Requirements Engineering

Shipping System

Submitted By: Daniel Jameson

Computing with Software Development

Date Submitted: 09/12/2022

**Table of Contents**

[1. Introduction/overview 4](#_Toc121229587)

[2. Functional Components 5](#_Toc121229588)

[3. User Requirements 6](#_Toc121229589)

[3.1. Manage Customers 6](#_Toc121229590)

[3.2. Manage Games 6](#_Toc121229591)

[3.3. Process Orders 6](#_Toc121229592)

[4. System Requirements 7](#_Toc121229593)

[4.1. System Level Use Case Diagram 7](#_Toc121229594)

[4.2. Manage Customers 8](#_Toc121229595)

[4.2.1. Register Customer 9](#_Toc121229596)

[4.2.2. Update Customer 11](#_Toc121229597)

[4.2.3. Deregister Customer 13](#_Toc121229598)

[4.3. Manage Games 15](#_Toc121229599)

[4.3.1. Create Game 15](#_Toc121229600)

[4.3.2. Update Game 18](#_Toc121229601)

[4.3.3. Deregister Game 21](#_Toc121229602)

[4.4. Process Orders 23](#_Toc121229603)

[4.4.1. Place Order 23](#_Toc121229604)

[4.4.2. Cancel Order 24](#_Toc121229605)

[4.4.3. Return Game 27](#_Toc121229606)

[4.4.4. Ship Order 29](#_Toc121229607)

[4.4.5. Issue Invoice 31](#_Toc121229608)

[4.4.6. Record Payment 33](#_Toc121229609)

[4.5. Administrative Functions 35](#_Toc121229610)

[4.5.1. Yearly Revenue Analysis 35](#_Toc121229611)

[4.5.2. Yearly Stock Analysis 37](#_Toc121229612)

[5. System Model 40](#_Toc121229613)

[5.1. Level-0 DFD 40](#_Toc121229614)

[5.2. Level-1 DFD 41](#_Toc121229615)

[5.3. Level-2 DFD (Process P1: Manage Customers) 42](#_Toc121229616)

[5.4. Level-2 DFD (Process P2: Manage Games) 43](#_Toc121229617)

[5.5. Level-2 DFD (Process P3: Process Orders) 44](#_Toc121229618)

[5.6. Level-2 DFD (Process P4: Administrative Functions) 45](#_Toc121229619)

[6. Data Model (Class Diagram) 46](#_Toc121229620)

[6.1. Class Diagram 46](#_Toc121229621)

[6.2. Relational Schema 46](#_Toc121229622)

[6.3. Database Schema 47](#_Toc121229623)

[7. Conclusion 48](#_Toc121229624)

# Introduction/overview

This is the System Requirements for a game distribution company. This system provides customers with the ability to manage their account, and all their details on the system. It will have facilities for the customer to manage their orders and provide relevant staff with the ability to manage products on the system and calculate performance metrics for the business.

# Functional Components

This represents the high-level use cases for ShippingSYS. It describes the modules and the functions within them in a short clear way.

# User Requirements

This System will track customer details, services payments and financial information.

## Manage Customers

* + 1. Register a customers details on the system.
    2. Update customers details edits information already given to the system.
    3. De-register a customer changes customer status to deregistered.

## Manage Games

* + 1. Create Game will place a game on the system.
    2. Update Game will change information on the system for a game.
    3. De-register Game will set a games status to deregistered.

## Process Orders

* + 1. Place Order allows customers to buy a game.
    2. Cancel Game will cancel a placed order.
    3. Return Game will let a customer return a shipped order.
    4. Ship Game will send a game to a customer.
    5. Send Invoices will send invoices to customers.
    6. Record Payment.
  1. **Administrative Functions**

3.4.1 Yearly Revenue Analysis will Produce yearly revenue report.

3.4.2 Yearly Stock Analysis will produce a yearly stock analysis.

# System Requirements

Module 1 - Customers. This module will provide customers with an interface to interact with the software system. It will provide the ability to register with the business, update a customers details and possibly de-register them depending on conditions.

Module 2 – Stock. This module will provide Management with the ability to manage their own stock. Its functions include the ability to create an entry for a stock item and store its details to the database, update item details and delete an entry for an item.

Module 3 – Orders. This module will provide customers with an interface to handle stock orders.

Module 4 – Admin. This module will provide the management with metrics to assess how the business is doing and adjust accordingly.

## System Level Use Case Diagram

The following system level use case diagram illustrates the high-level system requirements.

ShippingSYS

Manager

Customer

## Manage Customers

This module will handle how customers and management might possibly need to interact with the customer registration system. For the business to be able to facilitate transactions between the customer and the marketplace, ShippingSYS must have a section where a customer can handle account details.

### Register Customer

This function will provide the customer with the facility to register with the business. When registered, the customer is then able to buy from the business.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Register Customer | |
| **Use Case Id** | CST01 | |
| **Priority** | Rank 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | Provides potential customers with the ability to register an account. | |
| **Preconditions** | Must not be already existing customer. | |
| **Trigger** | Accessed from main menu. | |
| **Expected Scenario** | **Customer** | **System** |
|  | **Step 1:** Customer clicks register customer.  **Step 3:** Customer enters registration information:   * Surname. Varchar20 * Forename. Varchar 20 * Street. Varchar 20 * Town. Varchar 20 * County. * Country. * EIRcode. Varchar 8 * Phone number. Varchar 10 * E-mail address. Varchar 20 * Card Number – Numeric 16 | **Step 2 :** Displays User Interface  **Step 4:** Validate Data Entered:   * All data must be entered * Surname must be max length of 20 chars. * Forename must be max length of 20 chars. * Street must contain a max length of 20 chars. * Town must contain a max length of 20 chars. * County. * EIRcode must be valid format. * Password must be valid format. * Phone number must be 10 numeric characters. * Email address must be valid format. * Card number must be 16 digits.   **Step 5**: On confirmation, assign next CustID.  **Step 6**: Set status to registered. (‘R’)  **Step 7:** Save data to customers file:   * CustID * Surname * Forename * Street. * Town. * County. * EIRCode. * Phone. * E-mail address. * Card number. * Status   **Step 8:** Display Confirmation Message.  **Step 9:** System resets user interface. |
| **Alternate Scenarios** | **Customer** | **System** |
| **Invalid Data entered** |  | **Step 4:** Invalid data detected.  **Step 5:** Display an appropriate error message.  **Step 6:** Return to step 3. |
| **Conclusions** | Customer account has been created. | |
| **Post conditions** | Customer can now place orders on system. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Customer

This function provides a customer or staff with the facility to update their own details with regards to their account. E.g. changing email address, password etc.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Customer | |
| **Use Case Id** | CST02 | |
| **Priority** | Rank 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** | Manager | |
| **Description** | Customer wants to update their already existing data. | |
| **Preconditions** | Customer must already have an account and must be logged in. | |
| **Trigger** | Customer clicks on edit account information. | |
| **Expected Scenario** | **Customer** | **System** |
|  | **Step 1:** Customer opens update/delete account.  **Step 2:** Customer enters their name and selects their account from a list of accounts.  **Step 5:** Customer edits the required information:   * Surname. Varchar 20 * Forename. Varchar 20 * Street. Varchar 20 * Town. Varchar 20 * County. * EIRcode. Varchar 8 * Password. Varchar 20 * Phone number. Varchar 10 * E-mail address. Varchar 20 * Card information – Numeric 16 | **Step 3:** Retrieve customer details from the Customers file.  **Step 4**: Displays Customer Information.  **Step 6:** Validate Data Entered:   * All data must be entered. * Data entered must match account data. * Surname must be max length of 20 chars. * Forename must be max length of 20 chars. * Street must contain a max length of 20 chars. * Town must contain a max length of 20 chars. * County. * EIRcode must have 8 characters. * Password must be valid format. * Phone number must be 10 numeric characters. * Email address must be valid format. * Validate card information   **Step 7:** System updates information in the Customers file:   * Surname. * Forename. * Street. * Town. * County. * EIRcode. * Password. * Phone number. * E-mail address. * Card information.   **Step 8:** Display Confirmation Message.  **Step 9:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 5:** Invalid data detected  **Step 6:** Display an appropriate error message.  **Step 7:**  Return to step 5. |
| **Conclusions** | Customer details will update successfully. | |
| **Post conditions** | Information must be valid before exiting. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Deregister Customer

This field will provide the user with a facility to de-register their account with the business. Management may also wish to de-register an account in the case of non-payment or violation of other business rules.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Deregister Customer | |
| **Use Case Id** | CST03 | |
| **Priority** | Rank 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** | Manager | |
| **Description** | This function sets the status of a Customer to deregistered. | |
| **Preconditions** | Customer must have logged in. | |
| **Trigger** | Accessed from Account page. | |
| **Expected Scenario** | **Customer** | **System** |
|  | **Step 1:** Customer clicks update/delete account.  **Step 3:** Customer enters their name and selects their account from a list of accounts.  **Step 6:** Asks user to confirm account status change. | **Step 2:** Displays User Interface  **Step 4:** System retrieves information from Customers file.  **Step 5:** Validate Data Entered:   * Forename must be entered.   **Step 7:** System changes account status to ‘D’.  **Step 8:** Display Confirmation Message.  **Step 9:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 5:** Invalid data detected  **Step 6:** Display an appropriate error message.  **Step 7:**  Return to step 4 then ask user to update data. |
| **Conclusions** | Customer account will deregister successfully. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Games

This module will provide management with the ability to control their stock items. It will provide the ability to create a new entry for a stock item, update the details on a stock item and the ability to remove old stock from the website but not database.

### Create Game

This function lets management make an entry for a game on the system.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Create Game | |
| **Use Case Id** | GME1 | |
| **Priority** | Rank 2 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function will provide management with the ability to create a stock item to be entered to the database. | |
| **Preconditions** | Must not match already existing game entry. | |
| **Trigger** | Manager navigates from main page. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager Clicks on Register Game.  **Step 3:** Manager Enters new game details   * Title – Varchar 20 * Developer – Varchar 20 * Publisher – Varchar 20 * Genre – Varchar 20 * Price – Numeric 3 | **Step 2:** Displays User Interface  **Step 4:** Validate Data Entered:   * Title must be less than 20 chars. * Developer must be less than 20 chars. * Publisher must be 20 chars. * Genre must be less than 20 characters. * Description max 50 chars. * BuyPrice must be valid number. * Sale Price must be valid number. * Quantity must be valid integer.   **Step 5:** Assign next GameID.  **Step 6:** Set status to ‘R’.  **Step 7:** Save Game details to games file.  **Step 8:** Display Confirmation of entry.  **Step 9:** System resets user interface. |
| **Alternate Scenarios** | **Manager** | **System** |
| **Invalid Data entered** |  | **Step 4:** Invalid data detected  **Step 5:** Display an appropriate error message and clear invalid data fields.  **Step 6:** Return to step 4. |
| **Conclusions** | Game gets added to Games file. | |
| **Post conditions** | Item must not already exist. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Game

This function provides the manager with the means to update details for a game.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Game | |
| **Use Case Id** | GME2 | |
| **Priority** | Rank 2 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function will provide the user with the ability to update a games details in the database. | |
| **Preconditions** | Item must already exist in database. | |
| **Trigger** | Manager navigates here from Stock page. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager clicks on update game.  **Step 3:** Manager enters the name of the game they are trying to update.  **Step 6:** Manager enters the game details they wish to update.   * Title * Developer * Publisher * Genre * Description * Sale Price * Buying Price * Quantity * Status | **Step 2:** Displays User Interface  **Step 4:** Validate Data Entered:   * Title must be max 20 characters. * Title must not be blank.   **Step 5:** System retrieves game data from games file.  **Step 7:** Validate data entered by manager.   * Title must not be blank. * Developer must not be blank. * Publisher must not be blank. * Genre must not be blank. * Description must not be blank. * Sale Price must not be negative number. * Buying Price must not be negative number. * Quantity must not be negative. * Status can only be ‘R’ or ‘D’.   **Step 8:** System updates game data in accordance with what was entered.  **Step 8:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 6:** Invalid data detected  **Step 7:** Display an appropriate error message. Clear appropriate fields.  **Step 8:** Return to step 6. |
| **Conclusions** | Stock items details get updated. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Deregister Game

This function sets a stock items status to deregistered.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Deregister Game | |
| **Use Case Id** | GME3 | |
| **Priority** | Rank 2 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function will provide the user with the ability to deregister a game from the sales page. | |
| **Preconditions** | Item must already exist in database. | |
| **Trigger** | Manager navigates here from main menu. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager clicks update/delete game.  **Step 3:** Manager Enters game Title.  **Step 6:** Manager clicks on Deregister button. | **Step 2:** Displays User Interface  **Step 4:** Validate Title Entered:   * Title must not be blank.   **Step 5:** System retrieves game information from games file.  **Step 7:** Set Games status to ‘D’.  **Step 8:** Display Confirmation of status change.  **Step 9:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 4:** Invalid data detected  **Step 5:** Display an appropriate error message and clear fields.  **Step 6:** Return to step 4. |
| **Conclusions** | Game status changes to deregistered. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Process Orders

This module handles ordering by the customer and shipping. It provides a facility to place, review, and cancel order as well as handles invoices and logistics.

### Place Order

Place Order allows a user to purchase an item from shop.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Place Order | |
| **Use Case Id** | ORD01 | |
| **Priority** | Rank 3 | |
| **Source** | Manager | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function allows a user to place an order on the system. | |
| **Preconditions** | User has added item to the shopping cart. | |
| **Trigger** | Customer clicks on purchase items from accounts. | |
| **Expected Scenario** | **Customer** | **System** |
|  | **Step 1:** Customer clicks on Place Order.  **Step 4:** Customer enters their login details.  **Step 5:** Customer enters name of game to search for or selects all.  **Step 8:** Customer clicks on place order. | **Step 2:** Retrieves next available Order ID information from Orders file.  **Step 3:** Displays User Interface  **Step 6:** Selected **i**tem is added to shopping cart.  **Step 7:** System makes sure the amount of games in Cart does not exceed quantity in database.  **Step 9:** On successful order placement, the customer is informed of its success.  **Step 10:** System adds order information to Orders file. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 4:** Customer enters invalid login information.  **Step 5:** System informs user they need to fill out login information again.  **Step 6:** Clear login and password fields. |
| **Conclusions** | Customer completes purchase order. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Cancel Order

This section allows a customer to view an existing order and cancel it if necessary.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Cancel Order | |
| **Use Case Id** | ORD02 | |
| **Priority** | Rank 3 | |
| **Source** | Manager | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function allows a customer to cancel an order that is being processed. | |
| **Preconditions** | Goods must be paid for but not yet shipped. | |
| **Trigger** | Customer clicks on cancel order from orders page. | |
| **Expected Scenario** | **Customer** | **System** |
|  | **Step 1:** Customer opens the orders page.  **Step 4:** Customer enters number of order to be cancelled.  **Step 6:** Customer clicks button to confirm they are returning order. | **Step 2:** Retrieves information from Orders file.  **Step 3:** Displays User Interface  **Step 5:** Validate return number.   * Must match an existing transaction on same account. * Orders status must not be in shipping.   **Step 7:**  System changes orders status to ‘C’.  **Step 8:** System sends return information to Orders file and refunds payment.  **Step 9:** Display Confirmation Message.  **Step 10:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 4:** Invalid order number entered.  **Step 5:** Display error message.  **Step 6:** Return to step 4. |
| **Conclusions** | Customer successfully cancels an existing order. | |
| **Post conditions** |  | |
| **Business Rules** | Business has a 6 month return policy. Orders can only be cancelled if they are in transit or have not been assembled. | |
| **Implementation Constraints** |  | |

### Return Order

This function provides a customer with the means to return an already purchased and shipped item.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Return Order | |
| **Use Case Id** | ORD03 | |
| **Priority** | Rank 3 | |
| **Source** | Manager | |
| **Primary Business Actor** | Customer | |
| **Other Participating Actors** |  | |
| **Description** | This function allows a customer to return an order that has already been received. | |
| **Preconditions** | User must have a valid existing order. | |
| **Trigger** | Customer clicks on return game from manage orders page. | |
| **Expected Scenario** | **Customer** | **System** |
|  | **Step 1:** Customer opens the Manage Orders page.  **Step 3:** Customer enters login information.  **Step 4:** Customer clicks login button.  **Step 7:** Customer selects their order from a list of orders.  **Step 8:** Customer click Return order from a list of orders. | **Step 2:** Displays User Interface.  **Step 5:** Validate login information.   * Email must be valid email. * Password must match password that is associated with email.   **Step 6:** Customer is informed of successful login.  **Step 9:** System updates transaction status to ‘Returned’ on Orders file.  **Step 10:** Display Confirmation Message.  **Step 11:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  | **Step 4:** Invalid login information entered.  **Step 5:** Clear text fields and inform user.  **Step 6:** Return to step 4. |
| **Conclusions** | Customer successfully returns order. | |
| **Post conditions** |  | |
| **Business Rules** | Business has a 6 month return policy. Returns must not be over 6 months old. | |
| **Implementation Constraints** |  | |

### Ship Order

This section allows a manager to ship an existing order.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Ship Order | |
| **Use Case Id** | ORD04 | |
| **Priority** | Rank 3 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function provides management with the ability to generate shipping information and attach them to order. | |
| **Preconditions** | Goods must be paid for but must occur before shipping. | |
| **Trigger** | Manager clicks on ship order from Management Section in Orders. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager opens the Order shipping page.  **Step 3:** Manager clicks for a list of orders that need assembly.  **Step 6:** Worker clicks on the items in the order.  **Step 8:** Worker confirms assembly of order. | **Step 2:** Displays User Interface.  **Step 4:** System returns orders that need to be assembled.  **Step 5:** System generates information on each item in the order for assembly.  **Step 7:**  System prompts user if they have gathered all items in the order.  **Step 9:** System updates status to ‘In Transit’.  **Step 10:** Display Confirmation Message. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  |  |
| **Conclusions** | Customer successfully cancels an existing order. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Issue Invoice

This section provides management with the ability to issue an invoice for a purchased order.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Issue Invoice | |
| **Use Case Id** | ORD05 | |
| **Priority** | Rank 3 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function allows management to issue an electronic invoice for an order. | |
| **Preconditions** | Must occur on a shipped transaction. | |
| **Trigger** | Manager clicks on Invoices from Management Section in Orders. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager opens issue invoices page.  **Step 5:** Manager clicks order that needs receipt generated.  **Step 7:** Manager clicks on the order details generated in the previous step. | **Step 2:** System retrieves information from Orders file.  **Step 3:** Displays User Interface  **Step 4:** System displays orders that need invoices generated.  **Step 6:** System Generates info to go into receipt.  **Step 8:** System generates a receipt to an external file.  **Step 9:** System displays confirmation to the user. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  |  |
| **Conclusions** | Manager successfully issues an invoice for an order. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Administrative Functions

This section handles performance metrics, generating the associated information for business purposes.

### Yearly Revenue Analysis

This function allows management to generate and analyse revenue of the business.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Yearly Revenue Analysis | |
| **Use Case Id** | ADM01 | |
| **Priority** | Rank 4 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function allows management to generate a yearly revenue analysis. | |
| **Preconditions** | Must have 12 months available to analyse. | |
| **Trigger** | Manager clicks on function from Analysis page. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager opens Analysis page.  **Step 4:** Manager clicks button to generate yearly revenue statement.  **Step 7:** Manager closes function. | **Step 2:** Retrieves information from Orders file.  **Step 3:** Displays User Interface  **Step 5:** System generates a statement with information.   * Yearly profit. * Yearly expenditure. * Net income. * Asset value.   **Step 6:** Display information to user.  **Step 8:** System resets user interface. |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  |  |
| **Conclusions** | Manager can generate a yearly revenue analysis. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Yearly Stock Analysis

This section allows management to analyse a stock items performance in-depth, informing future decisions with regards to stocking the business.

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Yearly Stock Analysis | |
| **Use Case Id** | ADM02 | |
| **Priority** | Rank 4 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function gives an analysis on sales of individual stock items. | |
| **Preconditions** |  | |
| **Trigger** | Manager accesses from Administration section. | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager requests yearly stock analysis.  **Step 4:** Manager enters request for required information by type.   * Sort by Publisher. * Sort by Title. * Sort by Developer.   **Step 7:** Manager clicks close. | **Step 2:** System retrieves information from Orders file.  **Step 3:** Display User Interface.  **Step 5:** System produces requested information.   * Amount sold. * Time and date of sale. * Destination of sale. * Sales by Publisher. * Sales by Developer. * Sales by Title.   **Step 6:** Display information to user.  **Step 8:** Resets user interface |
| **Alternate Scenarios** | **Actor** | **System** |
| **Invalid Data entered** |  |  |
| **Conclusions** | Manager generates a stock analysis report. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD

ShippingSYS

Game Details

Invoice

## Level-1 DFD

P1

P2

Manage Games

Manage Customers

Customer Details

Game Details

Customer Details

D1 Customers File

Game Details

D2 Games File

Game Details

Order Details

D3 Orders File

Financial Details

P4

P3

Admin

Process Orders

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

Update Customer

P1.2

P1.1

Create Customer

Customer Details

Customer Details

Customer Details

D1 Customers File

Customer Details

P1.3

De-register Customer

## Level-2 DFD (Process P2: Manage Games)

Update Game

P2.2

De-register Game

P2.3

Game Details

Game Details

Create Game

P2.1

Game Details

D2 Games File

Game Details

## Level-2 DFD (Process P3: Process Orders)

Order Details

Order Details

Order Status

Order Status

Order Status

Order Details

D3 Orders File

Customer Details

D1 Customers File

Issue Invoice

P3.5

Ship Order

P3.4

Return Game

P3.3

Cancel Order

P3.2

Place Order

P3.1

Record Payment

P3.6

## Level-2 DFD (Process P4: Administrative Functions)

Financial Information

Financial Information

Yearly Revenue Analysis

P4.1

Yearly Stock Analysis

P4.2

D3 Orders File

# Data Model (Class Diagram)

The following information represents the database structure for ShippingSYS. It will contain the necessary information for the employees to construct a database for the project including the tables for customers, orders, games and administrative functions.

## Class Diagram

|  |
| --- |
| Orders |
| Order\_Id: Int(5) <pk>  Order\_Date: Date  Total: Decimal(6,2)  Status: Varchar(15)  Cust\_Id: Numeric(5) <fk> |

|  |
| --- |
| Customers |
| Cust\_Id: Numeric(5) <pk>  Surname: Varchar2(20)  Forename: Varchar2(20)  Town: Varchar2(20)  EIRcode: Char(8)  Password: Varchar2(20)  Phone: Char(10)  Email: Varchar2(50)  CardNumber: Char(16)  Status: Varchar2(15)  County: Varchar2(9) |

1..1

Has order

1..1

1..\*

Has Games

|  |
| --- |
| Games |
| Game\_Id: Int(5) <pk>  Title: Varchar2(20)  Developer: Varchar2(20)  Publisher: Varchar2(20)  Genre: Varchar2(10)  Description: Varchar2(50)  BuyPrice: Numeric(4,2)  SalePrice: Numeric(4,2)  Quantity: Numeric(3)  Status: Varchar2(15) |

|  |
| --- |
| OrderItems  \*..1 |
| Order\_Id: Numeric(5) <fk>  Cost : Numeric(4,2)  Game\_Id: Numeric(5) <fk> |

1..1

Has Game

1..1

## Relational Schema

**Customers** (CustID, Surname, Forename, Town, EIRcode, Password, Phone, Email, CardNumber, Status, County)

**Games** (GameID, Title, Developer, Publisher, Genre, Description , BuyPrice, Saleprice, Quantity, Status)

**Orders** (OrderID, Date, Cost, Status, Cust\_Id)

**Order\_Items** (Order\_Id, Cost, Game\_Id)

## Database Schema

**Relation Customers**

CustID Numeric(5) PRIMARY KEY

Surname Varchar2(20) NOT NULL

Forename Varchar2(20) NOT NULL

Town Varchar2(20) NOT NULL

County Varchar2(9) NOT NULL

EIRcode Varchar2(8) NOT NULL

Email Varchar2(20) NOT NULL UNIQUE

Phone Varchar2(10) NOT NULL

CardNumber Numeric (16)

Status Char(1)

Primary Key CustID;

**Relation Games**

GameID Numeric(5) PRIMARY KEY

Developer Varchar2(20) NOT NULL,

Publisher Varchar2(20) NOT NULL,

Genre Varchar2(20),

Description Varchar2(50),

SalePrice Numeric (4,2),

BuyPrice Numeric (4, 2),

Quantity Numeric (5),

Status Char(1),

Primary Key GameID

**Relation Orders**

OrderID Numeric(5) PRIMARY KEY,

Date Date NOT NULL,

Cost Numeric (4,2),

Status Char(1) NOT NULL,

CustID NOT NULL,

Primary Key OrderID

Foreign Key CustID on Customers

**Relation Order\_Items**

Order\_Id Numeric(5),

Cost Numeric(6,2),

Game\_Id Numeric(5),

FOREIGN KEY OrderID references Orders,

FOREIGN KEY CustId References Customers

# Conclusion

This online game ordering and shipping system provides its users with the ability to create and maintain their accounts with ShippingSYS and, from there, allows the customer to manage their orders on the system. It provides management with the ability to manage the products on sale, and to generate performance metrics for the system.