**Standards for Design Specification**

**Imor Hartnett X00118631**

**Conor Griffin X00111602**

**Lee Healy X00120179**

Contents

[**Input Design** 2](#_Toc443908717)

[**Output Design** 5](#_Toc443908720)

[**Database Design** 6](#_Toc443908721)

[**Entity Relationship Diagrams** 6](#_Toc443908722)

[**Table/Record Layouts** 7](#_Toc443908723)

[**Program Design** 8](#_Toc443908724)

[**Pseudocode** 8](#_Toc443908725)

[**Test Design** 11](#_Toc443908726)

[**Documentation Design** 11](#_Toc443908727)

## **Input Design**

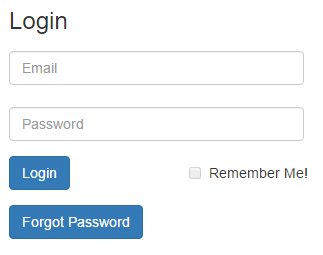
For our input design we chose to show the main input screens for both the customer and the administrator.

For the Customer he must be able to perform actions such as Logging In, Signing Up to the website, entering address details, entering card details to make a purchase, purchasing products.

For the Administrator he must also be able to Log in so he can check stock flow and look at reports.

**Customer Input Screens**

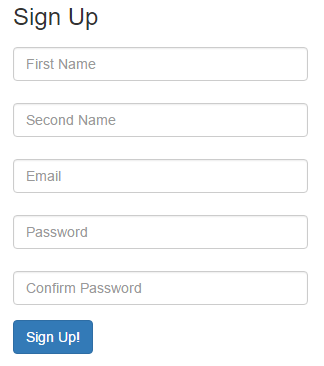
Logging into the website, user must enter his/her email and password..



## 

Signing up to the website:

User must enter details name, email, password..



User then must enter his/her address details, house number, street name, county, postal code.

## 

User must enter card details when buying something from the website, name on card, card number,expiry month, year and security number.

## 

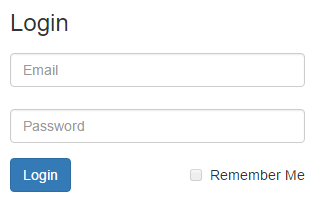
Customer enter’s amount of product to buy..

User can contact us or they can enter Information and register to our newsletter..

## 

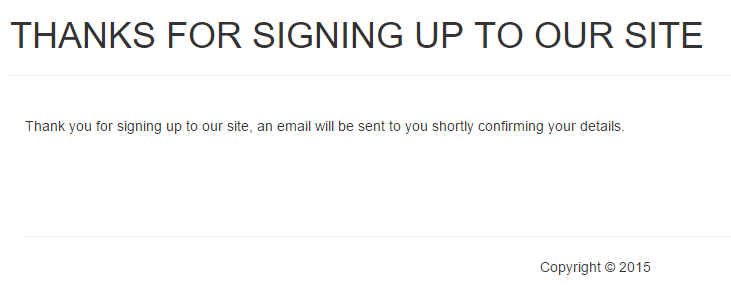
**Administrator Input Screens**

Admin must Login to access Stock details.

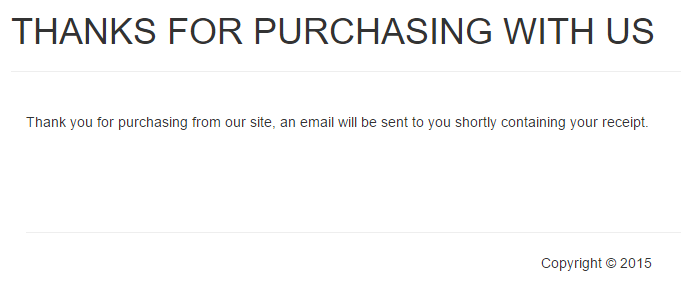


## **Output Design**

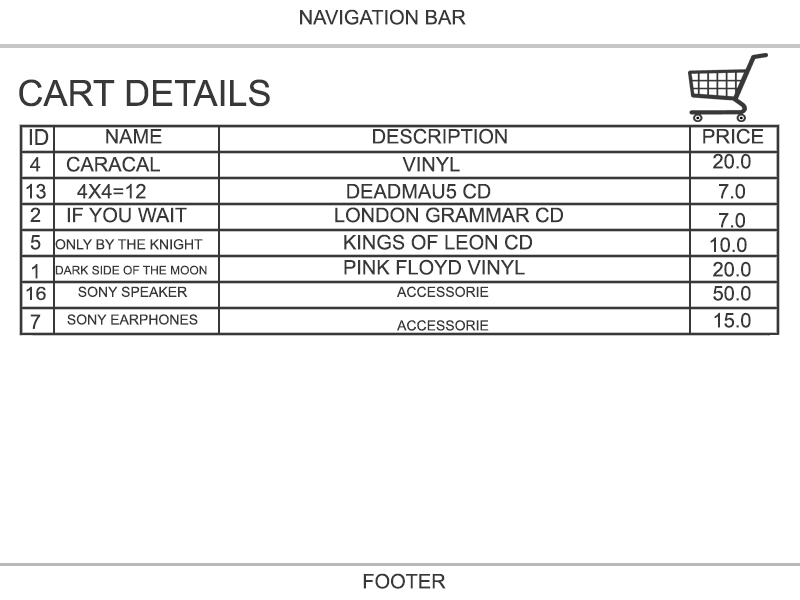
Customer signs up successfully..



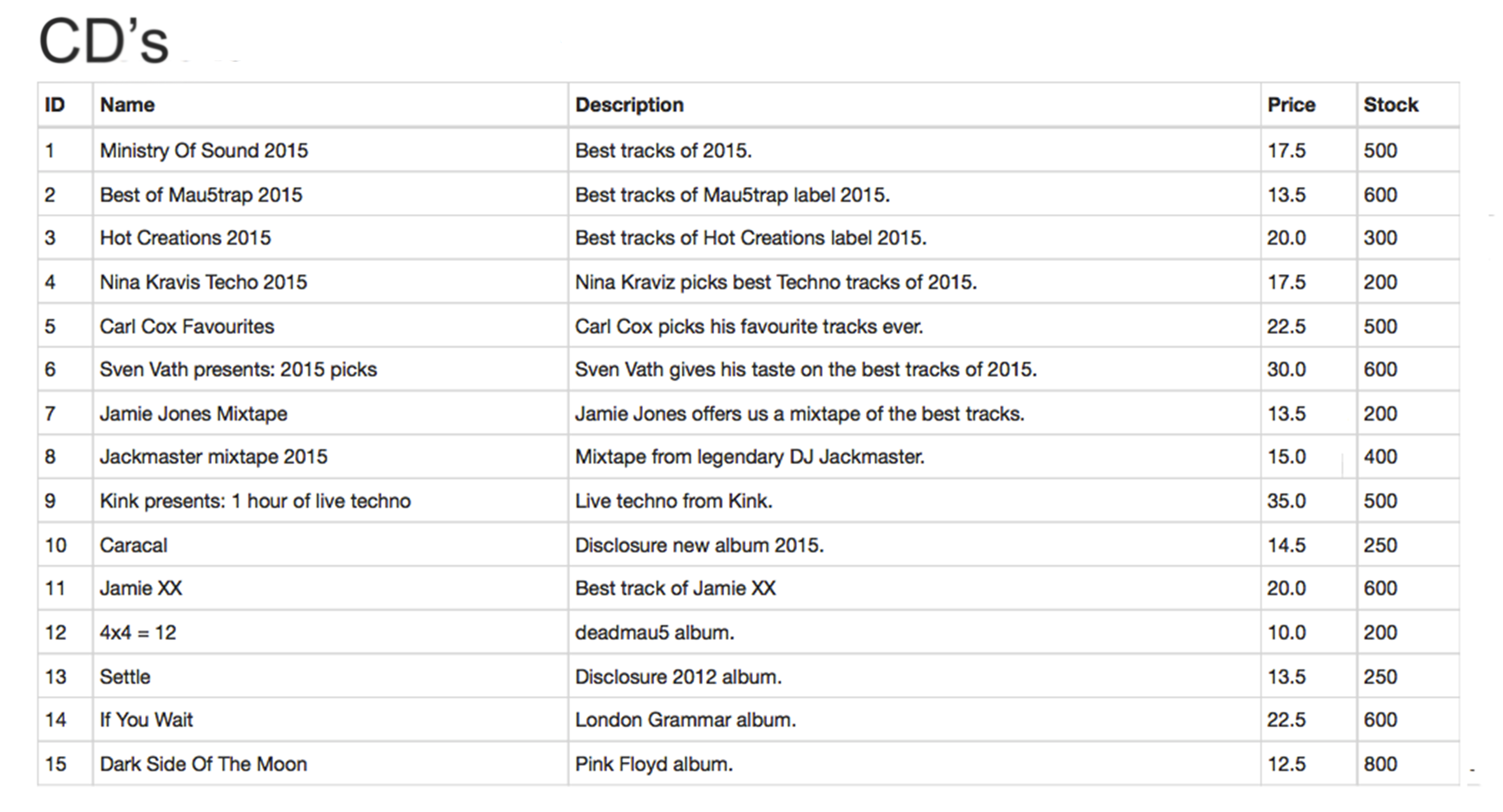
Successful purchase..



Customer can look at his/her cart..



Administrator checks stock



## **Database Design**

## **Entity Relationship Diagrams**



## **Table/Record Layouts**

**Customer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Customer\_ID | First\_Name | Last\_Name | D.O.B | Email | Password |
| Number, NOT NULL | VARCHAR  (25) | VARCHAR  (25) | VARCHAR(20) | VARCHAR(30) | VARCHAR  (20) |
| Primary Key |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| Credit\_Card\_Number | Credit\_Card\_ExMonth | Credit\_Card\_ExYear |
| Number | VARCHAR(10) | VARCHAR  (10) |

**Order**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Order\_ID | Customer\_ID | Quantity | Date\_Created | Total\_Cost |
| Number, NOT NULL | Number, NOT NULL | VARCHAR(10) | VARCHAR(15) | Number |
| Primary Key | Foreign Key |  |  |  |

**Stock**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stock\_ID | Name | Description | Price | Quantity |
| Number, NOT NULL | VARCHAR(25) | VARCHAR(255) | Number | VARCHAR(20) |
| Primary Key |  |  |  |  |

**Stock Order Details**

|  |  |
| --- | --- |
| Order\_ID | Stock\_ID |
| Number, NOT NULL | Number, NOT NULL |
| Primary Key  Foreign Key | Primary Key  Foreign Key |

**Sales**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sales\_ID | Admin\_ID | Name | Description | Price |
| Number, NOT NULL | Number, NOT NULL | VARCHAR(255) | VARCHAR(255) | Number, NOT NULL |
| Primary Key | Foreign Key |  |  |  |

**Admin**

|  |  |  |  |
| --- | --- | --- | --- |
| Admin\_ID | Name | Email | Password |
| Number, NOT NULL | VARCHAR(25) | VARCHAR(50) | VARCHAR(20) |
| Primary Key |  |  |  |

## **Program Design**

## **Pseudocode**

Login system

**Inputs: (Name: Type:)**

Username :String

Password :String

**Begin:**

**IF** (Username **==** EnteredUsername **&&** Password **==** EnteredPassword) **THEN**

Login Successful

**ELSE**

Login Failed.

**ENDIF**

**End.**

Remove Stock

*//Customer buys their items and the items are removed from stock*

**Inputs: (Name: Type:)**

stockAmount :int

itemsAmount :int

**Constants:**

LOW\_STOCK :int = 5

**Begin:**

**OUTPUT** (How many would you like to purchase?)

**INPUT** itemsAmount

stockAmount -= itemsAmount

**IF**(StockAmount **<=** LOW\_STOCK)

**OUTPUT** (to stock administrator) (Warning, low stock on item: )

**ELSE**

Stock removed successfully.

**END IF**

**End.**

PLACE ORDER

*//Customer goes online and views our products*

*//Customer sees a product they like and adds it to the basket*

*//The basket fills up with items that the customer wants*

*//The Customer then proceeds with this basket and makes the order*

**Inputs: (Name: Type:)**

basket :ArrayList *//ArrayList to hold objects (products) that the customer wants*

**Begin:**

**IF** (basket == null) **THEN**

**OUTPUT** Error: Cannot proceed with empty basket

**ELSE IF** (basket != null) **THEN**

**OUTPUT** Proceeding with items…

**End.**

*//All item prices will be added up to a total price and displayed to the Customer*

PAY ORDER

*//The total price for all items in the basket is displayed to the customer and they are then asked to enter their card details.*

**Inputs: (Name: Type:)**

cardNum :int

cardExMonth :int *//Number representation of the month e.g. feb = 2*

cardExYear :int *//Number representation of the year*

**Outputs: (Name: Type)**

totalPrice :double

**OUTPUT** (Total price for all items: **totalPrice**)

**OUTPUT** (Please enter your card number:)

**INPUT** (cardNum)

**OUTPUT** (Please enter your card’s expiry date:)

**INPUT** (cardExMonth) **INPUT** (cardExYear)

**OUTPUT** (Validating card details..)

CHECK STOCK

***//****If the stock of an item is below 5 then warn the Stock Administrator that it is low, if it is not low, the amount of items left should be displayed to the Stock Administrator*

**Outputs: (Name: Type:)**

certainItemStock :Object *//An object of stock in the system, e.g. A Speaker would be //a product object in an ArrayList*

amount :int *//Amount of stock of an item (e.g. 7 speakers)*

**Begin:**

**IF** (certainItemStock < 5) **THEN**

**OUTPUT** Warning: Stock for **certainItemStock** is low!

**OUTPUT** There are **amount** left of **certainItemStock**

**ELSE IF** (certainItemStock > 5) **THEN**

**OUTPUT** There are **amount** left of **certainItemStock**

**End.**

RESUPPLY STOCK

*//Stock Administrator emails the Supplier an email detailing what stock the store needs.*

**Outputs: (Name: Type:)**

stockItem :Object

supplier :String *//Suppliers name*

supplierEmail :String *//Supplier’s email*

**OUTPUT** (Stock level for **stockItem** is low)

**OUTPUT** (The supplier for this product is: **supplier**)

**OUTPUT** (To resupply stock, email the supplier at: **supplierEmail**)

*//Stock Administrator then Emails the supplier asking for more stock.*

EDIT STOCK

***//****Stock Administrator goes on and edits the products in the stock*

*//The Stock Administrator can edit the ID, Name, Description and Price of stock.*

**Inputs: (Name: Type:)**

newProductID :String

newProductName :String

newProductDescription :String

newProductPrice :String

**Outputs: (Name: Type:)**

productID :int

productName :String

productDescription :String

productPrice :double

**Begin:**

***//****Displayed in a form*

**OUTPUT** (productID)

**INTPUT** (newProductID)

**OUTPUT** (productName)

**INTPUT** (newProductName)

**OUTPUT** (productDescription)

**INTPUT** (newProductDescription)

**OUTPUT** (productPrice)

**INTPUT** (newProductPrice)

//*When ‘****Submit****’**is clicked*

Product edited successfully.

**OUTPUT** (Saved!)

**End.**

## 

## **Test Design**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case Matrix** | | | | | |
| **Test Case No** | **Test Case Name** | **Input** | **Expected Outcome** | **Actual Outcome** | **Result (Pass/Fail)** |
| 1 | verifyLogin | Username and password is entered | User is successfully logged in to the profile |  |  |
|  |  |  |  |  |  |
| 2 | checkStock | Manager enters product id into the database | Amount of stock at hand for the product is displayed to the user |  |  |
|  |  |  |  |  |  |
| 3 | login | Enter name and password. | Successfully logged in as customer. |  |  |
| 4 | signUp | Ask user for id and password that meets the set criteria. | Create account  which stores he user’s data in the database. |  |  |
| 5 | updateProfile | Prompt user to enter name and password | Profiles details are successfully changed |  |  |
|  |  |  |  |  |  |
| 6 | updateShippingInfo | New shipping address for item to be sent to | Details are successfully changed and saved. |  |  |
|  |  |  |  |  |  |
| 7 | addStock | Amount of stock to be added with the prodcts unique id | Stock is added to the database and new total is displayed to the user |  |  |
| 8 | deleteStock | Amount and id of stock to be deleted | New total count of stock is successfully updated. |  |  |
| 9 | updateStock | Updated stock information and stock id. | Stock is successfully changed and displays the updated correct information. |  |  |
|  |  |  |  |  |  |
| 10 | addCartItem | Items are placed into the user's cart. | User's cart is updated and the total price of products currently in the basket is displayed. |  |  |
| 11 | checkOut | User’s card details to be entered or customer chooses which payment method to use | All item, their quantity are total price of basket shown to the user |  |  |
| 12 | deleteCartItem | Users chooses how many items to delete. | The user's cart is successfully updated with the new subtotal being displayed |  |  |
| 13 | updateQuantity | Quantity of item changed in the cart | The user's cart is successfully updated with the new subtotal being displayed |  |  |
| 14 | viewCartDetails | Display what products are currently placed in the cart | Cart details including details on the products in the cart, total price of the cart and and included charges on top of product cost |  |  |
|  |  |  |  |  |  |
| 15 | placeOrder | User confirms payment of purchase | Order is passed through to back end of system to be prepared for delivery to the user |  |  |
|  |  |  |  |  |  |
| 16 | calcPrice | Prices of individual products added together | Total of the products are successfully summed together and the total is displayed to the user. |  |  |

## **Documentation Design**

1. User
   * Register as a customer………………………………………………………………………………
   * Login as a customer…………………………………………………………………………………..
   * Familiarizing the user with the website's layout…………………………………………
   * Browse products………………………………………………………………………………………..
2. Customer
   * Update profile information..……………………………………………………………
   * Change billing address….…………………………………………………………………
   * Add two or more shipping addresses………………………………………………
   * Placing an order………………………………………………………………………………
   * Deleting item from shopping cart…………………………………………………...
   * Cancelling an order before it is processed……………………………………….
   * Paying for an order………………………………………………………………………….
   * Adding two or more payment methods……………………………………….....
3. Admin
   * Add Stock………….……………………………………………………………………………………………
   * Delete Stock…………………………………………………………………………………………………..
   * Resupplying Stock…..……………………………………………………………………………………..
   * Update Stock Information…………………………………..………………………………………...
   * Checking Recent Sales Figures……………………………………………………………………….
   * Generate Reports……………………………….………………………………………………………….