# supply\_chain\_management.py

class Product:

def \_\_init\_\_(self, product\_id, name, quantity, price):

self.product\_id = product\_id

self.name = name

self.quantity = quantity

self.price = price

def \_\_str\_\_(self):

return f"{self.product\_id} - {self.name} | Qty: {self.quantity} | ₹{self.price}"

class Supplier:

def \_\_init\_\_(self, supplier\_id, name, contact):

self.supplier\_id = supplier\_id

self.name = name

self.contact = contact

def \_\_str\_\_(self):

return f"{self.supplier\_id} - {self.name} | Contact: {self.contact}"

class Order:

def \_\_init\_\_(self, order\_id, product, supplier, quantity):

self.order\_id = order\_id

self.product = product

self.supplier = supplier

self.quantity = quantity

self.status = "Pending"

def update\_status(self, status):

self.status = status

def \_\_str\_\_(self):

return f"Order #{self.order\_id} | Product: {self.product.name} | Qty: {self.quantity} | Supplier: {self.supplier.name} | Status: {self.status}"

class SupplyChainManager:

def \_\_init\_\_(self):

self.products = {}

self.suppliers = {}

self.orders = {}

self.next\_product\_id = 1

self.next\_supplier\_id = 1

self.next\_order\_id = 1

def add\_product(self, name, quantity, price):

pid = self.next\_product\_id

self.products[pid] = Product(pid, name, quantity, price)

self.next\_product\_id += 1

def add\_supplier(self, name, contact):

sid = self.next\_supplier\_id

self.suppliers[sid] = Supplier(sid, name, contact)

self.next\_supplier\_id += 1

def place\_order(self, product\_id, supplier\_id, quantity):

if product\_id not in self.products or supplier\_id not in self.suppliers:

print("Invalid product or supplier ID.")

return

order = Order(self.next\_order\_id, self.products[product\_id], self.suppliers[supplier\_id], quantity)

self.orders[self.next\_order\_id] = order

self.next\_order\_id += 1

print("Order placed successfully.")

def update\_order\_status(self, order\_id, status):

if order\_id in self.orders:

self.orders[order\_id].update\_status(status)

print("Order status updated.")

else:

print("Invalid order ID.")

def display\_products(self):

for product in self.products.values():

print(product)

def display\_suppliers(self):

for supplier in self.suppliers.values():

print(supplier)

def display\_orders(self):

for order in self.orders.values():

print(order)

def main():

scm = SupplyChainManager()

while True:

print("\n--- Supply Chain Management ---")

print("1. Add Product")

print("2. Add Supplier")

print("3. Place Order")

print("4. Update Order Status")

print("5. Show Products")

print("6. Show Suppliers")

print("7. Show Orders")

print("8. Exit")

choice = input("Enter your choice: ")

if choice == '1':

name = input("Product name: ")

quantity = int(input("Quantity: "))

price = float(input("Price: "))

scm.add\_product(name, quantity, price)

elif choice == '2':

name = input("Supplier name: ")

contact = input("Contact info: ")

scm.add\_supplier(name, contact)

elif choice == '3':

scm.display\_products()

pid = int(input("Enter product ID: "))

scm.display\_suppliers()

sid = int(input("Enter supplier ID: "))

quantity = int(input("Order quantity: "))

scm.place\_order(pid, sid, quantity)

elif choice == '4':

scm.display\_orders()

oid = int(input("Enter order ID: "))

status = input("New status (Pending/Shipped/Delivered): ")

scm.update\_order\_status(oid, status)

elif choice == '5':

scm.display\_products()

elif choice == '6':

scm.display\_suppliers()

elif choice == '7':

scm.display\_orders()

elif choice == '8':

print("Exiting...")

break

else:

print("Invalid choice!")

if \_\_name\_\_ == "\_\_main\_\_":

main()