AI ASSISTED -CODING

ASSIGNMENT-6.5

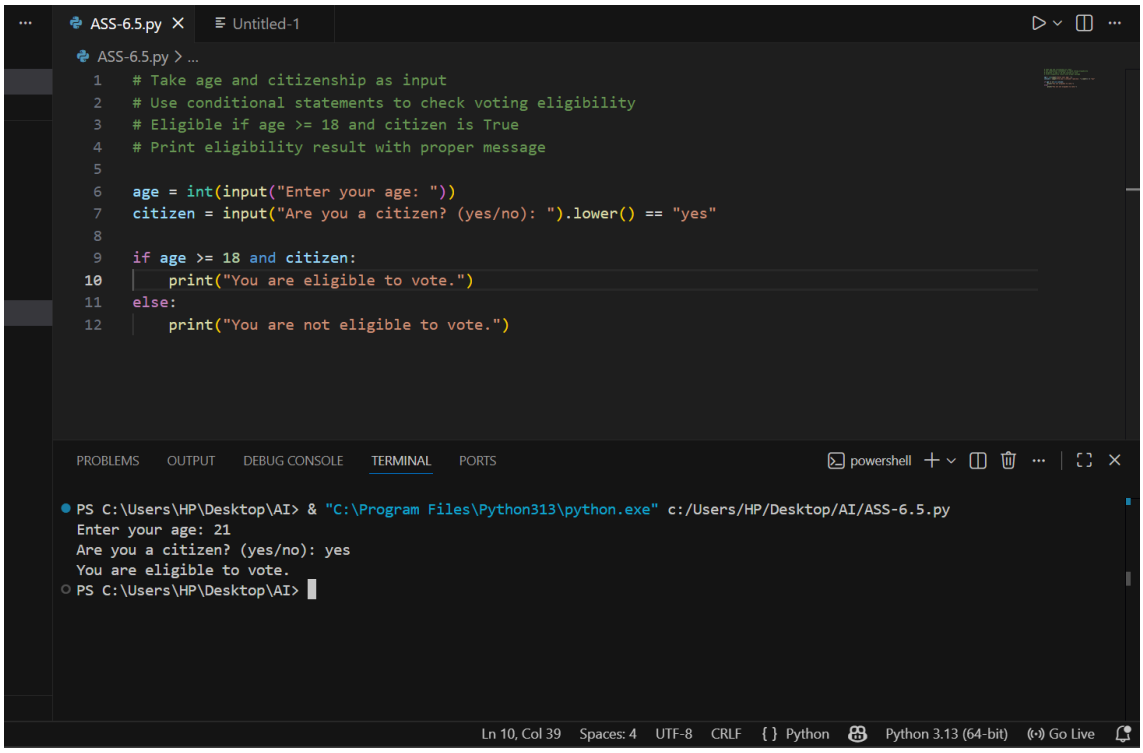
RUDROJU RUPA SRI

2303A51918

BATCH-30

Experiment 6: AI-Based Code Completion: Working with suggestions for classes, loops, conditionals
LO1. Use AI-based code completion tools to generate Python code involving classes, loops, and conditionals.
LO2. Interpret and explain AI-generated code line-by-line.
LO3. Identify errors, inefficiencies, or logical flaws in AI-suggested implementations.
LO4. Optimize AI-generated code for better readability and performance.
LO5. Demonstrate ethical and responsible use of AI tools in coding tasks.

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.



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

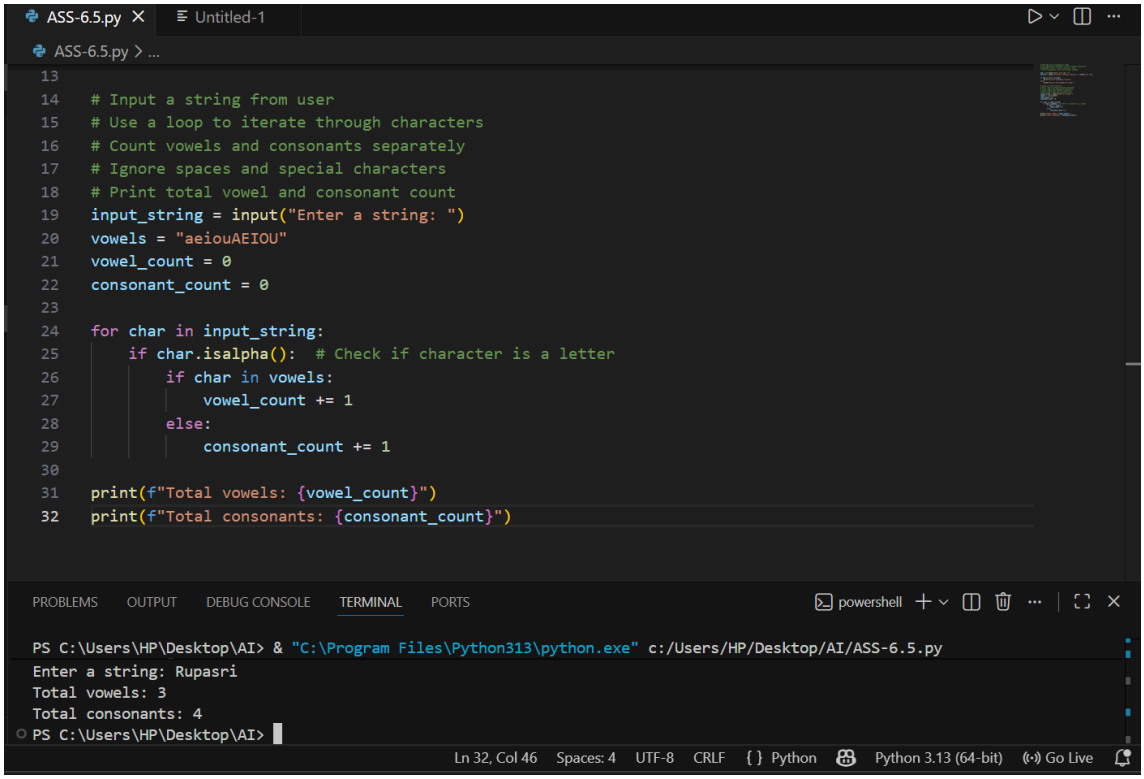
Task: Use an AI tool to process strings using loops.

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.



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

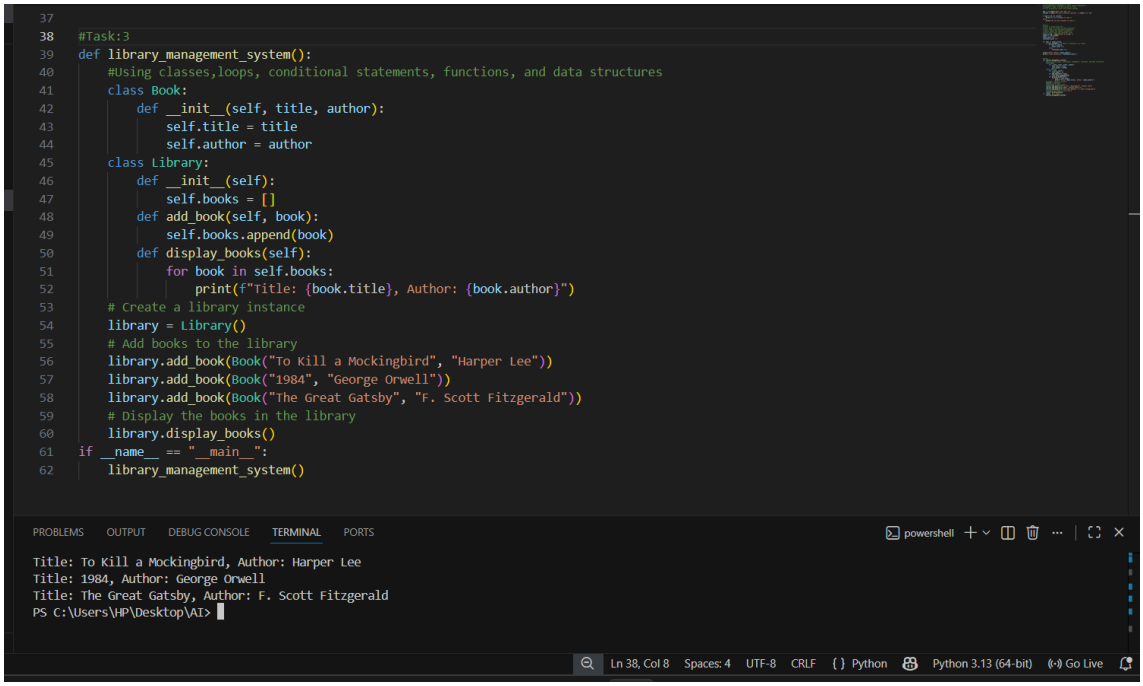
Task: Use an AI tool to generate a complete program using classes, loops, and conditionals.

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.



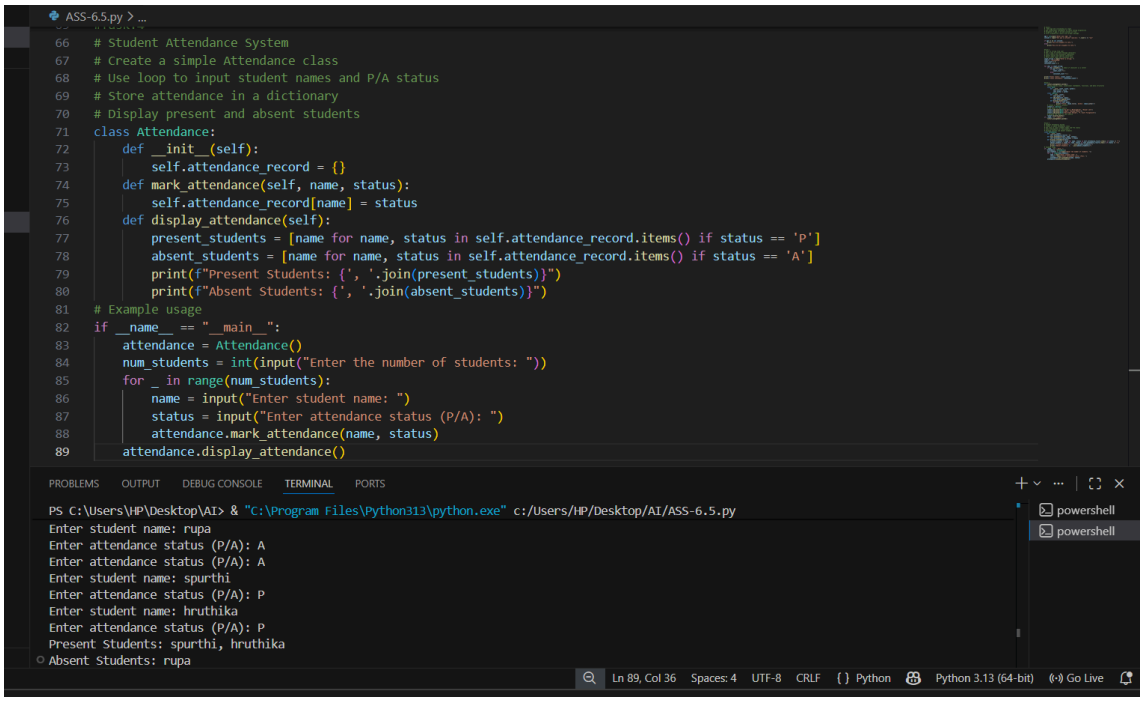
Task Description #4 (AI-Assisted Code Completion for Class-Based Attendance System)

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.



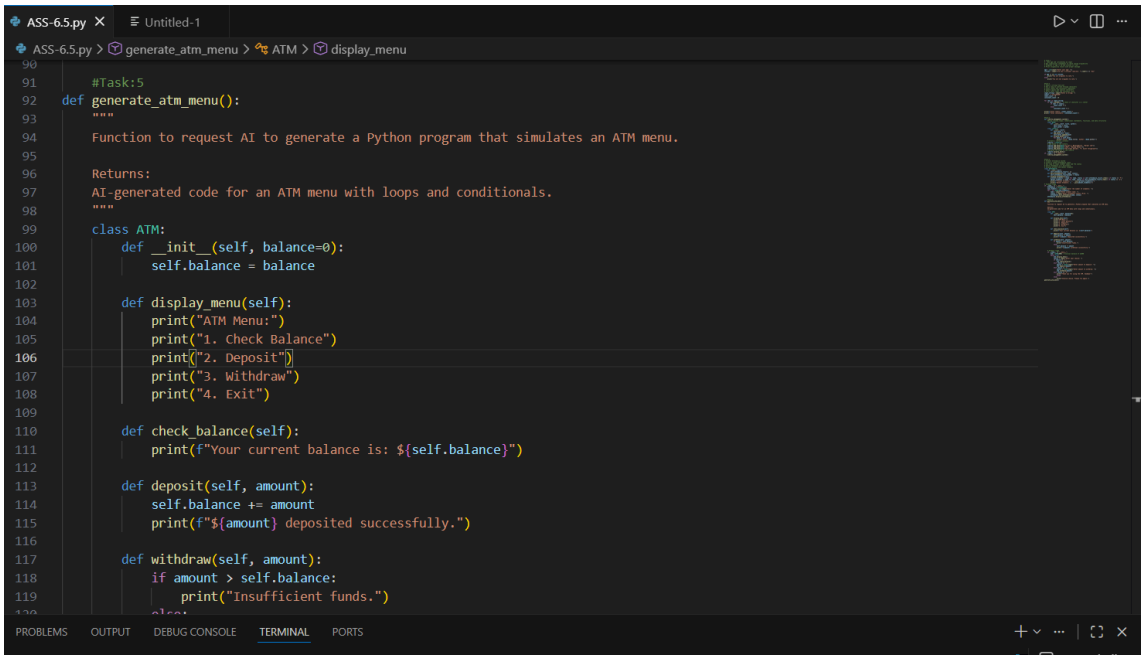
Task Description #5 (AI-Based Code Completion for Conditional Menu Navigation)

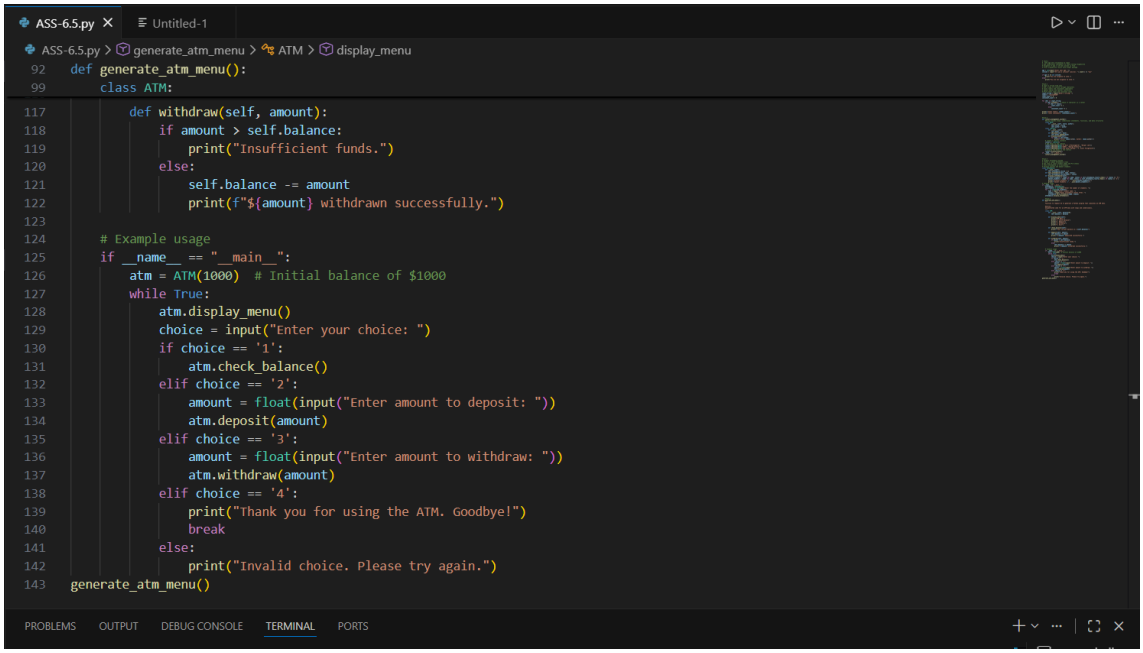
Task: Use an AI tool to complete a navigation menu.

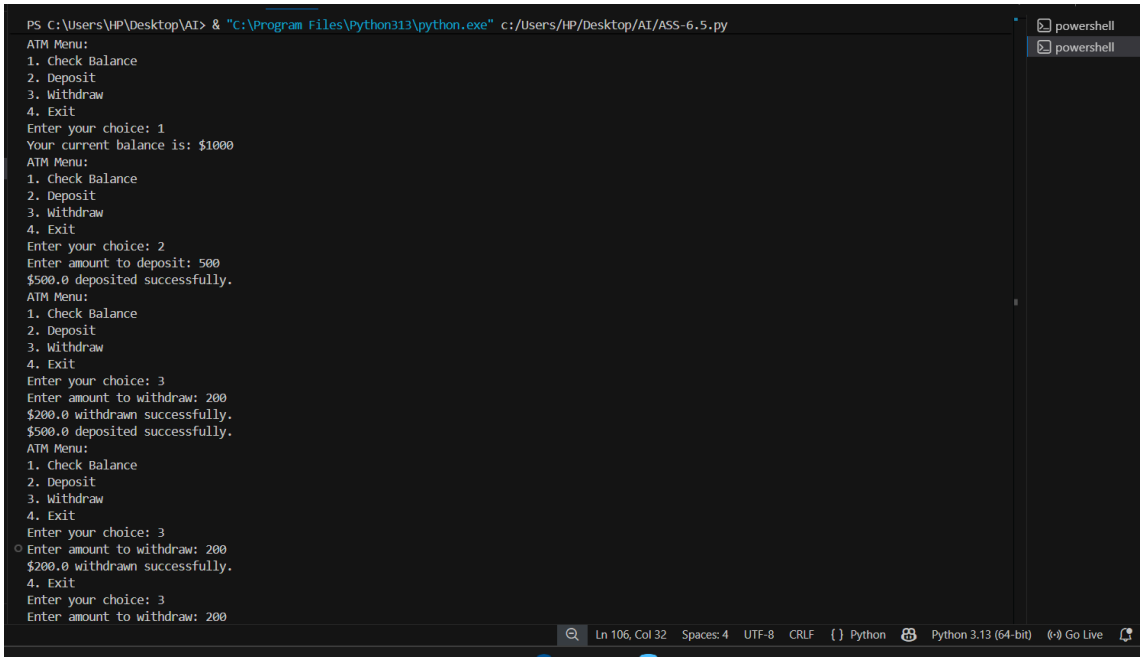
Prompt: “Generate a Python program using loops and conditionals to simulate an ATM menu.”

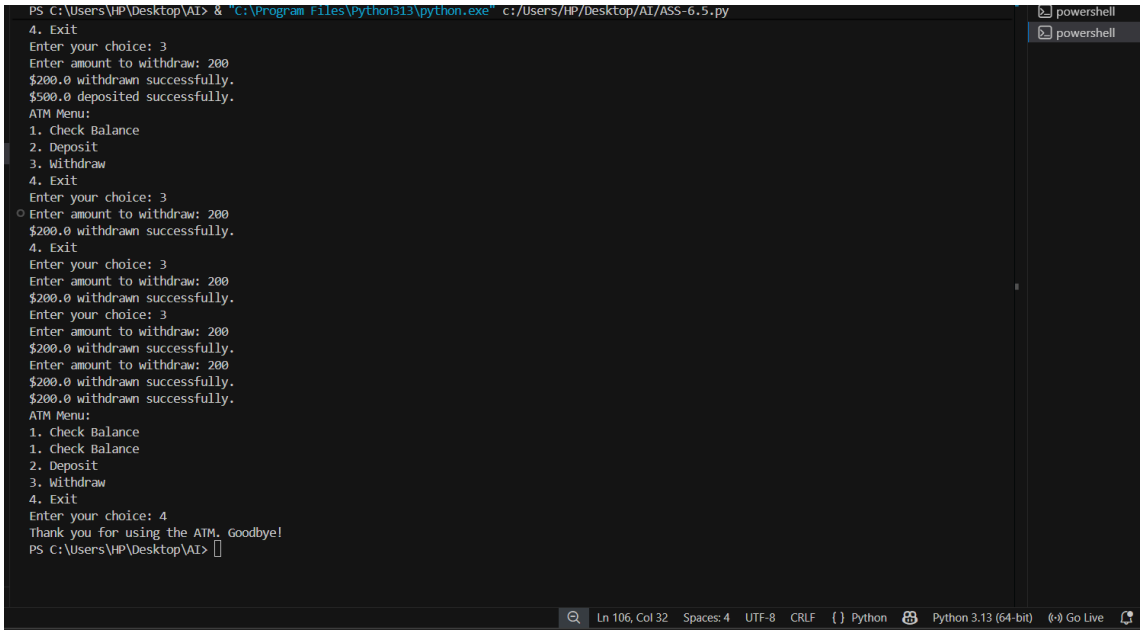
Expected Output:

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


ASS-6.5.py
Untitled-1
ASS-6.5.py > generate_atm_menu > ATM > display_menu
#Task:5
def generate_atm_menu():
"""
Function to request AI to generate a Python program that simulates an ATM menu.
Returns:
AI-generated code for an ATM menu with loops and conditionals.
"""
class ATM:
def __init__(self, balance=0):
self.balance = balance
def display_menu(self):
print("ATM Menu:")
print("1. Check Balance")
print("2. Deposit")
print("3. Withdraw")
print("4. Exit")
def check_balance(self):
print(f"Your current balance is: ${self.balance}")
def deposit(self, amount):
self.balance += amount
print(f"${amount} deposited successfully.")
def withdraw(self, amount):
if amount > self.balance:
print("Insufficient funds.")
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS


ASS-6.5.py
Untitled-1
ASS-6.5.py > generate_atm_menu > ATM > display_menu
def generate_atm_menu():
class ATM:
def withdraw(self, amount):
if amount > self.balance:
print("Insufficient funds.")
else:
self.balance -= amount
print(f"${amount} withdrawn successfully.")
# Example usage
if __name__ == "__main__":
atm = ATM(1000) # Initial balance of $1000
while True:
atm.display_menu()
choice = input("Enter your choice: ")
if choice == '1':
atm.check_balance()
elif choice == '2':
amount = float(input("Enter amount to deposit: "))
atm.deposit(amount)
elif choice == '3':
amount = float(input("Enter amount to withdraw: "))
atm.withdraw(amount)
elif choice == '4':
print("Thank you for using the ATM. Goodbye!")
break
else:
print("Invalid choice. Please try again.")
generate_atm_menu()
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS


PS C:\Users\HP\Desktop\AI> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/AI/ASS-6.5.py
ATM Menu:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 1
Your current balance is: $1000
ATM Menu:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 2
Enter amount to deposit: 500
$500.0 deposited successfully.
ATM Menu:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 3
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
$500.0 deposited successfully.
ATM Menu:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 3
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
4. Exit
Enter your choice: 3
Enter amount to withdraw: 200
powershell
powershell
Ln 106, Col 32
Spaces: 4
UTF-8
CRLF
{ } Python
Python 3.13 (64-bit)
Go Live


PS C:\Users\HP\Desktop\AI> & "C:\Program Files\Python313\python.exe" c:/Users/HP/Desktop/AI/ASS-6.5.py
4. Exit
Enter your choice: 3
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
$500.0 deposited successfully.
ATM Menu:
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 3
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
4. Exit
Enter your choice: 3
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
Enter your choice: 3
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
Enter amount to withdraw: 200
$200.0 withdrawn successfully.
$200.0 withdrawn successfully.
ATM Menu:
1. Check Balance
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter your choice: 4
Thank you for using the ATM. Goodbye!
PS C:\Users\HP\Desktop\AI>
powershell
powershell
Ln 106, Col 32
Spaces: 4
UTF-8
CRLF
{ } Python
Python 3.13 (64-bit)
Go Live