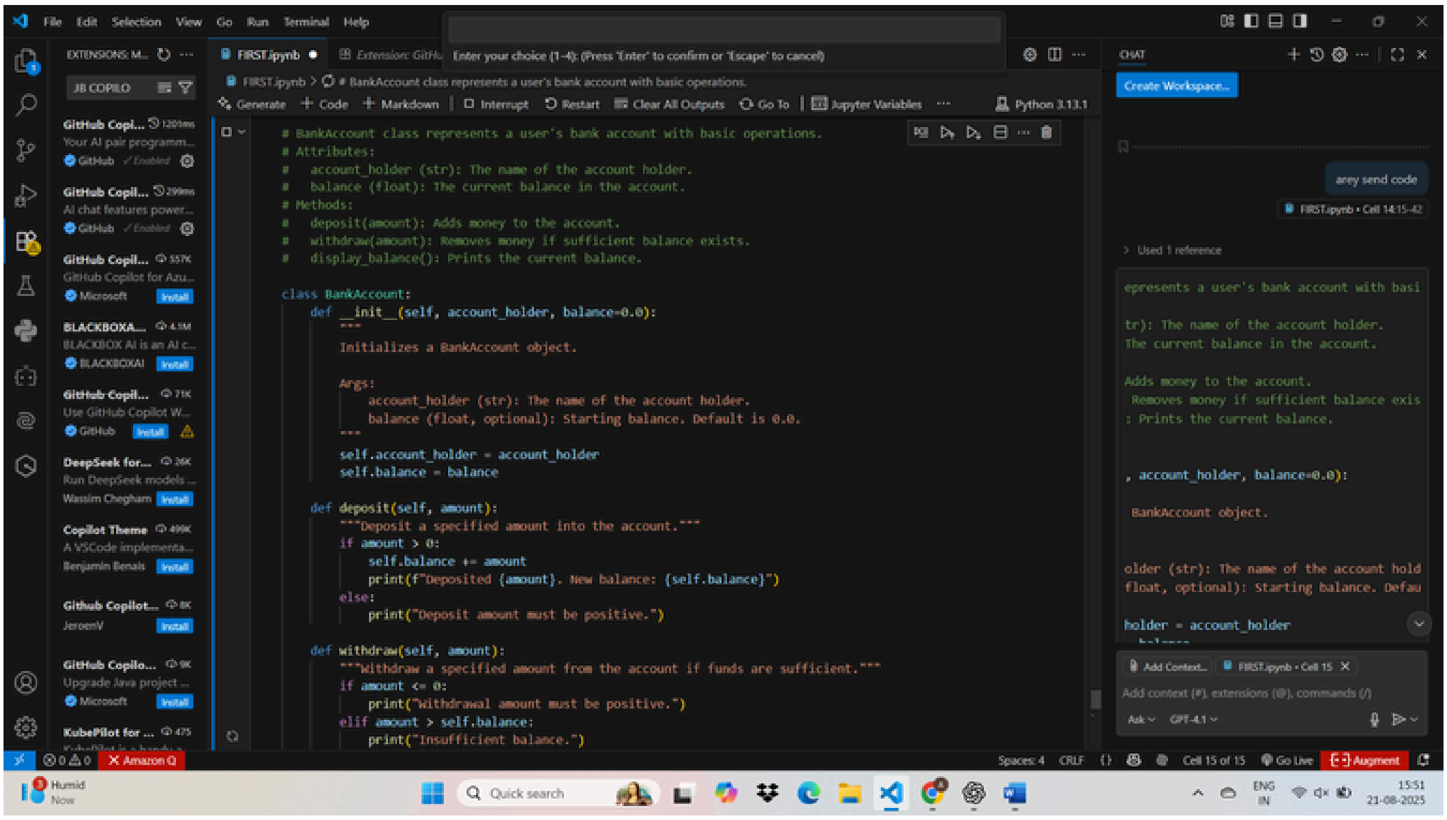
ASSIGNMENT-4.4

2403A51238

Chandana.T

TASK-1:

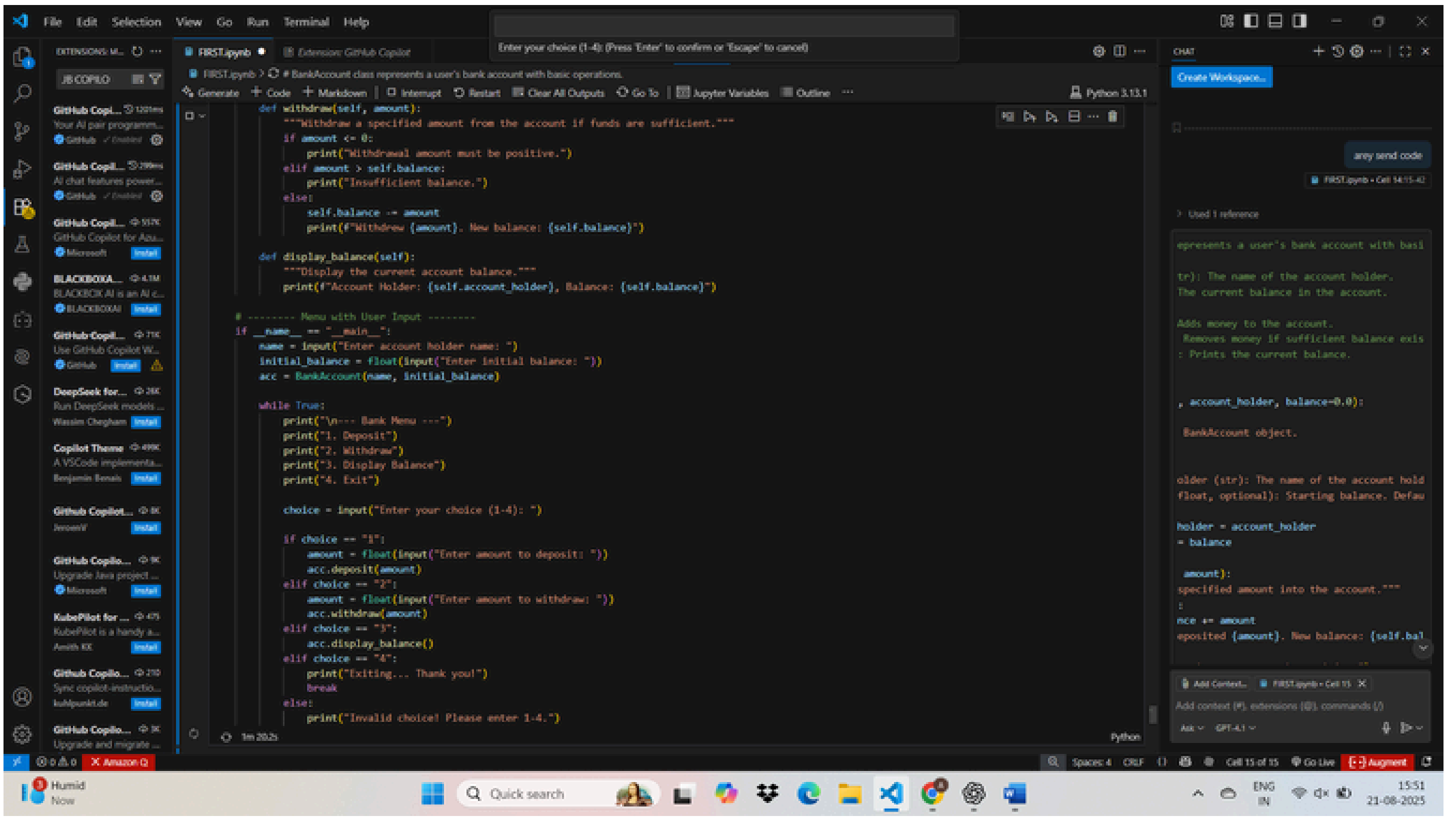
CODE:



Enter your choice (1-4): (Press 'Enter' to confirm or 'Escape' to cancel)

```python
# BankAccount class represents a user's bank account with basic operations.
# Attributes:
#   account_holder (str): The name of the account holder.
#   balance (float): The current balance in the account.
# Methods:
#   deposit(amount): Adds money to the account.
#   withdraw(amount): Removes money if sufficient balance exists.
#   display_balance(): Prints the current balance.

class BankAccount:
    def __init__(self, account_holder, balance=0.0):
        """
        Initializes a BankAccount object.

        Args:
            account_holder (str): The name of the account holder.
            balance (float, optional): Starting balance. Default is 0.0.
        """
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        """Deposit a specified amount into the account."""
        if amount > 0:
            self.balance += amount
            print(f"Deposited {amount}. New balance: {self.balance}")
        else:
            print("Deposit amount must be positive.")

    def withdraw(self, amount):
        """Withdraw a specified amount from the account if funds are sufficient."""
        if amount <= 0:
            print("Withdrawal amount must be positive.")
        elif amount > self.balance:
            print("Insufficient balance.")
```

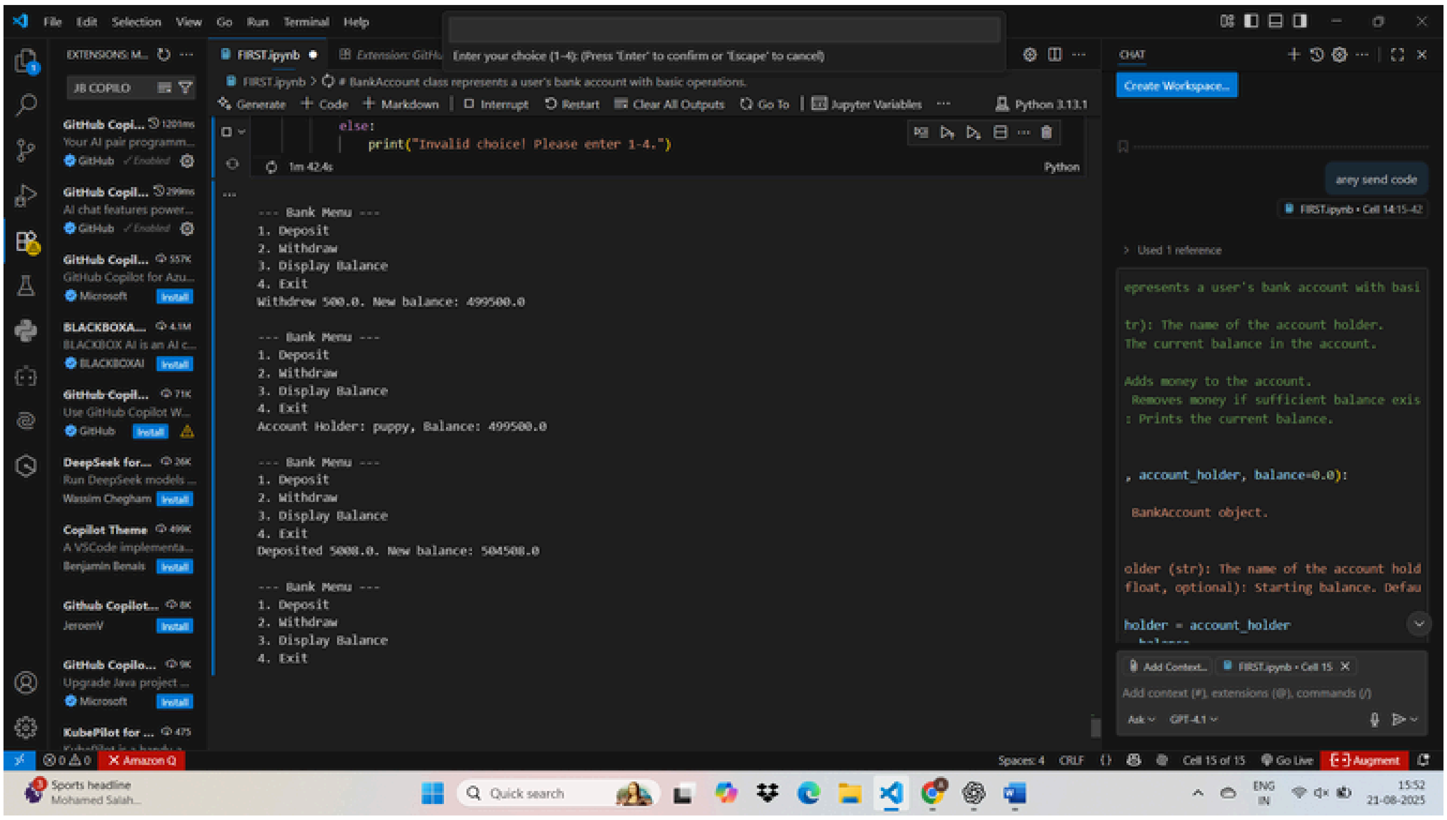
arey send code



```
Enter your choice (1-4): (Press 'Enter' to confirm or 'Escape' to cancel)
```

```python
    def withdraw(self, amount):
        """Withdraw a specified amount from the account if funds are sufficient."""
        if amount <= 0:
            print("Withdrawal amount must be positive.")
        elif amount > self.balance:
            print("Insufficient balance.")
        else:
            self.balance -= amount
            print(f"Withdrew {amount}. New balance: {self.balance}")

    def display_balance(self):
        """Display the current account balance."""
        print(f"Account Holder: {self.account_holder}, Balance: {self.balance}")

# -------- Menu with User Input --------
if __name__ == "__main__":
    name = input("Enter account holder name: ")
    initial_balance = float(input("Enter initial balance: "))
    acc = BankAccount(name, initial_balance)

    while True:
        print("\n--- Bank Menu ---")
        print("1. Deposit")
        print("2. Withdraw")
        print("3. Display Balance")
        print("4. Exit")

        choice = input("Enter your choice (1-4): ")

        if choice == "1":
            amount = float(input("Enter amount to deposit: "))
            acc.deposit(amount)
        elif choice == "2":
            amount = float(input("Enter amount to withdraw: "))
            acc.withdraw(amount)
        elif choice == "3":
            acc.display_balance()
        elif choice == "4":
            print("Exiting... Thank you!")
            break
        else:
            print("Invalid choice! Please enter 1-4.")
```

OUTPUT:

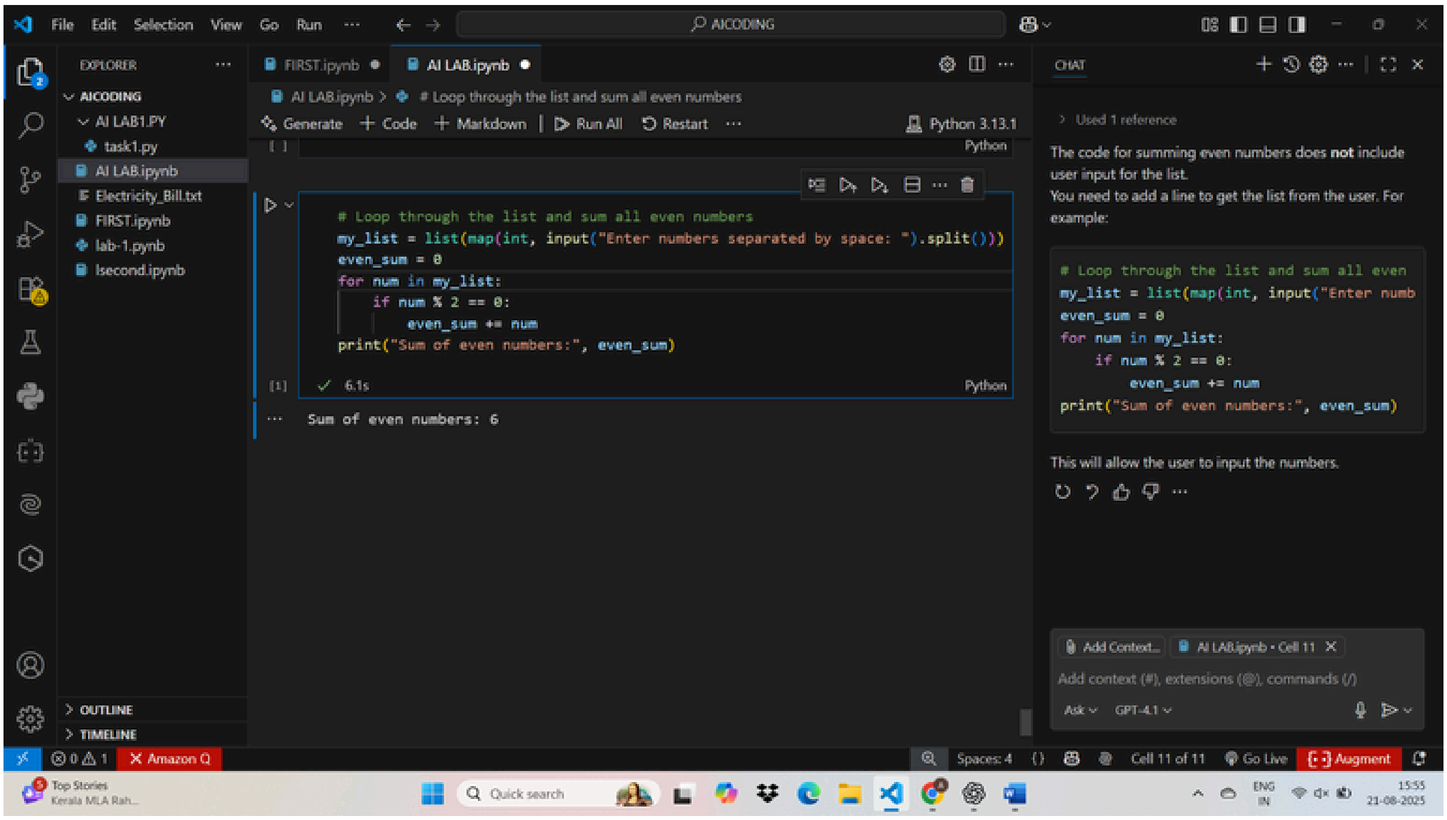


```
            else:
                print("Invalid choice! Please enter 1-4.")
```

```
--- Bank Menu ---
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Withdrew 500.0. New balance: 499500.0

--- Bank Menu ---
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Account Holder: puppy, Balance: 499500.0

--- Bank Menu ---
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Deposited 5000.0. New balance: 504500.0

--- Bank Menu ---
1. Deposit
2. Withdraw
3. Display Balance
4. Exit
```

TASK-2

CODE WITH OUTPUT:



```python
# Loop through the list and sum all even numbers
my_list = list(map(int, input("Enter numbers separated by space: ").split()))
even_sum = 0
for num in my_list:
    if num % 2 == 0:
        even_sum += num
print("Sum of even numbers:", even_sum)
```

```
Sum of even numbers: 6
```

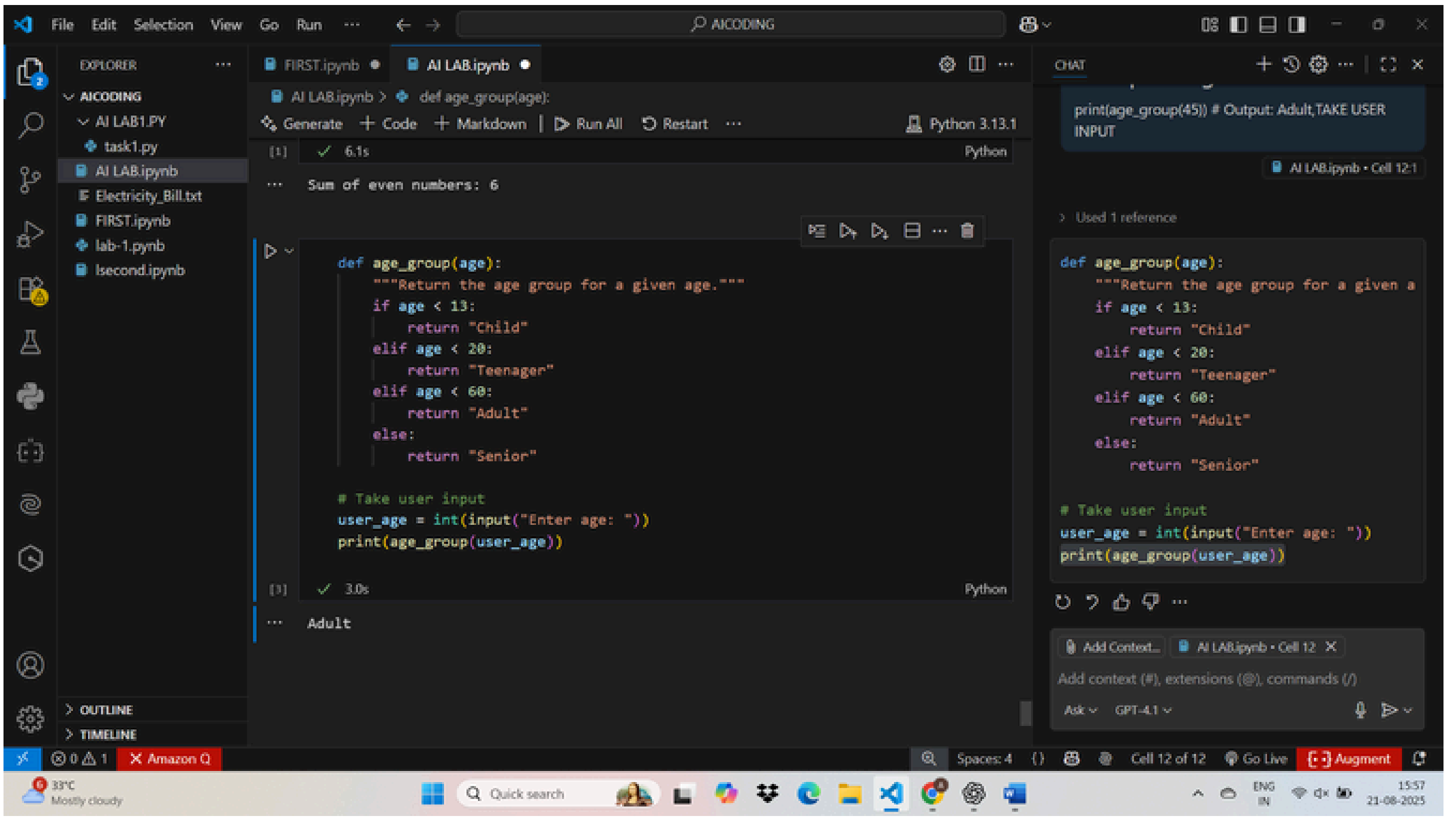
The code for summing even numbers does **not** include user input for the list.
You need to add a line to get the list from the user. For example:

```python
# Loop through the list and sum all even
my_list = list(map(int, input("Enter numb
even_sum = 0
for num in my_list:
    if num % 2 == 0:
        even_sum += num
print("Sum of even numbers:", even_sum)
```

This will allow the user to input the numbers.

TASK3:

CODE WITH OUTPUT:



```
Sum of even numbers: 6
```

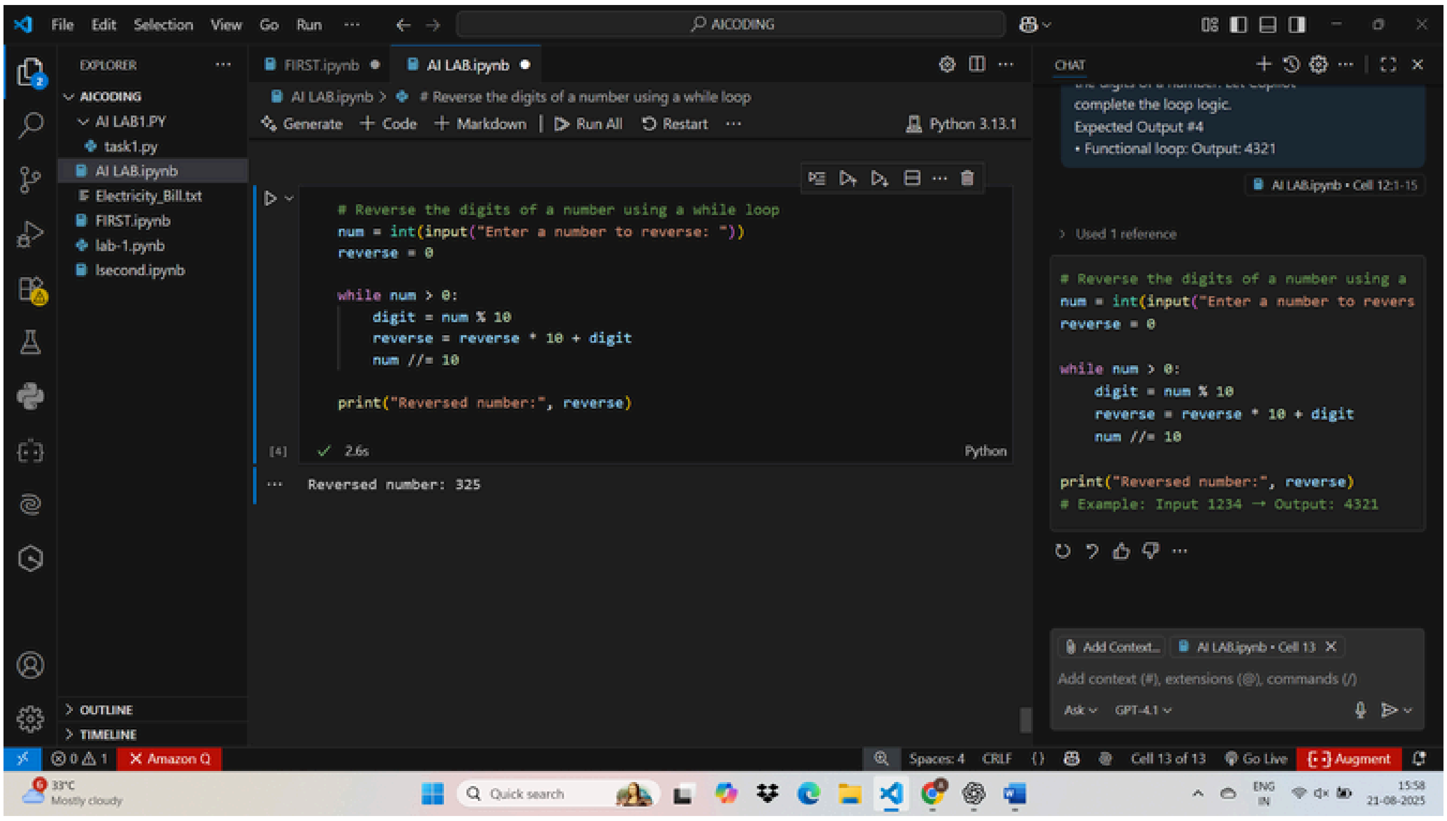
```python
def age_group(age):
    """Return the age group for a given age."""
    if age < 13:
        return "Child"
    elif age < 20:
        return "Teenager"
    elif age < 60:
        return "Adult"
    else:
        return "Senior"

# Take user input
user_age = int(input("Enter age: "))
print(age_group(user_age))
```

```
Adult
```

print(age_group(45)) # Output: Adult,TAKE USER INPUT

TASK-4:

CODE WITH OUTPUT



```python
# Reverse the digits of a number using a while loop
num = int(input("Enter a number to reverse: "))
reverse = 0

while num > 0:
    digit = num % 10
    reverse = reverse * 10 + digit
    num //= 10

print("Reversed number:", reverse)
```

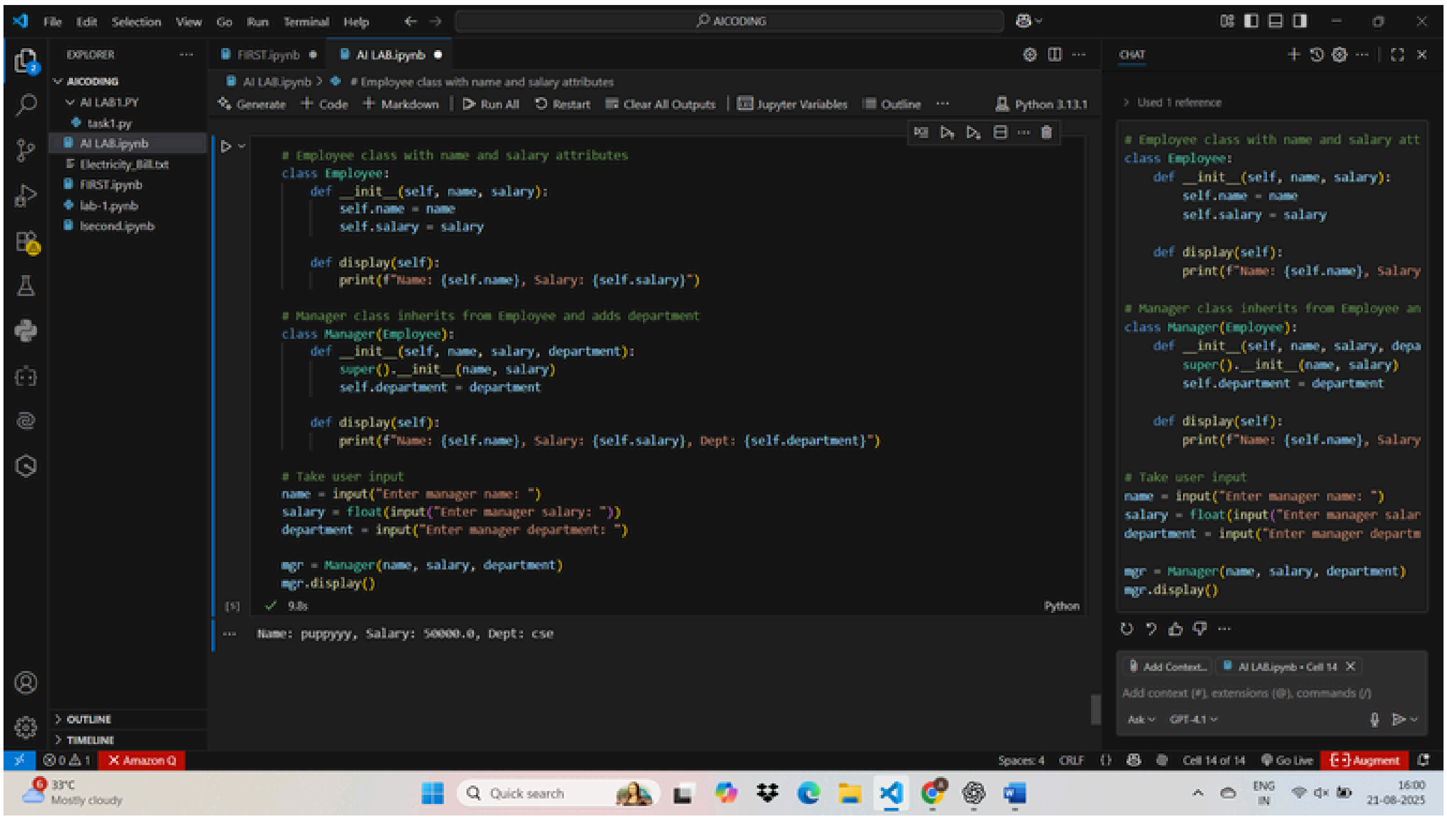
```
Reversed number: 325
```

complete the loop logic.
Expected Output #4
• Functional loop: Output: 4321

```python
# Reverse the digits of a number using a
num = int(input("Enter a number to revers
reverse = 0

while num > 0:
    digit = num % 10
    reverse = reverse * 10 + digit
    num //= 10

print("Reversed number:", reverse)
# Example: Input 1234 → Output: 4321
```

TASK 5:

Code with output:



File Edit Selection View Go Run Terminal Help
AICODING
EXPLORER
AICODING
AI LAB1.PY
task1.py
AI LAB.ipynb
Electricity_Bill.txt
FIRST.ipynb
lab-1.pynb
second.ipynb
FIRST.ipynb
AI LAB.ipynb
AI LAB.ipynb > # Employee class with name and salary attributes
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline
Python 3.13.1
CHAT
Used 1 reference
Add Context... AI LAB.ipynb • Cell 14
Add context (#), extensions (@), commands (/)
Ask GPT-4.1
OUTLINE
TIMELINE
Amazon Q
Spaces: 4 CRLF Cell 14 of 14 Go Live Augment
33°C Mostly cloudy
Quick search
ENG IN
16:00 21-08-2025

```python
# Employee class with name and salary attributes
class Employee:
    def __init__(self, name, salary):
        self.name = name
        self.salary = salary

    def display(self):
        print(f"Name: {self.name}, Salary: {self.salary}")

# Manager class inherits from Employee and adds department
class Manager(Employee):
    def __init__(self, name, salary, department):
        super().__init__(name, salary)
        self.department = department

    def display(self):
        print(f"Name: {self.name}, Salary: {self.salary}, Dept: {self.department}")

# Take user input
name = input("Enter manager name: ")
salary = float(input("Enter manager salary: "))
department = input("Enter manager department: ")

mgr = Manager(name, salary, department)
mgr.display()
```

[5] ✓ 9.8s

```
Name: puppyyy, Salary: 50000.0, Dept: cse
```