COLLEGE CODE: 3108

COLLEGE NAME: JEPPIAAR ENGINEERING COLLEGE

DEPARTMENT: INFORMATION TECHNOLOGY

STUDENT NM-ID: 1. KIRUTHIKA.V
F8E1ACF46AF3F43990A466B6E99363E7
2. DEEPIKHA.S
596DD063E8FE542A3F24D4956C43904B
3. D.S. DHARSHINI
7560A86070FD9C69A7742FC0FE6A7723

ROLL NO: 1. KIRUTHIKA.V – 23JEIT211
2. DEEPIKHA.S – 23JEIT124
3.D.S. DHARSHINI – 23JEIT145

DATE:16-05-2025

SUPPLY CHAIN MANAGEMENT – PROJECT NAME

SUBMITTED BY,

1. **V. KIRUTHIKA**
2. **S. DEEPIKHA**
3. **D.S. DHARSHINI**

Phase 5: Project Demonstration & Documentation

Title: Supply Chain Management

Abstract:

The Supply Chain Management project is designed to streamline and optimize the flow of goods, information, and finances across a distributed network of suppliers, manufacturers, and retailers. Utilizing advanced technologies such as artificial intelligence, IoT (Internet of Things), and ERP (Enterprise Resource Planning) integration, the system enhances visibility, accuracy, and responsiveness in supply chain operations. This final phase delivers a comprehensive overview of the system demonstration, technical documentation, performance metrics, source code, and testing reports. The solution ensures real-time tracking, automated decision-making, and secure data handling. Diagrams, system architecture snapshots, and workflow screenshots are provided to aid in understanding the end-to-end functionality.

Index

1. Project Demonstration

Overview:

The Supply Chain Management system will be demonstrated to stakeholders, showcasing its features such as inventory tracking, supplier integration, demand forecasting, and ERP connectivity.

Demonstration Details:

- **SYSTEM WALKTHROUGH**: A live tour of the platform from order generation to final delivery status updates.
- **REAL-TIME DATA VISIBILITY**: Display of real-time inventory levels, shipment tracking, and supplier inputs.
- **AI-DRIVEN FORECASTING**: Demonstration of AI models predicting demand based on historical data.
- **IOT INTEGRATION**: Sensors provide updates on warehouse conditions (temperature, stock levels).
- **PERFORMANCE METRICS**: System efficiency, load management, and accuracy of delivery timelines.
- **SECURITY & COMPLIANCE**: Data encryption and compliance with industry regulations will be shown.

Outcome:

The system's ability to improve logistics efficiency, reduce lead times, and ensure reliable data integration across supply chain nodes will be demonstrated effectively.

2. Project Documentation

Overview:

Complete documentation is provided to describe all system modules, technical choices, and usage instructions for stakeholders and administrators.

Documentation Sections:

- **SYSTEM ARCHITECTURE**: Diagrams showing connections among suppliers, warehouses, transport units, and ERP systems.
- **CODE DOCUMENTATION**: Annotated source code for data ingestion, analytics, and alert modules.
- **USER GUIDE**: Instructions for supply managers on tracking orders, managing suppliers, and generating reports.
- **ADMINISTRATOR GUIDE**: System maintenance, alerts configuration, and server-side monitoring.
- **TESTING REPORTS**: Evaluation results covering throughput, latency, and fault tolerance.

OUTCOME:

The documentation supports deployment, maintenance, and future scaling by providing a clear blueprint of the solution.

3.Feedback and Final Adjustments

Overview:

Constructive feedback from stakeholders and users will be used to finetune system functionality.

Steps:

- **FEEDBACK COLLECTION**: Through demo sessions, user interviews, and observation.
- **REFINEMENT**: Modifications based on feedback—e.g., dashboard UI, forecast adjustments.
- **FINAL TESTING**: Ensures reliability under peak load, inventory sync accuracy, and integration consistency.

Outcome:

Post-adjustment validation confirms the system's readiness for real world deployment across supply chains.

4.Final Project Report Submission

Overview:

A report summarizing all development phases, deliverables, and outcomes is compiled.

Report Sections:

- **EXECUTIVE SUMMARY**: High-level overview highlighting business impact and cost reduction potential.
- **PHASE BREAKDOWN**: Covers system design, module development, ERP interfacing, and risk management.
- **CHALLENGES & SOLUTIONS**: Discusses logistics delays, data integrity issues, and how they were mitigated.
- **OUTCOMES:** Documents readiness for deployment and measurable gains achieved.

Outcome:

A formal project report is submitted outlining the full lifecycle and project impact.

5.Project Handover and Future Works

Overview:

Official handover with a roadmap for scalability and enhancements.

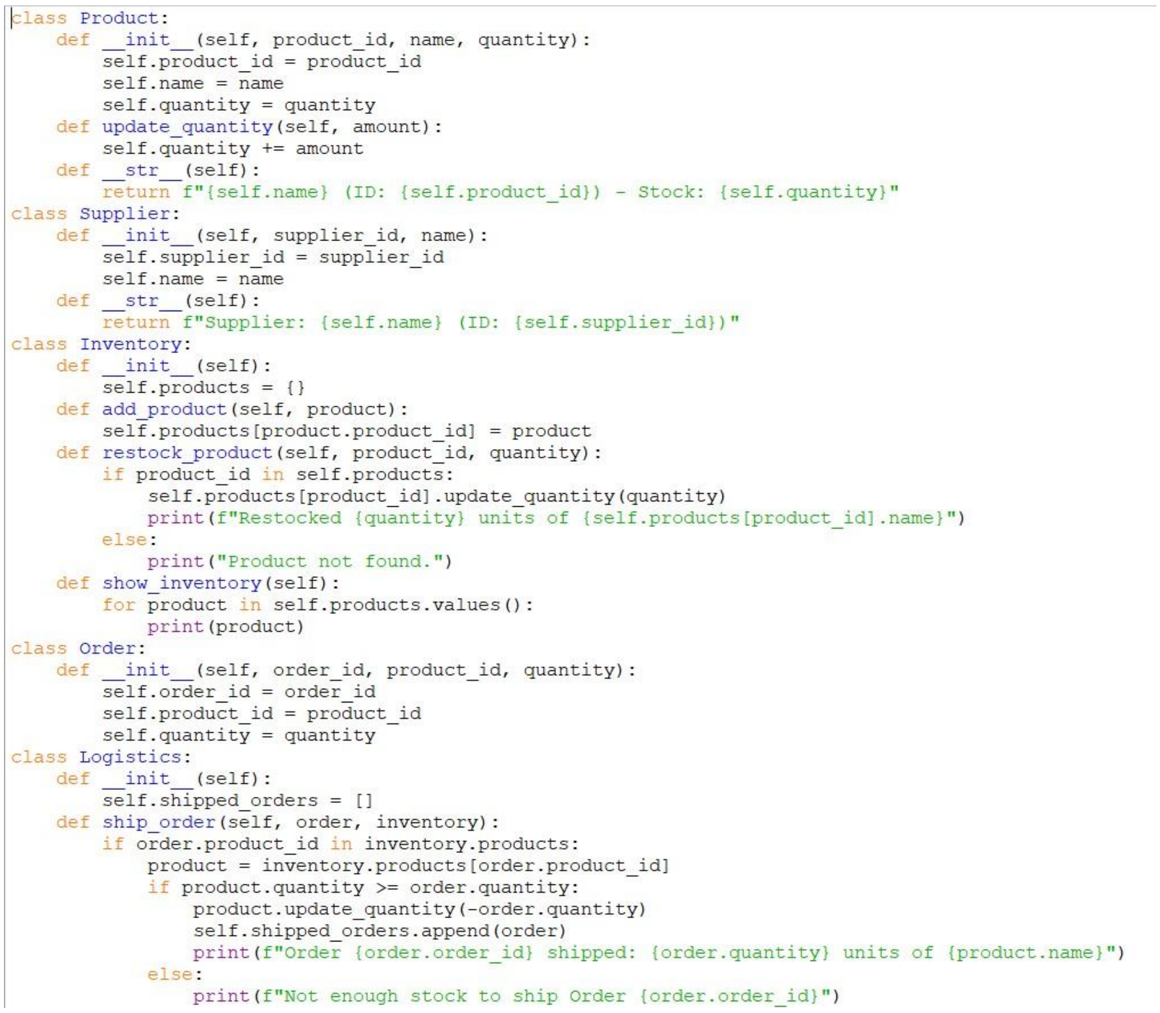
Handover Details:

• **NEXT STEPS**: Include extending to multi-location operations, AI model enhancements, and vendor onboarding automation.

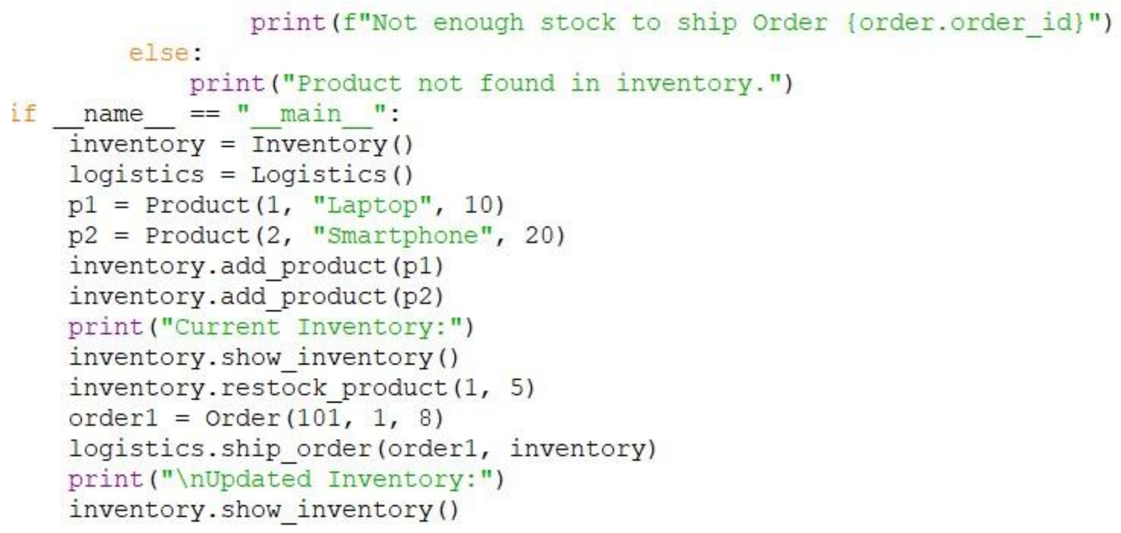
OUTCOME:

The system is handed over with technical documentation and future work suggestions to enhance supply chain agility.

Include Screenshots of source code and final working project.

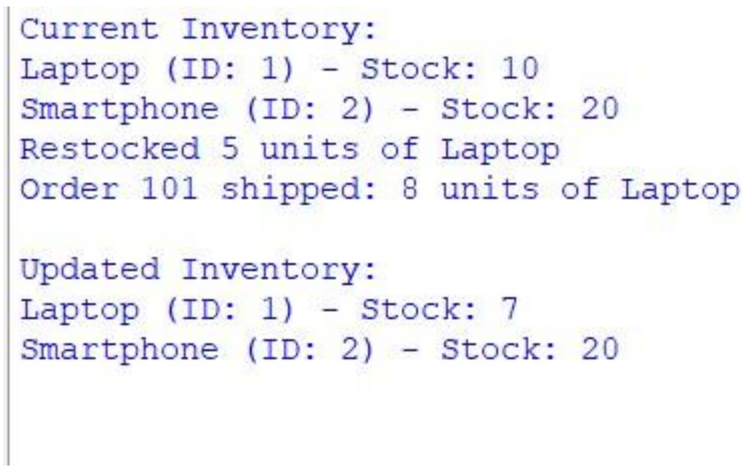


```python
class Product:
    def __init__(self, product_id, name, quantity):
        self.product_id = product_id
        self.name = name
        self.quantity = quantity
    def update_quantity(self, amount):
        self.quantity += amount
    def __str__(self):
        return f"{self.name} (ID: {self.product_id}) - Stock: {self.quantity}"
class Supplier:
    def __init__(self, supplier_id, name):
        self.supplier_id = supplier_id
        self.name = name
    def __str__(self):
        return f"Supplier: {self.name} (ID: {self.supplier_id})"
class Inventory:
    def __init__(self):
        self.products = {}
    def add_product(self, product):
        self.products[product.product_id] = product
    def restock_product(self, product_id, quantity):
        if product_id in self.products:
            self.products[product_id].update_quantity(quantity)
            print(f"Restocked {quantity} units of {self.products[product_id].name}")
        else:
            print("Product not found.")
    def show_inventory(self):
        for product in self.products.values():
            print(product)
class Order:
    def __init__(self, order_id, product_id, quantity):
        self.order_id = order_id
        self.product_id = product_id
        self.quantity = quantity
class Logistics:
    def __init__(self):
        self.shipped_orders = []
    def ship_order(self, order, inventory):
        if order.product_id in inventory.products:
            product = inventory.products[order.product_id]
            if product.quantity >= order.quantity:
                product.update_quantity(-order.quantity)
                self.shipped_orders.append(order)
                print(f"Order {order.order_id} shipped: {order.quantity} units of {product.name}")
            else:
                print(f"Not enough stock to ship Order {order.order_id}")
```



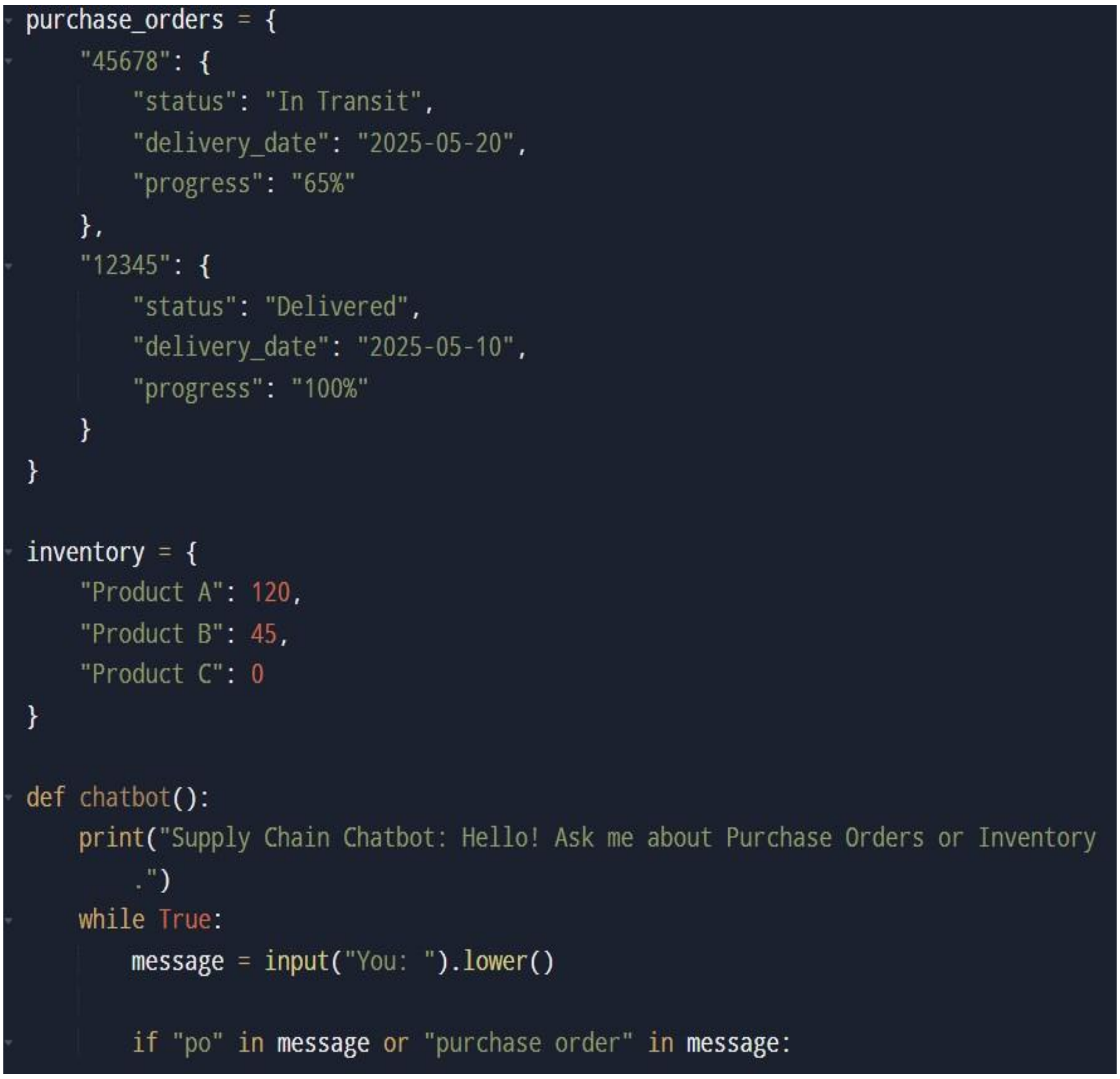
```
                print(f"Not enough stock to ship Order {order.order_id}")
        else:
            print("Product not found in inventory.")
if __name__ == "__main__":
    inventory = Inventory()
    logistics = Logistics()
    p1 = Product(1, "Laptop", 10)
    p2 = Product(2, "Smartphone", 20)
    inventory.add_product(p1)
    inventory.add_product(p2)
    print("Current Inventory:")
    inventory.show_inventory()
    inventory.restock_product(1, 5)
    order1 = Order(101, 1, 8)
    logistics.ship_order(order1, inventory)
    print("\nUpdated Inventory:")
    inventory.show_inventory()
```

OUTPUT:

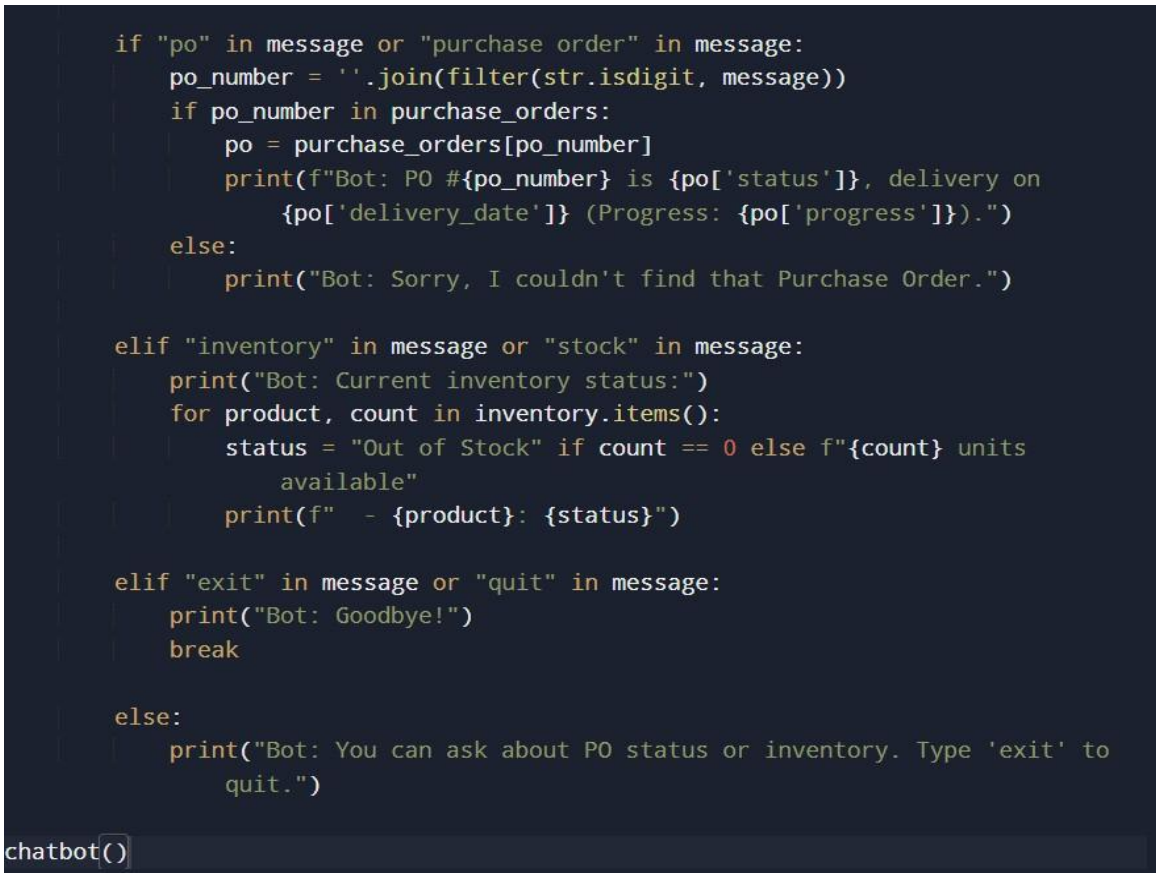


```
Current Inventory:
Laptop (ID: 1) - Stock: 10
Smartphone (ID: 2) - Stock: 20
Restocked 5 units of Laptop
Order 101 shipped: 8 units of Laptop

Updated Inventory:
Laptop (ID: 1) - Stock: 7
Smartphone (ID: 2) - Stock: 20
```

SOURCE CODE:

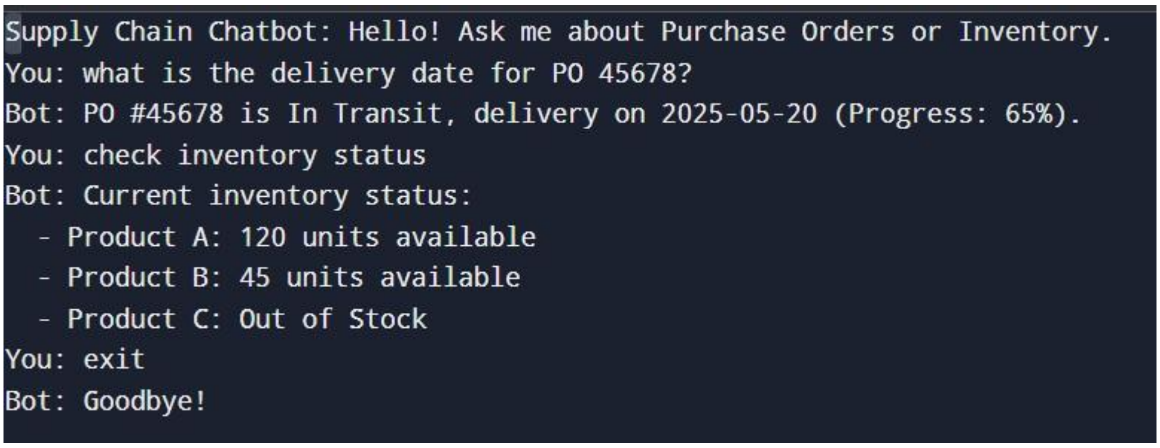


```python
purchase_orders = {
    "45678": {
        "status": "In Transit",
        "delivery_date": "2025-05-20",
        "progress": "65%"
    },
    "12345": {
        "status": "Delivered",
        "delivery_date": "2025-05-10",
        "progress": "100%"
    }
}

inventory = {
    "Product A": 120,
    "Product B": 45,
    "Product C": 0
}

def chatbot():
    print("Supply Chain Chatbot: Hello! Ask me about Purchase Orders or Inventory
        .")
    while True:
        message = input("You: ").lower()

        if "po" in message or "purchase order" in message:
```



```
        if "po" in message or "purchase order" in message:
            po_number = ''.join(filter(str.isdigit, message))
            if po_number in purchase_orders:
                po = purchase_orders[po_number]
                print(f"Bot: PO #{po_number} is {po['status']}, delivery on
                    {po['delivery_date']} (Progress: {po['progress']}).")
            else:
                print("Bot: Sorry, I couldn't find that Purchase Order.")

        elif "inventory" in message or "stock" in message:
            print("Bot: Current inventory status:")
            for product, count in inventory.items():
                status = "Out of Stock" if count == 0 else f"{count} units
                    available"
                print(f"  - {product}: {status}")

        elif "exit" in message or "quit" in message:
            print("Bot: Goodbye!")
            break

        else:
            print("Bot: You can ask about PO status or inventory. Type 'exit' to
                quit.")

chatbot()
```

OUTPUT: [CHATBOT RESPONSE]



```
Supply Chain Chatbot: Hello! Ask me about Purchase Orders or Inventory.
You: what is the delivery date for PO 45678?
Bot: PO #45678 is In Transit, delivery on 2025-05-20 (Progress: 65%).
You: check inventory status
Bot: Current inventory status:
  - Product A: 120 units available
  - Product B: 45 units available
  - Product C: Out of Stock
You: exit
Bot: Goodbye!
```