

Johann V. Calda

TB22

Activity 6 Screenshots

```
ItemManagementApplication.py X
C: > Users > User > Desktop > ItemManagementApplication.py > ...

1  class Item:
2      def __init__(self, item_id: int, name: str, description: str, price: float):
3          if not name.strip():
4              raise ValueError("Name cannot be empty.")
5          if price < 0:
6              raise ValueError("Price cannot be negative.")
7
8          self.id = item_id
9          self.name = name
10         self.description = description
11         self.price = price
12
13     def __str__(self):
14         return f"ID: {self.id}, Name: {self.name}, Description: {self.description}, Price: ${self.price:.2f}"
15
16
17 class ItemManager:
18     def __init__(self):
19         self.items = {}
20         self.next_id = 1
21
22     def create_item(self, name: str, description: str, price: float):
23         try:
24             item = Item(self.next_id, name, description, price)
25             self.items[self.next_id] = item
26             self.next_id += 1
27             print("Item added successfully.")
28         except ValueError as e:
29             print(f"Error: {e}")
30
31     def read_items(self):
32         if not self.items:
33             print("No items available.")
34         else:
35             for item in self.items.values():
36                 print(item)
37
38     def update_item(self, item_id: int, name: str, description: str, price: float):
39         if item_id in self.items:
```

```

38     def update_item(self, item_id: int, name: str = None, description: str = None, price: float = None):
39         if item_id not in self.items:
40             print("Item not found.")
41             return
42
43         item = self.items[item_id]
44         try:
45             if name:
46                 if not name.strip():
47                     raise ValueError("Name cannot be empty.")
48                 item.name = name
49             if description:
50                 item.description = description
51             if price is not None:
52                 if price < 0:
53                     raise ValueError("Price cannot be negative.")
54                 item.price = price
55             print("Item updated successfully.")
56         except ValueError as e:
57             print(f"Error: {e}")
58
59     def delete_item(self, item_id: int):
60         if item_id in self.items:
61             del self.items[item_id]
62             print("Item deleted successfully.")
63         else:
64             print("Item not found.")
65
66
67 def main():
68     manager = ItemManager()
69     while True:
70         print("\nItem Management System")
71         print("1. Create Item")
72         print("2. View Items")
73         print("3. Update Item")

```

```

74     print("4. Delete Item")
75     print("5. Exit")
76     choice = input("Choose an option: ")
77
78     if choice == "1":
79         name = input("Enter item name: ")
80         description = input("Enter item description: ")
81         try:
82             price = float(input("Enter item price: "))
83             manager.create_item(name, description, price)
84         except ValueError:
85             print("Invalid price. Please enter a number.")
86     elif choice == "2":
87         manager.read_items()
88     elif choice == "3":
89         try:
90             item_id = int(input("Enter item ID to update: "))
91             name = input("Enter new name (leave blank to keep current): ") or None
92             description = input("Enter new description (leave blank to keep current): ") or None
93             price_input = input("Enter new price (leave blank to keep current): ")
94             price = float(price_input) if price_input else None
95             manager.update_item(item_id, name, description, price)
96         except ValueError:
97             print("Invalid input.")
98     elif choice == "4":
99         try:
100             item_id = int(input("Enter item ID to delete: "))
101             manager.delete_item(item_id)
102         except ValueError:
103             print("Invalid ID.")
104     elif choice == "5":
105         print("Exiting...")
106         break
107     else:
108         print("Invalid choice, please try again.")
109

```

```

108         print("Invalid choice, please try again.")
109
110     if __name__ == "__main__":
111         main()
112

```

```
C:\Users\User\Desktop\Currency>C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/User/Desktop/ItemManagementApplication.py

Item Management System
1. Create Item
2. View Items
3. Update Item
4. Delete Item
5. Exit
Choose an option: 1
Enter item name: Good Sheet
Enter item description: This will bring you to Cloud Nine
Enter item price: 420
Item added successfully.

Item Management System
1. Create Item
2. View Items
3. Update Item
4. Delete Item
5. Exit
Choose an option: 2
ID: 1, Name: Good Sheet, Description: This will bring you to Cloud Nine, Price: $420.00

Item Management System
1. Create Item
2. View Items
3. Update Item
4. Delete Item
5. Exit
Choose an option: 3
Enter item ID to update: 1
Enter new name (leave blank to keep current): Snoop Dogg's Ice Cream
Enter new description (leave blank to keep current): Snoop Dogg's prototype of good sheet
```

```
Item Management System
1. Create Item
2. View Items
3. Update Item
4. Delete Item
5. Exit
Choose an option: 4
Enter item ID to delete: 1
Item deleted successfully.
```

```
Item Management System
1. Create Item
2. View Items
3. Update Item
4. Delete Item
5. Exit
Choose an option: █
```

Item Management System

1. Create Item

2. View Items

3. Update Item

4. Delete Item

5. Exit

Choose an option: 5

Exiting...

C:\Users\User\Desktop\Currency>