**TOPIC**

Inventory Management System of a Tech Shop called

Byte Bazaar.

**Agenda**

* Aim of the Project
* What is an Inventory Management System
* Key Features of an Inventory Management System
* How does an Inventory Management System Work
* Why should we use an Inventory Management System
* Importance of Inventory Management System
* How Does Inventory Management System Help us
* The Future of Inventory Management System
* Code of the Program
* Output
* Conclusion
* Bibliography

## Aim Of the Project

The aim of our project is to create a system which is useful for us in real life and not just theoretically. As programmers it is our duty to implement what we learn so that it helps us solve real world problems. By the use of Java and its values features like the concept of classes, objects, functions etc, we have created an inventory management system for a tech-based shop/organization.

## What is An Inventory Management System?

Spreadsheets, hand-counted stock levels and manual order placement have largely been replaced by advanced inventory tracking software. An inventory management system can simplify the process of ordering, storing and using inventory by automating end-to-end production, business management, demand forecasting and accounting.

## Key Features of an Inventory Management System

### Inventory tracking

Know exactly where inventory is across the supply chain.

### Order management

Customize pricing, send quotes, track orders and manage returns.

### Transfer management

Move product to where it's most valuable.

### Reporting and analytics

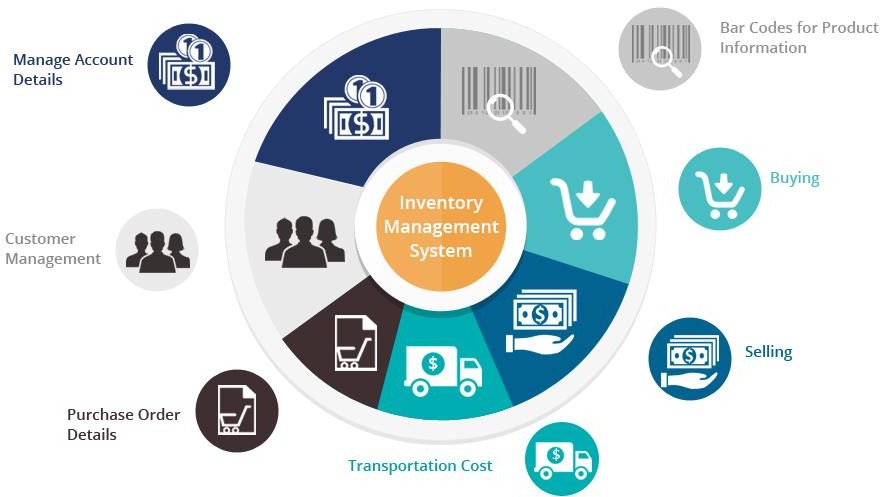
Evaluate patterns in processes to forecast future demand and sales.

### Purchasing

Create and manage purchase orders.

### Shipping capabilities

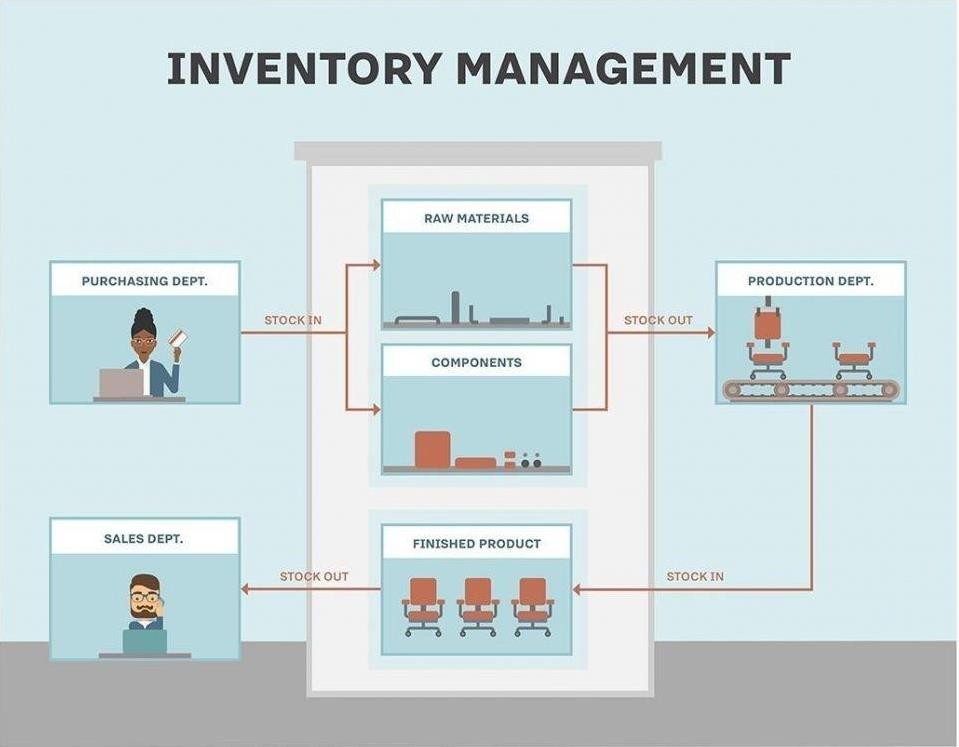
Automate shipping to reduce errors such as late deliveries or delivering incorrect packages.



## How Does an Inventory Management System Works

Inventory is the goods or materials a business intends to sell to customers for profit. Inventory management, a critical element of the supply chain, is the tracking of inventory from manufacturers to warehouses and from these facilities to a point of sale. The goal of inventory management is to have the right products in the right place at the right time. This requires inventory visibility — knowing when to order, how much to order and where to store stock. The basic steps of inventory management include:

1. Purchasing inventory: Ready-to-sell goods are purchased and delivered to the warehouse or directly to the point of sale.
2. Storing inventory: Inventory is stored until needed. Goods or materials are transferred across your fulfillment network until ready for shipment.
3. Profiting from inventory: The amount of product for sale is controlled. Finished goods are pulled to fulfill orders. Products are shipped to customers.



## Why is Inventory Management System Important

Inventory can be a company’s most important asset. Inventory management is where all the elements of the supply chain converge. Too little inventory when and where it's needed can create unhappy customers. But a large inventory has its own liabilities — the cost to store and insure it, and the risk of spoilage, theft and damage. Companies with complex supply chains and manufacturing processes must find the right balance between having too much inventory on hand or not enough.

## How Does an Inventory Management System Helps us

Both inventory management types have their benefits. However, as your business grows, going from a periodic inventory management strategy to a perpetual one will help reduce human error, optimize stock levels, and ultimately increase your bottom line.

## The Future of Inventory Management System

Globalization, technology and empowered consumers are changing the way businesses manage inventory. Supply chain operators will use technologies that provide significant insights into how supply chain performance can be improved. They’ll anticipate anomalies in logistics costs and performance before they occur and have insights into where automation can deliver significant scale advantages.

In the future, these technologies will continue to transform inventory management:

### Artificial intelligence

Intelligent, self-correcting AI will make inventory monitoring more accurate and reduce material waste.

### Internet of Things

Data from IoT sensors will provide insight into inventory location and status.

### Blockchain

Disparate parties will be connected through a unified and immutable record of all transactions.

### Intelligent order management

Supply chains will master inventory visibility with improved demand forecasting and automation.

### Quantum computing

Unprecedented computational power will solve previously unsolvable problems.

**Overview**

The ByteBazaar Inventory Management System offers a user-friendly interface for both sellers and customers to interact with inventory data, make purchases, and update product information. It includes features such as inventory tracking, order management, reporting, and more.

Libraries Used Java Swing Java Swing is a set of GUI (Graphical User Interface) components for Java programs, providing a rich set of tools for building interactive applications. In this project, Java Swing is used to create the graphical user interface for the inventory management system. Components such as buttons, panels, tabs, and tables are utilized to design the various screens and functionalities of the application.

javax.swing.JOptionPane The javax.swing.JOptionPane class is used to create pop-up dialog boxes for displaying messages, taking user input, and providing alerts. It is utilized in various parts of the application to prompt users for input, notify them of updates or errors, and display transaction information.

javax.swing.JTable and javax.swing.table.DefaultTableModel These classes are part of the Java Swing library and are used to create and manage tables in GUI applications. The javax.swing.JTable class represents a table component, while the javax.swing.table.DefaultTableModel class provides a default implementation of the TableModel interface. In this project, these classes are used to display product data in tabular format, allowing users to view and interact with inventory information.

How to Use

1. **Installation:** Clone the repository to your local machine.
2. **Compilation:** Compile the Java source files using a Java compiler.
3. **Execution:** Run the compiled Java program to launch the Inventory Management System.
4. **Usage:** Use the graphical user interface to interact with the system, perform tasks such as displaying products, updating inventory, making purchases, and viewing transactions.

## CODE FOR THE INVENTORY MANAGEMENT SYSTEM

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class InventoryManagementSystem extends JFrame {

    private static final long serialVersionUID = 1L;

    static int failure = 0;

    static int success = 0;

    static class Product {

        private int prod\_code;

        private String prod\_company;

        private String prod\_name;

        private double price;

        private String prod\_type;

        private int stock;

        Product(int u, String x, String y, String z, double w, int v) {

            prod\_code = u;

            prod\_name = x;

            prod\_company = y;

            prod\_type = z;

            price = w;

            stock = v;

        }

        Product(String x, String y) {

            prod\_company = x;

            prod\_type = y;

        }

        int search(Product x) {

            return prod\_company.equals(x.prod\_company) && prod\_type.equals(x.prod\_type) ? 1 : 0;

        }

        void noOfcopies(int required) {

            if (required > stock) {

                JOptionPane.showMessageDialog(null, "Required copies are not in stock");

                Trans(0);

            } else {

                JOptionPane.showMessageDialog(null, "Total cost of the product: " + required \* price +

                        "\nRemaining stock: " + (stock - required));

                stock = stock - required;

                Trans(1);

            }

        }

        void updatePrice(double newPrice) {

            price = newPrice;

        }

        void updateStock(int newStock) {

            stock = newStock;

        }

        Object[] getProductData() {

            return new Object[]{prod\_code, prod\_company, prod\_name, prod\_type, price, stock};

        }

    }

    static void Trans(int a) {

        if (a == 0)

            failure++;

        else

            success++;

    }

    static Product p1 = new Product(111, "Iphone14", "Apple", "Phone", 150000, 10);

    static Product p2 = new Product(112, "EOS90D", "Canon", "Camera", 120000, 15);

    static Product p3 = new Product(113, "Victus", "HP", "Laptop", 175000, 7);

    static Product p4 = new Product(114, "G102Prodigy", "Logitech", "Mouse", 2000, 30);

    static Product p5 = new Product(115, "Series8", "Apple", "SmartWatch", 85000, 18);

    private DefaultTableModel tableModel;

    public InventoryManagementSystem() {

        setTitle("Inventory Management System");

        setSize(800, 400);

        setDefaultCloseOperation(EXIT\_ON\_CLOSE);

        JTabbedPane tabbedPane = new JTabbedPane();

        getContentPane().add(tabbedPane, BorderLayout.CENTER);

        JPanel sellerPanel = createSellerPanel();

        JPanel customerPanel = createCustomerPanel();

        tabbedPane.addTab("Seller", sellerPanel);

        tabbedPane.addTab("Customer", customerPanel);

        JPanel productPanel = createProductPanel();

        tabbedPane.addTab("Products", productPanel);

    }

    private JPanel createSellerPanel() {

        JPanel sellerPanel = new JPanel();

        sellerPanel.setLayout(new GridLayout(4, 1));

        JButton displayProductsBtn = new JButton("Display all Products");

        displayProductsBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                JTabbedPane tabbedPane = (JTabbedPane) InventoryManagementSystem.this.getContentPane().getComponent(0);

                tabbedPane.setSelectedIndex(2); // Set index to the "Products" tab

            }

        });

        sellerPanel.add(displayProductsBtn);

        JButton updateProductsBtn = new JButton("Update");

        updateProductsBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                int productCode = Integer.parseInt(JOptionPane.showInputDialog("Enter the Product code which has to be updated: "));

                String choice = JOptionPane.showInputDialog("Select what to update\n1. Price\n2. Stock");

                if (choice.equals("1")) {

                    double newPrice = Double.parseDouble(JOptionPane.showInputDialog("Enter the new price: "));

                    switch (productCode) {

                        case 111:

                            p1.updatePrice(newPrice);

                            break;

                        case 112:

                            p2.updatePrice(newPrice);

                            break;

                        case 113:

                            p3.updatePrice(newPrice);

                            break;

                        case 114:

                            p4.updatePrice(newPrice);

                            break;

                        case 115:

                            p5.updatePrice(newPrice);

                            break;

                        default:

                            JOptionPane.showMessageDialog(null, "Invalid Product code");

                    }

                } else if (choice.equals("2")) {

                    int newStock = Integer.parseInt(JOptionPane.showInputDialog("Enter the new stock: "));

                    switch (productCode) {

                        case 111:

                            p1.updateStock(newStock);

                            break;

                        case 112:

                            p2.updateStock(newStock);

                            break;

                        case 113:

                            p3.updateStock(newStock);

                            break;

                        case 114:

                            p4.updateStock(newStock);

                            break;

                        case 115:

                            p5.updateStock(newStock);

                            break;

                        default:

                            JOptionPane.showMessageDialog(null, "Invalid Product code");

                    }

                } else {

                    JOptionPane.showMessageDialog(null, "Invalid choice");

                }

                updateProductTable();

            }

        });

        sellerPanel.add(updateProductsBtn);

        JButton viewTransactionsBtn = new JButton("View Transactions");

        viewTransactionsBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                JOptionPane.showMessageDialog(null, "Total failed transaction: " + failure + "\nTotal successful transaction: " + success);

            }

        });

        sellerPanel.add(viewTransactionsBtn);

        JButton exitSellerBtn = new JButton("Exit");

        exitSellerBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                dispose();

            }

        });

        sellerPanel.add(exitSellerBtn);

        return sellerPanel;

    }

    private JPanel createCustomerPanel() {

        JPanel customerPanel = new JPanel();

        customerPanel.setLayout(new GridLayout(4, 1));

        JButton displayProductsBtn = new JButton("Display all Products");

        displayProductsBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                JTabbedPane tabbedPane = (JTabbedPane) InventoryManagementSystem.this.getContentPane().getComponent(0);

                tabbedPane.setSelectedIndex(2); // Set index to the "Products" tab

            }

        });

        customerPanel.add(displayProductsBtn);

        JButton purchaseProductBtn = new JButton("Purchase a Product");

        purchaseProductBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                String prodCompany = JOptionPane.showInputDialog("Enter Product Company: ");

                String prodType = JOptionPane.showInputDialog("Enter Product Type: ");

                Product product = new Product(prodCompany, prodType);

                if (product.search(p1) == 1)

                    purchaseProduct(p1);

                else if (product.search(p2) == 1)

                    purchaseProduct(p2);

                else if (product.search(p3) == 1)

                    purchaseProduct(p3);

                else if (product.search(p4) == 1)

                    purchaseProduct(p4);

                else if (product.search(p5) == 1)

                    purchaseProduct(p5);

                else

                    JOptionPane.showMessageDialog(null, "This product is not available");

                updateProductTable();

            }

        });

        customerPanel.add(purchaseProductBtn);

        JButton exitCustomerBtn = new JButton("Exit");

        exitCustomerBtn.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                dispose();

            }

        });

        customerPanel.add(exitCustomerBtn);

        return customerPanel;

    }

    private JPanel createProductPanel() {

        JPanel productPanel = new JPanel();

        productPanel.setLayout(new BorderLayout());

        String[] columnNames = {"Product Code", "Company", "Name", "Type", "Price", "Stock"};

        Object[][] data = {

                {p1.prod\_code, p1.prod\_company, p1.prod\_name, p1.prod\_type, p1.price, p1.stock},

                {p2.prod\_code, p2.prod\_company, p2.prod\_name, p2.prod\_type, p2.price, p2.stock},

                {p3.prod\_code, p3.prod\_company, p3.prod\_name, p3.prod\_type, p3.price, p3.stock},

                {p4.prod\_code, p4.prod\_company, p4.prod\_name, p4.prod\_type, p4.price, p4.stock},

                {p5.prod\_code, p5.prod\_company, p5.prod\_name, p5.prod\_type, p5.price, p5.stock}

        };

        tableModel = new DefaultTableModel(data, columnNames);

        JTable table = new JTable(tableModel);

        JScrollPane scrollPane = new JScrollPane(table);

        productPanel.add(scrollPane, BorderLayout.CENTER);

        return productPanel;

    }

    private void purchaseProduct(Product product) {

        int required = Integer.parseInt(JOptionPane.showInputDialog("Enter required number of copies: "));

        product.noOfcopies(required);

    }

    private void updateProductTable() {

        tableModel.setValueAt(p1.stock, 0, 5);

        tableModel.setValueAt(p2.stock, 1, 5);

        tableModel.setValueAt(p3.stock, 2, 5);

        tableModel.setValueAt(p4.stock, 3, 5);

        tableModel.setValueAt(p5.stock, 4, 5);

    }

    public static void main(String[] args) {

        InventoryManagementSystem ims = new InventoryManagementSystem();

        ims.setVisible(true);

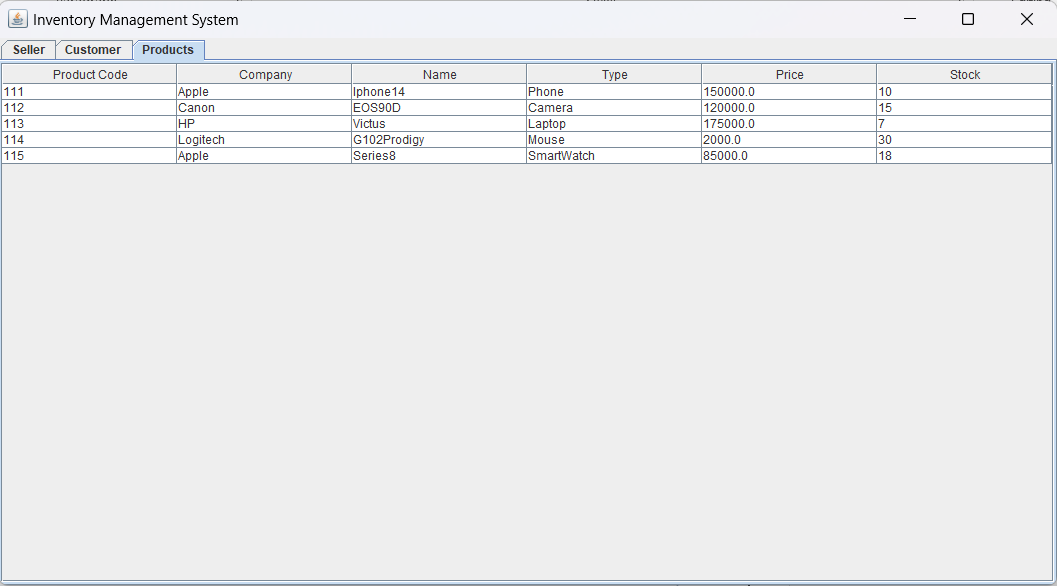
    }

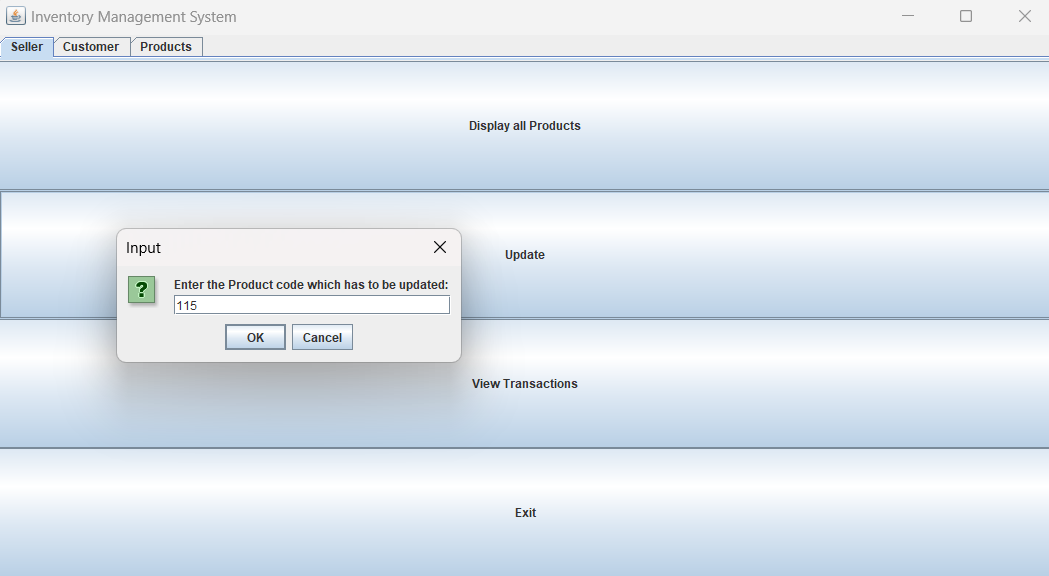
}

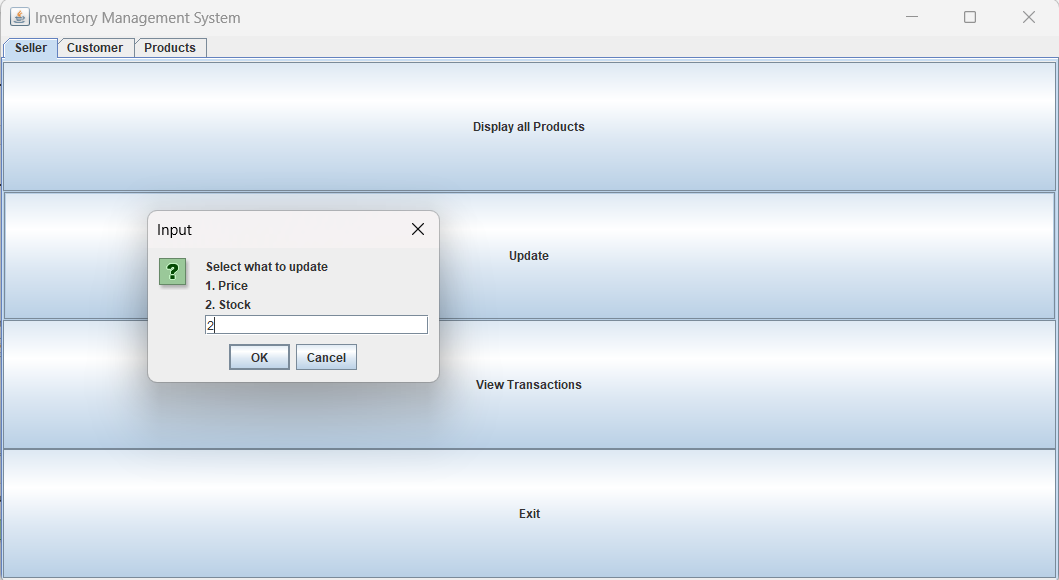
**OUTPUT:**

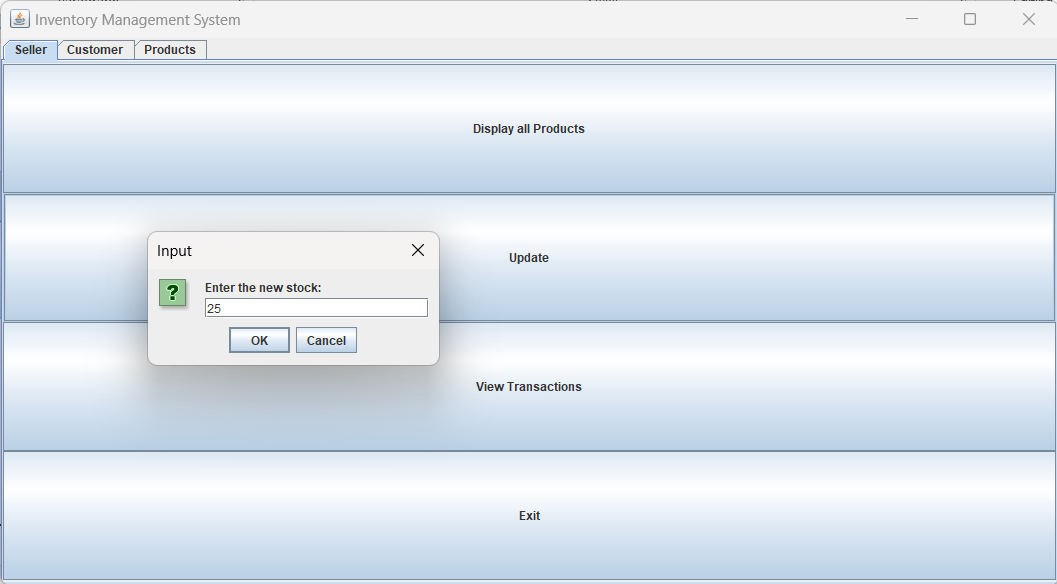


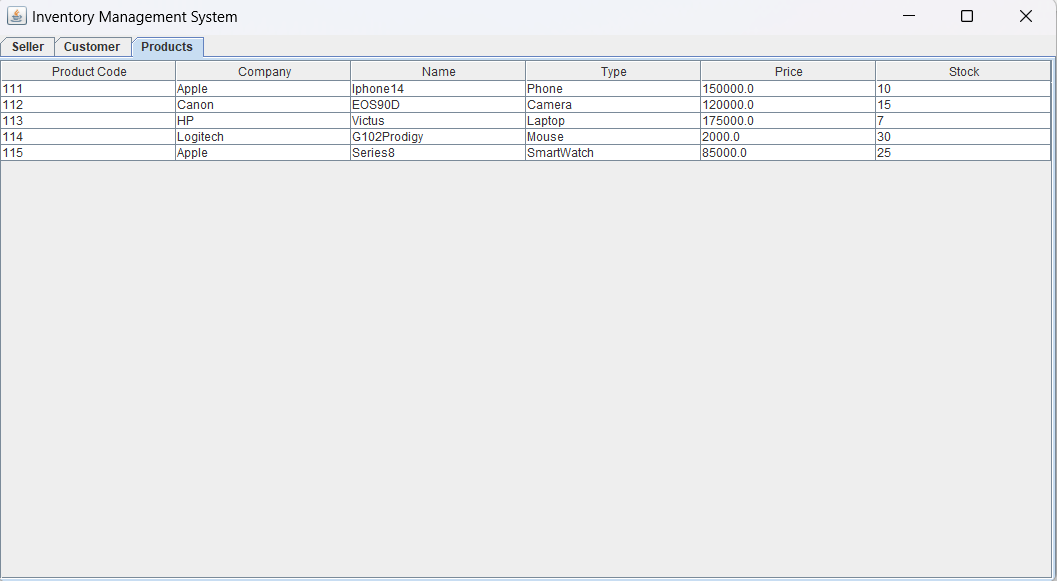












**Conclusion**

The conclusion of this project is that we should use new technologies into the real world as it is much better the old school practices that have been used from generation to generation in big companies, factories, organizations and firms as we know that programming and AI are the future of all the real-world solutions we should learn to use and adapt it.

## Bibliography

The reference links used are as follows:

[Message from FreshBooks](https://www.freshbooks.com/hub/productivity/inventory-management-system)

[What is inventory management and how does it work? | IBM](https://www.ibm.com/in-en/topics/inventory-management)

## PROJECT DONE BY:

1. Mahek Mushrif (A082).
2. Charles D’Souza (A063)

## GitHub Repository link:

<https://github.com/MahekMushrif-25/ByteBazaar.git>

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*