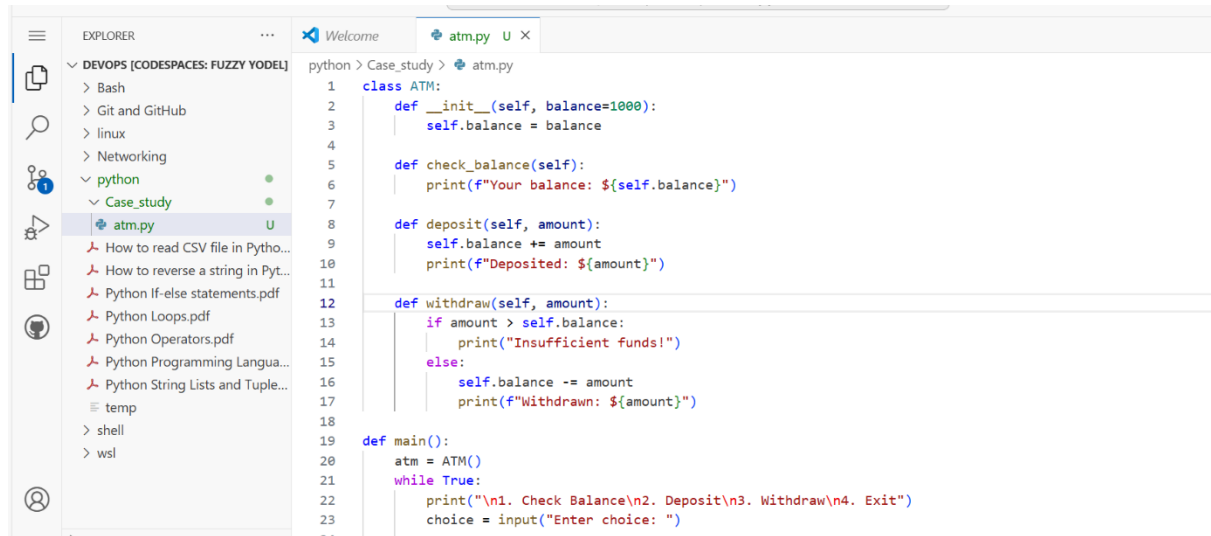
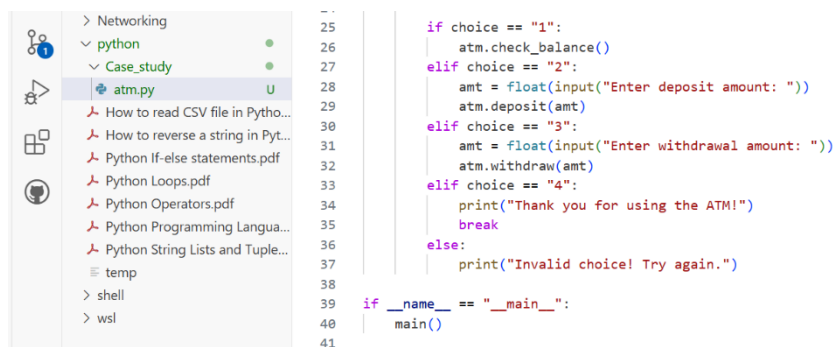


## CASE STUDY:

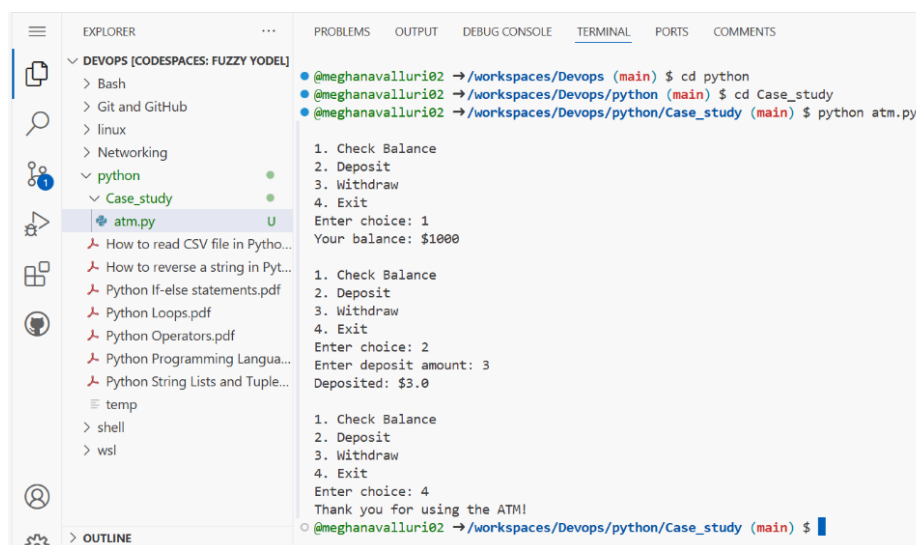
### 1. ATM Simulation System



```
python > Case_study > atm.py
1 class ATM:
2     def __init__(self, balance=1000):
3         self.balance = balance
4
5     def check_balance(self):
6         print(f"Your balance: ${self.balance}")
7
8     def deposit(self, amount):
9         self.balance += amount
10        print(f"Deposited: ${amount}")
11
12    def withdraw(self, amount):
13        if amount > self.balance:
14            print("Insufficient funds!")
15        else:
16            self.balance -= amount
17            print(f"Withdrawn: ${amount}")
18
19    def main():
20        atm = ATM()
21        while True:
22            print("\n1. Check Balance\n2. Deposit\n3. Withdraw\n4. Exit")
23            choice = input("Enter choice: ")
24
```



```
25
26     if choice == "1":
27         atm.check_balance()
28     elif choice == "2":
29         amt = float(input("Enter deposit amount: "))
30         atm.deposit(amt)
31     elif choice == "3":
32         amt = float(input("Enter withdrawal amount: "))
33         atm.withdraw(amt)
34     elif choice == "4":
35         print("Thank you for using the ATM!")
36         break
37     else:
38         print("Invalid choice! Try again.")
39
40 if __name__ == "__main__":
41     main()
42
```



```
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: 3
Deposited: $3.0

1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Enter choice: 4
Thank you for using the ATM!
```

## 2. E-commerce Order Management

The screenshot displays the Visual Studio Code interface for a Python project titled "E-commerce Order Management". The Explorer on the left shows the project structure, including a "Case\_study" folder containing "atm.py" and "E-commerce.py". The Editor shows the code for "E-commerce.py", which defines a "Product" class, a "ShoppingCart" class, and a "main" function. The code includes methods for adding products to the cart, viewing the cart, and checking out. The Terminal at the bottom shows the execution of the program, with user input for adding items to the cart and checking out.

```
python > Case_study > E-commerce.py
1 class Product:
2     def __init__(self, name, price):
3         self.name = name
4         self.price = price
5
6 class ShoppingCart:
7     def __init__(self):
8         self.cart = []
9
10    def add_product(self, product):
11        self.cart.append(product)
12        print(f"{product.name} added to cart!")
13
14    def view_cart(self):
15        if not self.cart:
16            print("Cart is empty!")
17        else:
18            print("\nShopping Cart:")
19            total = 0
20            for p in self.cart:
21                print(f"- {p.name}: ${p.price}")
22                total += p.price
23            print(f"Total: ${total}")
24
25    def checkout(self):
26        if not self.cart:
27            print("Cart is empty!")
28        else:
29            self.view_cart()
30            print("Proceeding to checkout...")
31
32    def main():
33        cart = ShoppingCart()
34        products = {
35            "1": Product("Laptop", 1000),
36            "2": Product("Headphones", 150),
37            "3": Product("Mouse", 50),
38        }
39
40        while True:
41            print("\n1. Add Laptop ($1000)\n2. Add Headphones ($150)\n3. Add Mouse ($50)\n4. View Cart\n5. Check
42            choice = input("Enter choice: ")
43
44            if choice in products:
45                cart.add_product(products[choice])
46            elif choice == "4":
47                cart.view_cart()
48
49            elif choice == "5":
50                cart.checkout()
51                break
52            elif choice == "6":
53                print("Thank you for shopping!")
54                break
55            else:
56                print("Invalid choice!")
57
58    if __name__ == "__main__":
59        main()
```

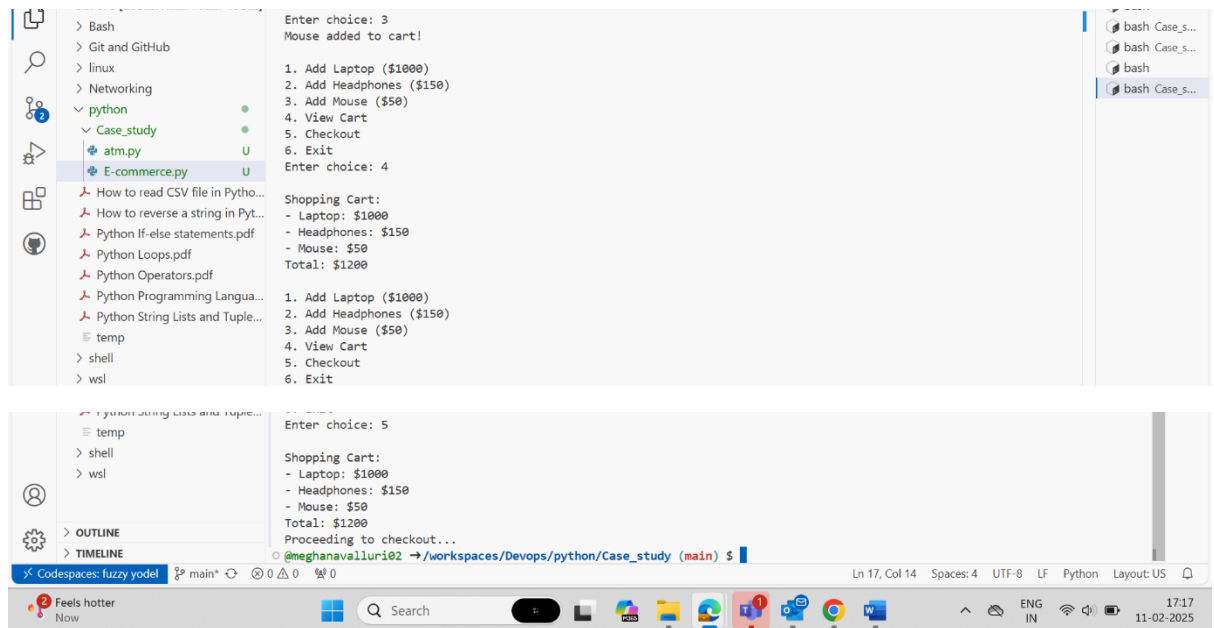
The Terminal output shows the execution of the program:

```
@meghanavalluri02 -> /workspaces/Devops (main) $ cd python
@meghanavalluri02 -> /workspaces/Devops/python (main) $ cd Case_study
@meghanavalluri02 -> /workspaces/Devops/python/Case_study (main) $ python E-commerce.py

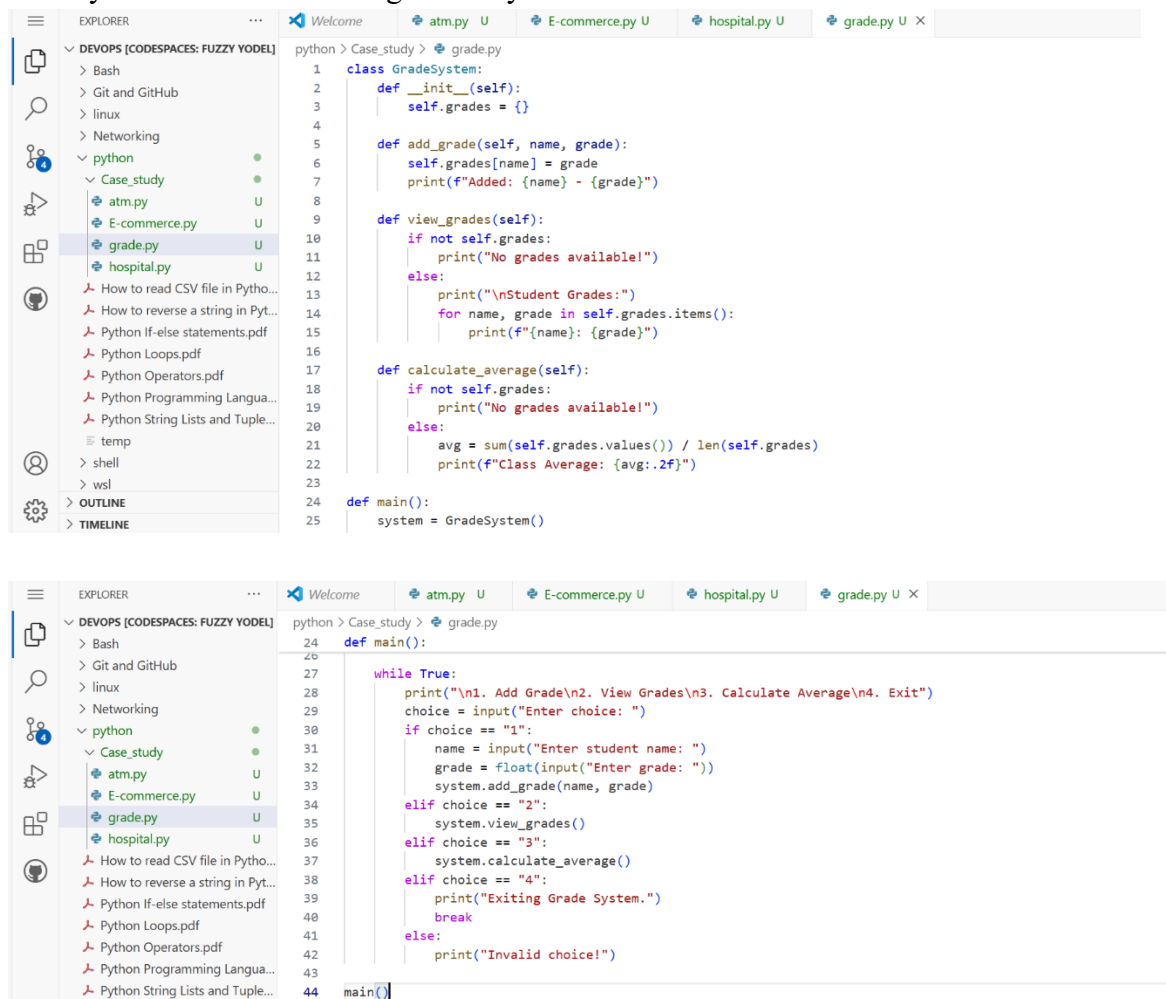
1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
5. Checkout
6. Exit
Enter choice: 1
Laptop added to cart!

1. Add Laptop ($1000)
2. Add Headphones ($150)
3. Add Mouse ($50)
4. View Cart
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
```



### 3. Case Study: Student Grade Management System



```

@meghanavalluri02 → /workspaces/Devops (main) $ cd python
@meghanavalluri02 → /workspaces/Devops/python (main) $ cd Case_study/
@meghanavalluri02 → /workspaces/Devops/python/Case_study (main) $ python grade.py

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 1
Enter student name: meghana
Enter grade: 8
Added: meghana - 8.0

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 1
Enter student name: meghana
Enter grade: 6
Added: meghana - 6.0

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

```

```

Student Grades:
meghana: 6.0

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 1
Enter student name: pravalika
Enter grade: 9
Added: pravalika - 9.0

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

Student Grades:
meghana: 6.0
pravalika: 9.0

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

```

## 4. Case Study: Hospital Patient Management

```

python > Case_study > hospital.py
1 class Hospital:
2     def __init__(self):
3         self.patients = {}
4
5     def add_patient(self, id, name, age, disease):
6         self.patients[id] = {"Name": name, "Age": age, "Disease": disease}
7         print(f"Patient {name} added!")
8
9     def view_patients(self):
10        if not self.patients:
11            print("No patients registered!")
12        else:
13            print("\nPatient Records:")
14            for id, details in self.patients.items():
15                print(f"ID: {id} - {details}")
16
17    def remove_patient(self, id):
18        if id in self.patients:
19            del self.patients[id]
20            print("Patient removed!")
21        else:
22            print("Patient not found!")
23
24    def main():
25        hospital = Hospital()

```

```
python > Case_study > hospital.py
24 def main():
25     while True:
26         print("\n1. Add Patient\n2. View Patients\n3. Remove Patient\n4. Exit")
27         choice = input("Enter choice: ")
28         if choice == "1":
29             id = input("Enter Patient ID: ")
30             name = input("Enter Name: ")
31             age = input("Enter Age: ")
32             disease = input("Enter Disease: ")
33             hospital.add_patient(id, name, age, disease)
34         elif choice == "2":
35             hospital.view_patients()
36         elif choice == "3":
37             id = input("Enter Patient ID to remove: ")
38             hospital.remove_patient(id)
39         elif choice == "4":
40             print("Exiting Hospital System.")
41             break
42         else:
43             print("Invalid choice!")
44     main()
```

```
python - Case_study
@meghanavalluri02 → /workspaces/Devops (main) $ cd python
@meghanavalluri02 → /workspaces/Devops/python (main) $ cd Case_study
@meghanavalluri02 → /workspaces/Devops/python/Case_study (main) $ python hospital.py

1. Add Patient
2. View Patients
3. Remove Patient
4. Exit
Enter choice: 1
Enter Patient ID: 111
Enter Name: Atlas
Enter Age: 22
Enter Disease: jaundice
Patient Atlas added!

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

Patient Records:
ID: 111 - {'Name': 'Atlas', 'Age': '22', 'Disease': 'jaundice'}

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