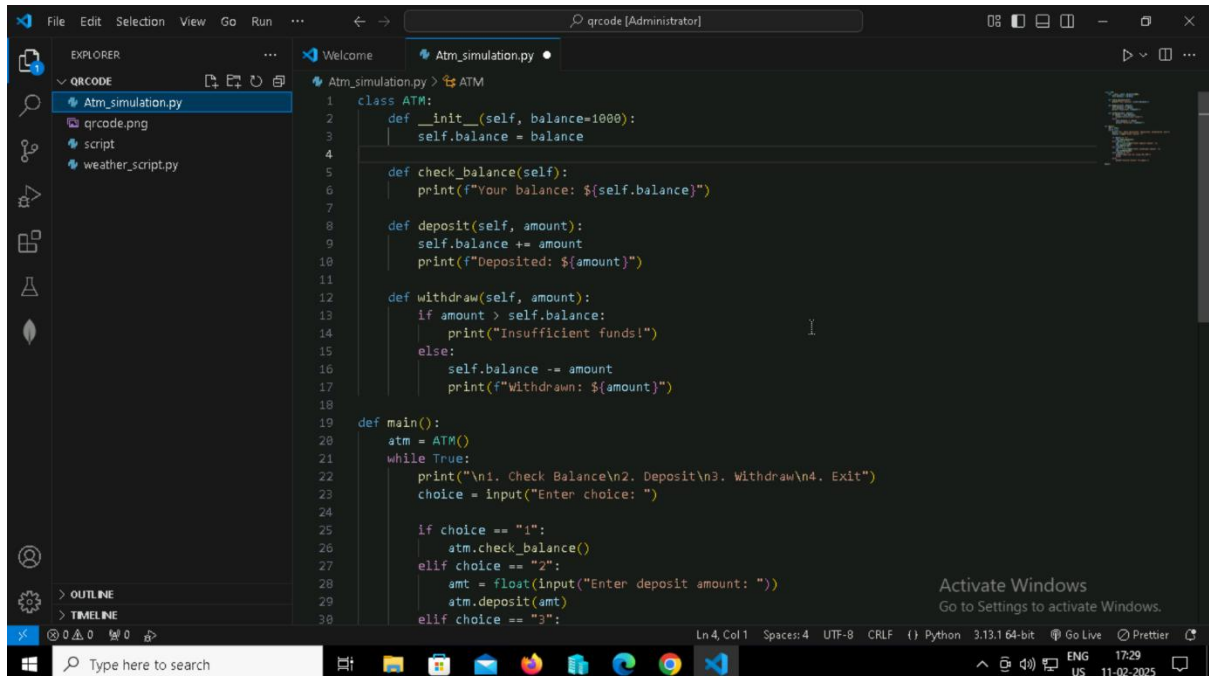


PYTHON CASE STUDIES

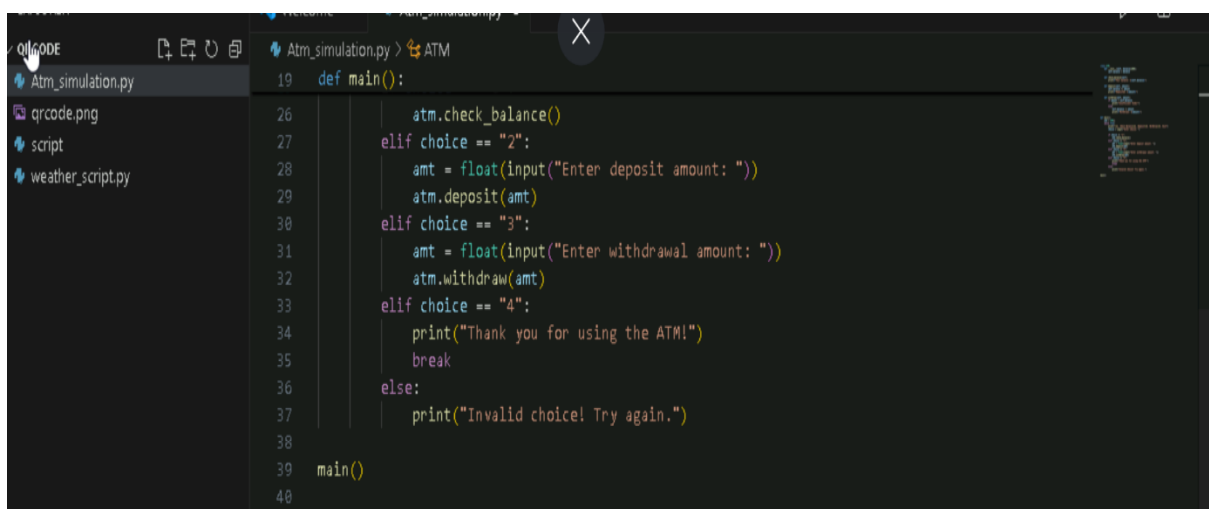
Case Study 1 :ATM Simulation System Program



The screenshot shows a Visual Studio Code editor window with the file explorer on the left displaying a project named 'QRCODE' containing files 'Atm_simulation.py', 'qrcode.png', 'script', and 'weather_script.py'. The main editor area shows the 'Atm_simulation.py' file with the following Python code:

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17            print(f"Withdrawn: ${amount}")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

The status bar at the bottom indicates 'Ln 4, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python 3.13.1 64-bit', 'Go Live', and 'Prettier'.



The screenshot shows a continuation of the Visual Studio Code editor window, focusing on the 'main()' function of the 'Atm_simulation.py' file. The code continues from line 19:

```
19 def main():
20
21     atm.check_balance()
22
23     elif choice == "2":
24         amt = float(input("Enter deposit amount: "))
25         atm.deposit(amt)
26
27     elif choice == "3":
28         amt = float(input("Enter withdrawal amount: "))
29         atm.withdraw(amt)
30
31     elif choice == "4":
32         print("Thank you for using the ATM!")
33         break
34
35     else:
36         print("Invalid choice! Try again.")
37
38
39 main()
40
```

The status bar at the bottom shows 'Ln 4, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python 3.13.1 64-bit', 'Go Live', and 'Prettier'.

The screenshot shows a Visual Studio Code editor with a file explorer on the left containing 'Atm_simulation.py', 'qrcode.png', 'script', and 'weather_script.py'. The main editor window displays the 'Atm_simulation.py' file with the following code:

```
19 def main():  
    Enter deposit amount: 500  
    Deposited: $500.0  
  
    1. Check Balance  
    2. Deposit  
    3. Withdraw  
    4. Exit  
    Enter choice: 3  
    Enter withdrawal amount: 2000  
    Insufficient funds!  
  
    1. Check Balance  
    2. Deposit  
    3. Withdraw  
    3. Withdraw  
    4. Exit  
    Enter choice: 1  
    Your balance: $1500.0  
  
    1. Check Balance  
    2. Deposit  
    3. Withdraw  
    4. Exit  
    4. Exit  
    Enter choice:  
    4. Exit  
    4. Exit  
    Enter choice:
```

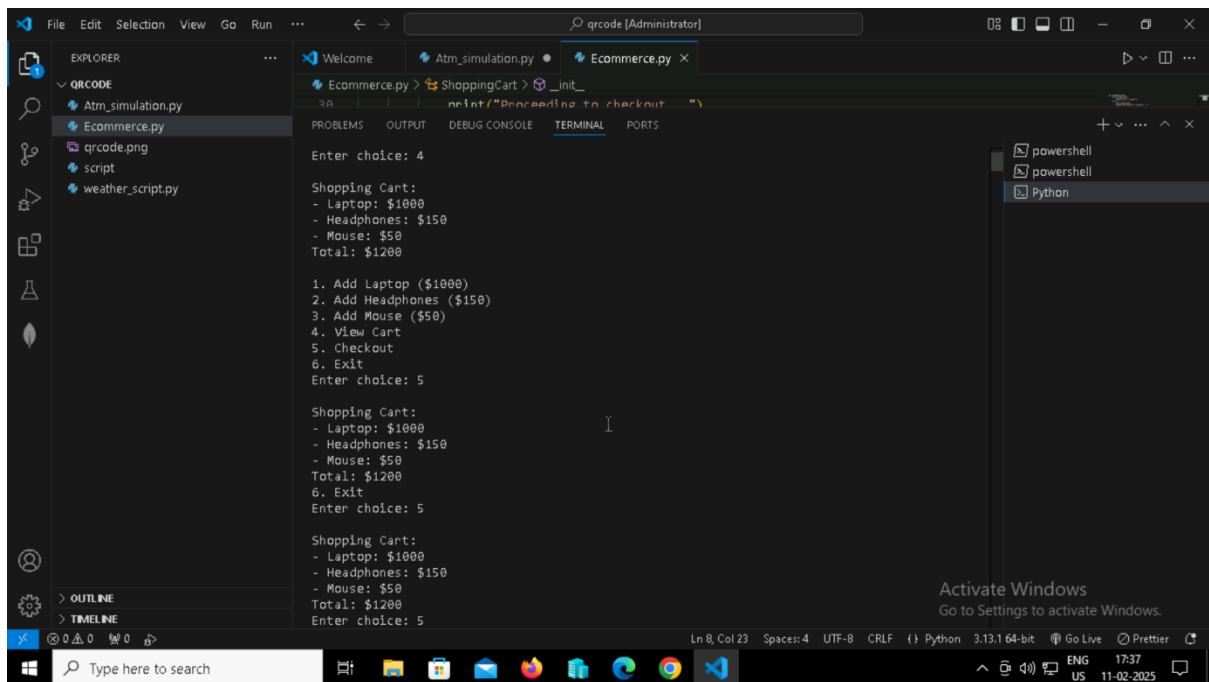
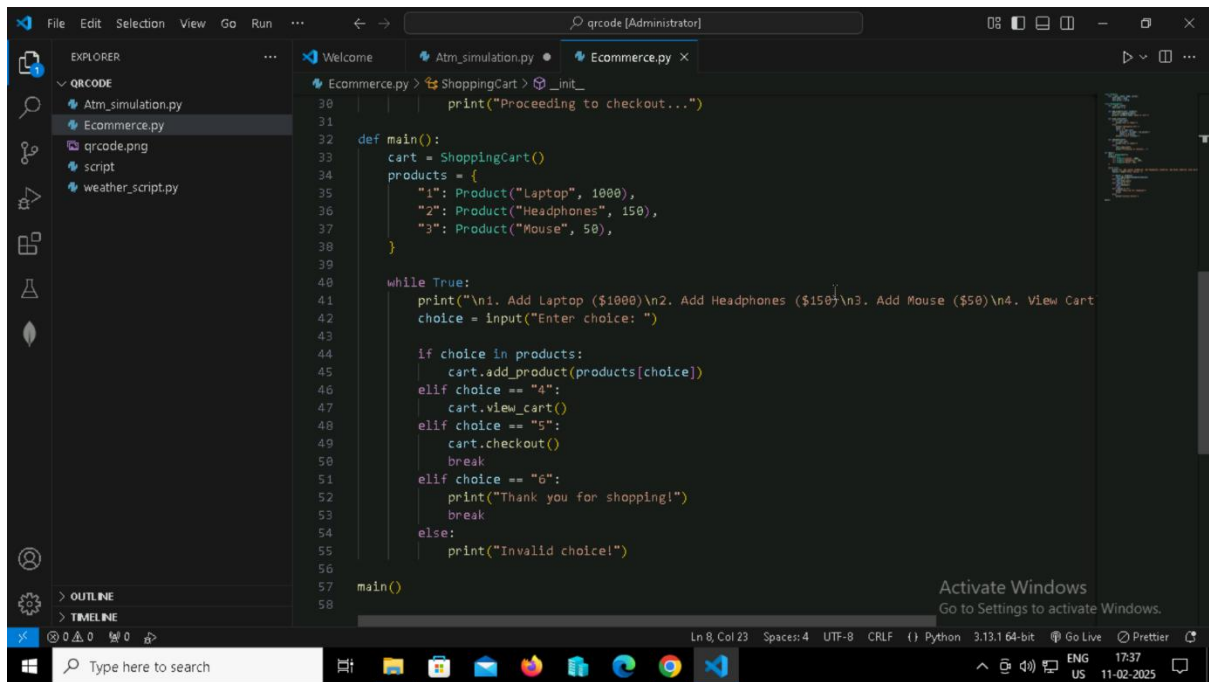
The terminal output shows the execution of the program, including the deposit of 500, the attempt to withdraw 2000 (insufficient funds), and the subsequent withdrawal of 1000, leaving a balance of 1500. The program then prompts for further actions.

Case study 2 Ecommerce Order Management System

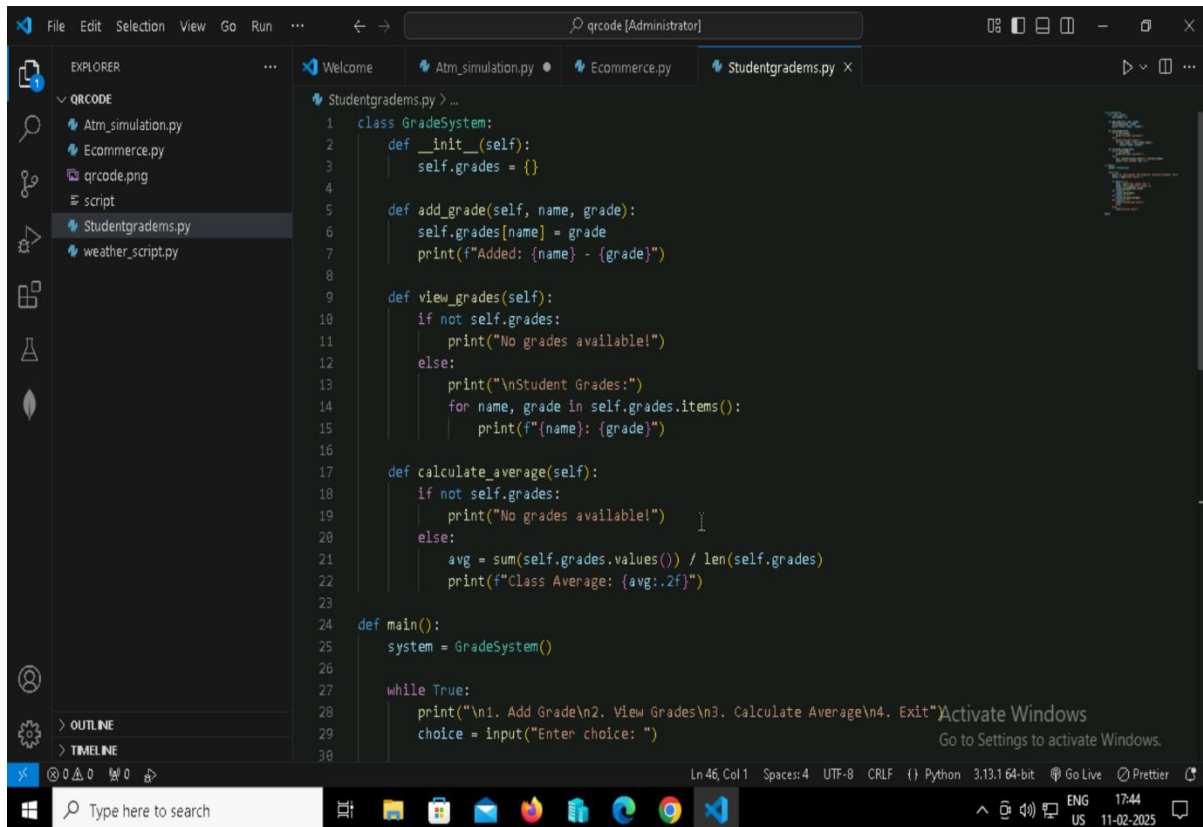
The screenshot shows a Visual Studio Code editor with a file explorer on the left containing 'Atm_simulation.py', 'Ecommerce.py', 'qrcode.png', 'script', and 'weather_script.py'. The main editor window displays the 'Ecommerce.py' file with the following code:

```
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...")
```

The terminal output shows the execution of the program, including the addition of a product to the cart, viewing the cart, and proceeding to checkout.



Case study 3 Student Grade Management System



The image shows a screenshot of a Visual Studio Code editor window. The Explorer panel on the left shows a project named 'QR CODE' with files: 'Atm_simulation.py', 'Ecommerce.py', 'qrcode.png', 'script', 'Studentgradems.py' (selected), and 'weather_script.py'. The main editor area displays the code for 'Studentgradems.py'. The code defines a 'GradeSystem' class with methods for adding, viewing, and calculating average grades, and a 'main' function that runs a loop with a menu.

```
1 class GradeSystem:
2     def __init__(self):
3         self.grades = {}
4
5     def add_grade(self, name, grade):
6         self.grades[name] = grade
7         print(f"Added: {name} - {grade}")
8
9     def view_grades(self):
10        if not self.grades:
11            print("No grades available!")
12        else:
13            print("\nStudent Grades:")
14            for name, grade in self.grades.items():
15                print(f"{name}: {grade}")
16
17    def calculate_average(self):
18        if not self.grades:
19            print("No grades available!")
20        else:
21            avg = sum(self.grades.values()) / len(self.grades)
22            print(f"Class Average: {avg:.2f}")
23
24    def main():
25        system = GradeSystem()
26
27        while True:
28            print("\n1. Add Grade\n2. View Grades\n3. Calculate Average\n4. Exit")
29            choice = input("Enter choice: ")
30
```

At the bottom of the editor, there is a status bar showing 'Ln 46, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python', '3.13.1 64-bit', 'Go Live', and 'Prettier'. A Windows taskbar is visible at the very bottom with the search bar and several application icons.

```
1 class GradeSystem:
2     def calculate_average(self):
3         avg = sum(self.grades.values()) / len(self.grades)
4         print(f"Class Average: {avg:.2f}")
5
6     def add_grade(self, name, grade):
7         self.grades[name] = grade
8
9     def view_grades(self):
10        print(self.grades)
11
12    def calculate_average(self):
13        avg = sum(self.grades.values()) / len(self.grades)
14        print(f"Class Average: {avg:.2f}")
15
16    def main():
17        system = GradeSystem()
18
19        while True:
20            print("\n1. Add Grade\n2. View Grades\n3. Calculate Average\n4. Exit")
21            choice = input("Enter choice: ")
22
23            if choice == "1":
24                name = input("Enter student name: ")
25                grade = float(input("Enter grade: "))
26                system.add_grade(name, grade)
27            elif choice == "2":
28                system.view_grades()
29            elif choice == "3":
30                system.calculate_average()
31            elif choice == "4":
32                print("Exiting Grade System.")
33                break
34            else:
35                print("Invalid choice!")
36
37    main()
38
```

```
4. Exit
Enter choice: 1
Enter student name: kk
Enter grade: 78
Added: kk - 78.0

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

Student Grades:
kk: 78.0

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 3
Class Average: 78.00

1. Add Grade
2. View Grades
3. Calculate Average
4. Exit
Enter choice: 4
Exiting Grade System.
PS C:\Users\Administrator\Desktop\qrcode>
```

The image shows a Windows 10 desktop environment. The primary application is Visual Studio Code, which is open to a Python file named 'Hospitalpatientms.py'. The code defines a 'Hospital' class with methods for adding, viewing, and removing patients, and a 'main' function to execute the program. The Explorer sidebar on the left shows the project structure, including a 'qrcode.png' file. The status bar at the bottom indicates the file is encoded in UTF-8 and is 3.13 KB in size. The taskbar at the bottom shows various application icons, including the Start menu, search bar, and several open applications like File Explorer, Edge, and VS Code. The system tray in the bottom right corner shows the date and time as 11-02-2025, 17:49.

