Requirements Engineering

Modestas Cepulis T00199237

Wholesale sales processing system

Submitted By: Modestas Cepulis (T-00199237)

Computing with Games Development

Date Submitted: 26/10/2018

**Table of Contents**

[1. Introduction/overview 3](#_Toc536524703)

[2. Functional Components 4](#_Toc536524704)

[3. User Requirements 5](#_Toc536524705)

[5. System Requirements 6](#_Toc536524706)

[5.1. System Level Use Case Diagram 6](#_Toc536524709)

[5.2. Manage Customers 7](#_Toc536524710)

[5.2.1. Record customer details 7](#_Toc536524715)

[5.2.2. Change customer details 11](#_Toc536524716)

[5.2.3. Remove customer details 14](#_Toc536524717)

[5.3. Process Orders 15](#_Toc536524718)

[5.3.1. New order details 15](#_Toc536524719)

[5.3.2. Change order details 20](#_Toc536524726)

[5.3.3. Dispatch order 23](#_Toc536524734)

[5.3.4. Cancel order 26](#_Toc536524743)

[5.4. Perform Admin 28](#_Toc536524744)

[5.4.1. Delivery analysis 28](#_Toc536524745)

[5.4.2. Analyse orders 30](#_Toc536524753)

[5.4.3. Analyse revenue 32](#_Toc536524762)

[5.5 Manage Stock 34](#_Toc536524763)

[5.5.1 New Stock 34](#_Toc536524774)

[5.5.2 Replenish Stock 37](#_Toc536524775)

[6. System Model 40](#_Toc536524776)

[6.1. Level-0 DFD 40](#_Toc536524779)

[6.2. Level-1 DFD 41](#_Toc536524780)

[6.3. Level-2 DFD (Process P1: Manage customers) 42](#_Toc536524781)

[6.4. Level-2 DFD (Process P2: Process Orders) 43](#_Toc536524782)

[6.5. Level-2 DFD (Process P3: Perform Admin) 44](#_Toc536524783)

[6.6. Level-2 DFD (Process P3: Manage Stock) 45](#_Toc536524784)

[7. Data Model (Class Diagram) 46](#_Toc536524785)

[7.1.1. Class Diagram 46](#_Toc536524789)

[7.2. Relational Schema 47](#_Toc536524790)

[7.3. Database Schema 48](#_Toc536524791)

[8. Conclusion 50](#_Toc536524792)

# Introduction/overview

Whole sales processing system is the process of making a product or service available for the consumer or business that needs it. The business needs to ensure that the products reach targeted customers in the most direct and cost efficient matter.

The System shows how the wholesale sales processing system works. System can be separated into four different categories - dealing with customers, processing orders, performing admin duties and managing the overall stock of the business.  
  
- Managing Customers  
 It can record, change or remove customer details. Thus giving the user an accessibility to the main functionality of the system, - ordering any goods from the business.  
  
- Processing Orders  
 Allows an existing customer to make, change, or cancel an order. If the order is cancelled its status is automatically set to ‘C’ (Cancelled).  
The admin has the responsibility to dispatch the orders. Once an order is dispatched, its status is set to ‘D’ (Dispatched).  
   
- Performing Admin  
 Allows the admin to analyse the deliveries, the revenue and orders made by the customers. Most of the data is shown in form of graphs.  
   
- Managing Stock  
 This process lets the Admin to add new stock to the system or replenish/replace missing stock.

# Functional Components

This section of the document presents the functional components of the proposed Wholesales Sales processing system.

# User Requirements

The purpose of user requirements is to describe the main functionalities of the system. It provides the user with information about the functions of the system.

1. **StockSYS will manage customers.**

1.1 StockSYS will register customer details.

1.2 StockSYS will allow customer details to be changed.

1.3 StockSYS will allow customer details to be removed.

1. **StockSYS will manage stock.**

2.1 StockSYS will allow to add new stock.

2.2 StockSYS will allow an order of new stock to be made.

1. **StockSYS will manage orders.**

3.1 StockSYS will allow an order to be made.

3.2 StockSYS will allow an order to be changed.

3.3 StockSYS will allow an order to be dispatched.

3.4 StockSYS will allow an order to be cancelled.

1. **StockSYS will perform administrative reporting.**

4.1 StockSYS will provide a list of daily deliveries.

4.2 StockSYS will provide an analysis on all orders.

4.3 StockSYS will provide a revenue analysis report.

# System Requirements

System requirements are all of the requirements at the system level that describe the functions which the system as a whole should make.



## System Level Use Case Diagram

The high-level functional components of the system are shown in the following system level Use Case diagram.

Customer

Admin

## Manage Customers



### Record customer details

This function records the details of a new **Customer** and saves it in the **Customers file**. It runs validation on user inputs and shows the confirmation or error message accordingly. It sets the current customer status to ‘Active’.

Customer

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Record Customer Details** | |
| **Use Case Id** | 1 | |
| **Priority** | 1 | |
| **Source** | Customer | |
|  |  | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function records the details of a new customer. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Customer invokes the Add Customer Details function  **Step 3:** Customer enters his ‘Customer’ details   * Username * Password * First name * Last name * E-mail address * Address * Street * Town * County * Eircode * DOB * Phone | **Step 2:** The system displays the UI.  **Step 4:** The system Validates the order details entered:   * All fields must be entered * First and last name must be alphabetic. * Address details must have a valid address or a valid post code * The phone number must be numeric and no longer or shorter than 10 characters. (starting with 08) * E-mail address needs to have a valid format * Date Of Birth must be in a valid format (DD-MM-YYYY) * The customer must be at least 18 years old   **Step 5:** The system assigns appropriate customer ID for the new customer created.  **Step 6:** The system sets this customer status to Active  **Step 7:** The system saves the customer details in the **Customers File:**   * CustID * First Name * Last Name * Address (Street, Town, County, Eircode) * Date Of Birth * E-mail address * Phone Number * Username * Password * Status   **Step 8:** The system displays a confirmation message  **Step 9:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 4:** Invalid data is entered  **Step 5:** The system displays an appropriate error message |
| **Conclusions** | The customer details are saved in the Customers file | |
| **Post conditions** | The customer now can access the Orders function | |
| **Business Rules** | The customer must be at least 18 years old | |
| **Implementation Constraints** |  | |

### Change customer details

This function allows the user to edit its customer details. The system runs a validation process before updating the customers details in the **Customers File.**

Customer

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Change Customer Details** | |
| **Use Case Id** | 2 | |
| **Priority** | 2 | |
| **Source** | Customer | |
|  |  | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function changes the details of an existing customer. | |
| **Preconditions** | Customer must have logged in | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The customer logs in to his/her account.  **Step 2:** The Customer invokes the ‘Change Customer Details’ function  **Step 4:** Customer enters his new ‘Customer’ details   * First name * Last name * Address * E-mail address * Street * Town * County * Eircode * DOB * Phone | **Step 3:** The system retrieves the customer details from the **Customers** file and displays on the UI in alphabetical order of customer name.  **Step 5:** The system Validates the customer details entered:   * All fields must be entered * First and last name must be alphabetic. * Address details must have a valid address or a valid post code * The phone number must be numeric and no longer or shorter than 10 characters. (starting with 08) * Date Of Birth must be in a valid format (DD-MM-YYYY) * The customer must be at least 18 years old   **Step 6:** The system updates the customer details in the **Customers File**   * First Name * Last Name * E-mail address * Address (Street, Town, County, Eircode) * Date Of Birth * Phone Number * Status * Username * Password   **Step 7:** The system displays a confirmation message  **Step 8:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 4:** Invalid data is entered  **Step 5:** The system displays an appropriate error message |
| **Conclusions** | The customer details are updated on the Customers file | |
| **Post conditions** |  | |
| **Business Rules** | Only active customers can be changed Customer cannot log in if inactive | |
| **Implementation Constraints** |  | |

### Remove customer details

This function allows the **Customer** to remove his/hers customer details from the system. The system then sets the status of the customer to Removed.

Customer

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Remove Customer Details** | |
| **Use Case Id** | 3 | |
| **Priority** | 2 | |
| **Source** | Customer | |
|  |  | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function sets the Customer status to *Removed* of an already existing customer. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Customer invokes the ‘Remove Customer Details’ function  **Step 3:** The Customer selects the customer name and details to be removed.  **Step 4:** The Customer confirms that the customer details are to be set for removal | **Step 2:** The system retrieves the customer details from the **Customers** file and displays on the UI in alphabetical order.  **Step 5:** The system sets the customer status to ‘Removed’ in the **Customers File**.  **Step 6:** The system displays a confirmation message  **Step 7:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The customer status is set to Removed in the Customers file | |
| **Post conditions** | The customer cannot access his account anymore | |
| **Business Rules** | The customer cannot remove his account if he/she owes money to the company | |
| **Implementation Constraints** |  | |

## Process Orders

### New order details

This function allows an existing customer to make an order. The **Customer** inputs the details of the order and the system validates them. If the details are valid, the system sends a confirmation e-mail to the customer and shows a confirmation message. The system saves the order details in the **Orders** and **Orders item** files.  
If the customer orders the item, the company must remake the item lost. The system must delete the information from the basket/cart and decrease the stock in the warehouse.  
Once the customer adds the item for ordering, the system must increase the total bill/cost of the items.

New Order activity diagram showing the main processes of the function.

System

Customer

Assign Order ID

Invoke New Order

Enter new order data

Display UI

Validate data

Valid?

N

Y

Add items to basket

More?

Y

N

Show total value

Show total value

Save Order details

Confirmation message

Continue

Reset UI

Customer

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **New Order** | |
| **Use Case Id** | 4 | |
| **Priority** | 1 | |
| **Source** | Customer | |
|  |  | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function records the details of a new order. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Customer invokes the New Order function  **Step 3:** Customer chooses the desired stock and enters the quantity wanted   * Item type * Amount of goods * Delivery details   **Step 4:** The Customer confirms the order details to be added | **Step 2:** The system displays the UI.  **Step 5:** The system Validates the order details entered:   * All fields must be entered * The order must have a valid Item type and a valid quantity (amount of goods) that have a status ‘A’ for available in stock * Item type must be alphabetic and at least 15 characters long * Amount of goods must be numeric and no longer than 7 characters long * Delivery details must have a valid delivery address, a valid post code and a valid delivery date * The date must have a valid format (dd-mm-yyyy)   **Step 6:** The system assigns an appropriate OrderID for the new order.  **Step 7:** The system calculates the cost of the item(s) by using a formula (Quantity \* Cost) and adds to the running total of the shopping cart  **Step 8:** The system adds the item details to the shopping cart  (If the customer wants to add more items, the system brings back the user to step 3)  **Step 9:** The system saves the order details in the **Orders File**   * The type of the Item ordered * The ID of the customer that made this order * The amount ordered * The delivery address * Street * Town * County * Eircode * The delivery date (Day, Month, Year, Preferable Time)   **Step 10:** The system saves each item in shopping cart to the **Order Items File**  **Step 11:** The system confirms that the order has been made by sending an e-mail to the customer  **Step 12:** The system displays a confirmation message  **Step 13:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 5:** Invalid data is entered  **Step 6:** The system displays an appropriate error message |
| **Conclusions** | The order details are saved in the Orders file | |
| **Post conditions** |  | |
| **Business Rules** | Only active stock (status = ‘A’ may be ordered) | |
| **Implementation Constraints** |  | |



### Change order details

This function allows an existing customer to change his/hers order details. The system validates the user input data. If the data is valid, the system displays a confirmation message. The new order details are updated in the **Orders** file.  
If the customer orders the item, the company must remake the item lost. The system must delete the information from the basket/cart and increase the stock in the warehouse.  
Once the customer adds the item for ordering, the system must increase the total bill/cost of the items.

Customer

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Change Order** | |
| **Use Case Id** | 5 | |
| **Priority** | 2 | |
| **Source** | Customer | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function changes the order details. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Customer invokes the Change Order function  **Step 3:** Customer chooses the new desired stock and enters the new quantity wanted   * Item type * Amount of goods * Delivery details | **Step 2:** The system retrieves the details of the order from the **Order Items** file and displays the UI in alphabetical order of description.  **Step 4:** The system Validates the order details entered:   * All fields must be entered * Item type must be alphabetic and at least 15 characters long * Amount of goods must be numeric and no longer than 7 characters long * Delivery details must have a valid delivery address, a valid post code and a valid delivery date * The date must have a valid format (dd-mm-yyyy)   **Step 5:** The system calculates the cost of the item(s) by using a formula (Quantity \* Cost) and adds to the running total of the shopping cart  **Step 6:** The customer sends the information to the Admin informing him that one or more of the items are gone and that he has to replace them.  **Step 7:** The system increases the total bill of the order.  **Step 8:** The system updates the order details in the **Orders File**   * The type of the Item ordered * The ID of the customer that made this order * The amount ordered * The delivery address * Street * Town, * County * Eircode * The delivery date (Day, Month, Year, Preferable Time)   **Step 9:** The system saves each item in shopping cart to the **Order Items File**  **Step 10:** The system confirms the order to the customer by sending an e-mail  **Step 11:** The system displays a confirmation message  **Step 12:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 4:** Invalid data is entered  **Step 5:** The system displays an appropriate error message |
| **Conclusions** | The order details are changed in the ‘Orders’ file | |
| **Post conditions** |  | |
| **Business Rules** | Not allowed to change the details if the order is already dispatched | |
| **Implementation Constraints** |  | |



### Dispatch order

This function allows the **Admin** to dispatch an order to the **Customer**. Before that happens the system validates if the specific order has the correct and required order details. If the order details are correct the system sets the order status to Dispatched ‘D’ and sends a confirmation message to the **Customer.**

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Dispatch Order** | |
| **Use Case Id** | 6 | |
| **Priority** | 2 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function dispatches the order to the customer. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Admin invokes the Dispatch Order function  **Step 3:** The Admin selects the order to be dispatched. | **Step 2:** The system retrieves the details of the order from the **Orders** file and displays the UI in alphabetical order of description.  **Step 4:** The system updates the status of an order to dispatched ‘D’ (Dispatched) in the **Orders File**  **Step 5:** The system sends an e-mail to the customer confirming that the order has been successfully dispatched.  **Step 6:** The system displays a confirmation message  **Step 7:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The order status is updated to dispatched ‘D’ in the Orders file | |
| **Post conditions** | The order is set to be dispatched | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |



### Cancel order

This function allows an existing **Customer** to cancel an order that he made before. The system sets the order status to cancelled ‘C’ (Cancelled) and displays a confirmation message.

Customer

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel Order** | |
| **Use Case Id** | 7 | |
| **Priority** | 2 | |
| **Source** | Customer | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function sets the current order status to cancelled ‘C’ | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Customer invokes the Cancel Order function  **Step 3:** The Customer selects the order to be removed. | **Step 2:** The system retrieves the details of the order from the **Orders** file and displays the UI in alphabetical order of description.  **Step 4:** The system sets the order status to cancelled ‘C’ in the **Orders File**  **Step 5:** The system sets the items to available that they could be dispatched for someone else.  **Step 6:** The system displays a confirmation message  **Step 7:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The order status is set to cancelled ‘C’ in the Orders file | |
| **Post conditions** | The customer can no longer access the order details | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

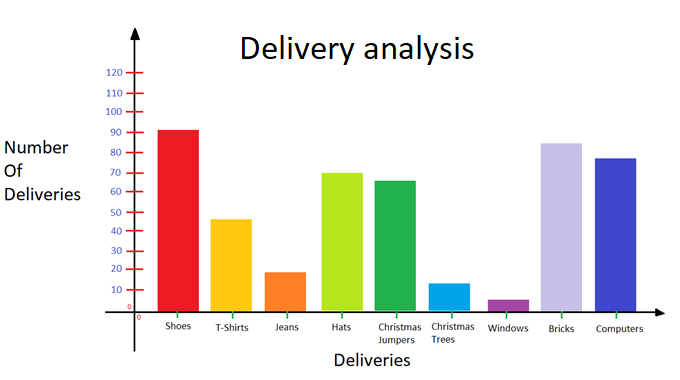
## Perform Admin

### Delivery analysis

This function allows the **Admin** to show the delivery analysis of the company. The system retrieves the order details from the **Orders** file and shows the results in a table.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Delivery analysis** | |
| **Use Case Id** | 8 | |
| **Priority** | 1 | |
| **Source** | Admin | |
|  |  | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This functions lists the daily deliveries made by the company. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Admin invokes the analyse deliveries function  **Step 3:** The Admin chooses what day deliveries he wants to show. | **Step 2:** The system retrieves the details of the deliveries from the **Orders** file.  **Step 4:** The system shows the daily deliveries made by the company from the **Orders File** in a table  **Step 5:** The system displays a confirmation message  **Step 6:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The daily deliveries are listed from the Orders file. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |



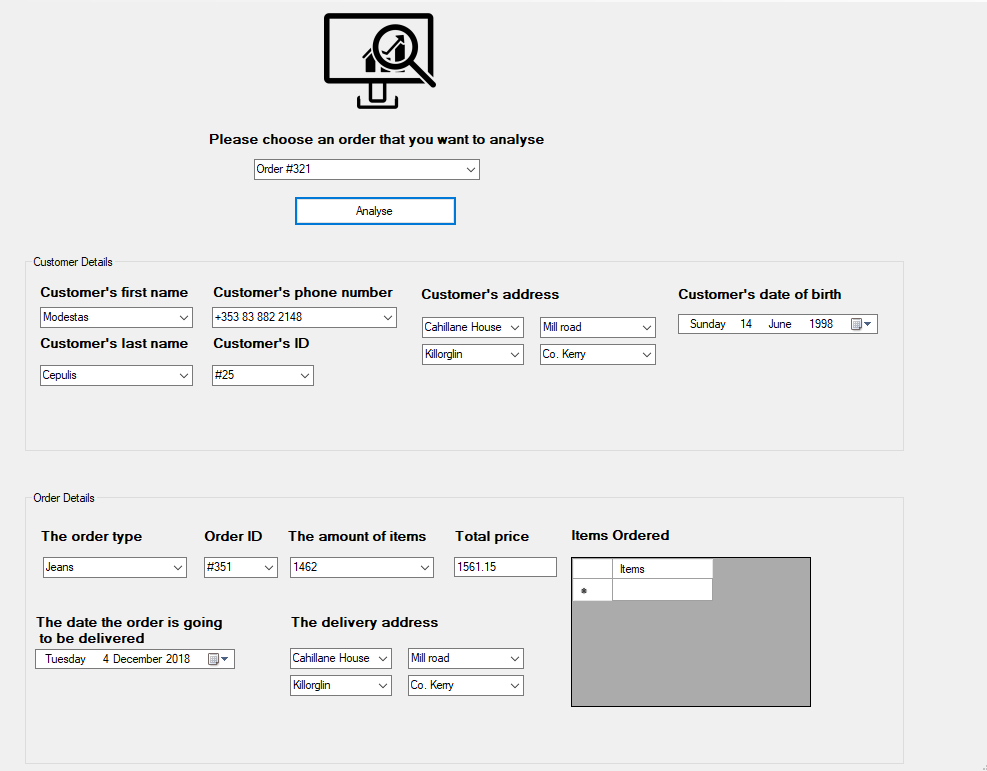


### Analyse orders

This function allows the **Admin** to analyse the orders made by **Customers**. The **Admin** enters the **Order ID**. The system lists the **Order** and **Customer** details from the **Orders** and **Customer files** in a table.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Analyse Orders** | |
| **Use Case Id** | 9 | |
| **Priority** | 2 | |
| **Source** | Admin | |
|  |  | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This functions analyses the orders made by the customers. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Admin invokes the Analyse Orders function  **Step 3:** Admin enters the Order ID that he wants to list   * Order ID | **Step 2:** The system retrieves the details of the orders from the **Orders** file and displays the UI in alphabetical order of description.  **Step 4:** The system lists the order and Customer details from the **Orders** and **Customers Files.**  **Step 5:** The system displays a confirmation message  **Step 6:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The Order and Customer details are listed. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |



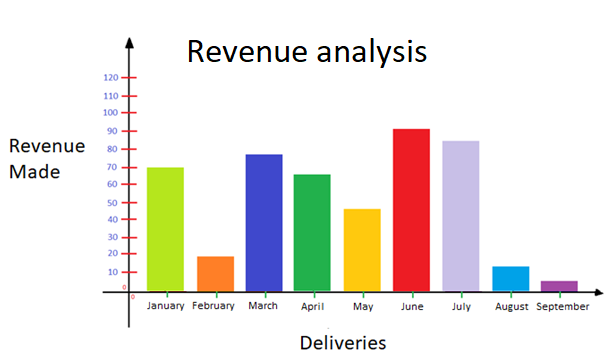


### Analyse revenue

This function allows the **Admin** to analyse the revenue made by the company. The system calculates the total revenue made by the company by getting the cost information from the **Orders** file and adding the total costs together. The system represents the revenue made by the company in a graph.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Analyse Revenue** | |
| **Use Case Id** | 10 | |
| **Priority** | 2 | |
| **Source** | Admin | |
|  |  | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This functions analyses the revenue made by the company. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Admin invokes the Analyse Revenue function  **Step 3:** The Admin confirms the order revenue details to be listed | **Step 2:** The system retrieves the details of the orders from the **Orders** file and displays the UI in alphabetical order of description.  **Step 4:** The system calculates the total revenue by getting the cost information from **Orders** file and adding them together.  **Step 5:** The system displays a graph showing the revenue made by the company.  **Step 6:** The system displays a confirmation message  **Step 7:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | The revenue made by the company is showed in a graph | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |



## 5.5 Manage Stock



### New Stock

This function allows the **Admin** to add new stock to the system.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **New Order** | |
| **Use Case Id** | 11 | |
| **Priority** | 1 | |
| **Source** | Admin | |
|  |  | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function records the details of a new stock. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Admin invokes the New Stock function  **Step 3:** Admin enters the details of a new stock   * Item type * Amount of goods * Added date   **Step 4:** The Admin confirms the stock details to be added | **Step 2:** The system displays the UI.  **Step 5:** The system Validates the stock details entered:   * All fields must be entered * The stock must have a valid Item type and a valid quantity (amount of goods) * Item type must be alphabetic and at least 15 characters long * Amount of goods must be numeric and no longer than 7 characters long * The date must have a valid format (dd-mm-yyyy)   **Step 6:** The system assigns an appropriate StockID for the new stock.  **Step 7:** The system saves the order details in the **Stock File**   * The type of the Stock added * StockID * The amount added * The date the stock was added (Day, Month, Year, Preferable Time)   **Step 8:** The system displays a confirmation message  **Step 9:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 5:** Invalid data is entered  **Step 6:** The system displays an appropriate error message |
| **Conclusions** | The Stock details are saved in the Stock file | |
| **Post conditions** |  | |
| **Business Rules** | A new stock order can only be added if it doesn’t already exist | |
| **Implementation Constraints** |  | |

### Replenish Stock

This function allows the **Admin** to choose what stock it needs to replenish/replace that is missing or been ordered by a **customer**.

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **New Order** | |
| **Use Case Id** | 12 | |
| **Priority** | 2 | |
| **Source** | Admin | |
|  |  | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function allows the admin to replace missing stock | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The Admin invokes the replenish stock function  **Step 3:** Admin chooses the desired stock and enters the quantity wanted   * Item type * Amount of goods   **Step 4:** The Admin confirms the stock details to be added | **Step 2:** The system displays the UI.  **Step 5:** The system Validates the stock details entered:   * All fields must be entered   Item type must be alphabetic and at least 15 characters long   * Amount of goods must be numeric and no longer than 7 characters long   **Step 6:** The system assigns an appropriate StockID for the new stock ordered.  **Step 7:** The system saves the stock details in the **Stock File**   * The type of the Item ordered * Stock ID * The amount ordered   **Step 8:** The system displays a confirmation message  **Step 9:** The system clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  | **Step 5:** Invalid data is entered  **Step 6:** The system displays an appropriate error message |
| **Conclusions** | The stock details are saved in the Stock file | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

# System Model

The following dataflow diagrams have been produced for the system:



## Level-0 DFD

Data flow diagram that shows the main functionality of the Wholesale sales processing system.

Orders

Stock

Customer

StockSYS

d

## Level-1 DFD

Data flow diagram that looks into detail what happens in the wholesale sales processing system.

Order details

Customer

Customer detailstails

Customer details

Customer details

Process Orders

P2

Manage Customers

P1

Order details

Customer details

Perfom Admin

P3

Customers file

D1

Order details

Order details

Orders file

D2

Stock Details

Orders Item file

D3

Stock Details

Manage Stock

P4

Stock file

D4

Stock Details

## Level-2 DFD (Process P1: Manage customers)

Customer

Customer details

Customer details

Customer details

Change Customer Details

P1.2

Record Customer Details

P1.1

Customers file

D1

Customer details

Customer details

Customer details

Remove Customer Details

P1.3

## Level-2 DFD (Process P2: Process Orders)

Customer

Order details

OrderID details

Change Order

P2.2

Order details

Order details

Add New Order

P2.1

Order Items details

Orders file

D2

Order details

Order details

Order details

Order details

Order details

Orders Item file

D3

Order Items details

Cancel Order

P2.4

Dispatch Order

P2.3

Order details

Order Items details

Order Receipt

Customer

## Level-2 DFD (Process P3: Perform Admin)

Order details

Admin

Orders file

D2

List daily deliveries

P3.1

Orders details

Order Items details

Order details

Analyse Orders

P3.2

Delivery detailsails

Orders Item file

D3

Order Items details

Customer details

Admin

Order Items details ails

Analyse revenue

P3.3

Customers file

D1

Revenue details

Admin

## Level-2 DFD (Process P3: Manage Stock)

Stock details

Stock details

Admin

Stock details

Stock details

Orders file

D2

Orders Item file

D3

Stock details

Stock details

New Stock

P4.1

Stock file

D4

Replenish Stock

P4.2

# Data Model (Class Diagram)

Brief introduction……



## Class Diagram

- Customer\_ID\*  
- First name  
- Second name  
- Address  
- D.O.B.  
- Phone number  
- Status  
- Username  
- Password  
- E-mail

Customers

- Amount   
- Item Type

Orders Items

- Order\_ID\*  
- Value  
- Date  
- Status

Orders

1

1..\*

1..\*

1..\*

1

1..\*

- Stock\_ID\*  
- Stock type  
- Amount  
- Date

Stock

## Relational Schema

CUSTOMERS (Customer\_ID\*, First\_name, Last\_name, Address, D.O.B., Phone Number, Status, E-mail address, Username, Password)  
  
ORDERS(Order\_ID\*,Order\_type, Amount, Delivery\_address, Delivery\_date, Status)  
  
ORDERS ITEMS(Amount, Item Type)  
  
STOCK(Stock\_ID\*, Stock\_type, Amount, Date)

## Database Schema

**Schema:** StockSYS

**Relation Customers**

Customer\_ID numeric (5) NOT NULL

First\_name char (15) NOT NULL

Last\_name char (15) NOT NULL

Address char (20) NOT NULL UNIQUE

Date\_Of\_Birth char (10)

Phone\_Number char (13) NOT NULL UNIQUE  
 Email\_address char (25) NOT NULL

Status char (1) NOT NULL

Username char (15) NOT NULL UNIQUE

Password char (15) NOT NULL UNIQUE

**Primary Key:** Customer\_ID

**Relation Stock** Stock\_ID numeric (3) NOT NULL  
 Stock\_Description char (7) NOT NULL

Cost\_Price numeric (5) NOT NULL

Sale\_Price numeric (5) NOT NULL

Stock\_Quantity numeric (8) NOT NULL  
 Stock\_Date char (10) NOT NULL

Status char(1) NOT NULL  
**Primary Key:** Stock\_ID

**Relation Orders**

Order\_ID numeric (5) NOT NULL

Order\_type char (10) NOT NULL

Amount numeric (8) NOT NULL

Value numeric (10) NOT NULL

Delivery\_address numeric (20) NOT NULL UNIQUE

Delivery\_date char (10) NOT NULL

Status char (1) NOT NULL

Customer\_ID numeric (5) NOT NULL

**Primary Key:** Order\_ID

**Foreign Key:** Customer\_ID References Customers  
  
**Relation Orders Items**

Amount numeric (8) NOT NULL

Item type char (7) NOT NULL

Order\_ID numeric (5) NOT NULL

**Foreign Key:** Order\_ID References Orders

# Conclusion

The wholesales sales processing system is a critical and a very complex system. It takes a lot of various activities in order for the system to work efficiently. It manages a lot of actions and includes different external entities and actors, such as, customers.  
  
Customers have the access to make orders from the business. If the order is made, the system validates if the ordered items are in stock and if they are, the order is processed.  
The system assigns an appropriate Order\_ID for the new order.   
When the customer is satisfied with the order, the customer proceeds to the check out and the system calculates the cost of the order items by using a formula (Quantity \* Cost) and adds to the running total of shopping cart.  
  
The customer has the ability to *change* or *cancel* the order at any time. If the order is cancelled by the admin or by the customer, its status automatically sets to ‘C’ (Cancelled).  
  
The main responsibility of the Admin is to ensure that the business product is up to the quality standards of the company.  
Only the admin is allowed to show or analyse all of the business orders made by the customer or the revenue made by the company. The data is mainly shown in form of graphs.  
  
If the stock is missing, it is the Admin’s responsibility to replenish/order the new stock required. As well as there is a choice for the Admin to add new stock to the system/business.