Inventory Management

Release Notes

Version 1.0

**Inventory Management:**

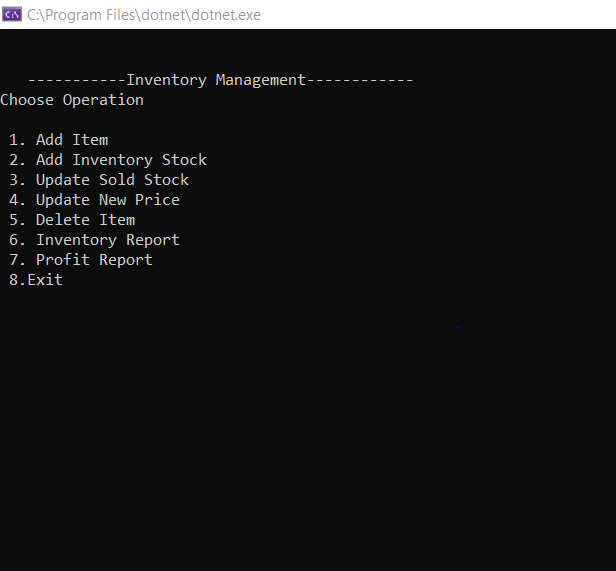
|  |  |  |
| --- | --- | --- |
| Date | Version | Author |
| 07/16/2019 | V1.0 | Haritha Manyala |
|  |  |  |

**Overview:**

The Inventory Management application is basically developed as a console application using Microsoft Technologies. This application allows the user to keep track of the products and current stock.

This application features below operations

1. Add Products and Prices.
2. Manage Product Purchases/Sales stock transactions
3. Update Selling prices
4. Delete Products
5. Reports
6. Inventory Report
7. Profit Report



**Technical Design :**

1. Application Implementation uses Command Design Pattern to achieve developing a robust application.

Command pattern is a data driven design pattern and falls under behavioural pattern category. A request is wrapped under an object as command and passed to invoker object. Invoker object looks for the appropriate encapsulated request object which can handle this command and passes the command to the corresponding object which executes the command.

The classes and objects of the Inventory management participating in this pattern are as below :

1. ICommand (ICommand.cs):

acts as an interface for executing an operation

1. ConcreteCommands (InventoryOperations.cs)

Defines a binding between a Receiver object and an action and implements Process() by invoking the corresponding operation(s) on Receiver

1. Client (Program.cs)

Creates a ConcreteCommand object and sets its receiver

1. Invoker (StockController.cs)

This class act as an User(Mobile Operator) who invokes the command to process the request

1. Receiver (Mobile.cs)

Upon receiving the requests performs the operations associated with carrying out the request.

This pattern helps in terms of extensibility as we can add new commands without changing the existing code adhering to Open and Close principal.

and further facilitates the development of highly cohesive modules with minimal coupling.

1. The results of the Inventory operations are simply stored in a JSON file with in the application acts as a back end.
2. Helper classes are created to perform basic IO operations like Reading File , Reading JSON file content.

**Environment Setup :**

The Console application is built on framework .NET Core 2.1.  
Make sure the JSON file path is configured in the appsettings.json file.

Requires Newtonsoft.JSON package for handling JSON data.

**Application Functionality:**

The User has to choose any operation out of 8 operations listed out on the Console. Based on the selected id of the operation the user can perform inventory action.

1. **Add Items :**

Upon choosing the Add Items operation user has to enter item name, Cost Price, Selling price and subsequently Items are added in the inventory.

1. **Add Inventory Stock :**

Upon choosing the operation user has to enter the existing item name and Purchased quantity and then the values are updated to the inventory.

1. **Update Sold Quantity :**

Upon choosing this operation the user has to input existing item name , quantity sold so far to maintain a record of sales transactions and inventory update.

1. **Update New Price:**

User can use this feature to update the current selling price to new value with a view to forecast the future sales.

1. **Delete Item:**

The user is chosen to delete the item permanently.

1. **Inventory Report:**

User can generate the report Available stock of all the items with current pricing values.

1. **Profit Report:**

User has a facility to pull the profit value report calculated based on the item previous Selling Price and current Selling Price.