

# Inventory Management System

Documentation By Nabila Bannay Khan

## Overview

The **Inventory Management System** is a Python-based application that enables you to manage various product types (like **Electronics** and **Clothing**) using Object-Oriented Programming (OOP) principles.

This project is a great example of:

- Clean OOP architecture
- Realistic use of abstraction, inheritance, and polymorphism
- User interaction via CLI
- Robust error handling with custom exceptions
- JSON-based data persistence

## System Requirement

Python 3.6 or higher

No external libraries required (uses built-in modules like json).

► How to Run 1. 2. 3. app Clone or download the project folder. Open terminal/command prompt. Run the file using: app.py.

## Key Features

- ✓ Add and manage multiple product types (Electronics, Clothing)
- ✓ Sell products and auto-adjust stock
- ✓ View individual products or the full inventory
- ✓ Save and load inventory to/from a file
- ✓ Fully interactive Command Line Interface (CLI)
- ✓ Custom exceptions for safer, more controlled error management

## OOP Principles Applied

Principle	Description
Abstraction	Product is an abstract base class with common product behavior
Inheritance	Electronics and Clothing inherit from Product
Polymorphism	Methods like <code>display_info()</code> and <code>to_dict()</code> behave differently by class
Encapsulation	Attributes are private with controlled access via getters/setters

## Class Design

### Product (Abstract Class)

Base class for all products.

- Private fields: `product_id`, `name`, `price`, `quantity`
- Common methods: `get_id()`, `get_price()`, `update_quantity()`
- Abstract methods: `display_info()`, `to_dict()`, `from_dict(data)`

### Electronics (Inherits from Product)

Adds electronics-specific attributes.

- Additional fields: `brand`, `warranty`
- Overrides: `display_info()`, `to_dict()`, `from_dict()`

### Clothing (Inherits from Product)

Adds clothing-specific attributes.

- Additional fields: `size`, `material`
- Overrides: `display_info()`, `to_dict()`, `from_dict()`

## Inventory

Manages all product-related operations.

- Add, sell, update, view, list products

- Save/load inventory to/from .json file
- Handles product lookup and validation

## Custom Exceptions

Exception Name	Trigger Condition
DuplicateProductError	Adding a product with an ID that already exists
InsufficientStockError	Selling more quantity than is available in stock
InvalidProductDataError	Loading file with corrupt or missing product data

## CLI Menu System

===== Inventory Menu =====

1. Add Product
2. Sell Product
3. View Product
4. Save Inventory to File
5. Load Inventory from File
6. List All Products
7. Exit

Each option is interactive, validates user input, and provides meaningful feedback.

## File Operations

### Save Inventory

- Saves product data to a JSON file.
- Each product is converted using its `to_dict()` method.

### Load Inventory

- Loads product data from a JSON file.

- Automatically re-creates Electronics or Clothing objects.

Example JSON:

json

CopyEdit





```
[  
  
  {  
  
    "type": "Electronics",  
  
    "product_id": "E101",  
  
    "name": "Gaming Laptop",  
  
    "price": 1800,  
  
    "quantity": 5,  
  
    "brand": "Asus",  
  
    "warranty": 24  
  
  },  
  
  {  
  
    "type": "Clothing",  
  
    "product_id": "C202",  
  
    "name": "Denim Jacket",  
  
    "price": 60,  
  
    "quantity": 12,  
  
    "size": "L",  
  
    "material": "Denim"  
  
  }  
  
]
```



## Sample Use Cases

Scenario	Example Input
Add Electronics	Electronics, E001, Laptop, 1500, 10, Dell, 24
Add Clothing	Clothing, C001, T-Shirt, 20, 50, M, Cotton
Sell Product	Product ID: E001, Quantity: 2
Save Inventory to File	File name: inventory.json
Load Inventory from File	File name: inventory.json

## Validation & Edge Case Handling

-  Prevents adding products with duplicate IDs
-  Gracefully handles insufficient stock errors
-  Catches invalid/missing fields in product data when loading
-  Handles bad input (like string instead of integer for quantity)