Thumati Pavan Venkata Narendra Kumar

EMP-ID-289219

PYTHON-CASE-STUDIES

1. ATM Simulation System.

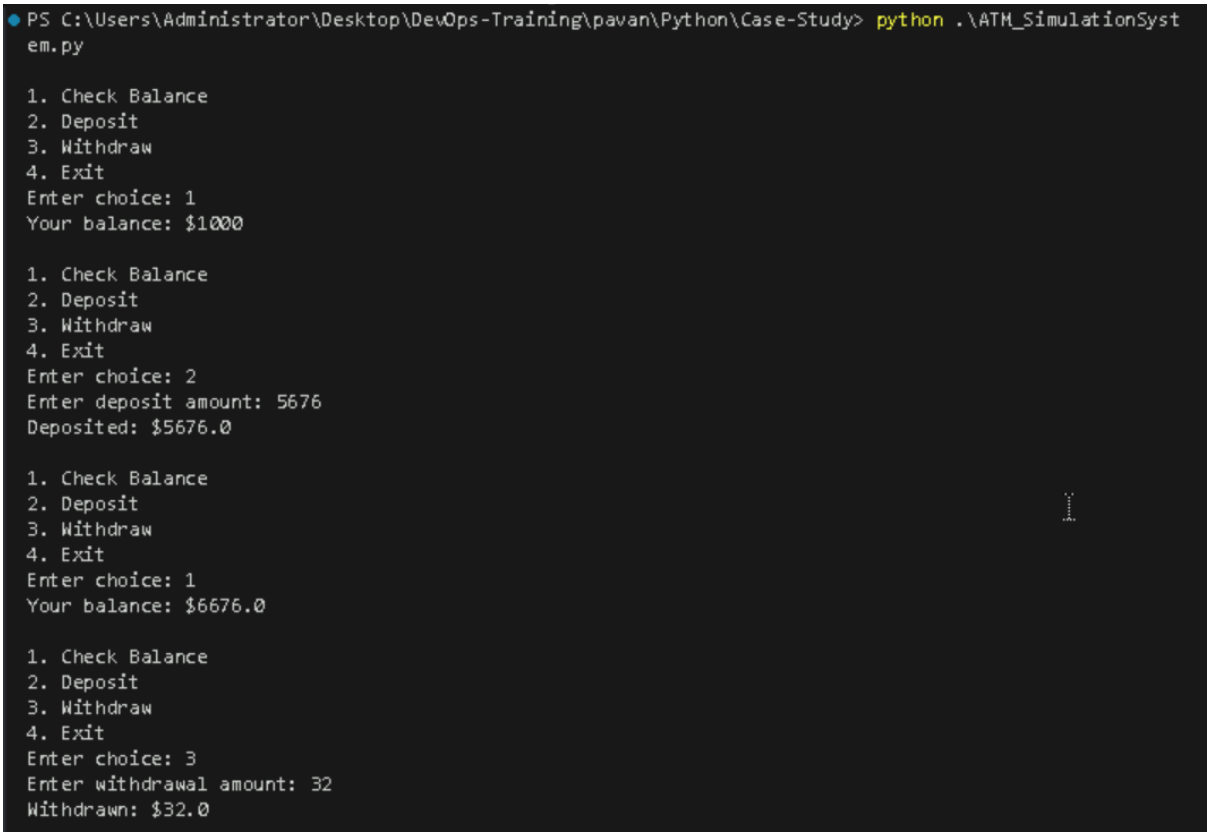
Code:

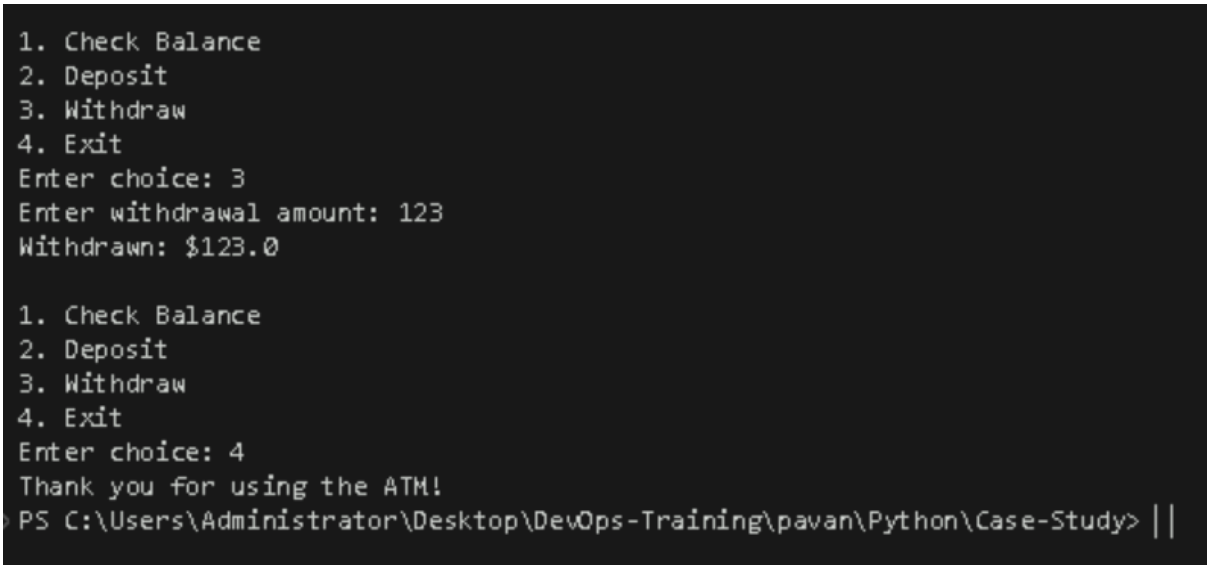
ATM_SimulationSystem.py

pavan > Python > Case-Study > ATM_SimulationSystem.py

```python
class ATM:
    def __init__(self, balance=1000):
        self.balance = balance
    def check_balance(self):
        print(f"Your balance: ${self.balance}")
    def deposit(self, amount):
        self.balance += amount
        print(f"Deposited: ${amount}")
    def withdraw(self, amount):
        if amount > self.balance:
            print("Insufficient funds!")
        else:
            self.balance -= amount
            print(f"Withdrawn: ${amount}")
def main():
    atm = ATM()
    while True:
        print("\n1. Check Balance\n2. Deposit\n3. Withdraw\n4. Exit")
        choice = input("Enter choice: ")
        if choice == "1":
            atm.check_balance()
        elif choice == "2":
            amt = float(input("Enter deposit amount: "))
            atm.deposit(amt)
        elif choice == "3":
            amt = float(input("Enter withdrawal amount: "))
            atm.withdraw(amt)
        elif choice == "4":
            print("Thank you for using the ATM!")
            break
        else:
            print("Invalid choice! Try again.")
main()
```

Output:


PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study> python .\ATM_SimulationSyst
em.py
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 1
Your balance: $1000
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 2
Enter deposit amount: 5676
Deposited: $5676.0
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 1
Your balance: $6676.0
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 3
Enter withdrawal amount: 32
Withdrawn: $32.0


1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 3
Enter withdrawal amount: 123
Withdrawn: $123.0
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 4
Thank you for using the ATM!
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study>

2. E-commerce Order Management.

Code:

E-commerceOrderManagement.py

pavan > Python > Case-Study > E-commerceOrderManagement.py

```python
class Product:
    def __init__(self, name, price):
        self.name = name
        self.price = price
class ShoppingCart:
    def __init__(self):
        self.cart = []
    def add_product(self, product):
        self.cart.append(product)
        print(f"{product.name} added to cart!")
    def view_cart(self):
        if not self.cart:
            print("Cart is empty!")
        else:
            print("\nShopping Cart:")
            total = 0
            for p in self.cart:
                print(f"- {p.name}: ${p.price}")
                total += p.price
            print(f"Total: ${total}")
    def checkout(self):
        if not self.cart:
            print("Cart is empty!")
        else:
            self.view_cart()
            print("Proceeding to checkout...")
```

E-commerceOrderManagement.py

pavan > Python > Case-Study > E-commerceOrderManagement.py

```python
class Product:...
class ShoppingCart:...
def main():
    cart = ShoppingCart()
    products = {
        "1": Product("Laptop", 1000),
        "2": Product("Headphones", 150),
        "3": Product("Mouse", 50),
    }
    while True:
        print("\n1. Add Laptop ($1000)\n2. Add Headphones ($150)\n3. Add Mouse ($50)\n4. View Cart
        choice = input("Enter choice: ")
        if choice in products:
            cart.add_product(products[choice])
        elif choice == "4":
            cart.view_cart()
        elif choice == "5":
            cart.checkout()
            break
        elif choice == "6":
            print("Thank you for shopping!")
            break
        else:
            print("Invalid choice!")

main()
```

Output:

```
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study> python .\E-commerceOrderManagement.py

 1. Add Laptop ($1000)
 2. Add Headphones ($150)
 3. Add Mouse ($50)
 4. View Cart
 5. Checkout
 6. Exit
 Enter choice: 4
 Cart is empty!

 1. Add Laptop ($1000)
 2. Add Headphones ($150)
 3. Add Mouse ($50)
 4. View Cart
 5. Checkout
 6. Exit
 Enter choice: 1
 Laptop added to cart!

 1. Add Laptop ($1000)
 2. Add Headphones ($150)
 3. Add Mouse ($50)
 4. View Cart
 1. Add Laptop ($1000)
 2. Add Headphones ($150)
 3. Add Mouse ($50)
 4. View Cart
 5. Checkout
 6. Exit
 Enter choice: 1
 Laptop added to cart!
```

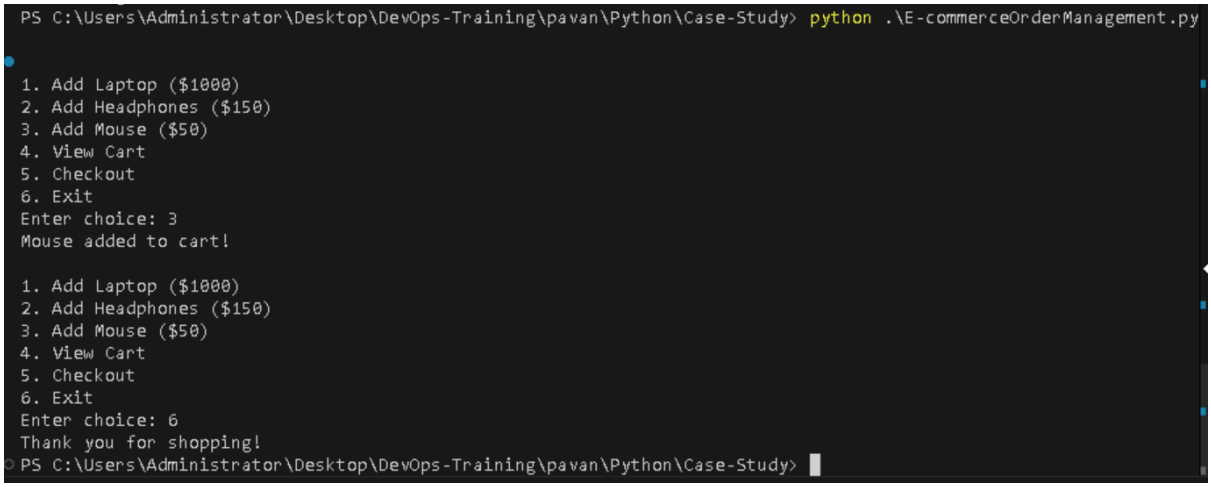
```
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study> python .\E-commerceOrderManagement.py


1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
5. Checkout
6. Exit
Enter choice: 4
Cart is empty!

1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
5. Checkout
6. Exit
Enter choice: 2
Headphones added to cart!

1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
5. Checkout
6. Exit
Enter choice: 5

Shopping Cart:
- Headphones: $150
Total: $150
Proceeding to checkout...
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study>
```


PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study> python .\E-commerceOrderManagement.py
1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
5. Checkout
6. Exit
Enter choice: 3
Mouse added to cart!
1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
5. Checkout
6. Exit
Enter choice: 6
Thank you for shopping!
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study>

3. Student Grade Management System.

Code:

studentGradeManagementSystem.py

pavan > Python > Case-Study > studentGradeManagementSystem.py

```python
class GradeSystem:
    def __init__(self):
        self.grades = {}

    def add_grade(self, name, grade):
        self.grades[name] = grade
        print(f"Added: {name} - {grade}")

    def view_grades(self):
        if not self.grades:
            print("No grades available!")
        else:
            print("\nStudent Grades:")
            for name, grade in self.grades.items():
                print(f"{name}: {grade}")

    def calculate_average(self):
        if not self.grades:
            print("No grades available!")
        else:
            avg = sum(self.grades.values()) / len(self.grades)
            print(f"Class Average: {avg:.2f}")

```

studentGradeManagementSystem.py

pavan > Python > Case-Study > studentGradeManagementSystem.py

```python
def main():
    system = GradeSystem()

    while True:
        print("\n1. Add Grade\n2. View Grades\n3. Calculate Average\n4. Exit")
        choice = input("Enter choice: ")

        if choice == "1":
            name = input("Enter student name: ")
            grade = float(input("Enter grade: "))
            system.add_grade(name, grade)
        elif choice == "2":
            system.view_grades()
        elif choice == "3":
            system.calculate_average()
        elif choice == "4":
            print("Exiting Grade System.")
            break
        else:
            print("Invalid choice!")

if __name__ == "__main__":
    main()
```

Output:

```
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study> python .\s
tudentGradeManagementSystem.py

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 2
No grades available!

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 1
Enter student name: Ram
Enter grade: 89
Added: Ram - 89.0

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 1
Enter student name: Yonith
Enter grade: 100
Added: Yonith - 100.0
```

```
1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 2

Student Grades:
Ram: 89.0
Yonith: 100.0

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 3
Class Average: 94.50

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 4
Exiting Grade System.
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study>
```

4. Hospital Patient Management.

Code:

hospitalPatientManagement.py

pavan > Python > Case-Study > hospitalPatientManagement.py

```python
class Hospital:
    def __init__(self):
        self.patients = {}

    def add_patient(self, id, name, age, disease):
        self.patients[id] = {"Name": name, "Age": age, "Disease": disease}
        print(f"Patient {name} added!")

    def view_patients(self):
        if not self.patients:
            print("No patients registered!")
        else:
            print("\nPatient Records:")
            for id, details in self.patients.items():
                print(f"ID: {id} - {details}")

    def remove_patient(self, id):
        if id in self.patients:
            del self.patients[id]
            print("Patient removed!")
        else:
            print("Patient not found!")

```

hospitalPatientManagement.py

pavan > Python > Case-Study > hospitalPatientManagement.py

```python
> class Hospital: ···

def main():
    hospital = Hospital()

    while True:
        print("\n1. Add Patient\n2. View Patients\n3. Remove Patient\n4. Exit")
        choice = input("Enter choice: ")

        if choice == "1":
            id = input("Enter Patient ID: ")
            name = input("Enter Name: ")
            age = input("Enter Age: ")
            disease = input("Enter Disease: ")
            hospital.add_patient(id, name, age, disease)
        elif choice == "2":
            hospital.view_patients()
        elif choice == "3":
            id = input("Enter Patient ID to remove: ")
            hospital.remove_patient(id)
        elif choice == "4":
            print("Exiting Hospital System.")
            break
        else:
            print("Invalid choice!")

if __name__ == "__main__":
    main()

```

Output:

```
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study>
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study> python .\hospitalPatientManagement.py


1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 2
No patients registered!

1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 1
Enter Patient ID: 101
Enter Name: Varma
Enter Age: 28
Enter Disease: Asthma
Patient Varma added!

1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 1
Enter Patient ID: 103
Enter Name: Kunal
Enter Age: 34
Enter Disease: Flu
Patient Kunal added!
```

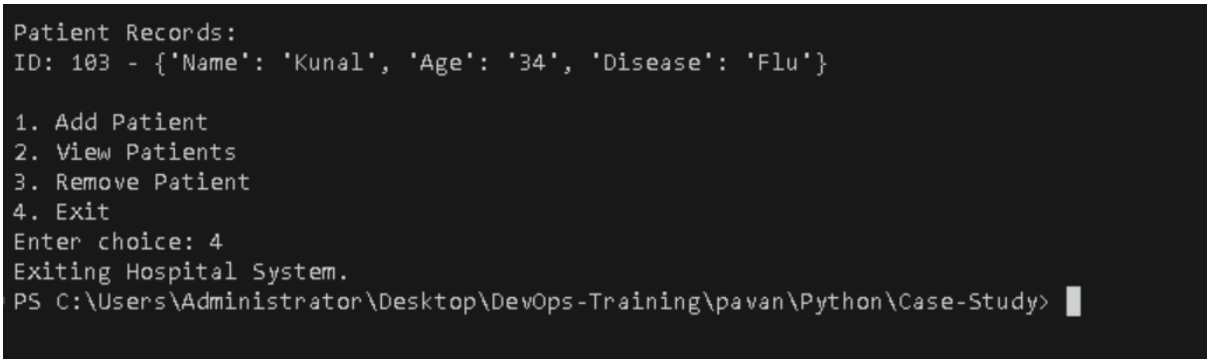
```
1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 2

Patient Records:
ID: 101 - {'Name': 'Varma', 'Age': '28', 'Disease': 'Asthma'}
ID: 103 - {'Name': 'Kunal', 'Age': '34', 'Disease': 'Flu'}

1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 3
Enter Patient ID to remove: 102
Patient not found!

1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 3
Enter Patient ID to remove: 101
Patient removed!

1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 2
```


Patient Records:
ID: 103 - {'Name': 'Kunal', 'Age': '34', 'Disease': 'Flu'}
1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 4
Exiting Hospital System.
PS C:\Users\Administrator\Desktop\DevOps-Training\pavan\Python\Case-Study>