Requirements Specification Document

Inventory Management System

Inventoria

March 3rd, 2020

Table of contents

List of Tables

List of Figures

Revision History

Name	Date	Reason for Changes	RSD Version
Team	10-Feb-2020	Requirements document 1.0	0.0
Team	14-Feb-2020	Modified document to conform to marker and client feedback.	0.1
Amaan	14-Feb-2020	Added use cases for sections 3.1 and 3.5.	0.2
Cameron	14-Feb-2020	Added use cases for section 3.6.	0.3
Jonathan	14-Feb-2020	Added use cases for section 3.2.	0.4
Omar	14-Feb-2020	Added use cases for section 3.4.	0.5
Elizabeth	14-Feb-2020	Added use cases for section 3.3.	0.6
Cameron	16-Feb-2020	Review of Requirements Specification Document	0.7
Siddhant	17-Feb-2020	Added use case diagram.	0.8
Team	17-Feb-2020	Draft of the Requirements Specification Document.	0.9
Team	18-Feb-2020	The final version of the Requirements Specification Document 1.0.	1.0
Team	25-Feb-2020	Modified document to conform to client feedback.	1.1
		Added the context diagram, level 1, and level 2.	1.2
Amaan and Siddhant	02-March-2020	Added ER diagram and data dictionary.	1.3
Team	02-March-2020	The draft version of the Requirements Specification Document 2.0.	1.4
Team	02-March-2020	The final version of the Requirements Specification Document 2.0.	2.0

1 Introduction

1.1 Purpose

The purpose of the following Requirements Specification Document (RSD 2.0) is to describe the functionalities for Kevin's Convenience Store's Inventory Management System. The client organization, Kevin's Convenience Store found that their employees had a hard time maintaining and tracking product information, keep up with the expiration dates and find any discrepancies using spreadsheets. This system is a solution to eliminate the need for physical spreadsheets by having all product information digitized. The functionalities in this document represent the main features of the Inventory Management System's methods of adding and removing inventory, managing orders, and user interaction. It will be integrated with existing systems and processes in place.

1.2 Project Scope

The scope of the solution system will cover the software used to manage inventory, new hardware needed to collect and interpret data, and the interactions required between the current systems and the new system. This resulting system is intended to be a simpler and cheaper alternative to the current, larger and more expensive systems used in major grocery store chains. The desired outcome for the solution system will increase inventory accuracy, lower product losses, and optimize the ordering of products.

1.3 Glossary of Terms

Term	Definition
Administrator	A user with special access due to their responsibilities, privileges, or position.
DFD	Data Flow Diagram, a diagram with the purpose of showing the flow of data in and outside the system.
Discrepancy	An inconsistency arising from the comparison between two or more facts.
ER Diagram	Entity Relationship Diagram, a diagram to define the relationship between entities within the system.
Inventory	The number of items of a given product that is currently at the store location.
Item	A single instance of a product.
Order summary	A PDF file that contains the list of products which is used to place orders to distributors

POS	Point of Sales, the machine at which the merchant conducts a transaction with a customer.
Product	An article or substance that is manufactured or refined for sale.
SKU	The number assigned to a product by a retail store to identify the price, product options and manufacturer of the merchandise.
UI	User Interface, space where interactions between humans and machines.

1.4 References

[1] *Request for Proposal*, Kevins's Convenience Corner, 2020 [Online]. Available: RFP - Inventory System.pdf. [Accessed: 25- Jan- 2020].

[2] "Kevin's Convenience Corner", 2020. [Online]. Available: https://sites.google.com/view/kevins-convenience-corner/home. [Accessed: 25- Feb- 2020].

[3] *Requirements Specification Document 1.1,* Kevins's Convenience Corner, 2020 [Online]. Available: RSD1.1Group9.pdf. [Accessed: 25- Feb- 2020].

1.5 Overview

This document, RSD 2.0, is the second revision of the Requirements Specification Document written for Kevin's Convenience Corner's Inventory Management System. The document begins by providing an overall description of the system. This section helps to understand the main features the system will have, the targeted users and the operating environment. This section also contains some of the assumptions that have been made and the dependencies for the solution system. The next section describes the six main system features with their corresponding functional requirements and use cases. Each feature has also been given a priority. In the following section, the External Interface Requirements section defines the different interfaces required to interact with the system. Other non-functional requirements follow in the next section, defining requirements for performance, security and software quality. The sixth section showcases data flow diagrams for the Inventory Management System at three levels. Finally, the entity-relationship diagram and the corresponding data dictionary are included to visualize the various relationships of the Inventory Management System.

2 Overall Description

This section gives a brief overview of the Inventory Management System's requirements specification document. The section provides information regarding the business goals for the system and the information about the problem domain.

2.1 Product Perspective

To improve and digitize the existing inventory records the organization management system, a convenience store inventory management system is needed. It can be integrated with existing systems and processes in place to synchronize the various systems of the store. This will synchronize all product information and their quantities promptly, reducing the frequency of product shortages and the number of products that expire before they are sold.

2.2 Product Features

The solution system is a web-based inventory subsystem. Users can use it to add inventory or remove damaged and expired goods. The system will record the products' information, including shelf life, expiration date reminders, SKU, and track the total quantity. Order summary lists and shipping arrival dates will be recorded in the system. When the expiry date of a product is approaching, or when the inventory of a product is low, the system will notify the user and provide them with the relevant notification.

2.3 User Classes and Characteristics

Two types of users interact with the system: support staff and store managers. Each user has different privileges of the system based on functional and administrative requirements.

2.3.1 Support Staff

The support staff will be able to use the web application to check the information of products, scan-in inventory or scan-out expired/damaged products using the products' barcode. Actions such as scan-in and scan-out that modifies the overall inventory system require administrator approval. As a security measure, the support staff has the least privileges.

2.3.2 Store Managers (Administrator)

The store managers will interact with the system with full administrative privileges, interacting with the web portal to manage the overall system. Store managers can manually modify inventory data, place orders to distributors, manipulate expiry dates, as well as manage account privileges. Store managers also have the responsibility of approving or denying modification or deletion requests made by support staff.

2.4 Operating Environment

This system is web-based and supports accessibility through desktop and mobile devices. The system will contain iOS and android functionalities to increase usability across mobile devices. The web and mobile user interfaces share the same UI functionality and database. The mobile application will have a dynamically sized UI to increase the support of mobile users. Product

information is gathered by scanning the product barcode using a scanner which is uploaded to the database upon completion.

2.5 Design and Implementation Constraints

The digital system made by the developers would be subject to the following constraints:

1. The system must be accessible to users with physical disabilities and color blindness. Therefore, the system UI must employ the use of symbols, larger buttons, a minimalistic interface, as well as avoid using color to convey actions or context.
2. The system will be accessible on both mobile devices and desktop computers. This means that the website will be mobile friendly and scale down effectively along with its interface and menus to fit the device's dimensions.
3. The system must make use of the software infrastructure already in place at the store including any established databases or tools.
4. The solution system must integrate with the current POS system.
5. Inventoria would be responsible for maintaining the designed software following deployment. This includes performing regular maintenance checks, security updates, and software updates.

2.6 Assumptions and dependencies

2.6.1 Assumptions

For the scope of this project, the following assumptions are made:

1. The expiry date of a product, except for fresh produce, would be embedded in the barcode of that particular product.
2. The store staff would be responsible for dealing with products nearing their expiration date and in the event of discrepancies arising in data.
3. Each product has one unique barcode which is equivalent throughout duplicates of that product.
4. Kevin's Convenience Store would be responsible for any direct communication with the distributors.

2.6.2 Dependencies

The system would have the following dependencies:

1. All Kevin's Convenience Store branches must have a stable internet connection to maintain accurate and real-time data distribution between the systems.
2. The system would utilize a new database to store the current inventory and associated product information.
3. The system would utilize the current POS system to update the inventory.

3 System Features

This section describes the six main features of the system which are relevant to its primary operations. This section contains uses-cases for each feature, as shown in **Figure 3.1** below.

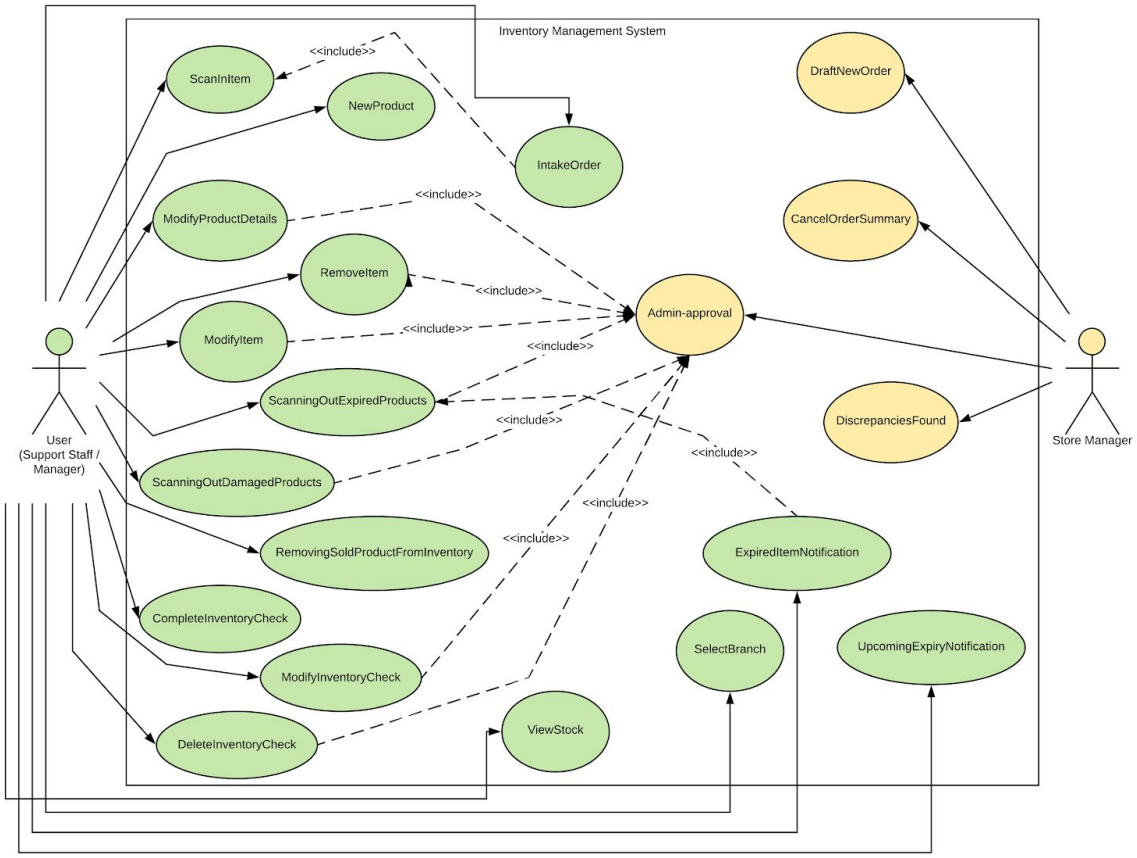

Figure 3.1. Use case diagram.

3.1 Scan-in Inventory

3.1.1 Description and Priority

This feature will allow users to scan-in inventory that was previously ordered and received into the system for tracking. Users will be able to scan or select a product and enter the number of products being added. The barcode of the product will contain the expiry dates. If the expiry date is unavailable, the user will manually enter the batch expiry date. This feature is the entry point for all tracked products and is of **high priority**.

3.1.2 Functional Requirements

REQ-1-1: The user must be able to add products by scanning the product barcode.

REQ-1-2: The user must be able to add products by finding them in a categorized product list, by sorting them in alphabetical order, or by searching the product list.

REQ-1-3: The number of products being added must be specified by the user.

REQ-1-4: The user must specify the category of the product, the SKU, and the expiry date notification timeframe if the product has not been entered before.

REQ-1-5: The user must enter an expiry date for perishables if not obtained from the barcode.

REQ-1-6: The support staff user must request approval from the store manager to allow the modification of inventory products.

REQ-1-7: The store manager must be able to approve or decline support staff requests to modify or remove inventory and products.

3.1.3 Use Cases

UC-1-1: ScanInItem
ID: UC-1-1
Description: The user can enter the product into the inventory management system.
Actor(s): User (support staff or store manager).
Preconditions: None.
Main Flow: 1. The user selects “add item”. 2. The user scans the barcode of the item. 3. If the item is a perishable or a dairy product then 3.1. If the expiry date is retrieved from the barcode then 3.1.1. The expiry date is entered into the item information. 3.2. Else 3.2.1. The user enters the expiry date. 4. The item quantity is selected. 5. The user confirms the new item entry.
Postconditions: The inventory is added.
Alternative Flow(s):

2 b) NewProduct
5 b) Cancel

AF-1-1: NewProduct
ID: AF-1-1
Description: The products category, SKU, name, description, and expiry notification length (if applicable) are entered by the user.
Actor(s): User (support staff or store manager).
Preconditions: The product doesn't exist in the system.
Alternate Flow: 1. The user scans the barcode of the product. 2. The user enters the product's name and description. 3. The system generates an SKU for the product. 4. The user selects the product category. 5. The user selects the product's expiry notification length. 6. The user "confirms" the product's details.
Postconditions: The product is created.
Alternative Flow(s): 5 b) Cancel current inventory transaction

UC-1-2: ModifyProductDetails
ID: UC-1-2
Description: The product's category, SKU, name, description, and expiry notification length (if applicable) can be modified by the user.
Actor(s): User (support staff or store manager).
Preconditions: The product exists in the system.
Main Flow: 1. The user can find the product in the "inventory tab". 2. The product can be found in alphabetical order, by category, by scanning the barcode or using the search bar. 3. The user selects the product and navigates to the "product details" tab. 4. The user selects the product detail they wish to modify and enters the change.

5. If the user wants to edit the SKU then
    5.1. The SKU must be unique.
    5.2. While the SKU is not unique
        5.2.1. The user is unable to change the SKU to the entered value.
6. The user “confirms” the changes.
7. If the user is a support staff then
    7.1. Include(AdminApproval).

Postconditions: The product’s information is updated.

Alternative Flow(s):
6 b) Cancel

UC-1-3: RemoveItem
ID: UC-1-3
Description: Users can remove items that have been added.
Actor(s): User (support staff or store manager).
Preconditions: The item has been added.
Main Flow: 1. The user can find the product in the “inventory tab”. 2. The product can be found in alphabetical order, by category, by scanning the barcode or using the search bar. 3. The user selects the product and navigates to the “inventory history” tab. 4. The user selects the inventory entry that they want to modify and selects discard. 5. The user “confirms” the entry. 6. If the user is a support staff then 6.1. Include(AdminApproval).
Postconditions: The item entry has been updated.
Alternative Flow(s): 5 b) Cancel

UC-1-4: ModifyItem
ID: UC-1-4
Description: Users can modify the details of inventory that have been added.

Actor(s): User (support staff or store manager).

Preconditions: The product exists in the system.

Main Flow:

1. The user can find the product in the “inventory tab”.
2. The product can be found in alphabetical order, by category, by scanning the barcode or using the search bar.
3. The user selects the product and navigates to the “inventory history” tab which shows the current product entries, the intaken orders, the expired and damaged items that were removed.
4. The user selects the inventory entry that they want to modify.
5. The user may modify the item quantity.
6. If the category contains an expiry date then
    6.1. The user may enter a modified expiry date.
7. If the user made modifications then
    7.1. The user can “confirm” the changes.
8. If the user is a support staff then
    8.1. Include(AdminApproval).

Postconditions: The inventory entry has been updated.

Alternative Flow(s):
7.1 b) DiscardChanges

UC-1-5: AdminApproval

ID: UC-1-5

Description: A store manager can approve, decline, or ask for more details when a support staff requests a change.

Actor(s): Store manager.

Preconditions: Support staff request to modify or remove an item.

Main Flow:

1. The store manager goes to the “request” tab.
2. The store manager can view the changes made by the support staff.
3. If the store manager approves of the change then
    3.1. The store manager “approves” the request.
4. If the store manager requires more details then
    4.1. The store manager submits a comment/question for clarification.
5. If the store manager doesn't agree with the changes then

5.1. The store manager “declines” the request.
Postconditions: The requesting support staff is notified of the decision of the change.
Alternative Flow(s): None.

3.2 Scan-out Sold/Discarded Inventory

3.2.1 Description and Priority

Scanning out inventory is a crucial feature for the overall inventory management system. This feature will allow products to be scanned out of the system and remove them from the current store inventory. This may be done for two reasons: the product has been purchased or the product has been damaged. The resulting actions are to either remove the product from inventory and categorize it under Sales or to remove the product from inventory and categorize it under losses. The requirement priority of this feature is a **high priority**.

3.2.2 Functional Requirements

REQ-2-1: The product once sold through the POS system must be removed from inventory, then categorized under Sales.

REQ-2-2: Damaged or expired products can be scanned to remove it from inventory, then categorized appropriately.

REQ-2-3: Products removed from inventory through scanning must be categorized under either Expired or Damaged.

REQ-2-4: Products can be scanned out of inventory by store managers without waiting for approval.

REQ-2-5: Support staff must request administrative approval to remove items from inventory.

3.2.3 Use cases

UC-2-1: ScanningOutExpiredProducts
ID: UC-2-1
Description: Items that are expired can be scanned to remove them from the current inventory.

Actor(s): User (support staff or store manager).

Preconditions: The actor is removing expired items from the shelf and has a scanner in hand.

Main Flow:

1. The user locates the barcode on the product.
2. The user selects ‘Remove items from inventory’ on the scanner.
3. The user selects the reason for removal as ‘expired’ through the scanner interface.
4. The user scans the barcode of the product.
5. If the user would like to remove multiple items of the scanned product
    5.1. The user changes the number of items section from 1 to the desired amount.
6. The user verifies the product details and selects ‘confirm’ on the scanner.
7. If the user is a support staff then
    7.1. Include(AdminApproval).
8. The items are removed from inventory and categorized under “expired”.

Postconditions: The items have been removed from inventory.

Alternative Flow(s):
5.1 b) DiscardChanges

UC-2-2: ScanningOutDamagedProducts

ID: UC-2-2

Description: Products that are damaged can be scanned to remove them from the current inventory.

Actor(s): User (support staff or store manager).

Preconditions: The actor is removing a damaged product from the shelf and has a scanner in hand.

Main Flow:

1. The user locates the barcode on the product.
2. The user selects ‘Remove items from inventory’ on the scanner.
3. The user selects the reason for removal as ‘damaged’ through the scanner interface.
4. The user scans the product’s barcode.
5. The user adds a description of the damages to the ‘damage’ section.
6. If the user would like to remove multiple items of the scanned product
    6.1. The user changes the number of items section from 1 to the desired amount.

7. The user verifies the product details and selects ‘confirm’. 8. If the user is a support staff then 8.1. Include(AdminApproval). 9. The items are removed from inventory and categorized under “damaged”.
Postconditions: The items have been removed from inventory.
Alternative Flow(s): 5 b) Cancel

UC-2-3: RemovingSoldProductFromInventory
ID: UC-2-3
Description: The product is sold to store customers through POS.
Actor(s): User (support staff or store manager).
Preconditions: The user completes a transaction with the POS system.
Main Flow: 1. The sold items are removed from inventory and categorized under “sold”.
Postconditions: The items are removed from the inventory.
Alternative Flow(s): None.

3.3 Manage Orders

3.3.1 Description and Priority

The manage order feature notifies the store manager when a product in inventory is below its minimum amount; this number is set up when the product is first scanned into the inventory system. This feature also helps store managers to draft a PDF order summary, by combining the necessary product information. Every past order summaries will be accessible to the store manager. This feature is a **medium priority.**

3.3.2 Functional Requirements

REQ-3-1: The store manager must get a reminder when the quantity of a product becomes less than the minimum number.

REQ-3-2: A PDF summary that includes the quantity, the total cost of the products and the estimated arrival date will be produced when requested.

REQ-3-3: The estimated arrival date of a product that has been ordered must be indicated on the product's information page, visible to all users.

REQ-3-4: The store manager must be able to manually change the order arrival date in the order summary to improve accuracy. This will help during inventory and discrepancy checks.

REQ-3-5: All the order summaries created will be accessible to the store managers in a section called Order History.

3.3.3 Use cases

UC-3-1: DraftNewOrder
ID: UC-3-1
Description: The store manager drafts a new order.
Actor(s): Store manager.
Preconditions: The store manager has received low inventory reminders or new products are determined to be needed.
Main Flow: 1. The store manager selects the Draft Order option. 2. The store manager enters the products and the quantity required for each product. 3. A PDF order summary is created with the necessary information of the products. 4. New products' information will be added to the inventory database. The new and existing products will have the estimated arrival date on the product's information page. 5. The produced PDF will be saved in the Order History section for future access.
Postconditions: A new order summary is produced with the necessary products.
Alternative Flow(s): 2 b) Cancel

UC-3-2: IntakeOrder
ID: UC-3-2
Description: The ordered items have arrived and are ready to be added to the shelves.
Actor(s): User (support staff or store manager).

Preconditions: An order from the distributors has arrived.
Main Flow: 1. The user selects the Order History tab. 2. A list of all the order summaries is displayed with the most recent order on top. 3. The user will select the summary for the products that have arrived. 4. Include(ScanInItem). 5. The user “confirms” the quantity that was received. 6. If the item quantity doesn’t match the expected quantity then 6.1. The user selects the “Modify” option and enters the actual quantity.
Postconditions: The inventory is updated.
Alternative Flow(s): 4 b) Cancel

UC-3-3: DeleteOrderSummary
ID: UC-3-3
Description: The store manager deletes an order summary.
Actor(s): Store manager.
Preconditions: The Order History tab is not empty.
Main Flow: 1. The store manager selects the Order History tab. 2. The store manager is presented with the list of all the order summaries. 3. The store manager selects the order summary he wants to remove. 4. The store manager selects Delete. 5. The store manager confirms he/she would like to Delete the order. 6. The selected order summary is now removed from the Order History tab.
Postconditions: The removed order summary is not present in the Order History tab.
Alternative Flow(s): 5 b) Cancel

3.4 Manage discrepancies - Discrepancy Sniffer

3.4.1 Description and Priority

The system with the responsibility of alerting the ordering manager of any discrepancies in the inventory is called the ‘Discrepancy Sniffer’. The Discrepancy Sniffer’s algorithm will allow for any discrepancies in product inventory to surface based on inconsistency in the amount of product originally received, the amount of product sold, and the amount of product still on the shelf. This will be facilitated using information made available by Sections 3.1 and 3.2 defined above. The Discrepancy Sniffer will also accumulate the amount of revenue lost due to the missing products not being sold. This feature is a **low priority.**

3.4.2 Functional Requirements

REQ-4-1: The store manager must be able to see which products are missing from the store and the quantities of each.

REQ-4-2: The store manager must be able to see the total amount of revenue lost due to discrepancies in inventory.

REQ-4-3: The store manager must be able to differentiate between product loss due to expiration and product loss use to theft and mishandling.

REQ-4-4: The store manager must be able to filter discrepancies by time.

3.4.3 Use cases

UC-4-1: CompleteInventoryCheck
ID: UC-4-1
Description: A user completes an inventory check.
Actor(s): User (support staff or store manager).
Preconditions: None.
Main Flow: 1. The user selects “Complete Inventory Check”. 2. The user enters the number of products present in inventory. 3. The inventory can be organized by category, or in ascending or descending alphabetically. 4. If not all the inventory amounts are entered then 4.1. The inventory check is categorized as a “partial inventory check.” 4.2. A partial inventory check if confirmed is placed in a partial inventory check tab. 5. Else 5.1. The inventory check is categorized as a “complete inventory check.”

5.2. A “complete inventory check” if confirmed is placed in a complete inventory check tab.

6. The user confirms the details of the Inventory check.

Postconditions: The store manager is notified of the inventory check completion.

Alternative Flow(s):
4 b) Cancel

UC-4-2: ModifyInventoryCheck
ID: UC-4-2
Description: A user modifies a completed inventory check.
Actor(s): User (support staff or store manager).
Preconditions: A Inventory check has been completed.
Main Flow: 1. The user selects “Completed Inventory Checks”. 2. The user selects the Inventory Check they would like to modify. 3. The user modifies the quantity of an entered product. 4. The user confirms the details of the modified Inventory check. 5. If the user is a support staff then 5.1. Include(AdminApproval).
Postconditions: The Inventory Check history is updated.
Alternative Flow(s): 4 b) Cancel

UC-4-3: DeleteInventoryCheck
ID: UC-4-3
Description: A user deletes a completed inventory check.
Actor(s): User (support staff or store manager).
Preconditions: A Inventory check has been completed.
Main Flow:

1. The user selects “Completed Inventory Checks”. 2. The user selects the Inventory Check they wish to delete. 3. The user confirms the deletion of the Inventory Check 4. If the user is a support staff then 4.1. Include(AdminApproval).
Postconditions: The Inventory Check history is updated.
Alternative Flow(s): 4.1 b) Cancel

UC-4-4: DiscrepanciesFound
ID: UC-4-4
Description: After a user completes an inventory check, store managers can view the amount of inventory that was unaccounted for.
Actor(s): Store manager.
Preconditions: Inventory check has been completed by the support staff or store manager.
Main Flow: 1. The store manager is notified of the completed inventory check. 2. The missing products tab contains the items that are in the system but not accounted for. 3. If products are missing then 3.1. They can be organized in alphabetical order, by category, the number of units missing.
Postconditions: The store manager is notified of the discrepancies.
Alternative Flow(s): None.

3.5 Manage product expiry

3.5.1 Description and Priority

Expiry dates that are collected when inventory is entered will be tracked. Inventory that is close to expiry or has expired will be presented to the staff as an alert when they ask for a product summary. Once presented, these alerts will remain until dismissed. It is the user’s responsibility to remove expired products from the shelf when notified. This feature is important to the clients,

however, it requires no system action other than a notification so the priority has been rated as **medium priority**.

3.5.2 Functional Requirements

REQ-5-1: A notification will be created for products that are at their specified expiry notification date.

REQ-5-2: Users will be notified when products are expired.

REQ-5-3: Expired product notifications will require confirmation of removal before being removed from inventory.

3.5.3 Use Cases

UC-5-1: UpcomingExpiryNotification
ID: UC-5-1
Description: The product is the specified number of days away from expiry and the user is notified.
Actor(s): User (support staff or store manager).
Preconditions: The user receives the product expiry notification (i.e. the product is the specified number of days away from expiry).
Main Flow: 1. The user navigates to the “upcoming expiry tab”. 2. The upcoming expiry items are shown with the product information and the time remaining before the product’s expiration. 3. The user acknowledges and “dismisses” the notification.
Postconditions: The notification has been moved to the notification history tab.
Alternative Flow(s): None.

UC-5-2: ExpiredItemNotification
ID: UC-5-2
Description: The system notifies users that a product is expired, and the user must confirm the product was removed to update the inventory.

Actor(s): User (support staff or store manager).
Preconditions: The product has expired.
Main Flow: 1. The user receives the expired product notification or navigates to the “expired products tab”. 2. The user is presented with a notification that contains the product information. 3. Include(ScanningOutExpiredProducts). 4. While the user has not confirmed the item’s removal: 4.1. The notification remains in the “expired products tab”.
Postconditions: The notification is moved to the notification history tab and the inventory is updated.
Alternative Flow(s): None.

3.6 View Current Inventory

3.6.1 Description and Priority

The system will allow for the current inventory to be viewed through the software’s user interface. This information will be displayed to the device in a manner that is easy to read for all users. This feature is a **high priority**.

3.6.2 Functional Requirements

REQ-6-1: Users will be able to view the current inventory of all stores.

REQ-6-2: Users will be able to sort the inventory view by keyword, expiry date, or a section.

REQ-6-3: The default product inventory viewed by a user will be based on the employees’ primary work location.

REQ-6-4: Users will be able to search product inventory through the product barcode.

3.6.3 Use cases

UC-6-1: SelectBranch
ID: UC-6-1
Description: The user selects Kevin’s Convenience Store branch they would like to view the current inventory of.
Actor(s): User (support staff or store manager).

Preconditions: Product information is available for the desired branch.
Main Flow: 1. The user opens the inventory page. 2. The user selects the desired Kevin's Convenience Store branch. 3. The user accesses the inventory page of the desired branch.
Postconditions: None.
Alternative Flow(s): None.

UC-6-2: ViewInventory
ID: UC-6-2
Description: Product information such as the product's name, the amount in inventory, and the product's expiry date is included in the inventory displayed to the user.
Actor(s): User (support staff or store manager).
Preconditions: Product information is available.
Main Flow: 1. The user opens the inventory page. 2. The user navigates to the product of interest by sorting inventory by category, sorting inventory alphabetically, or searching for a product. 3. The user selects the product of interest to open the product information page. 4. The product page displays the current inventory, inventory history, and product information.
Postconditions: None.
Alternative Flow(s): None.

4 External Interface Requirements

This section describes external user, hardware, software and communication interfaces required by Kevin's Convenience Store branch to run the Inventory Management System.

4.1 User Interfaces

The web-based application places a heavy emphasis on mobile hardware that requires a touch-screen style input, where a user will be able to interact with the application through

standardized digit motions across the device's screen. However, desktop users are expected to use the application as well, thereby necessitating standard mouse and keyboard input. Upon starting the application the user will require to log in through the company website. Two separate user types, store managers and support staff will have slightly varied options. Store managers will be able to draft orders to distributors through the application as well as change prices. The support staff will only be able to enter and view inventory/product information into the store database. A "menu" button will allow easy switching between windows. To scan information from product shelves/shipments, the user will hit a button from the app to bring up the scanner. The user will scan-in the barcode which automatically populates the product page with the appropriate information. The user will make any additions or changes where appropriate. The UI will avoid the use of colors to convey context. Additionally, larger buttons will be implemented as part of the standard UI to accommodate the members of Kevin's Convenience Corner with physical disabilities.

4.2 Hardware Interfaces

The application is web-based and focuses on mobile devices with a functional scanner peripheral. Desktop access will share the same features and UI. Navigation through the application will be accessible through touch-screen interface or, if on desktop, mouse, and keyboard. The device must also be able to maintain a constant internet connection to allow for real-time updating of the inventory database. However, in the event of an internet connection being unavailable, the application will require access to the device's local storage to ensure user input is maintained. Data entered will be uploaded as soon as a connection becomes available.

4.3 Software Interfaces

Upon entering login information, the app will connect with Kevin's Convenience Corner database and display appropriate windows, buttons, and menus based on user administrative privileges.

4.3.1 Product Scanning

Upon entering the scanning menu the user will press a button to prompt the scanning functionality. The user will scan the product's barcode whereby the software will bring up the appropriate product from the database. The scanning menu will close automatically and barcode information will auto-populate the product's information page. The user will be able to manually enter data for fields where information is unavailable or inaccurate (expiry, quantity, etc.). Once the user submits the completed page, the application will upload the new product information to the inventory database and display an empty product page.

4.3.2 Inventory Management

The application will connect to the inventory database whereby the user will be able to scroll through product data tables containing immediately pertinent information. Inventories of alternate stores are viewable through a menu. A product search function via manually entering key terms or through a product scanner will be available to the user. Should the user wish to investigate the details of a particular product further, they can either touch or double click on the product of interest to bring up the product page. End of day inventory checks will identify discrepancies and notify management.

4.3.3 Product Ordering and Tracking

Order numbers, distributer, and estimated delivery dates will be summarized and displayed in a data table similar to that displayed in 4.3.2 (Inventory Management). Creating a new order or modifying an existing order requires the user to either touch or double click on the create new order button/order of interest to bring up the order information page. The user will be able to manually change all data except ‘communication history’ on this page should they have the appropriate administrative privileges. Due to the inability to interact with distributor databases, the quality of information on this page will be the responsibility of the user. Should the distributor have an associated email that accepts product orders, the user will be able to email the pdf directly to the company’s email. The email will have a new message composed containing a pdf copy of the order in addition to the distributor’s email auto-populated. Once sent, the page will return to the orders data table along containing a newly populated date of last communication.

4.4 Communications Interfaces

The application will have a constant internet connection to maintain real-time database management as well as access to the device’s local storage for temporary storage should the event of an internet connection failure occur. Additionally, the application will require direct access to both the third-party POS system and inventory databases to ensure logged inventory and prices are maintained in real-time. Interaction with the company website and email is required for security, product ordering, and tracking features. The ability to print specific pages of interest requires a connection to a local network printer.

5 Other Non-Functional Requirements

5.1 Performance Requirements

The system will be capable of handling a request such as summarizing an inventory report with up to 5000 varying products. Each listed product will contain an associated count as well as any other pertinent information. Requests of this size are likely to occur every time the system is

used and must be responsive within 500ms. When edits are being made by the user, the system will be capable of updating the database within 100ms. When a sale is being finalized using the POS, the system will update the database within 10ms to remove sold products from the inventory. The store is open up to 128 hours a week, requiring round-the-clock runtimes. Therefore, the application should be run and tested for 140 hours before release.

5.2 Safety Requirements

At the current state of the project, there have been no safety requirements defined.

5.3 Security Requirements

Access to data is restricted to the employees of Kevin's Convenience Store. This may be implemented using a login feature. The login feature will require staff to enter their user ID and password for authentication. Financial security is of low concern as the current POS system prevents access to payment information. This system will only convey product sales. The system must have a daily end-of-day scan for any bugs and vulnerabilities. If any unusual activity is discovered, the store manager and an Inventoria security engineer will be notified with the error type. In addition to the error type, Inventoria's security engineer will receive a detailed event history to better address the issue. The system will also be subjected to regular system updates.

5.4 Software Quality Attributes

Our system is concerned with three primary quality attributes:

Accessibility - The system must be accessible for physically disabled employees as well as support for color blindness.

Learnability - Since the system simplifies the current procedure for the employees, the system must be learned by the employees within one workday.

Scalability - The system must be able to accommodate and integrate the databases of new branches, as the company expands.

6 Data Flow Diagrams

This section showcases the data flow in and out of the Inventory Management System in reference to the additional systems, users and actors it interacts with.

6.1 Context Diagram (DFD Level 0)

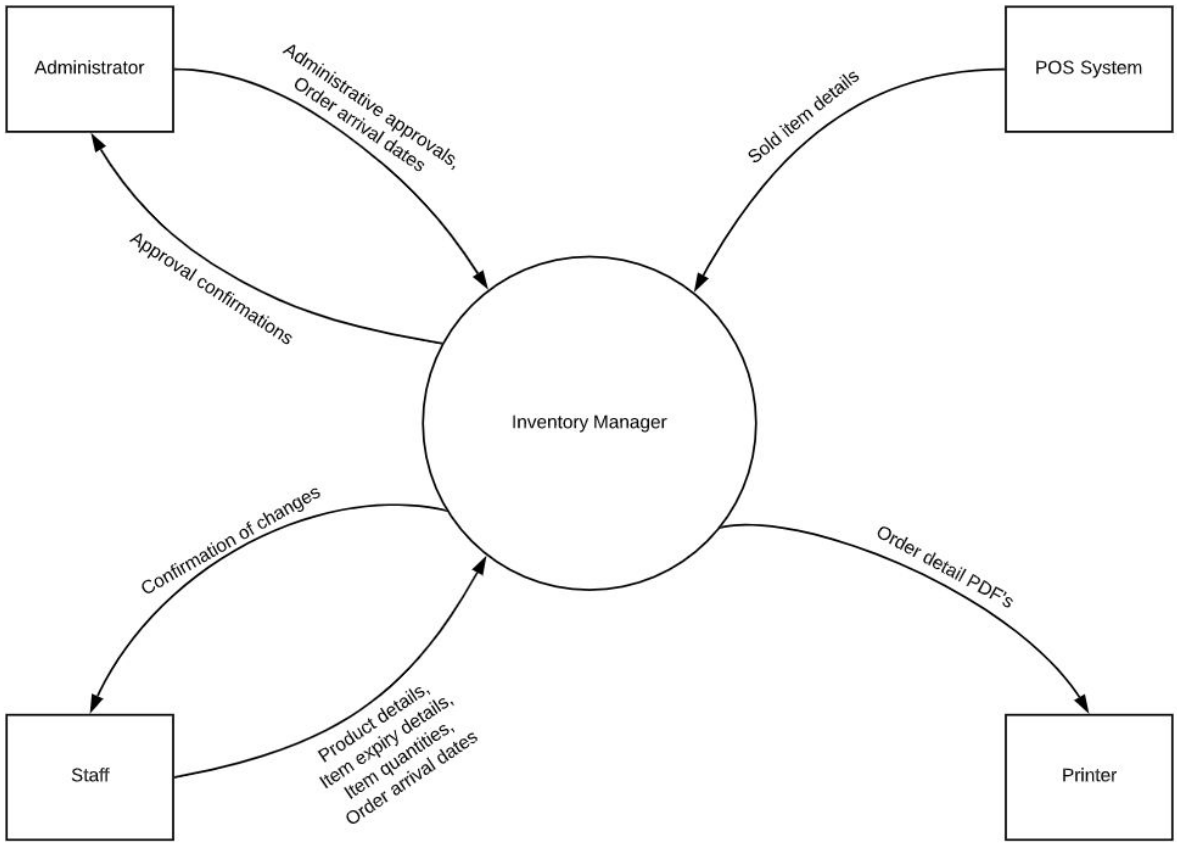

Figure 6.1 Context Diagram:
In the system level DFD, it shows that the main data flow is from the Staff to the Inventory Manager.

6.2 DFD Level 1

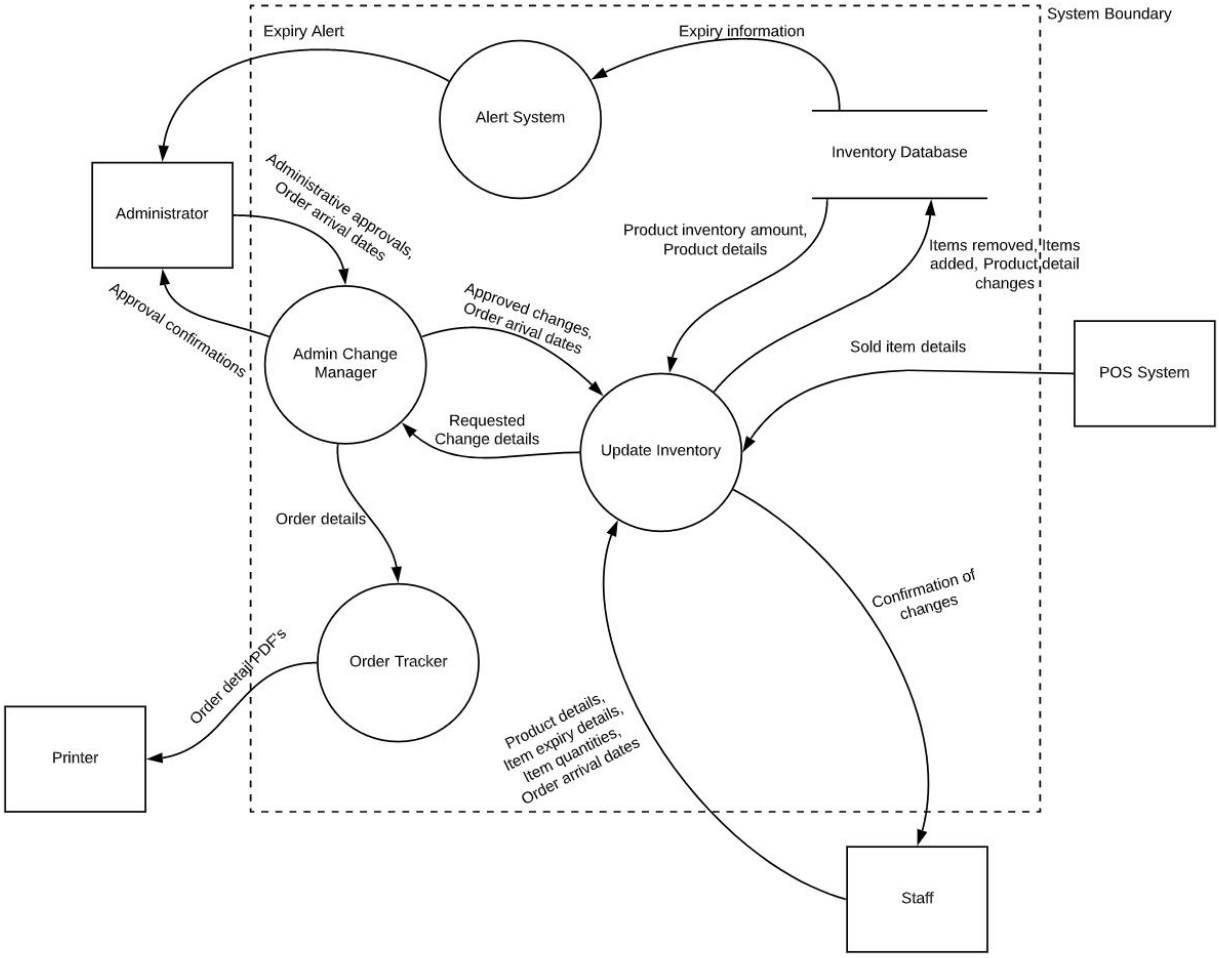

Figure 6.2 DFD Level 1:
This figure is a visual representation of the data flow from both within the system and with external terminators.

6.3 DFD Level 2

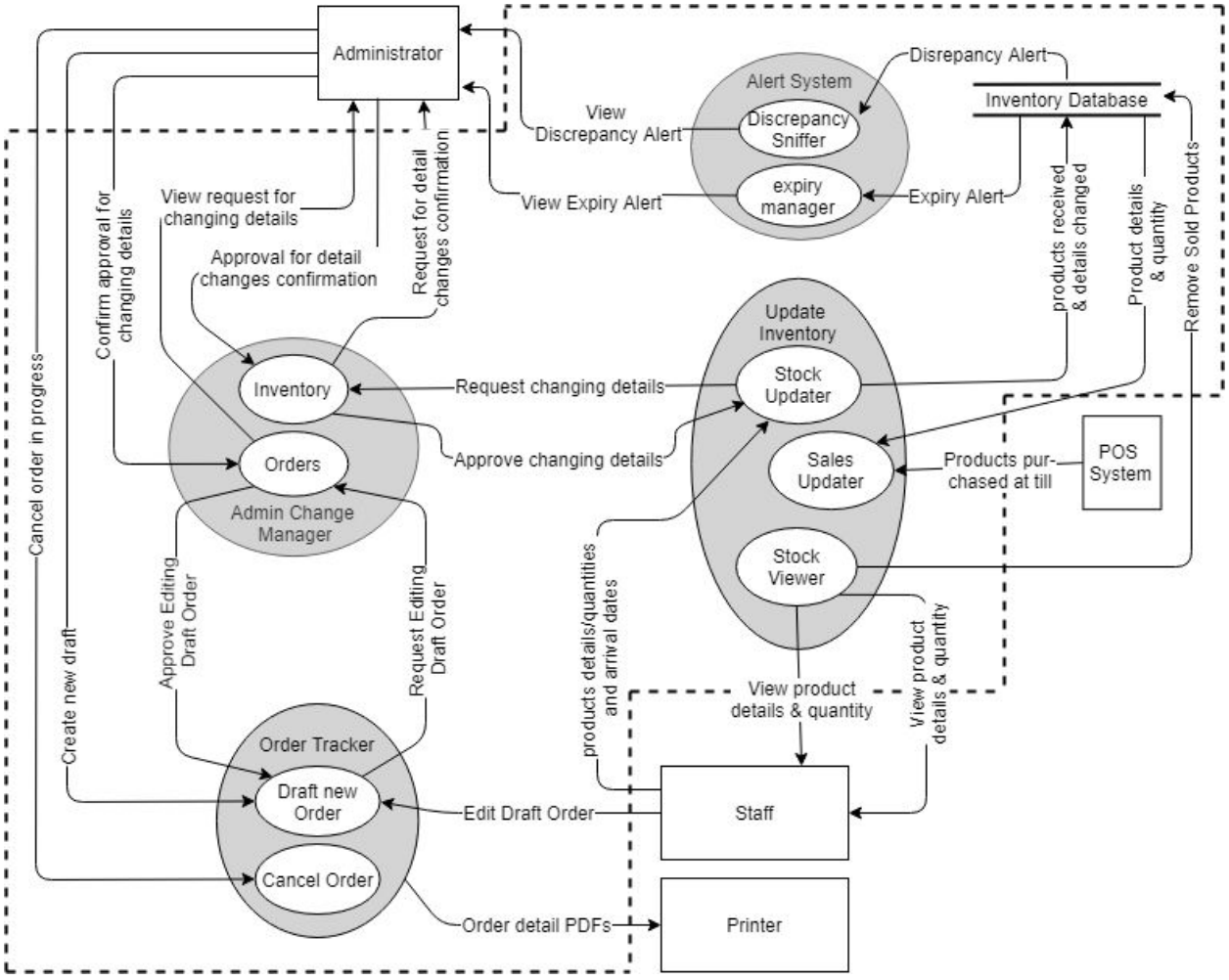

Figure 6.3 DFD Level 2:

Represents all subsystems of Inventory Manager. The interactions between User (Staff and Administrator), Printer, Inventory Database and POS system with subsystems is outlined in further detail. From top to bottom, left to right: Alert System and its alerts to the Administrator, Admin Change Manager and changing inventory values following Administrator approval, Update Inventory and change requests from Staff, and Order Tracker and its creation and manipulation of orders by an Administrator.

7 Entity-Relationship Diagrams

This section describes the relationships between the existing features, users and objects (together known as entities) for the problem domain of the inventory Management System.

7.1 Entity-Relationship Diagram

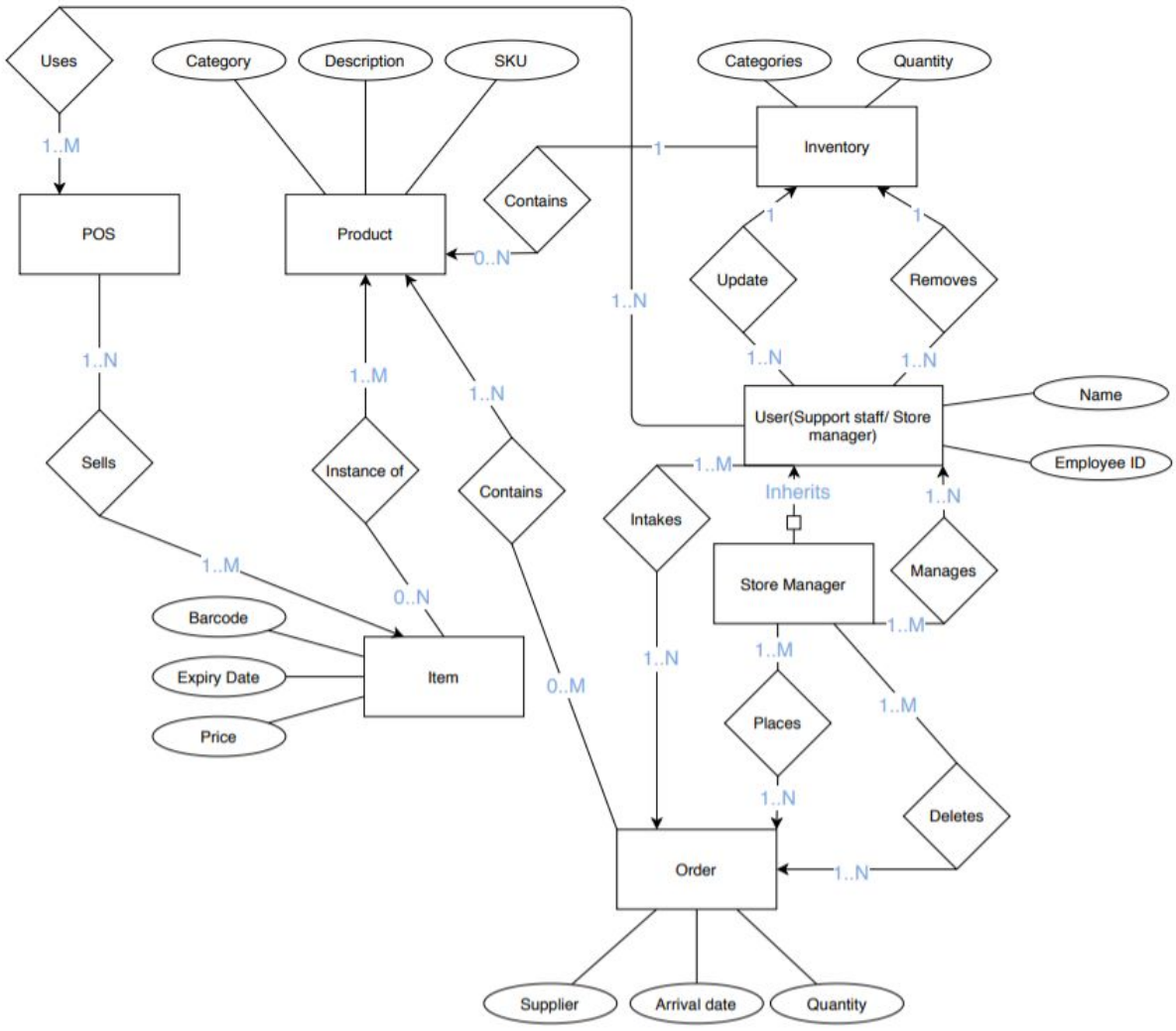

Figure 7.1 Entity-Relationship Diagram

The Data Dictionary is located in section 7.2. THe ER diagram uses Chen's notation with arrows to indicate the relationship direction. The Store Manager Entity is inherited from the User Entity which entitles the store manager to all the basic permissions of the User Entity plus added privileges. These added privileges include managing the User Entity and also placing/deleting an order. The POS Entity has no attributes, since it is outside our problem scope. An item is an instance of the Product Entity which is contained inside the Inventory Entity.

7.2 Data Dictionary

Entity Class	Entity Type
Product	Category {“Dairy”| “Produce”	“Household supplies”	“Dry goods”	“Pharmacy”	“Other”} , <<<*SKU*>>> ,“Description”.
Item	<<<*Barcode*>>>, <<<*Expiry Date*>>>, <<*Price*>>
Order	(Product), <<<*Arrival Date*>>>, <*Quantity*>, “Supplier”
Inventory	Category{“Ordered”	“In Stock”	“Sold”	“Damaged”	“Expired”}, <*Quantity*>, (Product)
User(Support staff/ store manager)	“Name”, <<<*ID*>>>>

Type	Representation
String	“Value”;
Enumeration	{“Value”	“Value2”	“Value3”|…..| “ValueN”};
Integer	*<Integer Value>;*
Decimal	*<<Decimal Value>>;*
Alphanumeric	*<<<Alphanumeric Value>>>;* Allows “a”....”z”|”A”.....”Z”|”0”.....”9”| “-”| “.”| “/”;
Reference to a defined entity	(Entity)

8 Appendix

This appendix includes additional information regarding hardware and software specifications, specifically what is unknown as of this report.

8.1 Issues List

- The scanner to be used for scanning barcodes is to be determined since the current systems for scanning barcodes have not been investigated enough to make an informed choice.