IMS (Inventory Management System)

Detailed Design

Version 0.5

**Revision History**

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 11/29/22 | 0.1 | Created UI designs (page mockups) for core pages, Created original draft outline for document | Priya Rivera |
| 11/29/22 | 0.1 | Architecture section 5.1-5.2.3 | Connor Svelling-Pescatore |
| 11/30/22 | 0.1 | Created initial document and added draft Class diagram | John Atkins |
| 12/02/22 | 0.1 | Added outline of Class Descriptions | John Atkins |
| 12/02/22 | 0.1 | Fix font, page breaks, rewrite intro | Larry Yang |
| 12/03/22 | 0.1 | Add class descriptions from 2.1 to 2.8 | Larry Yang |
| 12/04/22 | 0.2 | Highlighted sections to update | Connor Svelling-Pescatore |
| 12/04/22 | 0.3 | Added references, edit sections | Larry Yang |
| 12/04/22 | 0.4 | Added relationships to class diagrams, added CustomerUser and MerchantUser child classes. | Sean Kilroy |
| 12/05/22 | 0.4 | Updated all UI screens based on team feedback, Removed sections 4 - 6 from ToC; General review and clean up; Added UI descriptions | Priya Rivera |
| 12/05/22 | 0.4 | Removed architecture section, updated ToC to include MerchantUser and CustomerUser, completed class descriptions from 2.9-2.16 | Connor Svelling-Pescatore |
| 12/5/22 | 0.5 | Document review. Added figure labels. Suggestions for UI adjustments. | John Atkins |

**Table of Contents**

[**1. Introduction**](#_g9yuw4y45b6b) **6**

[**Related Documents**](#_b33fjudf6jz4) **6**

[**2. System Design**](#_lnrqfr1e8ykl) **6**

[**2.1 IMS Classes**](#_nmf14n) **6**

[2.2 InventoryManager Class](#_p3l5heg86k8a) 10

[2.2.1 Description](#_my6asy4xf4lh) 10

[2.2.2 Operations](#_2r0uhxc) 10

[**2.3 OrderManager**](#_1664s55) **10**

[2.3.1 Overview](#_3q5sasy) 10

[2.3.2 Description](#_i5jigifhoz4i) 11

[2.3.3 Operations](#_1x0gk37) 11

[**2.4 CustomerManager**](#_4h042r0) **11**

[2.4.1 Overview](#_2w5ecyt) 11

[2.4.2 Description](#_eyg2a4i0zjjc) 11

[2.4.3 Operations](#_1302m92) 12

[**2.5 Auth**](#_3mzq4wv) **12**

[2.5.1 Overview](#_7m80jhcbgikg) 12

[2.5.2 Description](#_2szc72q) 12

[2.5.3 Attributes](#_184mhaj) 12

[2.5.4 Operations](#_6vw6z5iho36q) 12

[**2.6 Report**](#_3s49zyc) **13**

[2.6.1 Overview](#_14qyib809ia8) 13

[2.6.2 Operations](#_xsm49n5yxua0) 13

[**2.7 InventoryItem**](#_1yyy98l) **13**

[2.7.1 Overview](#_kz9wlefl3z5g) 13

[2.7.2 Description](#_i6c2l1hyv6qk) 14

[2.7.3 Attributes](#_ruuc8jujjyxg) 14

[2.7.4 Operations](#_nhlzhj2bdt4z) 14

[**2.8 PurchaseOrder**](#_1ksv4uv) **14**

[2.8.1 Overview](#_1fiv12cbaes9) 14

[2.8.2 Description](#_4cmhg48) 15

[2.8.3 Attributes](#_9yoeccw4fxgc) 15

[2.8.4 Operations](#_16x20ju) 15

[**2.9 Customer**](#_3qwpj7n) **16**

[2.9.1 Overview](#_k5ge9qbmd3qx) 16

[2.9.2 Description](#_4n7gjtielgsa) 16

[2.9.3 Attributes](#_3w19e94) 16

[2.9.4 Operations](#_zaabi9vqzaub) 16

[**2.10 Address**](#_2b6jogx) **16**

[2.10.1 Overview](#_fqhb6f5i1k4k) 17

[2.10.2 Description](#_ui4mgs7gkq82) 17

[2.10.3 Attributes](#_1ci93xb) 17

[2.10.4 Operations](#_o53tifcwhva3) 17

[**2.11 Shipping**](#_4ekz59m) **17**

[2.11.1 Overview](#_w6nm64q0oxk3) 18

[2.11.2 Description](#_4t3kcl7agiu8) 18

[2.11.3 Attributes](#_1e03kqp) 18

[2.11.4 Operations](#_h777iqca14wm) 18

[**2.12 ShippingVendor**](#_3xzr3ei) **18**

[2.12.1 Overview](#_6zl06fjde42c) 19

[2.12.2 Description](#_m1nmzhz6wmm2) 19

[2.12.3 Attributes](#_33zd5kd) 19

[2.12.4 Operations](#_fygsise2wxub) 19

[**2.13 Invoice**](#_1j4nfs6) **19**

[2.13.1 Overview](#_r71qf26k19xs) 19

[2.13.2 Description](#_m4mdfprucpsu) 20

[2.13.3 Attributes](#_1o97atn) 20

[2.13.4 Operations](#_uc694bc95kro) 20

[**2.14 User**](#_2ne53p9) **20**

[2.14.1 Overview](#_89zx06rreniq) 20

[2.14.2 Description](#_u26sxykebwse) 21

[2.14.3 Attributes](#_4ddeoix) 21

[2.14.4 Operations](#_lwg89cild4gv) 21

[2.15 **MerchantUser**](#_hde8li6incue) 21

[2.15.1 Overview](#_mitolefw2kca) 21

[2.15.2 Description](#_q73gghsd6mhu) 21

[2.15.3 Attributes](#_wxtl37omhbrt) 22

[2.15.4 Operations](#_vxdmq1migh00) 22

[2.16 **CustomerUser**](#_j2qjirvtkvl8) 22

[2.16.1 Overview](#_vnv4e1cr4efz) 22

[2.16.2 Description](#_6ualro5veydh) 22

[2.16.3 Attributes](#_t71hpvdpxyu2) 22

[2.16.4 Operations](#_4wb5hcap7iky) 22

[**3. USER INTERFACE DESIGN**](#_1q7ozz1) **23**

# 

# Introduction

The Inventory Management System (IMS) is a business-to-business (B2B) cloud based software application solution that caters to a seller who acts as a buyer for clients.

The seller acquires inventory, negotiates sales with clients, generates purchase orders, takes payment, and ships. Currently the seller manages inventory in spreadsheets, which is cumbersome to use on mobile devices and makes it difficult to determine the current state of inventory. There will also be a read-only view for buyers to review order status. In a future release this view will be integrated with payment and shipping APIs.

The system will provide various interfaces to integrate with the seller’s and buyer’s workflow and processes. For small and medium businesses, this may include manual processes, forms, and spreadsheets.

# Related Documents

* [IMS Project Plan, Rev. 0.2](https://github.com/larryang/IS663-inventory-management-system/blob/main/documentation/IMS-Project-Plan.docx)
* [IMS Use-Case Specification, Rev. 0.3](https://github.com/larryang/IS663-inventory-management-system/blob/main/documentation/IMS-Use-Case-Specification.docx)
* [IMS Project Architecture Document, Rev. 1.0](https://github.com/larryang/IS663-inventory-management-system/blob/main/documentation/IMS_architecture_final.docx)

# System Design

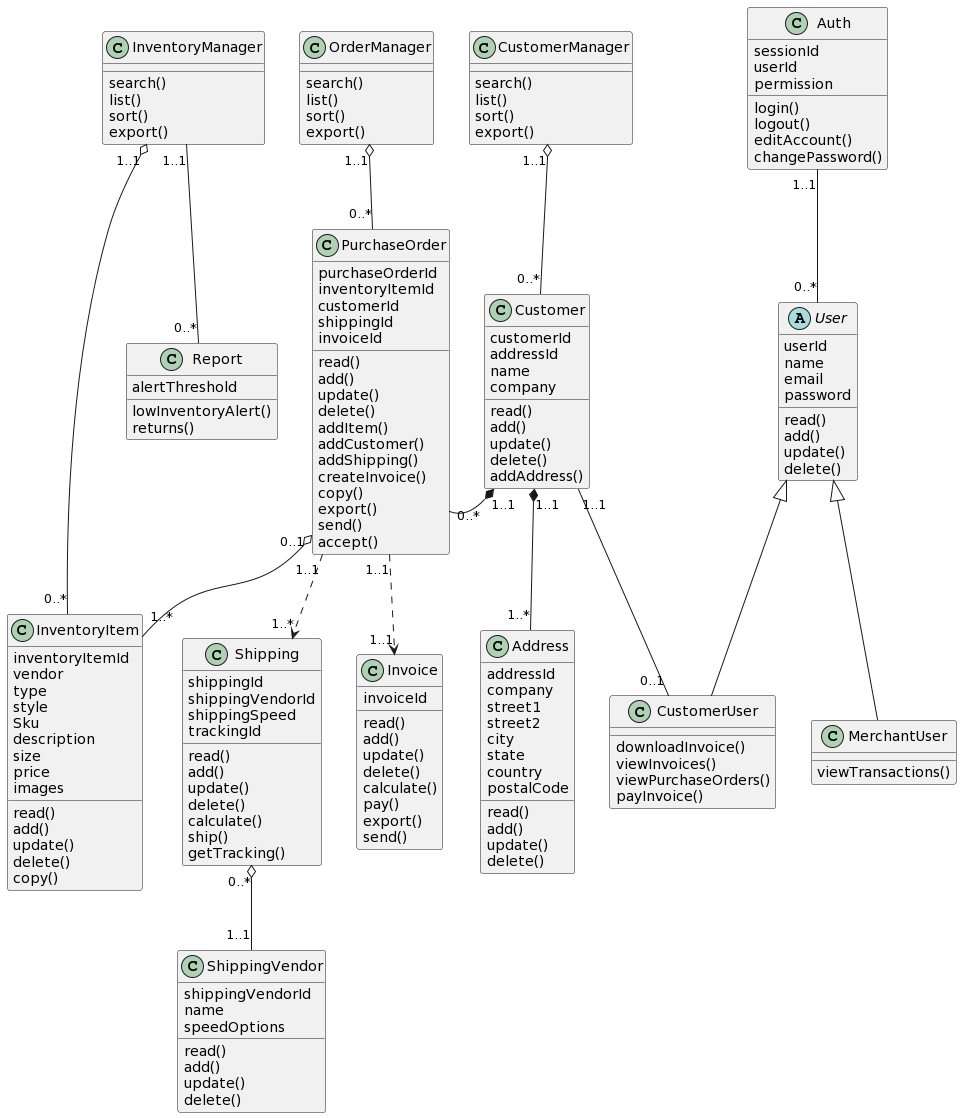
The system uses a Model View Controller (MVC) software architecture pattern for web based applications, and is implemented in an Object Oriented Programming language. This document provides the design of the classes and interfaces of the IMS system. It provides an overall class diagram, notated as a UML class diagram, and precisely describes how each class is related to the others.   
  
Further sections of the document provide an outline of each class. This includes a description, the purpose of each class, its attributes, and its operations/methods.

The latter sections of the document will provide information for the client graphic user interface. It has reference screenshots and descriptions of how the interface is to be used.

## IMS Classes

The following overview diagram, [Diagram 1. IMS Class Diagram](#qafah72eneiw), includes all classes for the IMS system. The attributes, operations, relations, and comparison within classes are shown.

The naming convention is Camel Case, with class names using upper or Pascal camel case, and attributes and methods using lower camel case.



**Diagram 1. IMS Class Diagram**

| Class | InventoryManager | CustomerManager | OrderManager |
| --- | --- | --- | --- |
| Attribute |  |  |  |
| Operation | search  list  sort  export | search  list  sort  export | search  list  sort  export |

| Class | InventoryItem | Customer | Address | PurchaseOrder |
| --- | --- | --- | --- | --- |
| Attribute | inventoryItemId  vendor  type  style  sku  description  size  price  images | customerId  addressId  name  company | addressId  company  street1  street2  city  state  country  postalCode | purchaseOrderId  inventoryItemId  customerId  shippingId  invoiceId |
| Operation | read  add  update  delete  copy | read  add  update  delete  addAddress | read  add  update  delete | read  add  update  delete  addItem  addCustomer  addShipping  createInvoice  copy  export  send  accept |

| Class | Shipping | ShippingVendor | Invoice |
| --- | --- | --- | --- |
| Attribute | shippingId  shippingVendorId  shippingSpeed  trackingId | shippingVendorId  name  speedOptions | invoiceId |
| Operation | read  add  update  delete  calculate  ship  getTracking | read  add  update  delete | read  add  update  delete  calculate  pay  export  send |

| Class | Auth | User | Report |
| --- | --- | --- | --- |
| Attribute | sessionId  userId  permission | userId  name  email  password | alertThreshold |
| Operation | login  logout  editAccount  changePassword | read  add  update  delete | lowInventoryAlert  returns |

| Class | MerchantUser | CustomerUser |
| --- | --- | --- |
| Attribute |  |  |
| Operation | viewTransactions | downloadInvoices  viewInvoices  viewPurchaseOrders  payInvoice |

## InventoryManager Class

The InventoryManager class in the IMS encapsulates the methods used to retrieve inventory information.

Overview

| **InventoryManager** | |
| --- | --- |
| **Attributes** | **Methods** |
| none | search |
|  | list |
|  | sort |
|  | export |

### Description

From the InventoryManager class, access the inventory information. It has zero or more instances of **InventoryItem**.

### Operations

| Method | Description |
| --- | --- |
| search | List inventory items whose attributes match search criteria. |
| list | List all inventory items. |
| sort | List all inventory items, sorted by input criteria. |
| export | Export list of inventory items. |

## OrderManager

The OrderManager class in the IMS encapsulates the methods used to retrieve order information.

### Overview

| **OrderManager** | |
| --- | --- |
| **Attributes** | **Methods** |
| none | search |
|  | list |
|  | sort |
|  | export |

### Description

From the OrderManager class, access order information. It contains zero or more instances of **PurchaseOrder**.

### Operations

| Method | Description |
| --- | --- |
| search | List purchase orders whose attributes match search criteria. |
| list | List all purchase orders. |
| sort | List all purchase orders, sorted by input criteria. |
| export | Export list of purchase orders. |

## CustomerManager

The CustomerManager class in the IMS encapsulates the methods used to retrieve customer information.

### Overview

| **CustomerManager** | |
| --- | --- |
| **Attributes** | **Methods** |
| none | search |
|  | list |
|  | sort |
|  | export |

### Description

From the CustomerManager class, access the customer information. It has zero or more instances of the **Customer** class.

### Operations

| Method | Description |
| --- | --- |
| search | List customers whose attributes match search criteria. |
| list | List all customers |
| sort | List all customers, sorted by input criteria. |
| export | Export list of customers. |

## Auth

Auth class in the IMS manages user sessions.

### Overview

| **Auth** | |
| --- | --- |
| **Attributes** | **Methods** |
| sessionId | login |
| userId | logout |
| permission | editAccount |
|  | changePassword |

### Description

Auth class manages user sessions, and provides user authentication, authorization, and accounting. It manages user account information. It contains one or more instances of **User**.

### Attributes

Whenever a user connects, it is associated with a **userId** and with a unique session identification, **sessionId**. The system then can know what **permission** to assign.

### Operations

| Method | Description |
| --- | --- |
| login | Create a session. |
| logout | End a session. |
| editAccount | Update **User** instance information. |
| changePassword | Update **User** instance password. |

## Report

The Report class generates alerts and reports.

### Overview

| **Report** | |
| --- | --- |
| **Attributes** | **Methods** |
| alertThreshold | lowInventoryAlert |
|  | returns |

### Operations

| Method | Description |
| --- | --- |
| lowInventoryAlert | Generate alerts upon low inventory criteria. |
| returns | Generate a report for returns. |

## InventoryItem

InventoryItem in the IMS represents the physical object in inventory.

### Overview

| **InventoryItem** | |
| --- | --- |
| **Attributes** | **Methods** |
| inventoryItemId | read |
| vendor | add |
| type | update |
| style | delete |
| SKU | copy |
| description |  |
| size |  |
| price |  |
| images |  |

### Description

The InventoryItem class is an abstract representation of the physical item.

Zero or more instances of **InventoryItem** are associated with an instance **InventoryManager**, and one or more are associated with an instance of **PurchaseManager.**

### Attributes

Each instance of InventoryItem has a unique **inventoryItemId**. Associated with the item is the **vendor, type, style, SKU,** a text **description**, **size, price**, and **images.**

### Operations

| Method | Description |
| --- | --- |
| read | Read inventory attributes. |
| add | Add inventory item. |
| update | Update inventory item attributes. |
| delete | Delete inventory item. |
| copy | Copy inventory item attributes, and change inventoryItemId. |

## PurchaseOrder

The PurchaseOrder in the IMS represents a purchase order.

### Overview

| **PurchaseOrder** | |
| --- | --- |
| **Attributes** | **Methods** |
| purchaseOrderID | read |
| inventoryItemId | add |
| customerID | update |
| shippingID | delete |
| invoiceID | addItem |
|  | addCustomer |
|  | addShipping |
|  | createInvoice |
|  | copy |
|  | export |
|  | send |
|  | accept |

### Description

The PurchaseOrder class abstracts the information in a PurchaseOrder. Zero or more instances of **PurchaseOrder** are associated with **OrderManager**. A PurchaseOrder is associated with an instance of **InventoryItem, Customer**, **Shipping**, and **Invoice**.

### Attributes

An instance of PurchaseOrder is uniquely identified by its **purchaseOrderId**. It is associated with instances of **InventoryItem, Customer**, **Shipping**, and **Invoice** by **inventoryItemId, customerId, ShippingId,** and **invoiceId.**

### Operations

| Method | Description |
| --- | --- |
| read | Read attributes. |
| add | Add instance of PurchaseOrder. |
| update | Update attributes. |
| delete | Delete instance of PurchaseOrder. |
| addItem | Associate InventoryItem with PurchaseOrder. |
| addCustomer | Associate Customer with PurchaseOrder. |
| addShipping | Associate Shipping with PurchaseOrder. |
| createInvoice | Create instance of Invoice associated with Purchase Order. |
| copy | Copy PurchaseOrder attributes and create new PurchaseOrder with different purchaseOrderId. |
| export | Export PurchaseOrder attributes into an external format. |
| send | Send purchase order from buyer. |
| accept | Accept purchase order from buyer. |

## Customer

### Overview

| **Customer** | |
| --- | --- |
| **Attributes** | **Methods** |
| customerID | read |
| addressID | add |
| name | update |
| company | delete |
|  | addAddress |

### Description

The Customer class abstracts the information for a Customer. Zero or more instances of **Customer** are associated with **CustomerManager**. A Customer is associated with an instance of **CustomerUser, Address**, and **PurchaseOrder**. Customer stores very basic attributes of the customer and is primarily used to link a purchase order to their address and a user account so the customer can view their purchase status.

### Attributes

An instance of Customer is uniquely identified by its **customerID**. It is associated with instances of **Address** by addressID.

### Operations

| Method | Description |
| --- | --- |
| read | Retrieve information about a specific customer. |
| add | Create a new instance of the Customer class. |
| update | Modify the information about an existing customer. |
| delete | Remove a customer from the list of registered customers. |
| addAddress | Add a new shipping address to the customer's account. |

## Address

### Overview

| **Address** | |
| --- | --- |
| **Attributes** | **Methods** |
| addressID | read |
| company | add |
| street1 | update |
| street2 | delete |
| city |  |
| state |  |
| country |  |
| postalCode |  |

### Description

The Address class abstracts the information for an Address. One or more instances of **Address** are associated with a **Customer**. This will store information about a Customer’s address(es) to be used in the Shipping stage of purchasing.

### Attributes

An instance of Address is uniquely identified by its **addressID**. It is associated with instances of Customerby addressID.

### Operations

| Method | Description |
| --- | --- |
| read | Retrieve information about a specific address. |
| add | Create a new instance of the address class. |
| update | Modify the information about an existing address. |
| delete | Remove a shipping vendor from the list of available vendors. |

## Shipping

### Overview

| **Shipping** | |
| --- | --- |
| **Attributes** | **Methods** |
| shippingID | read |
| shippingVendorId | add |
| shippingSpeed | update |
| trackingID | delete |
|  | calculate |
|  | ship |
|  | getTracking |

### Description

The Shipping class abstracts the information for Shipping. One instance of **Shipping** is associated with a **PurchaseOrder**. A Shipping is associated with an instance of **Customer**. Shipping stores the information relating to the details of the method, vendor, and tracking for items purchased by the customer.

### Attributes

An instance of Shipping is uniquely identified by its **shippingID**. It is associated with instances of PurchaseOrderby shippingID and ShippingVendor by shippingVendorID.

### Operations

| Method | Description |
| --- | --- |
| read | Retrieve information about a specific shipping vendor. |
| add | Create a new instance of the ShippingVendor class. |
| update | Modify the information about an existing shipping vendor. |
| delete | Remove a shipping vendor from the list of available vendors. |
| calculate | Determine the cost of shipping for a specific order. |
| ship | Process the shipping for an order, updating the order's status and generating any necessary shipping labels or tracking information. |
| getTracking | Retrieve the tracking information for an order, if available, to allow the customer to track the progress of their shipment. |

## ShippingVendor

### Overview

| **ShippingVendor** | |
| --- | --- |
| **Attributes** | **Methods** |
| shippingVendorId | read |
| name | add |
| speedOptions | update |
|  | delete |

### Description

The Shipping class abstracts the information for a ShippingVendor. One instance of **ShippingVendor** is associated with an instance of **Shipping**. This class stores the details necessary to display the shipping vendor’s name to the user and the shipping options available to the user during the ordering phase.

### Attributes

An instance of ShippingVendor is uniquely identified by its **shippingVendorID**. It is associated with instances of Shippingby shippingVendorID.

### Operations

| Method | Description |
| --- | --- |
| read | Retrieve information about a specific shipping vendor. |
| add | Create a new instance of the ShippingVendor class. |
| update | Modify the information about an existing shipping vendor. |
| delete | Remove a shipping vendor from the list of available vendors. |

## Invoice

### Overview

| **Invoice** | |
| --- | --- |
| **Attributes** | **Methods** |
| invoiceID | read |
|  | add |
|  | update |
|  | delete |
|  | calculate |
|  | pay |
|  | export |
|  | send |

### Description

The Invoice class provides a multitude of methods in order to generate an invoice for a customer based on the information present for the PurchaseOrder associated with the Invoice.

### Attributes

An instance of **Invoice** is uniquely identified by its **invoiceID**. It is associated with an instance of **PurchaseOrder** by invoiceID.

### Operations

| Method | Description |
| --- | --- |
| read | Retrieve information about a specific invoice. |
| add | Create a new instance of the Invoice class, which represents a new invoice for a customer's order. |
| update | Modify the information about an existing invoice. |
| delete | Remove an invoice. |
| calculate | Determine the total amount due for an invoice, based on the items included in the order and any applicable taxes or fees. |
| pay | Process a payment for an invoice, updating the invoice's payment status. |
| export | Generate a printable or downloadable version of the invoice, in a format such as PDF or Excel. |
| send | Email the invoice to the customer. |

## User

### Overview

| **User** | |
| --- | --- |
| **Attributes** | **Methods** |
| userID | read |
| name | add |
| email | update |
| password | delete |

### Description

A User instance includes information about the user and allows the customer to access a portal where they can interact with orders and invoices assigned to them. The class is meant to be implemented as either a CustomerUser or MerchantUser.

### Attributes

An instance of User is uniquely identified by its userID. The password for the user is not stored as plain text, rather, a hashing algorithm baked into the program is used to hash passwords.

### Operations

| Method | Description |
| --- | --- |
| read | Retrieve information about a specific user, such as their account details. |
| add | Create a new instance of the User class, which represents a new user who can access the program or system. |
| update | Modify the information about an existing user, such as their password. |
| delete | Remove a user from the list of registered users. |

## MerchantUser

### Overview

| **MerchantUser** | |
| --- | --- |
| **Attributes** | **Methods** |
|  | viewTransactions() |

### 

### Description

The MerchantUser class is an implementation of the User class. It inherits its attributes and methods and adds the ability to view all transactions in the system. It is intended to be an administrative view.

### Attributes

The **MerchantUser** class is uniquely identified by **userID**. See the section on User for a detailed description of the parent class.

### Operations

| Method | Description |
| --- | --- |
| viewTransactions | Retrieve a list of transactions in the system including information such as the date, amount, and status of each transaction. |

## 

## CustomerUser

### Overview

| **CustomerUser** | |
| --- | --- |
| **Attributes** | **Methods** |
|  | downloadInvoices |
|  | viewInvoices |
|  | viewPurchaseOrders |
|  | payInvoices |

### Description

The CustomerUser class is an implementation of the User class. It inherits its attributes and methods and adds the ability to manage orders directly related to the customer. It is intended to be a limited view into the application.

### Attributes

The **CustomerUser** class is uniquely identified by **userID**. See the section on User for a detailed description of the parent class.

### Operations

| Method | Description |
| --- | --- |
| downloadInvoices | Generate a downloadable version of the customer user's invoices, in a format such as PDF or Excel. |
| viewInvoices | Retrieve a list of the customer user's invoices. |
| viewPurchaseOrders | Retrieve a list of the customer user's purchase orders. |
| payInvoices | Process a payment for one or more of the customer user's invoices. |

# [USER INTERFACE DESIGN](https://lucid.app/lucidspark/3684f005-0c58-4033-aee0-40db589270ad/edit?viewport_loc=-13441%2C-4309%2C8633%2C4805%2C0_0&invitationId=inv_dd6421e4-c835-4cd6-9fa8-d34e7c8fa611)

Core pages for the end users will include a Dashboard, In Stock, and Purchase Order for Phase 1. Phase 2 will potentially include Reporting (to provide high level analytics of orders, i.e. orders per month). [Link to Lucid mock ups found here.](https://lucid.app/lucidspark/3684f005-0c58-4033-aee0-40db589270ad/edit?invitationId=inv_dd6421e4-c835-4cd6-9fa8-d34e7c8fa611#)

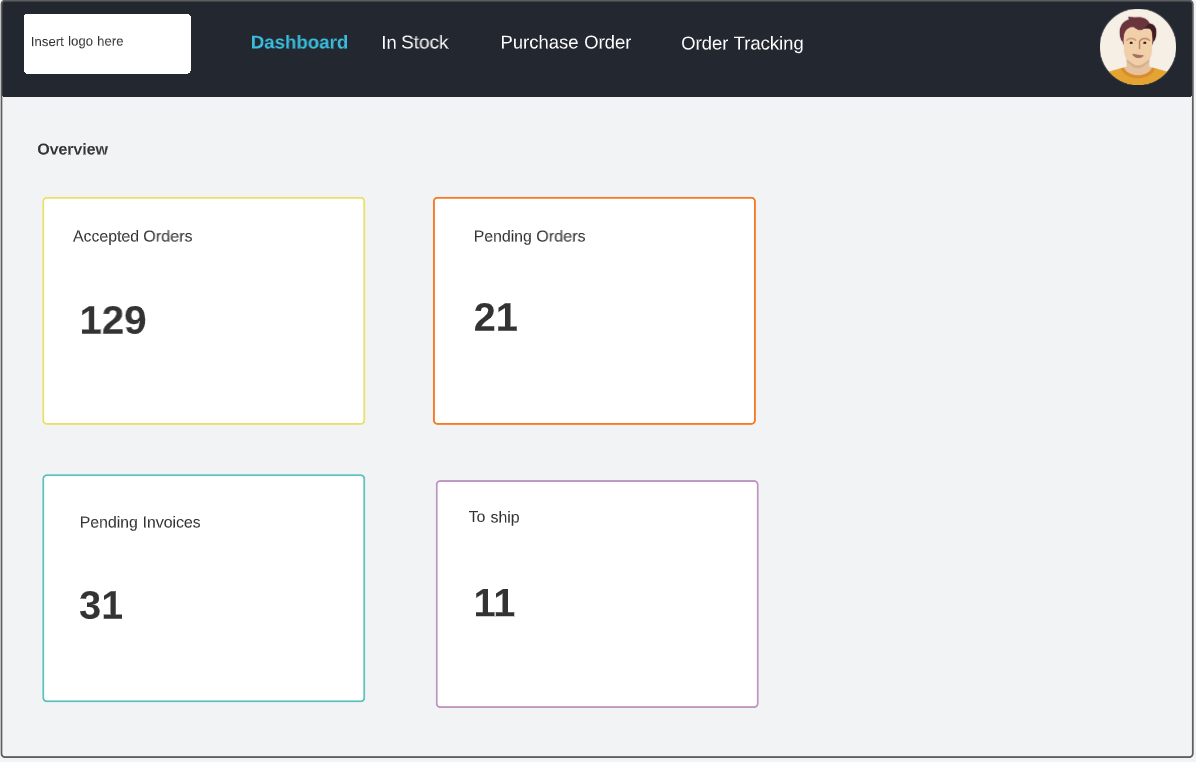
The dashboard provides a high level overview of the IMS (Accepted Orders, Pending orders, Pending invoices, To ship Orders). Tiles on the dashboard highlight key details for the merchant and are easily visible. (*See Figure A)*

The In Stock inventory lists the item’s: class (accessories, clothing, shoes, jewelry), subclass (jacket, dress, necklace), department (mens, womens), sku, vendor, style, color, quantity, lot number, retail price, and wholesale price. A search bar allows the merchant to search for any items as they type details of attributes associated with it. The “status” drop down allows the merchant to filter via order status (i.e. processing, complete). The “create order” button will take the merchant to the Purchase order page. (*See Figure B)*

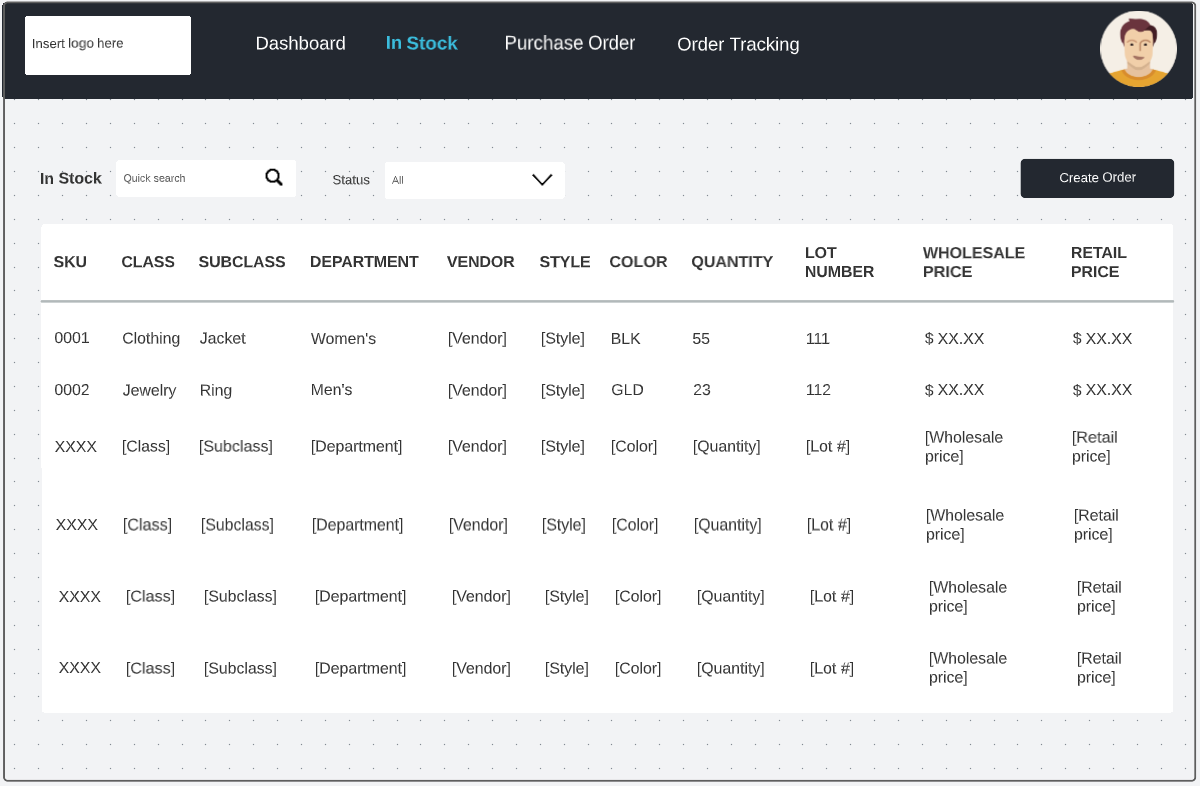
The Purchase Order page provides an inventory order form with the necessary fields for order. There is also a “bulk import” functionality where the end user can upload an XLS document when working with large orders. The “send invoice” button will send an invoice to the customer who will then pay it, and then the Merchant will complete the order. The merchant can user drop downs to select previously used Client information (names, addresses) or enter new client information. *(See Figure C)*

The order tracking page will provide a detailed view of the items reflected in the dashboard (orders placed, pending orders, canceled, and delivered orders). Items will be filtered within the appropriate tabs for the merchant. *(See Figure D)*

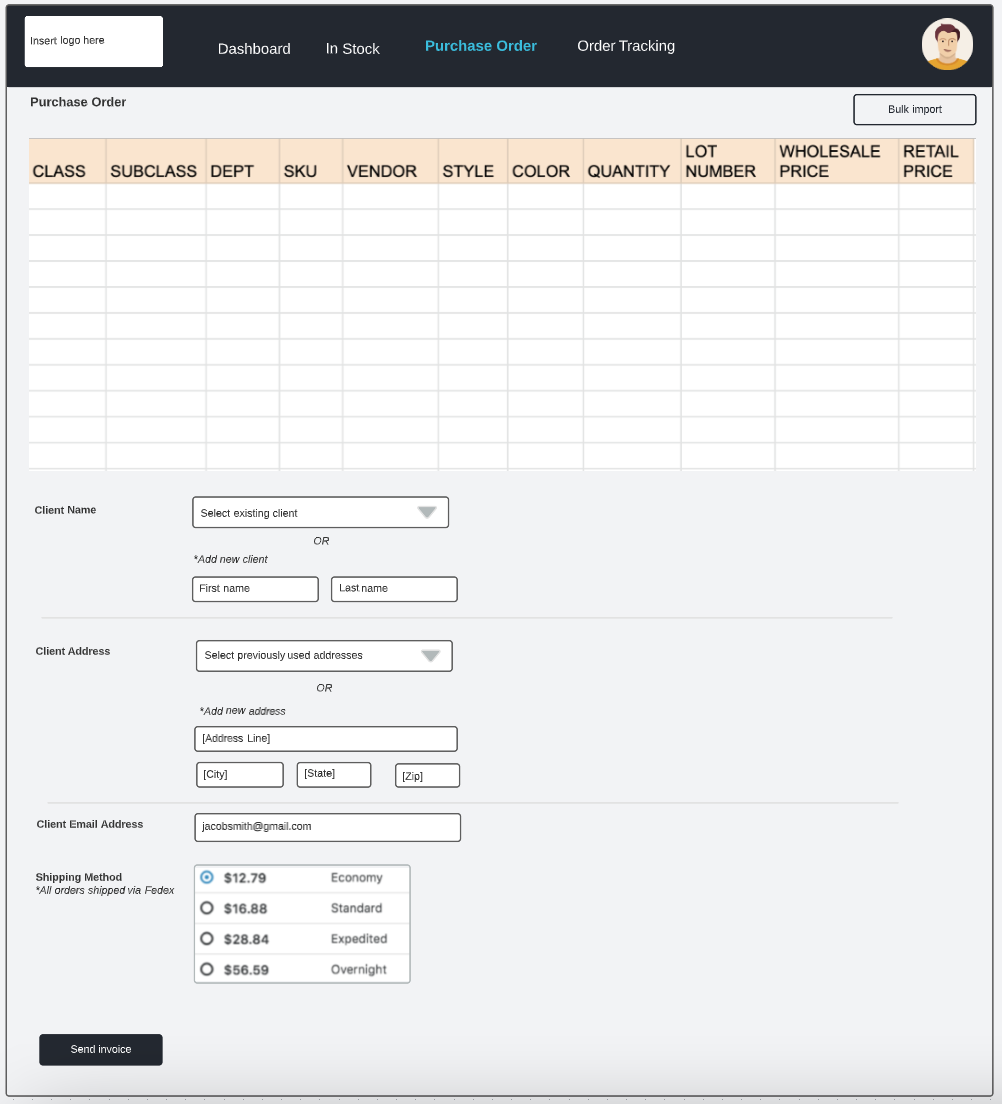
**Figure A. Dashboard view**



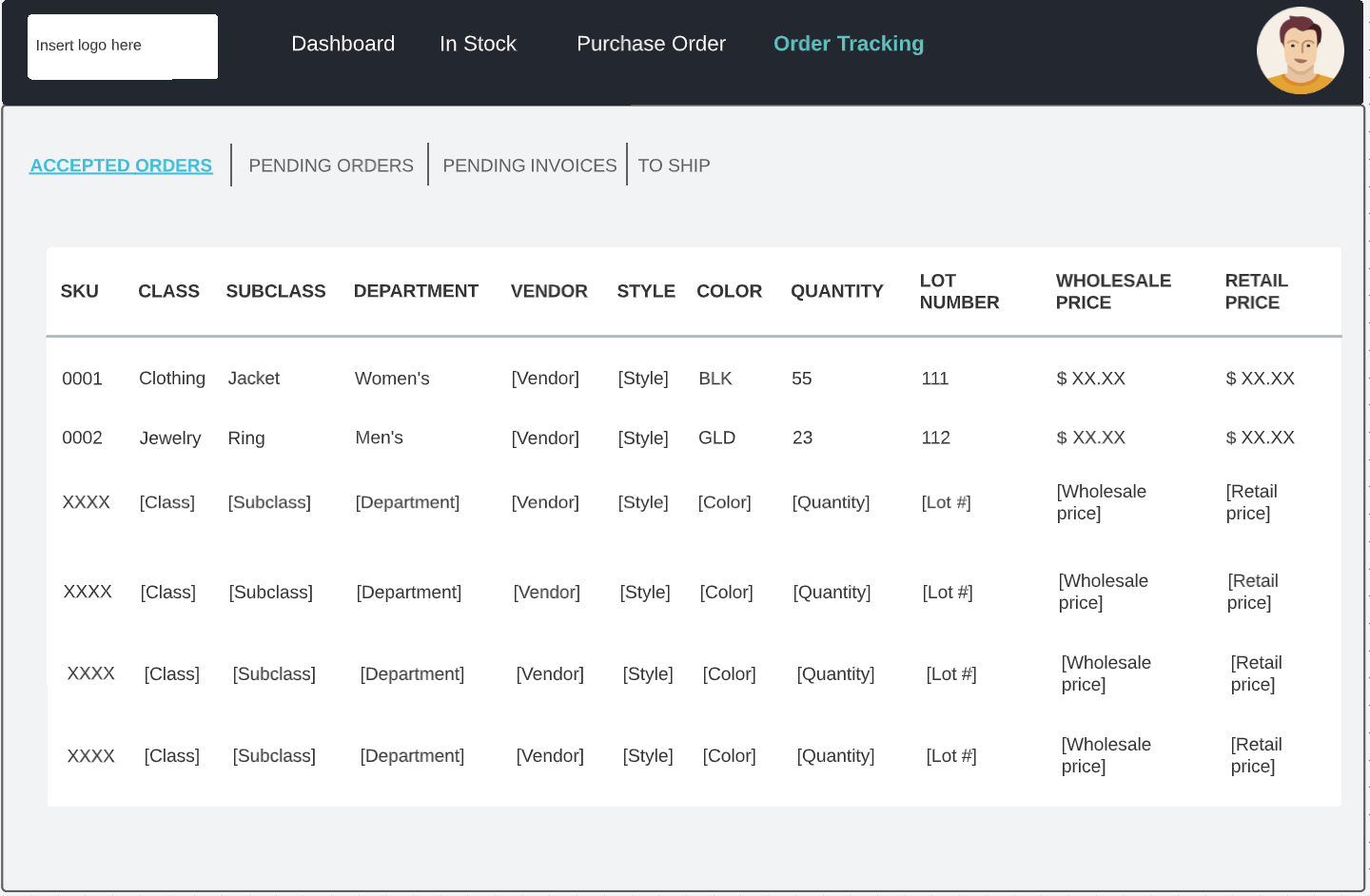
**Figure B. Inventory/Stock view**

****

**Figure C. Purchase Order view**

****

**Figure D. Order Tracking view**

****