****

Asia Pacific College

School of Computing and Information Technology

3 Humabon Place, Magallanes Vlg., Makati City

Inventory Management System

By:

Caguioa, Armersson

Gregorio, Felix

Romblon, Daveson

Mendoza, Karlo

BSIT-MI142

Prof. Manuel Sebastian Sanchez

CSPROJ Instructor

December 07, 2016

1. **Introduction**

**Project Context**

SMC is one of the top schools in the Philippines offering Basic Education and College. They are supervised by Southville International School and Colleges, a school with international character with international accreditation from the Western Association of Schools and Colleges.

Inventory management software is a computer-based system for tracking inventory levels, orders, sales and deliveries. Companies use inventory management software to avoid product overstock and outages. It is a tool for organizing inventory data that before was generally stored in hard-copy form or in spreadsheets.

**Purpose and Description**

The purpose of this project is to create a web-based inventory system that features:

1. Asset tracking: When a product is in the store, it can be tracked via item number.
2. Service management: To track sales and purchases. This way, they can see the in and out of items.
3. Inventory optimization: A fully automated demand forecasting and inventory optimization system to attain key inventory optimization metrics such as: Reorder point, Order quantity, Lead demand, Stock cover, and others.

**Objectives**

The objective of this project is to eliminate loads of manual work and errors that could be costly for the business and lightens the operational load by automating and streamlining processes.

**Scope and Limitation**

The scopes of the proposed system are sales, inventory, and reporting modules. The sales module covers the selling of items. Inventory module keeps track of the assets and purchases. Reporting module covers the summary reports on inventory, sales and purchase.

1. **Review of Related Literature/System**

Whether you are tracking inventory used to perform a service or sold to customers, using an inventory system provides staff accountability and minimizes inventory stockouts and shrinkage.

Clearly Inventory is an online inventory system that offers Low-Cost Inventory Tracking that features total control of user access, data management, data filters for easy search, and display inventory and sales reports.

TradeGecko manage all your sales, stock, accounting, shipping and customer data from a single place. Inventory control to improve tracking and control over inventory activities and stock movements. Inventory optimization to anticipate demand and receive reorder alerts in time so you'll never experience stockouts again.

Megaventory is web-based inventory, order management, CRM and reporting system for small to medium businesses with multiple locations. Add multiple users with variable permissions, track inventory levels, prepare sales and purchase orders, sales quotes, handle inbound and outbound goods and regain overall the control of your company.

Rapid Inventory, powered by Accucode, is a cloud- based inventory management system that seamlessly integrates with QuickBooks Pro, Premier and Enterprise. Rapid Inventory will make it easy to track your inventory across multiple warehouses and locations, lot numbers, expiration dates and hold reasons. With our QuickBooks inventory management software, you will process more transactions in less time and eliminate costly errors, improving your inventory accuracy and lowering your costs.

1. **Technical Background**

Web-based Server System Requirements

1. Minimum Hardware Requirements

Processor: Dual Core CPU 2.0GHz

Instruction Set: 64-bit

RAM: 4GB

HDD: 250GB

1. Operating System

Windows 7 SP1 (32-bit/64-bit); OR

Microsoft Windows Server 2008 SP2 (32-bit); OR

Microsoft Windows Server 2012 (64-bit)

1. Database Requirements

XAMPP Control Panel

1. Server

Apache

Notes:

The system must be accessible by desktop computers and laptops (connected through LAN) within the campus.

The system will utilize Apache as its Web-Based Server System.

The system must be able to retrieve, store and view data.

1. **Methodology, Results and Discussion**

**Context Diagram:**

Payment

Customer

Shop Clerk

AR number

Order Details

Payment

Purchase Details

Admin

Item Details

Management Report

Cashier

Proposed System

Order Information

**Data Flow Diagram:**

Management Reports

Purchase

Items

Item Information

Item Information

Customer

Order Info

Payment

Payment

Item list

Purchase Details

Item Updates

Items

Item Details

7

Manage Items

Inventory data

6

Generate Report

Quantity Added

Ordered items

Quantity released

4

Purchase stock

Admin

Order

Order list

Orders

Shop Clerk

Inventory

Items quantity

5

Update Inventory

Sales

Sales data

3

Update sales

Order Info

Receipt

AR number

Cashier

2

Process order

1

Order item

**Event Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Event | Trigger | Source | Use case | Response | Destination |
| Customer inquiries for the item | Item inquiry | Customer | Create order | Input customers order | Cashier |
| Customer pays for the ordered Item | Pay order | Customer | Get payment | Receive the payment of Customer | Cashier |
| Payment is cleared | Issue Receipt | Cashier | Get Receipt | Customer get the receipt | Customer |
| New order created | Store order | Cashier | View order list | View order details | Shop Clerk |
| Receive order Details | Release Item | Shop Clerk | Get item | Customer claims item | Customer |
| Admin manage item | Add, Edit, Remove items | Admin | Manage item | Update item details | System |
| Restock items | Purchase stock | Admin | Purchase Stock | Update item quantity | System |
| Shop Closing | Generate report | System | View Report | Display management reports | Admin |

**Use case:**

****

Admin

Shop Clerk

Cashier

Customer

**Use case full description: (order)**

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Item Inquiry | |
| Scenario: | Customer checks if order is available | |
| Triggering Event: | Customer ask for item details and availability | |
| Actors: | Customer, Cashier | |
| Related Use Cases: | Create Order | |
| Stakeholders: | Customer | |
| Preconditions: | Item details and availability must be given | |
| Postconditions: | Cashier informs the customer about the item status | |
| Flow of Activities: | Actor | System |
| 1. login  2. Customer orders item.  3. Cashier checks item status  4.Cashier informs the customer | 3.1 System will display item details |
| Exception Conditions: | Item inquired does not exist | |

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Create Order | |
| Scenario: | Cashier creates new order | |
| Triggering Event: | Item Inquiry | |
| Brief Description: | New order will be added to the order list | |
| Actors: | Customer, Cashier | |
| Related Use Cases: | Get payment | |
| Stakeholders: | Shop Clerk | |
| Preconditions: | Cashier would get the order information from the customer | |
| Postconditions: | New order is added to the order list | |
| Flow of Activities: | Actor | System |
| 1. login  2. Cashier inputs customer order. | 2.1 System stores order |
| Exception Conditions: | Customer cancels order  Incorrect data inputted | |

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Get payment | |
| Scenario: | Customer pay the total amount of his order | |
| Triggering Event: | Customer pays order | |
| Brief Description: | Cashier receives payment | |
| Actors: | Customer, Cashier | |
| Related Use Cases: | Issue Receipt | |
| Stakeholders: | Customer, Cashier | |
| Preconditions: | Cashier informs the customer the amount to be paid | |
| Postconditions: | Customer gives his payment | |
| Flow of Activities: | Actor | System |
| 2.1 Cashier informs the customer the amount to be paid | 1. Compute for the amount  2. Display amount |
| Exception Conditions: |  | |

**Use case full description: (Payment)**

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Issue Receipt | |
| Scenario: | Cashier gives the receipt | |
| Triggering Event: | Get payment | |
| Brief Description: | System would generate receipt | |
| Actors: | Customer, Cashier | |
| Related Use Cases: | Get receipt | |
| Stakeholders: | Customer, Cashier | |
| Preconditions: | Payment Successful | |
| Postconditions: | Prints Receipt | |
| Flow of Activities: | Actor | System |
| 1.1 Cashier prints receipt | 1. Generate Receipt |
| Exception Conditions: |  | |

**Use case full description: (Item Pickup)**

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Release item | |
| Scenario: | Give the item to the customer | |
| Triggering Event: | View order list | |
| Brief Description: | Shop clerk gives the item displayed on the order list | |
| Actors: | Customer, shop clerk | |
| Related Use Cases: | Get item | |
| Stakeholders: | Customer, Shop Clerk | |
| Preconditions: | Item released must match the order details | |
| Postconditions: | Shop clerk give the item to the customer | |
| Flow of Activities: | Actor | System |
| 1 login  2.1 Shop Clerk view orders  3. Shop clerk gives the item | 2. System display order list |
| Exception Conditions: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name: | View Order list | |
| Scenario: | Shop clerk checks for new order | |
| Triggering Event: | Create order | |
| Brief Description: | Order list displays item to be released | |
| Actors: | Customer, shop clerk | |
| Related Use Cases: | Release item | |
| Stakeholders: | Customer, Shop Clerk | |
| Preconditions: | Order must be inputted on the system | |
| Postconditions: | Displays order list | |
| Flow of Activities: | Actor | System |
| 1. login  2.1 Shop Clerk view orders | 2. System display order list |
| Exception Conditions: |  | |

**Use case full description: (Inventory)**

|  |  |  |
| --- | --- | --- |
| Use Case Name: | View reports | |
| Scenario: | Admin views the report before shop closes | |
| Triggering Event: | Shop closing | |
| Brief Description: | System generates report | |
| Actors: | Admin | |
| Related Use Cases: | Login | |
| Stakeholders: | Admin | |
| Preconditions: | Login as admin | |
| Postconditions: | System displays reports | |
| Flow of Activities: | Actor | System |
| 1. login  2. view reports | 2.1 Display reports |
| Exception Conditions: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Manage item | |
| Scenario: | Add, edit, remove item | |
| Triggering Event: | Item details needs to be updated | |
| Brief Description: | Admin manages the items details on the system | |
| Actors: | Admin | |
| Related Use Cases: | Login | |
| Stakeholders: | Customer, cashier, shop clerk | |
| Preconditions: | Login as admin | |
| Postconditions: | Updates item details | |
| Flow of Activities: | Actor | System |
| 1. login  2. manage items | 2.1 save changes |
| Exception Conditions: |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name: | Purchase stock | |
| Scenario: | Add item quantity | |
| Triggering Event: | Admin purchased item | |
| Brief Description: | Admin add item quantity | |
| Actors: | Admin | |
| Related Use Cases: | Login | |
| Stakeholders: | Customer, cashier, shop clerk | |
| Preconditions: | Login as admin | |
| Postconditions: | Updates item quantity | |
| Flow of Activities: | Actor | System |
| 1. login  2. add purchase details | 2.1 record details  3. updates inventory |
| Exception Conditions: |  | |

|  |
| --- |
| Inventory |
| Item  Quantity  Location |
| getItemDetails()  getUpdate() |

Class Diagram:

|  |
| --- |
| Cashier |
| Name Address  Views  number |

|  |
| --- |
| Admin |
| Name Address  number |

|  |
| --- |
| Item |
| Name Detail  1,\* in 1  Price |

Views

|  |
| --- |
| Order |
| Item  1,\* in 1  Quantity  Price |

|  |
| --- |
| Purchase |
| Item  Quantity  Price |

|  |
| --- |
| Sales |
| Order  Date |
| getOrderDetails() |

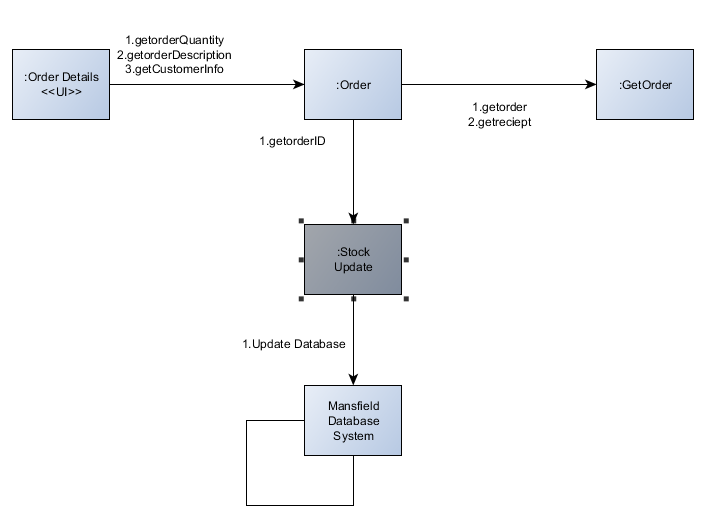
Input

Input

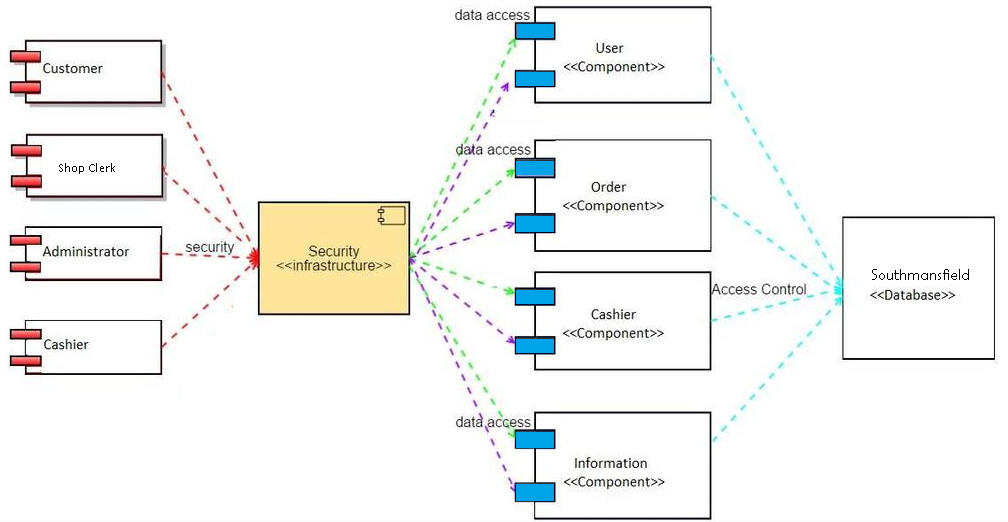
Manage

Updates

Communication Diagram:

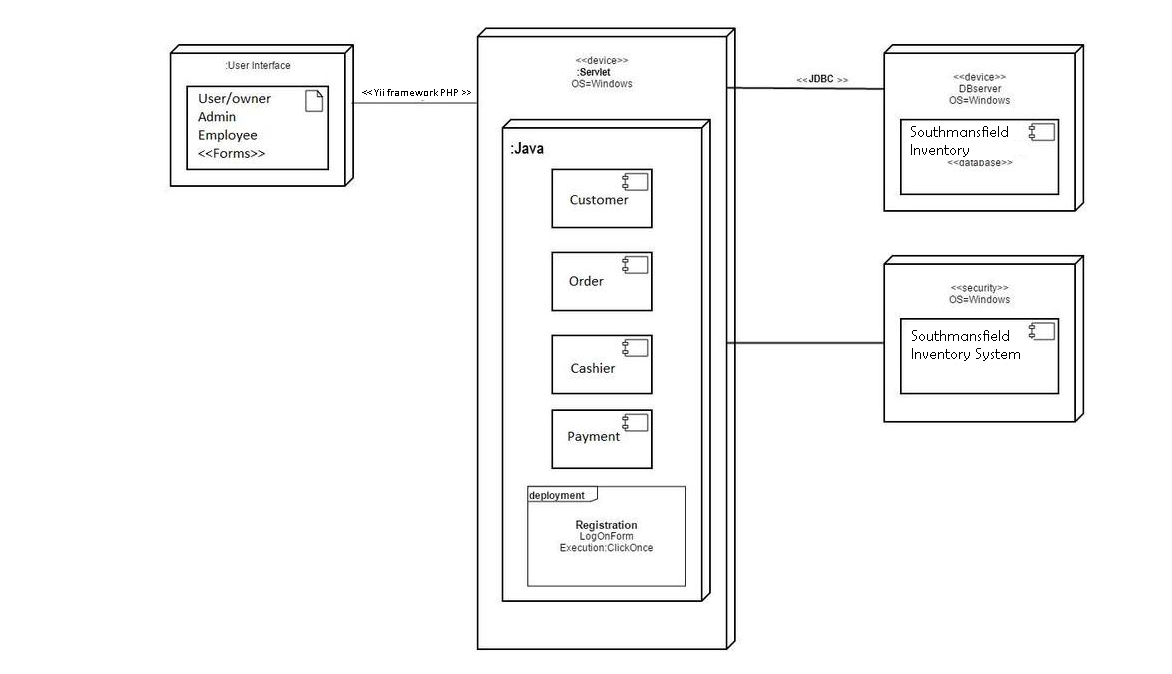


Component Diagram:

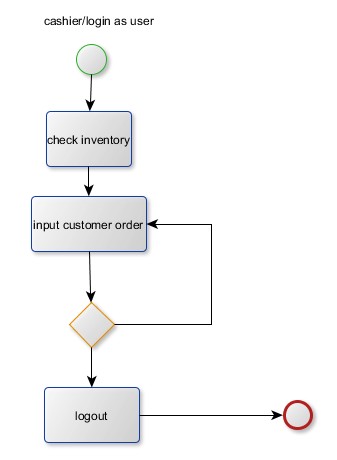
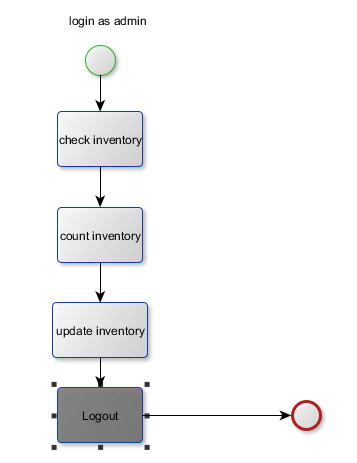


Composite Structure Diagram:

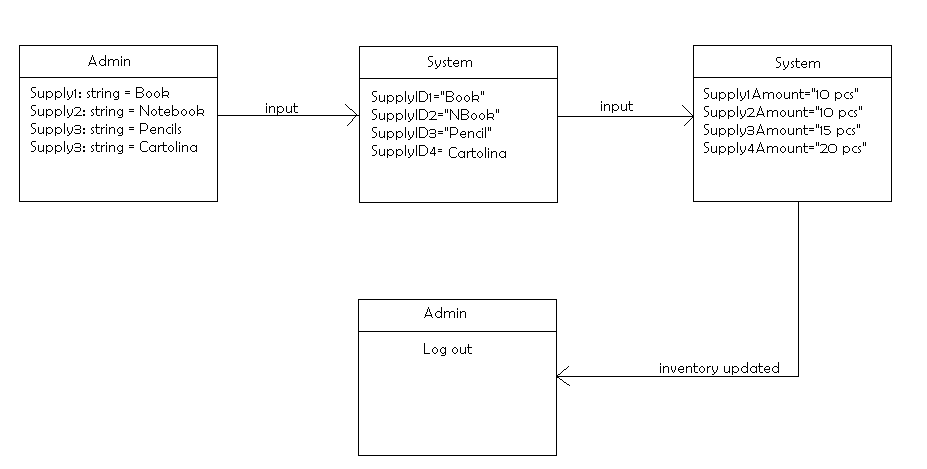
Deployment Diagram:

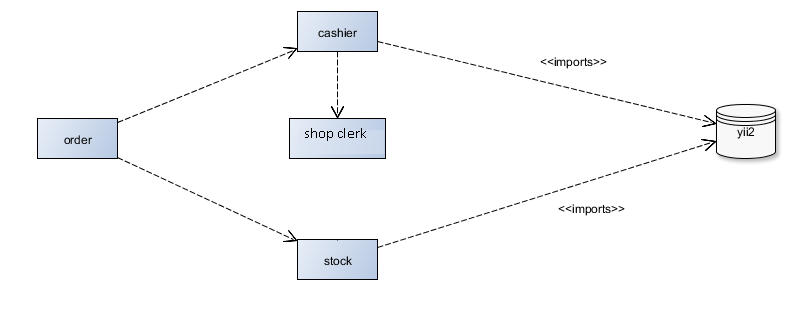


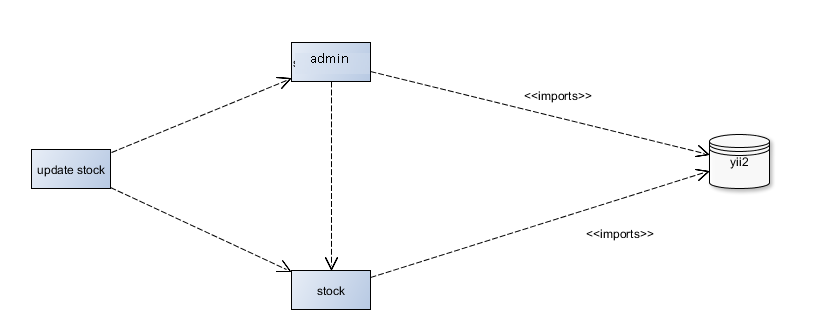
Interaction Overview Diagram:

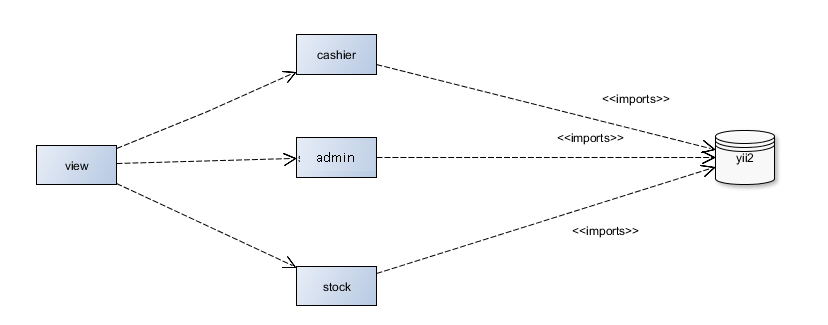


Object Diagram:

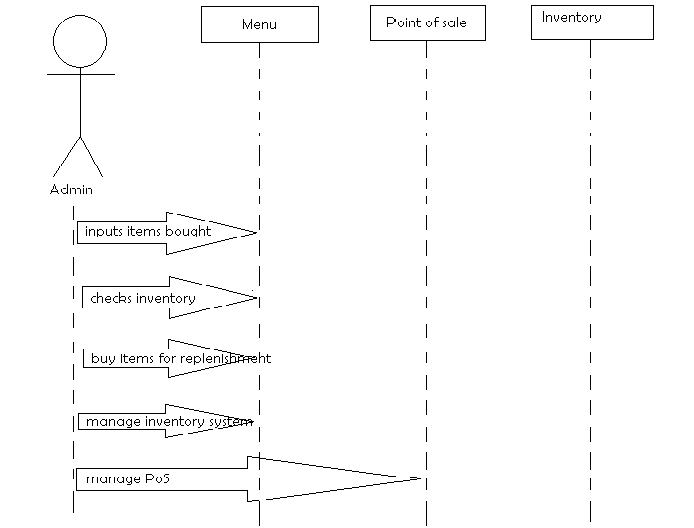


Package Diagram:

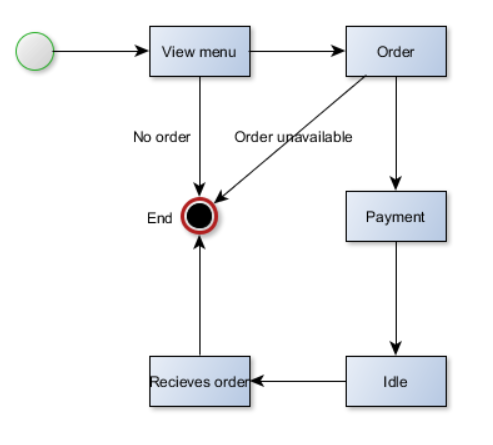
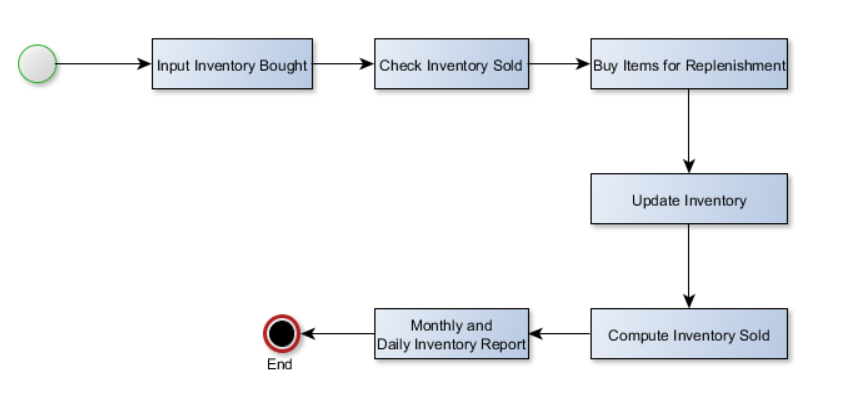




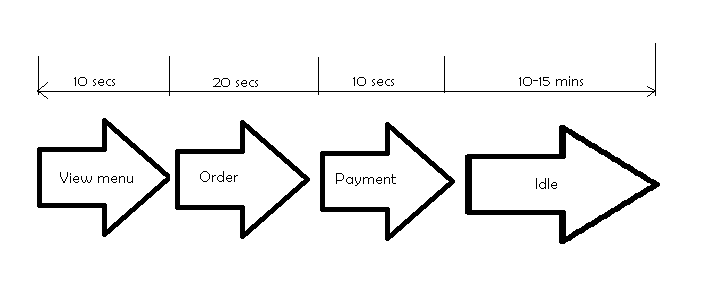
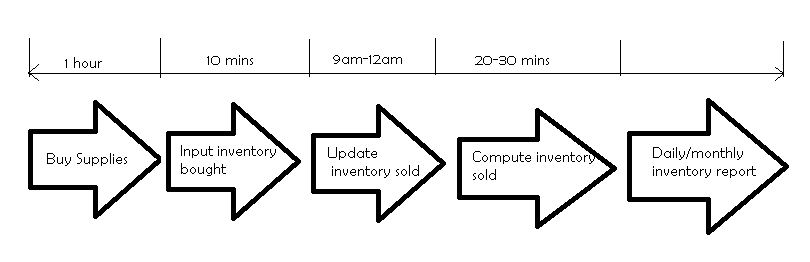
Sequence Diagram:



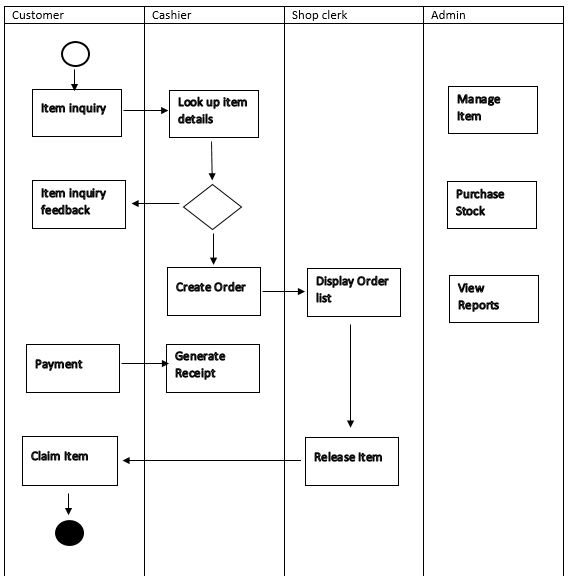
State Machine Diagram:



Timing Diagram:



**Activity Diagram:**



**ERD:**

