System Requirements Specification Index

For

Inventory Management System Console Application

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

TABLE OF CONTENTS

1 PROJECT ABSTRACT

EZRetail Solutions, a retail consultancy based in Chicago, is developing a lightweight inventory management tool for small and mid-sized businesses. Many of their clients struggle with tracking inventory efficiently, leading to frequent stockouts and overstock issues. To address this, we are building a Python console application that processes inventory data using optimized loop implementations. The system will handle product filtering, searching, counting, and calculations through parallel lists, ensuring accurate and efficient inventory management. Designed for businesses with limited technical resources, this tool will help retailers streamline stock tracking, identify low-stock items, and generate key inventory insights without requiring complex software solutions.

2 BUSINESS REQUIREMENTS:

Screen Name	Console input screen
Problem Statement	1. The application must handle batch processing of inventory data efficiently. 2. Products below a specified stock threshold must be identified. 3. The system should compute total inventory valuation. 4. Users should be able to search for products using partial name matches. 5. The application must produce various inventory reports based on category, low stock, and valuation.

3 CONSTRAINTS

3.1 INPUT REQUIREMENTS

1. Inventory Records:
   - Stored as a list of dictionaries in inventory_records
   - Each record contains: `product_id`, `name`, `category`, `quantity`, `price`.
   - Example: [{"product_id": "P001", "name": "Rice 5kg", "category": "Grocery", "quantity": 45, "price": 250.00}]
2. Low Stock Threshold:
   - Must be stored as integer in variable low_stock
   - Must be between 1 and 100
   - Example: 10
3. Search Term:
   - Must be stored as string in variable search_term
   - Example: "Rice"
4. Report Type:
   - Must be stored as integer in variable report_type
   - 1: Low stock report
   - 2: Category summary
   - 3: Valuation report
   - 4: Reorder list
   - Example: 2

3.2 CALCULATION CONSTRAINTS

1. **Inventory Filtering (Low Stock):**
   - Must use a loop to iterate through inventory_records
   - Must identify items where quantity <= low_stock
   - Return list of low stock items
   - Example: All items with quantity <= 10
2. **Product Search:**
   - Must use a loop to iterate through inventory_records

- Must find items where search_term is in name (case-insensitive)
- Return list of matching items
- Example: search_term="rice", return all items with "rice" in name

3. Category Summary:
    - Must use a loop to iterate through inventory_records
    - Must group items by category and count them
    - Return dictionary with categories as keys and counts as values
    - Example: {"Grocery": 15, "Electronics": 8}
4. Inventory Valuation:
    - Must use a loop to iterate through inventory_records
    - Must calculate total value for each item (quantity * price)
    - Return total valuation
    - Example: Sum of all (quantity * price)
5. Reorder List:
    - Must use a loop with conditions to iterate through inventory_records
    - Skip items where quantity > low_stock
    - Calculate reorder amount (3 * low_stock - quantity)
    - Return reorder list with item name and reorder quantity
    - Example: [{"name": "Rice 5kg", "reorder": 25}]

3.3 Output Constraints

1. Display Format:
    - Show "Inventory Management System"
    - Show "Report Type: {report_name}"
    - Show report contents in tabular format
2. Low Stock Report:
    - Table columns: ID, Name, Quantity, Category
    - Show count of low stock items
    - Show "Critical" for items with quantity < low_stock/2
3. Category Summary:
    - Table columns: Category, Item Count, % of Inventory
    - Sort by Item Count (descending)
    - Show percentages with 1 decimal place

4. Valuation Report:
    - - Table columns: Category, Items Count, Total Value
    - - Show grand total at bottom
    - - Show currency symbol (₹) for values
5. Reorder List:
    - - Table columns: ID, Name, Current Stock, Reorder Quantity
    - - Sort by Reorder Quantity (descending)
    - - Show estimated cost (Reorder Quantity * price)

4. TEMPLATE CODE STRUCTURE:

1. Processing Functions:

    - find_low_stock_items(inventory_records, low_stock)
    - search_products(inventory_records, search_term)
    - summarize_categories(inventory_records)
    - calculate_inventory_value(inventory_records)
    - generate_reorder_list(inventory_records, low_stock)

2. Input Section:

    - Load inventory data
    - Get low stock threshold
    - Get search term (if needed)
    - Get report type

3. Processing Section:

    - Execute appropriate processing function
    - Format results for display

4. Output Section:
    - Display report header
    - Show tabular data
    - Display summary statistics

5. EXECUTION STEPS TO FOLLOW:

1. Run the program
2. Load or enter inventory data
3. Set low stock threshold
4. Select report type
5. Enter search term (if applicable)
6. View generated report