California State University of Fullerton
ISDS 454-01 #12173
Professor Peng Liu

Inventory Management System Final Report

Group 7:
Adrian Cruz, Linda Huynh, Donna Nguyen,
Anthony Nava, Chris Nguyen, Julian Posadas

Due Date: 12/09/2024

Table of Contents

Introduction

This deliverable outlines the implementation of an Inventory Management System designed to enhance warehouse operations by enabling streamlined tracking, management, and inventory control. By providing real-time monitoring of inventory levels and item movements, this system ensures accuracy and allows for faster, data-driven decisions to maintain a competitive edge.

With features like searching, browsing, item addition/deletion, location tracking, and update logging, the system optimizes day-to-day tasks, reducing the time and effort required for manual inventory checks. Non-functional requirements, including performance, security, and usability, ensure the system is robust, user-friendly, and capable of supporting the warehouse's evolving needs.

By automating data capture and real-time tracking, this system minimizes the risk of stockouts, overstocking, and errors, directly impacting productivity and reducing waste. Inventory Specialists can rely on accurate data for demand planning and stock replenishment to enhance overall supply chain efficiency. This increased visibility also improves forecasting accuracy, allowing the business to align inventory levels with market demand better.

In summary, this Inventory Management System will be instrumental in advancing warehouse efficiency, accuracy, and security. Its enhanced tracking, automated data capture, and intuitive design reduce operational bottlenecks, facilitating improved workflow and real-time data for strategic decision-making. With this system, the warehouse can expect a substantial increase in operational efficiency, accuracy, and responsiveness to changing demands.

Requirements Definitions

Functional Requirements

1. <u>Authenticate User</u>
   The system will prompt users to log in with valid credentials to access the main menu.
2. <u>Search</u>
   The system will allow users to search inventory by SKU or Name and check if the physical inventory count matches the count in the database.
3. <u>Browse</u>
   The system will allow users to browse the available inventory by category.
4. <u>Add Items</u>
   The system will allow users to add items to the existing inventory.
5. <u>Delete Items</u>
   The system will allow users to delete items from the existing inventory.
6. <u>Locate Items</u>
   The system will allow users to locate inventory locations.
7. <u>Track Last Update/Change</u>
   The system will allow users to track when an inventory item was last updated/changed.
8. <u>Update Item Information</u>
   The system will allow users to modify the information in the database to match the physical inventory when needed.

Nonfunctional Requirements

1. <u>Operational</u>
   - **1.1.** The inventory management system will be constructed to facilitate searches by item name, SKU, and supplier.
   - **1.2.** The system will run on any in-store kiosk/employee scanners.
   - **1.3.** The system will integrate with the organization's existing operations, such as shipping and logistics.
2. <u>Performance</u>
   - **2.1.** Latency speeds will be monitored and kept at an acceptable level.
   - **2.2.** The system will be updated in real-time and online 24/7.
3. <u>Security</u>
   - **3.1.** All sensitive supplier information and data will be encrypted and secured.
   - **3.2.** The system will only be accessible from within the company and requires employee user IDs and passwords.
   - **3.3.** Each device within the system will have antivirus protection that evaluates the system’s integrity and scans any imported files or inserted drives.
4. <u>Cultural and Political</u>
   - **4.1.** The system will only be used within the US; therefore, customization related to global applications is not required.
   - **4.2.** The system will only operate in English.

Use Cases

Use Case Name: Authenticate User	**ID:** UC-1	**Priority:** High
Actor: Inventory Specialist		
Description: This use case describes how the Inventory Specialist can log into the Inventory Management System by authenticating themselves using valid credentials to gain access.		
Trigger: An Inventory Specialist wants to access the Inventory Management System		
Type: ☑External ☐Temporal		
Preconditions: 1. The Inventory Specialist is registered with a unique login ID and password.		
Normal Course: 1.0 Enter Authorized Credentials 1. The Inventory Specialist selects the "employee ID" field 2. The Inventory Specialist enters their employee ID into the system. 3. The Inventory Specialist selects the "password" field. 4. The Inventory Specialist enters their password into the system. 5. The Inventory Specialist selects the "login" button.		
Postconditions: 1. The Inventory Specialist will be authenticated and directed to the Main Menu to access the system fully.		
Exceptions: E1: The login fields are incorrect 1. Restart the Normal Course using a valid employee ID and password.		

Use Case Name: Inventory Search	**ID:** UC-2	**Priority:** High
Actor: Inventory Specialist		
Description: This use case describes how the Inventory Specialist can search the warehouse inventory by SKU or Name and view the current information about that item.		
Trigger: An Inventory Specialist wants to search the warehouse for a specific item.		
Type: [✓ External] ☐ Temporal		
Preconditions: 1. The Inventory Specialist is logged into their account. 2. The Inventory Specialist is authorized to access inventory records.		
Normal Course: 1.0 Item Search 1. The Inventory Specialist selects the "Search" option from the inventory management system. 2. The Inventory Specialist types either the item Name or SKU. 3. The system displays all relevant search results. 4. The Inventory Specialist selects the item desired. 5. The system displays the items' Name, SKU, Quantity, and location.		
Postconditions: 1. The Inventory Specialist will have the option to "View More…" details and/or features if desired.		
Exceptions: E1: The item desired does not appear in the results 1. Select "Cancel". 2. Restart Normal Course using a different search term.		

Use Case Name: Browse Data of Available Inventory by Category	**ID:** UC-3	**Priority:** High

Actor: Inventory Specialist

Description: This use case describes how a warehouse clerk may use the system to see all available inventory of a certain category of finished goods, parts, or ingredients.

Trigger: The warehouse clerk receives an inventory data request for a certain category of finished goods or parts.

Type: ☑ External ☐ Temporal

Preconditions:

1. The inventory management system must be up-to-date and online
2. The warehouse clerk is authorized to access inventory records

Normal Course:
1.0 Warehouse clerk browses available inventory by category

1. The warehouse clerk receives an inventory data request for a specific category of finished goods, parts, or ingredients.
2. The warehouse clerk is authenticated to access inventory records by providing valid credentials.
3. The system allows the clerk access to the warehouse inventory records.
4. The warehouse clerk sorts the inventory records by category.
5. The system returns inventory records organized by category.
6. The warehouse clerk selects the relevant category.
7. The system returns a record of all items in the category, the inventory count of each item, the inventory count of all items in the category, and other relevant information.
8. The warehouse clerk exports the data into a printable PDF
9. The system records the inventory data request

Postconditions:

1. The inventory data request is completed.
2. The system creates a log of the completed inventory data request.

Exceptions:
E1: The Warehouse Clerk does not have authorization to view inventory records. (occurs at step 2)

1. The system does not allow the warehouse clerk to view inventory records when invalid or unauthorized credentials are provided.
2. The warehouse clerk must contact warehouse management to obtain proper clearance or delegate the task to another warehouse clerk with clearance.
3. The normal course may be restarted after the clerk obtains clearance or delegates the task to someone with clearance.
4. If the system is offline or encounters an error, the inventory changes will not be saved, and the user will receive an error message.

Use Case Name: Adding New Items to Inventory	**ID:** UC-4	**Priority:** High
Actor: Inventory Specialist		
Description: This use case describes how the system enables users to add new items to the existing inventory. This feature updates inventory records, facilitating proper stock control and decision-making.		
Trigger: A new item SKU enters the warehouse and must be added to the inventory.		
Type: ☑️External 🔲Temporal		
Preconditions: 1. The user must be logged into the system with appropriate permissions to modify the inventory. 2. The inventory must not exist in the system.		
Normal Course: 1.0 Warehouse Clerk selects the ‘Add Item’ option. 1. The user navigates to the inventory management section and selects the “Add Item” function. 2. The system displays the Add Item form. 3. The system prompts the user to enter details for the new item, including but not limited to: 1. Item name 2. Quantity 3. SKU (Stock Keeping Unit) 4. The user inputs all the required information into the provided fields. 5. The system checks if all mandatory fields are filled and validates the input (following proper formatting rules). 6. The user reviews the input and clicks ‘Confirm’ to add the items to the inventory. 7. The system stores the new item in the inventory database and displays a success message. 8. The user is notified that the new item has been successfully added to the inventory.		
Postconditions: 1. The inventory is updated to reflect the added or deleted items. 2. A record of changes made to the inventory may be stored for audit purposes.		
Exceptions: 1. If the system is offline or encounters an error, the inventory changes will not be saved, and the user will receive an error message.		

Use Case Name: Deleting Items from Inventory	**ID:** UC-5	**Priority:** High
Actor: Inventory Specialist		
Description: This use case describes how the system allows users to delete items that are no longer needed or available. This feature updates inventory records, facilitating proper stock control and decision-making.		
Trigger: An item is no longer needed or available and must be removed from the inventory.		
Type: ☑️External 🔲Temporal		
Preconditions: 1. The user must be logged into the system with appropriate permissions to modify the inventory. 2. The inventory must already exist in the system.		
Normal Course: 1.0 Warehouse Clerk selects the ‘Delete Item’ option. 1. The user navigates to the inventory management section and selects the ‘Delete Item’ function. 2. The system shows all current inventory items, along with relevant details (e.g., item name, quantity). 3. The user clicks on the item they wish to remove from the inventory. 4. The system prompts the user to confirm the deletion of the selected item. 5. The user confirms by clicking ‘Delete’ or equivalent. 6. The system deletes the selected item from the inventory database and displays a success message. 7. The user is notified that the item has been successfully deleted from the inventory.		
Postconditions: 1. The inventory is updated to reflect the added or deleted items. 2. A record of changes made to the inventory may be stored for audit purposes.		
Exceptions: 1. If the system is offline or encounters an error, the inventory changes will not be saved, and the user will receive an error message.		

Use Case Name: Locate Items	**ID:** UC-6	**Priority:** High
Actor: Inventory Specialist		
Description: This use case describes how an Inventory Specialist can track an item's location within the warehouse.		
Trigger: A warehouse clerk receives a request for the location of a specific item.		
Type: ☑External ☐Temporal		
Preconditions: 1. The Inventory Specialist is logged into their account. 2. The Inventory Specialist is authorized to access inventory records.		
Normal Course: 1.0 Track Item Location 1. The Inventory Specialist locates the desired item in the system. 2. The Inventory Specialist selects the desired item from the search results. 3. The system displays the Aisle, Section, Department, Floor Level, and Rack/Bin for that item. 4. The Inventory Specialist exports/prints the location data.		
Postconditions: 1. The Inventory Special will have the option to "View More…" details and/or features if desired.		
Exceptions: 1. The inventory location will not be tracked if the system is offline or encounters an error.		

Use Case Name: Tracking Last Update/Change of Inventory Items	**ID:** UC-7	**Priority:** High

Actor: Inventory Specialist

Description: The system allows users to track the date and time when an inventory item was last updated or changed. This includes any additions, deletions, or modifications to the item's details, such as quantity, price, or description. This ensures users have a complete view of inventory activity for auditing and reporting purposes.

Trigger:

Type: ☑External ☐Temporal

Preconditions:

1. The user must be logged into the system with appropriate permissions to modify the inventory.
2. The inventory items must already exist in the system.
3. Any modification to the inventory, such as adding, deleting, or updating items, triggers an update to the 'Last Updated' timestamp.

Normal Course:

1.0 Warehouse Clerk navigates to the Inventory section.

1. The user opens the inventory management section from the main menu.
2. The system shows a list of all inventory items, including item details like name, quantity, and SKU.
3. For each item, the system can add a 'Last Updated' column showing the date and time of the item from the dropdown menu that was last changed. This information includes:
    1. The date and time of the last modification
    2. The type of modification (e.g., added, deleted, or updated)
4. The user clicks on a specific inventory item to view detailed information about the changes made to it over time.
5. The system shows a detailed log of changes made to that item, including:
    1. The user who made the change
    2. What was changed (e.g., quantity, price, description)
    3. Date and time of each change
6. The user can track the item's modification history to ensure accuracy and to monitor how often and why changes are made.
    1. The user can filter or search for items based on when they were last updated.

Postconditions:

1. The system maintains accurate and up-to-date records of all modifications to inventory items, and users can see when an item was last updated.

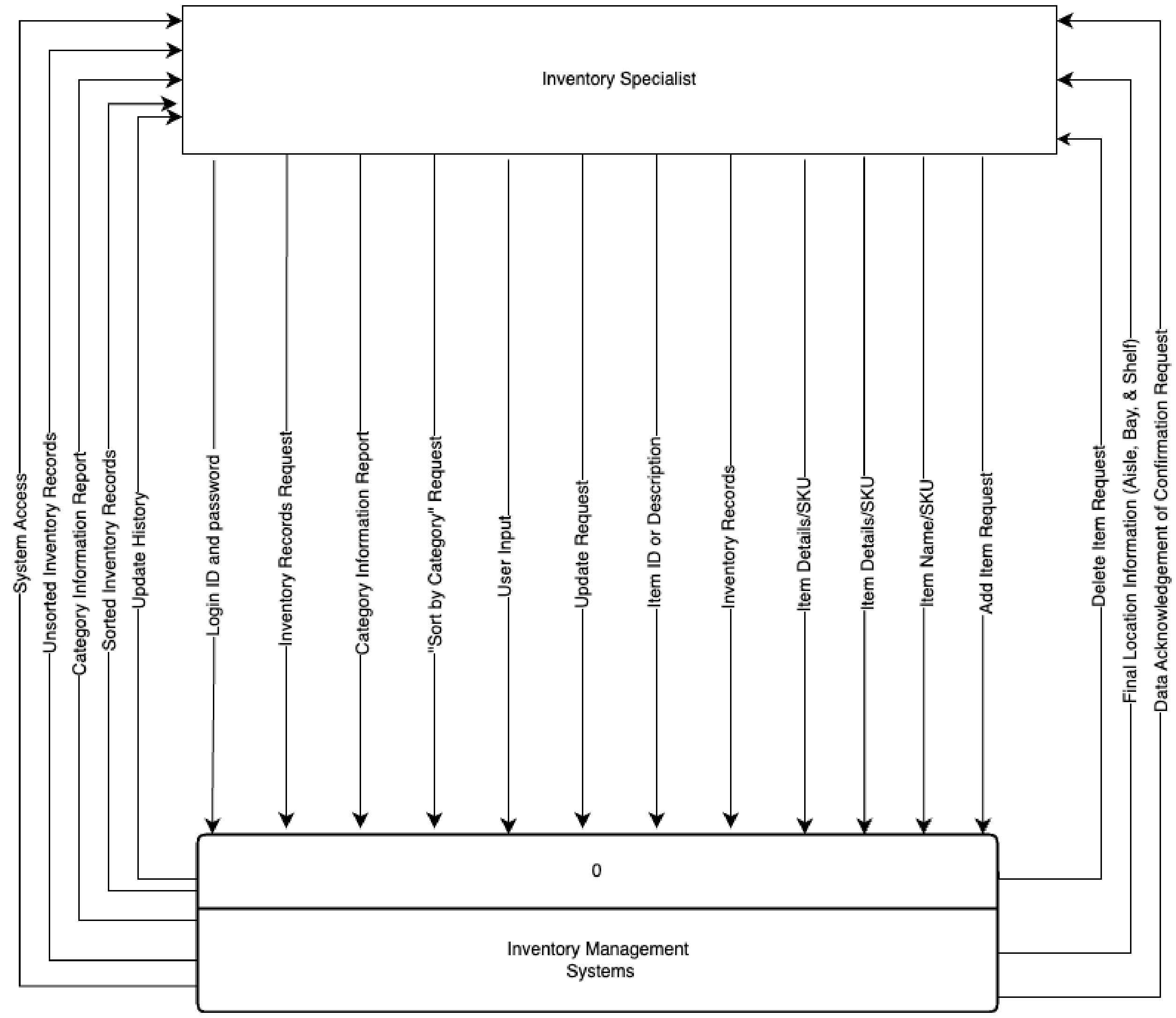
Exceptions:

1. The system will not allow the user to update or change any of the Log of Changes information since those activities were recorded automatically by the Inventory Management System.

Use Case Name: Update Inventory Item	**ID:** UC-8	**Priority:** High
Actor: Inventory Specialist		
Description: This use case describes how to update important inventory item information in the system.		
Trigger: The user initiates this process by selecting “add new item” or “edit item” from the inventory management dashboard.		
Type: [✓ External] ☐ Temporal		
Preconditions: 1. The Inventory Specialist is logged into the inventory management system. 2. The user has permission to add or make edits to items.		
Normal Course: 1. The user selects the “Update Item” option from the inventory management. 2. The system allows the Inventory Specialist to input details. 3. The user enters relevant information (Item name, SKU, quantity, description, location, supplier information, expiration date, etc.) 4. User reviews inputted information. 5. The user clicks “Save” to save information. 6. The system stores item information in the database. 7. The user is returned to the main menu.		
Postconditions: 1. The item is retrievable for future reference		
Exceptions: 1. If the system encounters a database error while saving item information, the user is prompted to try again.		

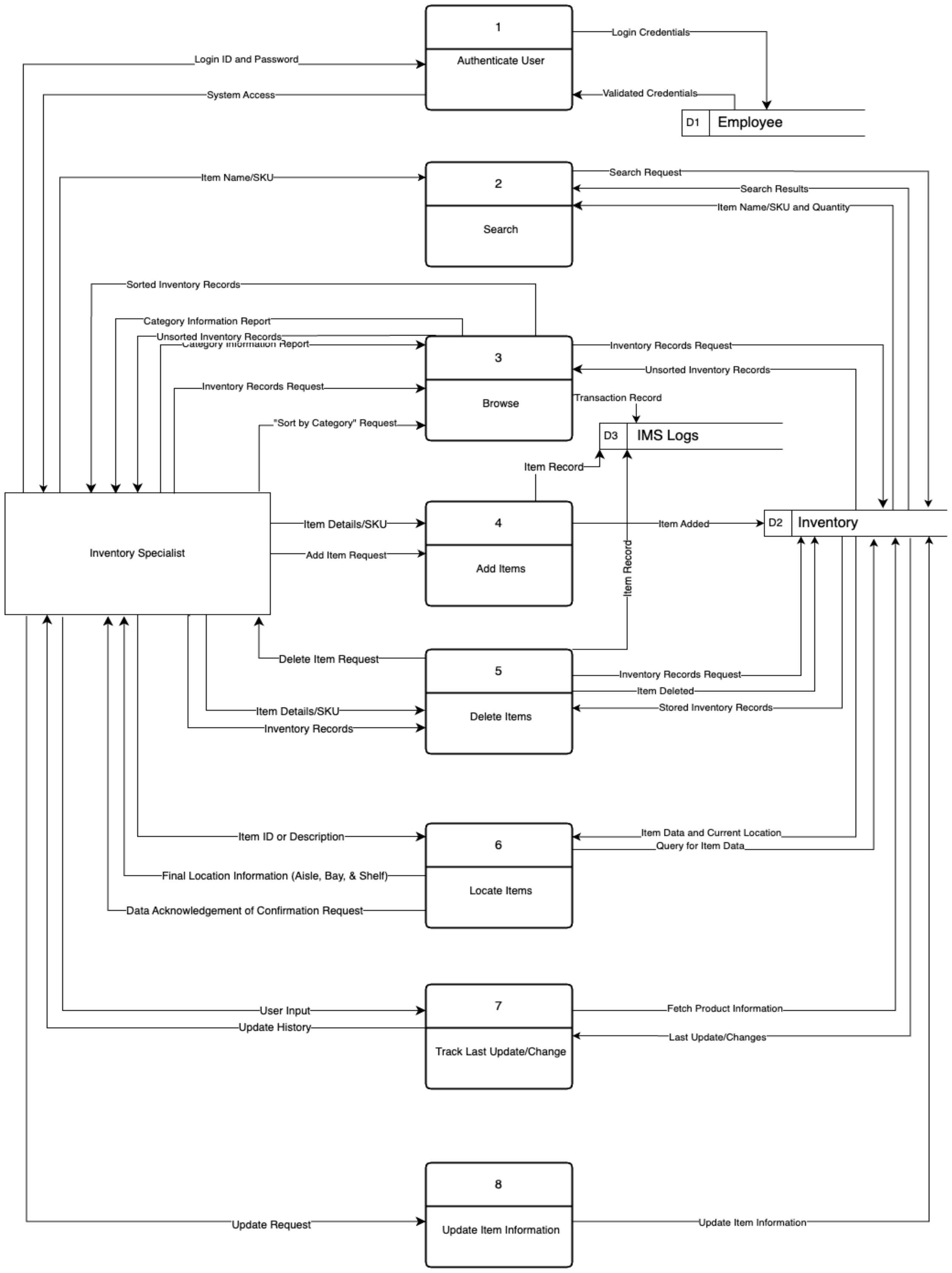
Data Flow Diagrams

Context Level DFD

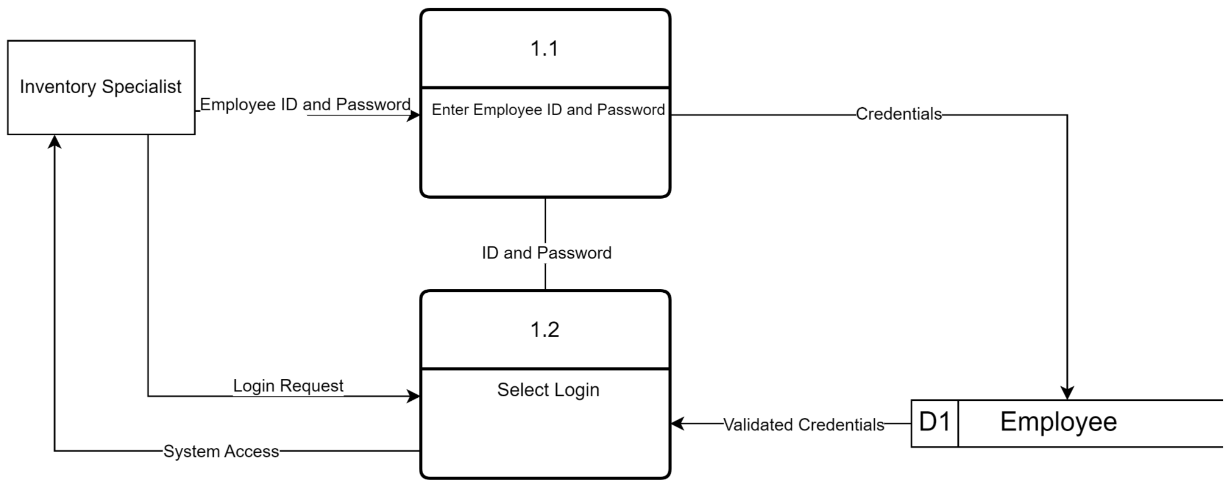


Inventory Specialist
System Access
Unsorted Inventory Records
Category Information Report
Sorted Inventory Records
Update History
Login ID and password
Inventory Records Request
Category Information Report
"Sort by Category" Request
User Input
Update Request
Item ID or Description
Inventory Records
Item Details/SKU
Item Details/SKU
Item Name/SKU
Add Item Request
Delete Item Request
Final Location Information (Aisle, Bay, & Shelf)
Data Acknowledgement of Confirmation Request
0
Inventory Management Systems

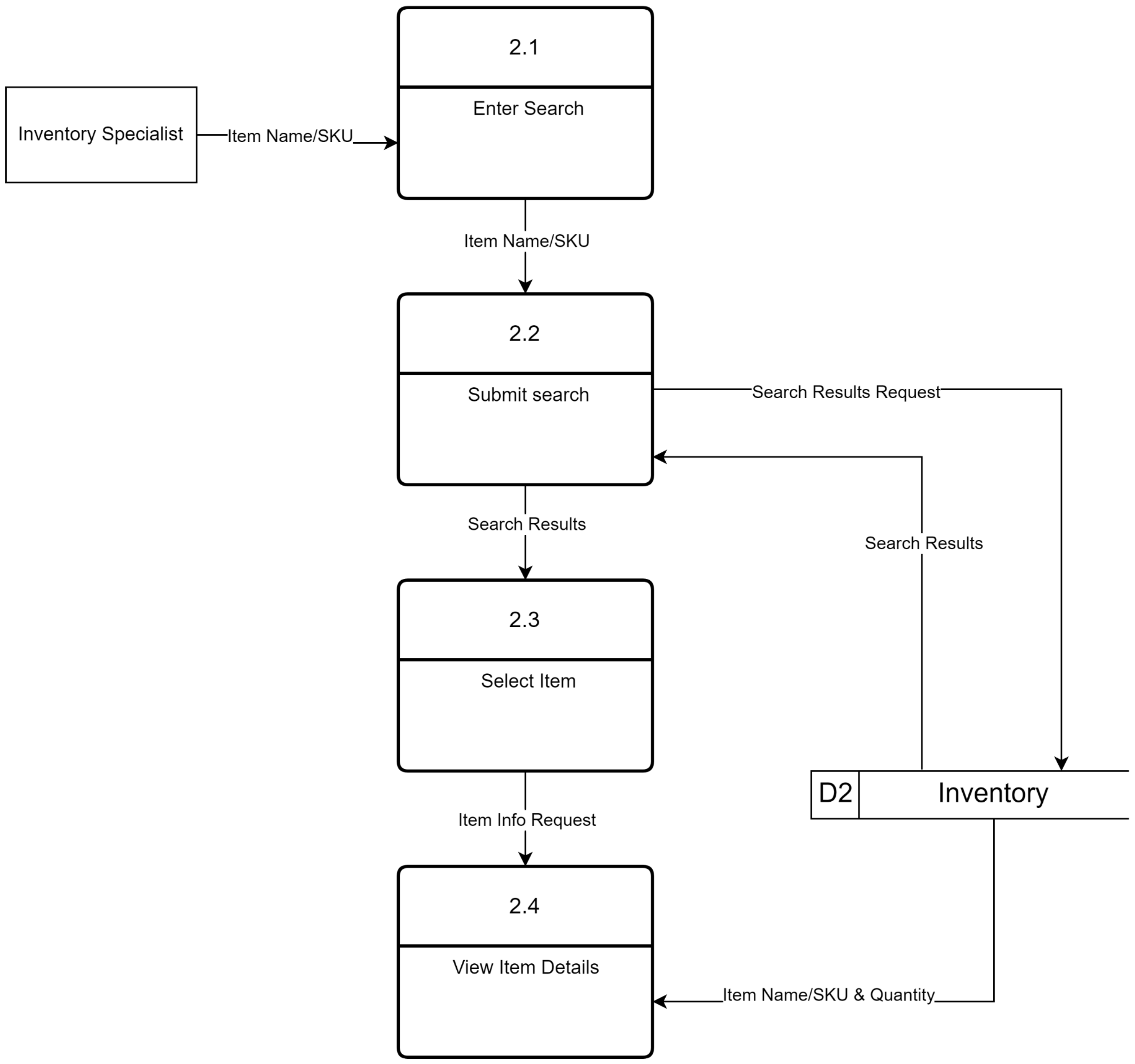
Level 0 DFD


1
Authenticate User
Login Credentials
Validated Credentials
Login ID and Password
System Access
D1
Employee
2
Search
Item Name/SKU
Search Request
Search Results
Item Name/SKU and Quantity
Sorted Inventory Records
Category Information Report
Unsorted Inventory Records
Category Information Report
3
Browse
Inventory Records Request
"Sort by Category" Request
Inventory Records Request
Unsorted Inventory Records
Transaction Record
D3
IMS Logs
Item Record
Inventory Specialist
Item Details/SKU
Add Item Request
4
Add Items
Item Added
D2
Inventory
Item Record
Delete Item Request
5
Delete Items
Item Details/SKU
Inventory Records
Inventory Records Request
Item Deleted
Stored Inventory Records
6
Locate Items
Item ID or Description
Final Location Information (Aisle, Bay, & Shelf)
Data Acknowledgement of Confirmation Request
Item Data and Current Location
Query for Item Data
7
Track Last Update/Change
User Input
Update History
Fetch Product Information
Last Update/Changes
8
Update Item Information
Update Request
Update Item Information

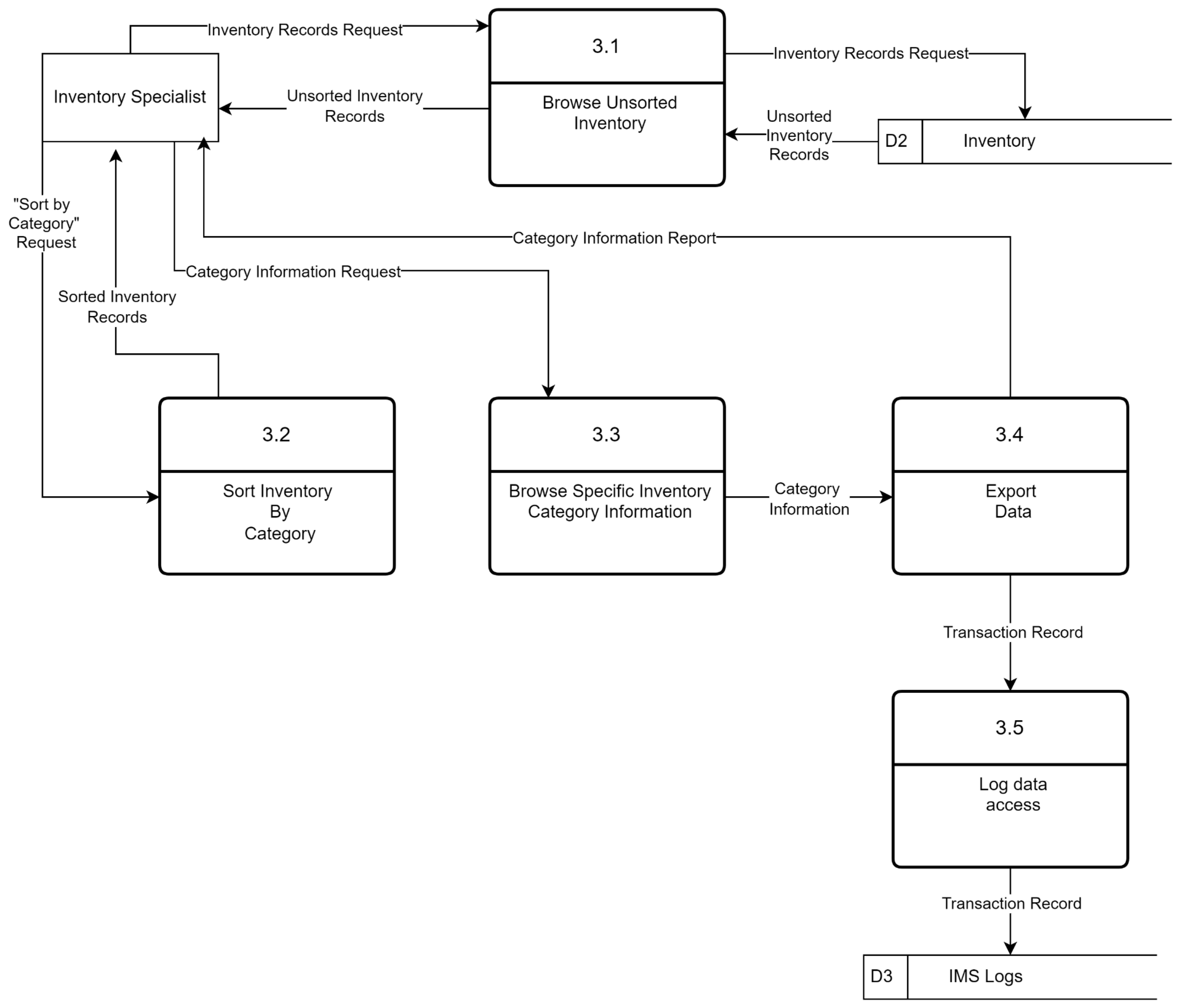
Level 1 DFD: UC#1 - Authenticate User



Level 1 DFD: UC#2 - Search

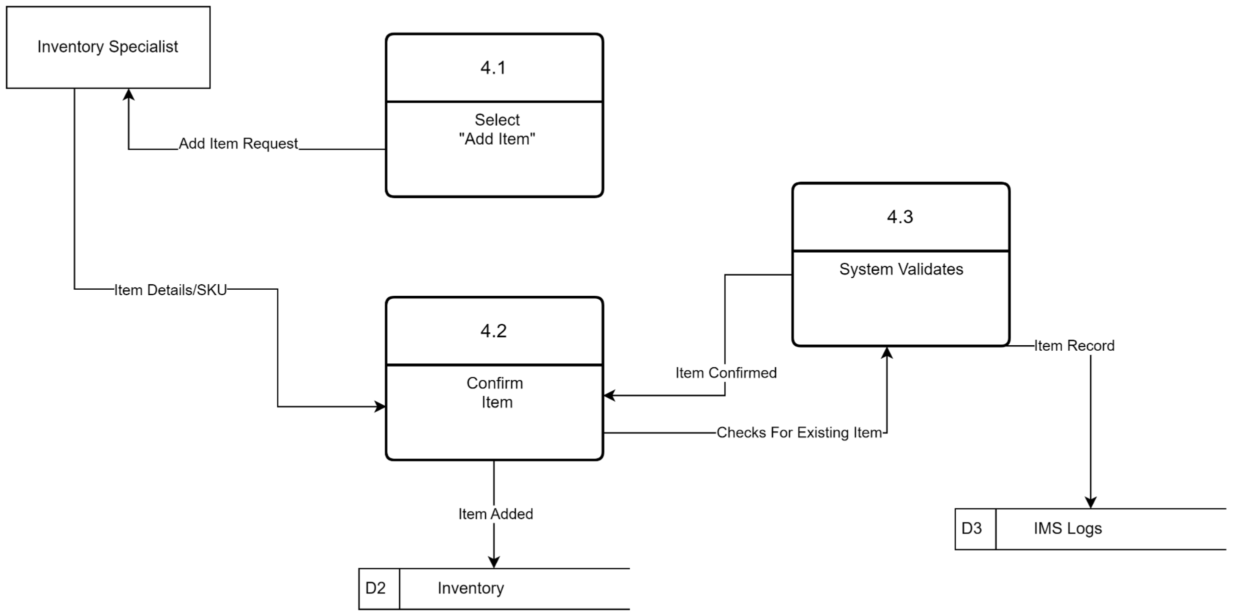


Level 1 DFD: UC#3 - Browse

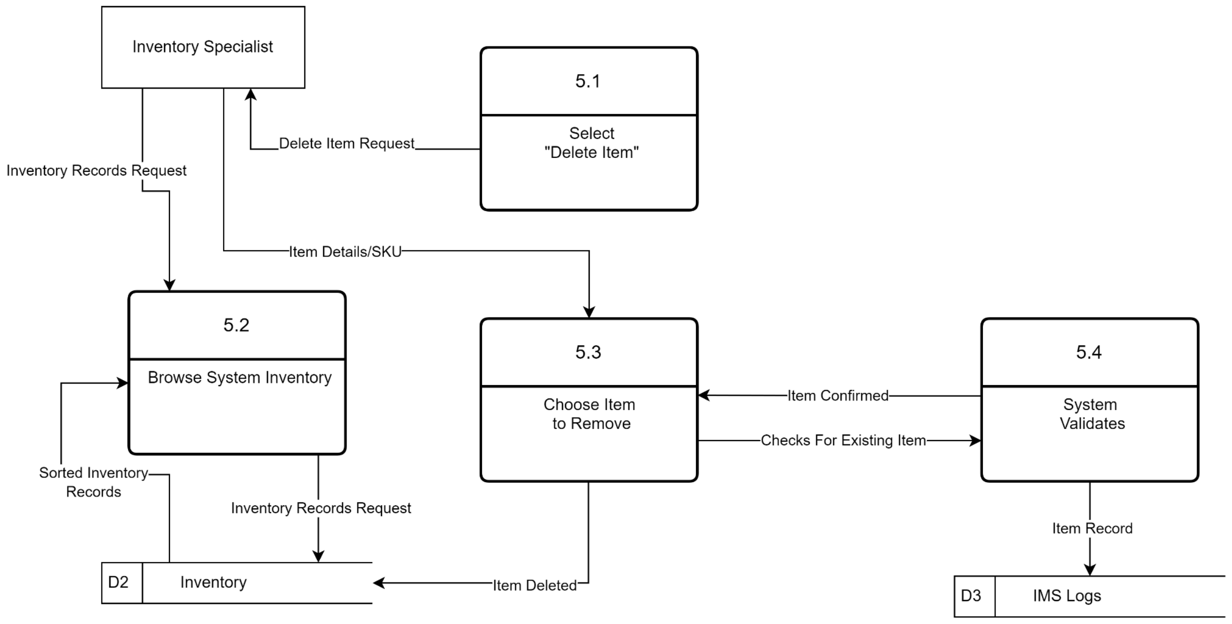


Inventory Specialist
Inventory Records Request
3.1
Browse Unsorted Inventory
Unsorted Inventory Records
Inventory Records Request
Unsorted Inventory Records
D2
Inventory
"Sort by Category" Request
Sorted Inventory Records
Category Information Request
Category Information Report
3.2
Sort Inventory By Category
3.3
Browse Specific Inventory Category Information
Category Information
3.4
Export Data
Transaction Record
3.5
Log data access
Transaction Record
D3
IMS Logs

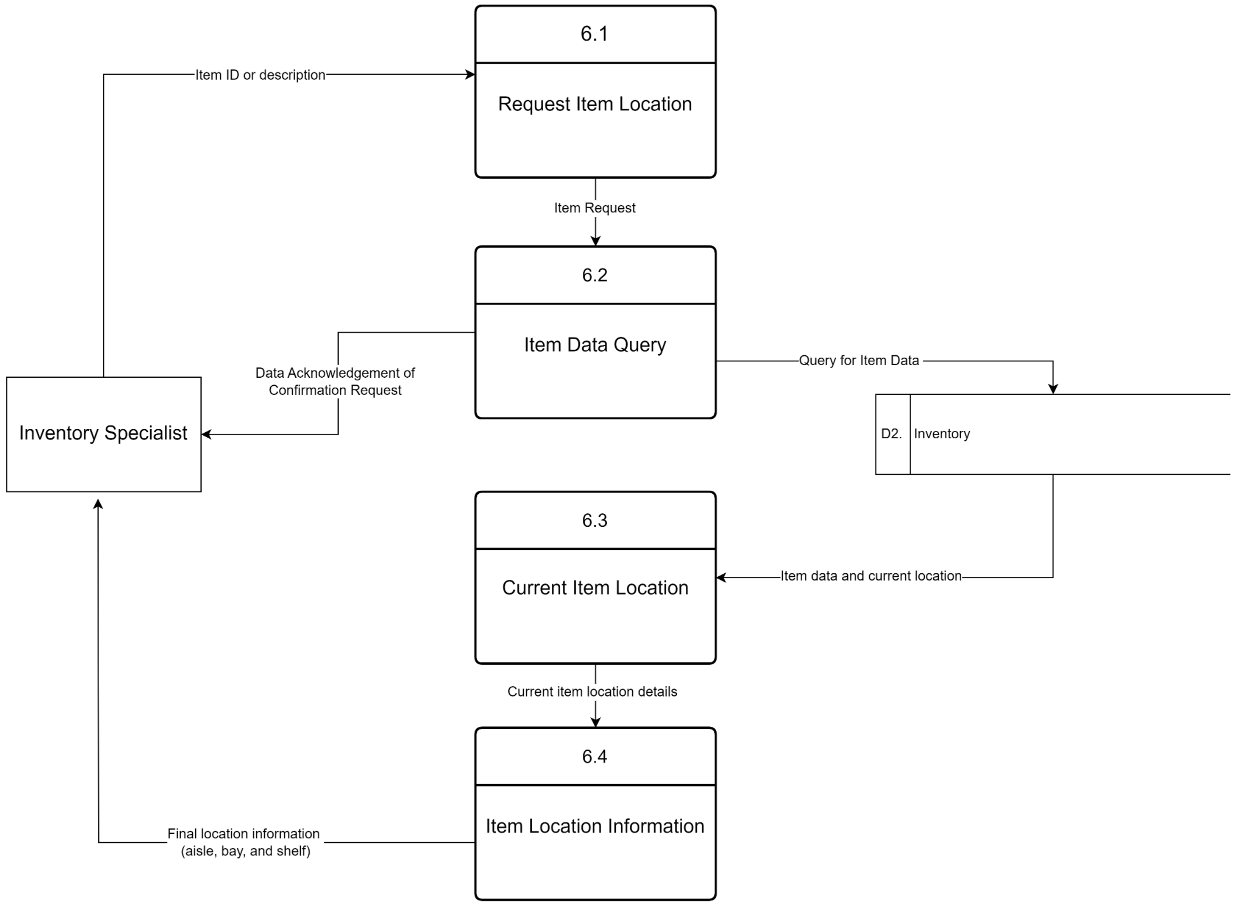
Level 1 DFD: UC#4 - Add Items



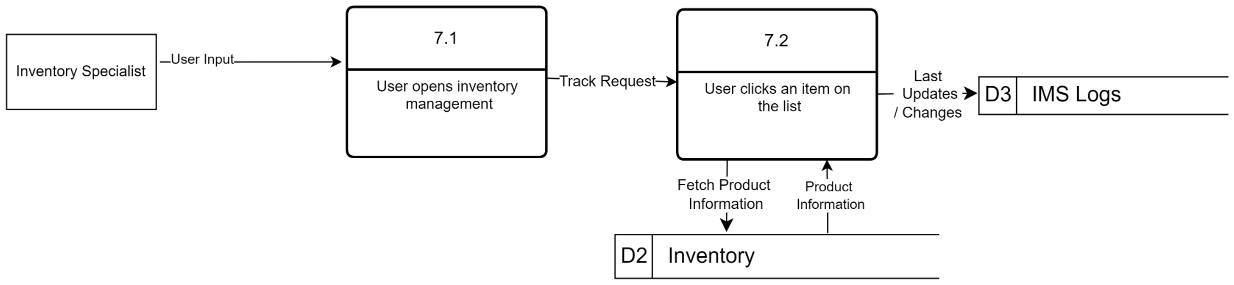
Level 1 DFD: UC#5 - Delete Items



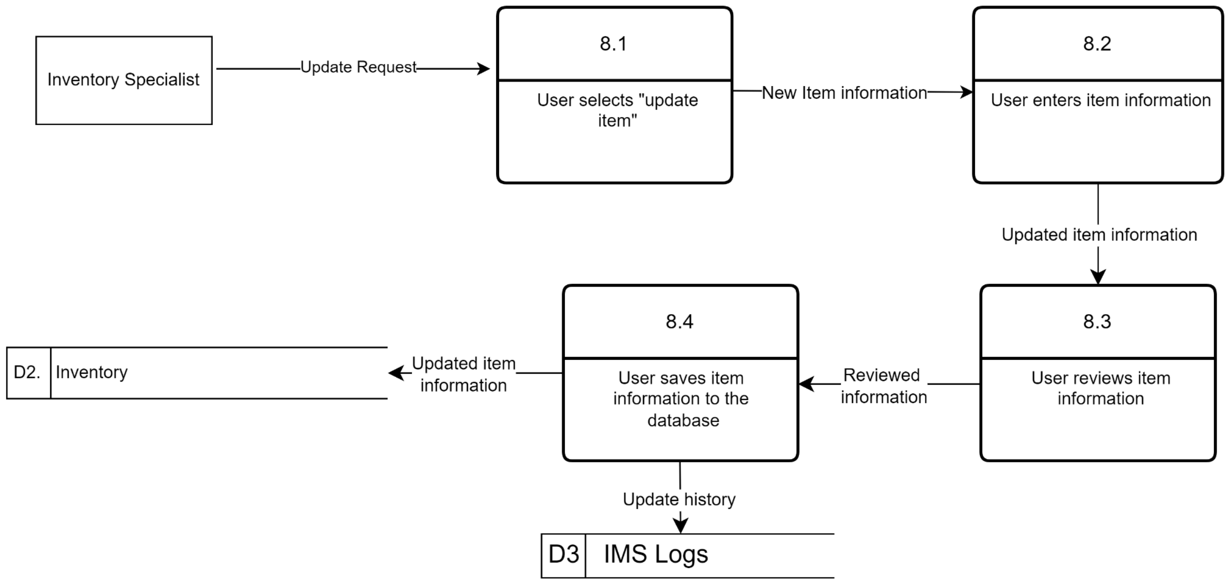
Level 1 DFD: UC#6 - Locate Items


6.1
Request Item Location
Item ID or description
Item Request
6.2
Item Data Query
Data Acknowledgement of Confirmation Request
Query for Item Data
Inventory Specialist
D2.
Inventory
6.3
Current Item Location
Item data and current location
Current item location details
6.4
Item Location Information
Final location information (aisle, bay, and shelf)

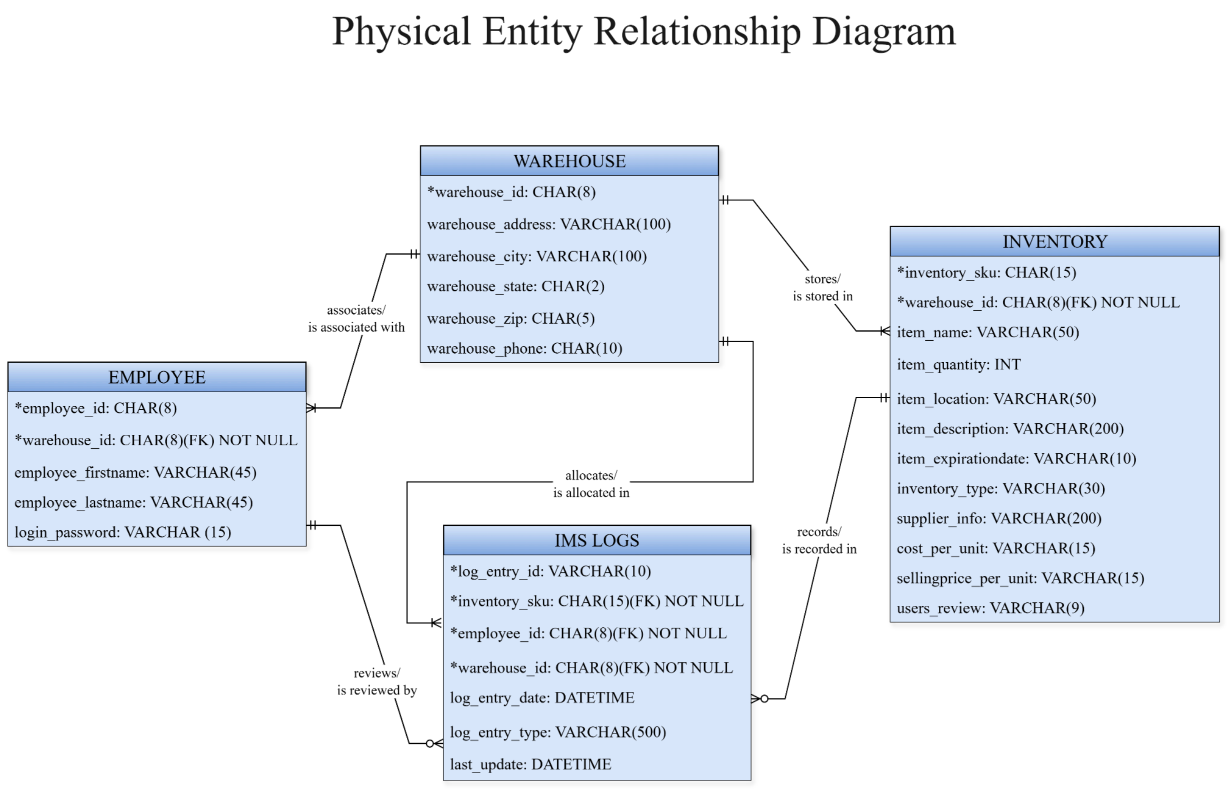
Level 1 DFD: UC#7 - Track Last Update/Change



Level 1 DFD: UC#8 - Update Item Information



Physical Entity Relationship Diagram

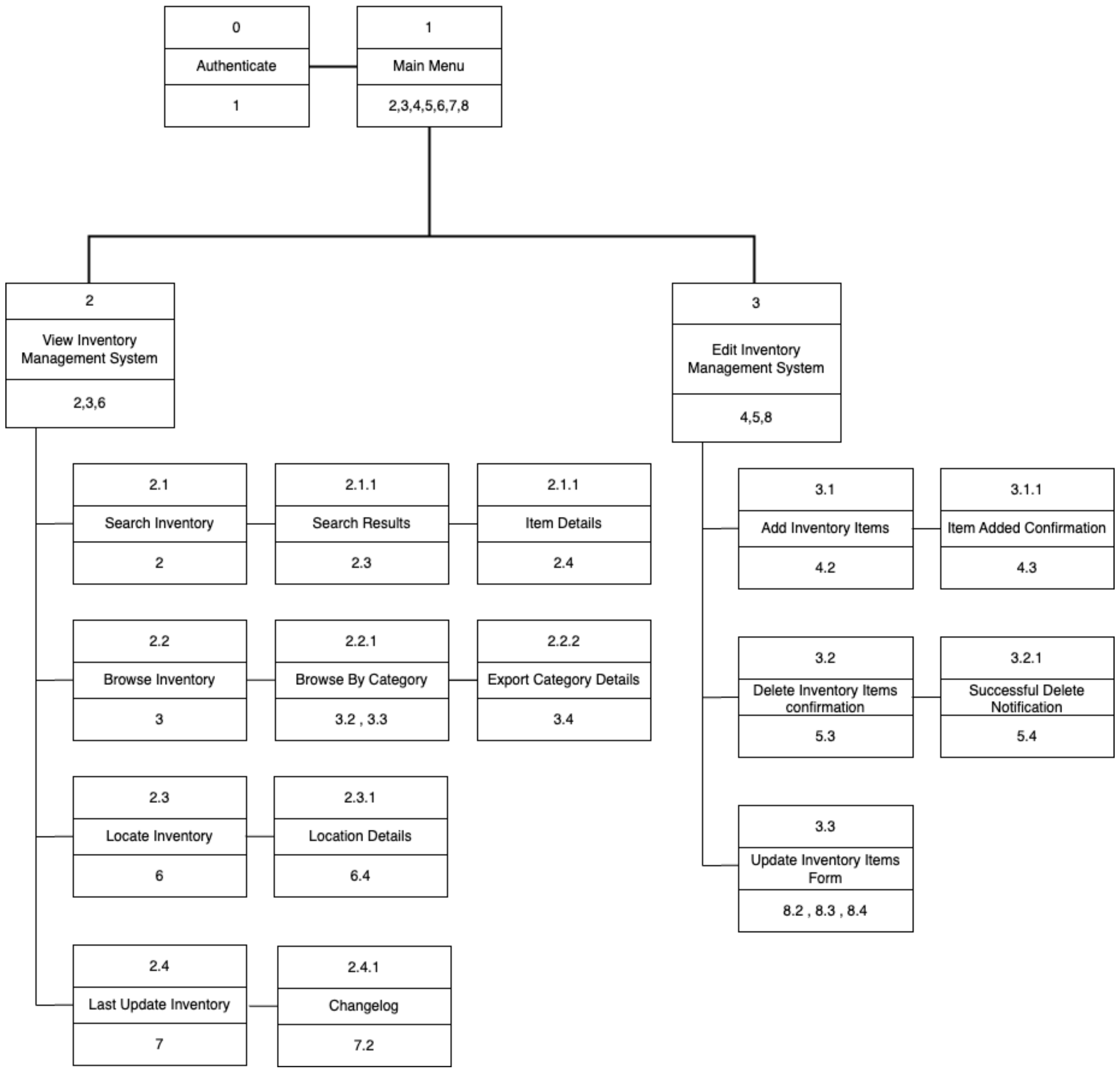

Physical Entity Relationship Diagram
WAREHOUSE
*warehouse_id: CHAR(8)
warehouse_address: VARCHAR(100)
warehouse_city: VARCHAR(100)
warehouse_state: CHAR(2)
warehouse_zip: CHAR(5)
warehouse_phone: CHAR(10)
INVENTORY
*inventory_sku: CHAR(15)
*warehouse_id: CHAR(8)(FK) NOT NULL
item_name: VARCHAR(50)
item_quantity: INT
item_location: VARCHAR(50)
item_description: VARCHAR(200)
item_expirationdate: VARCHAR(10)
inventory_type: VARCHAR(30)
supplier_info: VARCHAR(200)
cost_per_unit: VARCHAR(15)
sellingprice_per_unit: VARCHAR(15)
users_review: VARCHAR(9)
EMPLOYEE
*employee_id: CHAR(8)
*warehouse_id: CHAR(8)(FK) NOT NULL
employee_firstname: VARCHAR(45)
employee_lastname: VARCHAR(45)
login_password: VARCHAR (15)
IMS LOGS
*log_entry_id: VARCHAR(10)
*inventory_sku: CHAR(15)(FK) NOT NULL
*employee_id: CHAR(8)(FK) NOT NULL
*warehouse_id: CHAR(8)(FK) NOT NULL
log_entry_date: DATETIME
log_entry_type: VARCHAR(500)
last_update: DATETIME
associates/
is associated with
stores/
is stored in
allocates/
is allocated in
records/
is recorded in
reviews/
is reviewed by

Data Dictionary

Field Name	Data Type	Data Format	Field Size	Description	Example
employee_id	CHAR	NNNNNNNN	8	Unique number ID for employees	12345678
employee_firstname	VARCHAR		45	Employee’s first name	John
employee_lastname	VARCHAR		45	Employee’s surname	Smith
login_pasword	VARCHAR		15	Passcode accompanied with user Login ID	CSUFTitans2024
warehouse_id	CHAR	NNNNNNNN	8	Unique number ID for warehouse	00000000
warehouse_address	VARCHAR		100	Warehouse street # & address	123 Business Blvd
warehouse_city	VARCHAR		2	Warehouse city	Fullerton
warehouse_state	CHAR	NN	2	Warehouse state	CA
warehouse_zip	CHAR	NNNNN	5	Warehouse zip code	55577
warehouse_phone	CHAR	NNNNNNNNNN	10	Warehouse's main phone number	1234567890
log_entry_id	VARCHAR	NNNNNNNNNN	10	Unique ID for log entries	0000000000
inventory_sku	CHAR	NNNNNNNNNN NNNNN	15	Item’s stock keeping unit number to track stock level	000000000000000
log_entry_date	DATETIME	YYYY-MM-DD HH:MI:SS		Date an activity entry was posted onto the log	10/21/2024 - 01:17:54
item_name	VARCHAR		50	Product name/title	T-shirt
item_quantity	INT	NNNNNN	6	Amount of items that are available	100 or 100,000
item_location	VARCHAR		50	A specific location in a warehouse	Aisle 10
log_entry_type	VARCHAR		500	Details on the log entry	Item moved
last_update	DATETIME	YYYY-MM-DD HH:MI:SS		Time of last update	11/01/2024 - 22:11:33
item_description	VARCHAR		200	Details about product	Black color, XS
item_expirationdate	DATA	MM/DD/YYYY		Date of item’s expiration	11/02/2024
inventory_type	VARCHAR		30	FERT, HALB, WIP, etc…	Semifinished Goods
supplier_info	VARCHAR		200	Supplier’s name, number,.. etc	Lowe’s 888-999-1111
cost_per_unit	VARCHAR	$####.##	15	Original costs of good	$10.99
sellingprice_per_unit	VARCHAR	$####.##	15	Selling price per good	$75.95
users_review	TINYINT	# star	7	User review from 1 to 5 star	2 stars

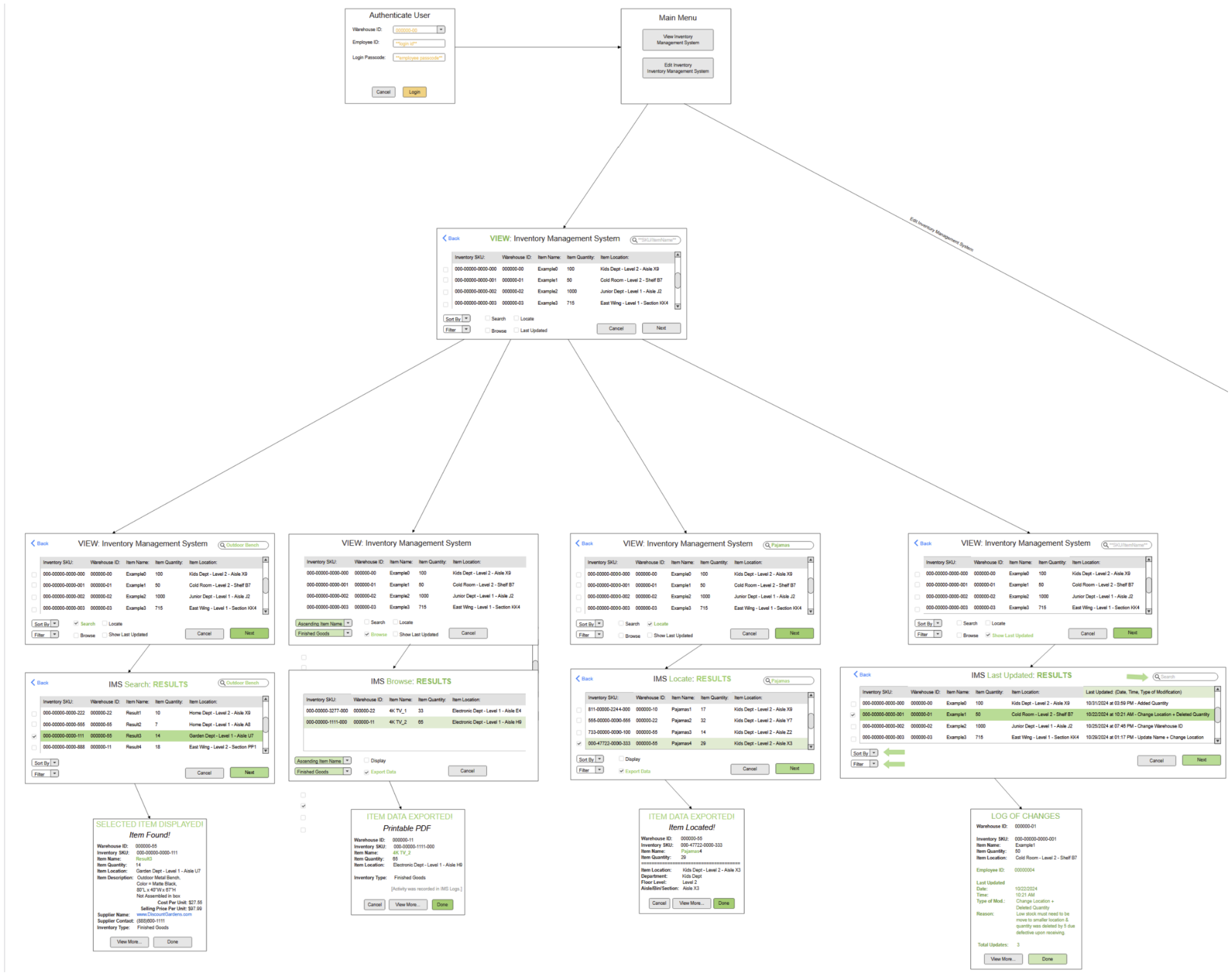
Virtual User Interface

Interface Structure Diagram



Prototype - Storyboarding

<u>Part 1: View Inventory Management System</u>



	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
☐	000-00000-0000-000	000000-00	Example0	100	Kids Dept - Level 2 - Aisle X9
☐	000-00000-0000-001	000000-01	Example1	50	Cold Room - Level 2 - Shelf B7
☐	000-00000-0000-002	000000-02	Example2	1000	Junior Dept - Level 1 - Aisle J2
☐	000-00000-0000-003	000000-03	Example3	715	East Wing - Level 1 - Section KK4

IMS Search: RESULTS

	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
☐	000-00000-0000-222	000000-22	Result1	10	Home Dept - Level 2 - Aisle X9
☐	000-00000-0000-555	000000-55	Result2	7	Home Dept - Level 1 - Aisle A8
✓	000-00000-0000-111	000000-55	Result3	14	Garden Dept - Level 1 - Aisle U7
☐	000-00000-0000-888	000000-11	Result4	18	East Wing - Level 2 - Section PP1

SELECTED ITEM DISPLAYED!
Item Found!

Warehouse ID:	000000-55
Inventory SKU:	000-00000-0000-111
Item Name:	Result3
Item Quantity:	14
Item Location:	Garden Dept - Level 1 - Aisle U7
Item Description:	Outdoor Metal Bench, Color = Matte Black, 80"L x 40"W x 67"H Not Assembled in box
Cost Per Unit:	$27.55
Selling Price Per Unit:	$97.99
Supplier Name:	www.DiscountGardens.com
Supplier Contact:	(888)600-1111
Inventory Type:	Finished Goods

IMS Browse: RESULTS

Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
000-00000-3277-000	000000-22	4K TV_1	33	Electronic Dept - Level 1 - Aisle E4
000-00000-1111-000	000000-11	4K TV_2	65	Electronic Dept - Level 1 - Aisle H9

ITEM DATA EXPORTED!
Printable PDF

Warehouse ID:	000000-11
Inventory SKU:	000-00000-1111-000
Item Name:	4K TV_2
Item Quantity:	65
Item Location:	Electronic Dept - Level 1 - Aisle H9
Inventory Type:	Finished Goods

[Activity was recorded in IMS Logs.]

IMS Locate: RESULTS

	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
☐	811-00000-2244-000	000000-10	Pajamas1	17	Kids Dept - Level 2 - Aisle X9
☐	555-00000-0000-555	000000-22	Pajamas2	32	Kids Dept - Level 2 - Aisle Y7
☐	733-00000-0000-100	000000-55	Pajamas3	14	Kids Dept - Level 2 - Aisle Z2
✓	000-47722-0000-333	000000-55	Pajamas4	29	Kids Dept - Level 2 - Aisle X3

ITEM DATA EXPORTED!
Item Located!

Warehouse ID:	000000-55
Inventory SKU:	000-47722-0000-333
Item Name:	Pajamas4
Item Quantity:	29
Item Location:	Kids Dept - Level 2 - Aisle X3
Department:	Kids Dept
Floor Level:	Level 2
Aisle/Bin/Section:	Aisle X3

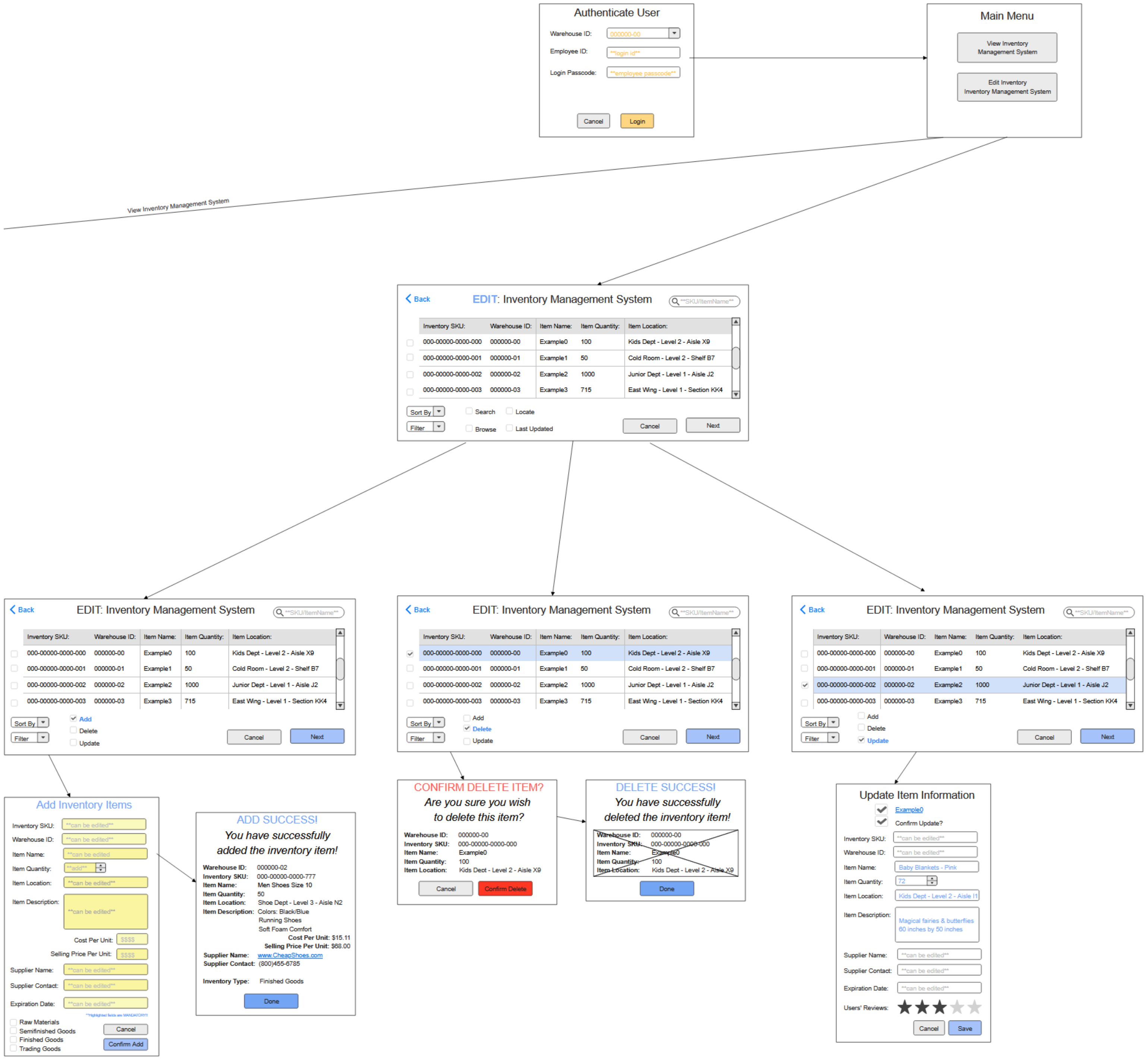
IMS Last Updated: RESULTS

	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:	Last Updated: (Date, Time, Type of Modification)
☐	000-00000-0000-000	000000-00	Example0	100	Kids Dept - Level 2 - Aisle X9	10/31/2024 at 03:59 PM - Added Quantity
✓	000-00000-0000-001	000000-01	Example1	50	Cold Room - Level 2 - Shelf B7	10/22/2024 at 10:21 AM - Change Location + Deleted Quantity
☐	000-00000-0000-002	000000-02	Example2	1000	Junior Dept - Level 1 - Aisle J2	10/25/2024 at 07:46 PM - Change Warehouse ID
☐	000-00000-0000-003	000000-03	Example3	715	East Wing - Level 1 - Section KK4	10/29/2024 at 01:17 PM - Update Name + Change Location

LOG OF CHANGES

Warehouse ID:	000000-01
Inventory SKU:	000-00000-0000-001
Item Name:	Example1
Item Quantity:	50
Item Location:	Cold Room - Level 2 - Shelf B7
Employee ID:	00000004
Last Updated	
Date:	10/22/2024
Time:	10:21 AM
Type of Mod.:	Change Location + Deleted Quantity
Reason:	Low stock must need to be move to smaller location & quantity was deleted by 5 due defective upon receiving.
Total Updates:	3

Part 2: Edit Inventory Management System



Authenticate User
Warehouse ID: 000000-00
Employee ID: **login id**
Login Passcode: **employee passcode**
Cancel
Login
Main Menu
View Inventory Management System
Edit Inventory Inventory Management System
View Inventory Management System
Back
EDIT: Inventory Management System
SKU/ItemName
Sort By
Filter
Search
Locate
Browse
Last Updated
Cancel
Next

	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
☐	000-00000-0000-000	000000-00	Example0	100	Kids Dept - Level 2 - Aisle X9
☐	000-00000-0000-001	000000-01	Example1	50	Cold Room - Level 2 - Shelf B7
☐	000-00000-0000-002	000000-02	Example2	1000	Junior Dept - Level 1 - Aisle J2
☐	000-00000-0000-003	000000-03	Example3	715	East Wing - Level 1 - Section KK4

Back
EDIT: Inventory Management System
SKU/ItemName
Sort By
Filter
✓ Add
☐ Delete
☐ Update
Cancel
Next

	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
☐	000-00000-0000-000	000000-00	Example0	100	Kids Dept - Level 2 - Aisle X9
☐	000-00000-0000-001	000000-01	Example1	50	Cold Room - Level 2 - Shelf B7
☐	000-00000-0000-002	000000-02	Example2	1000	Junior Dept - Level 1 - Aisle J2
☐	000-00000-0000-003	000000-03	Example3	715	East Wing - Level 1 - Section KK4

Add Inventory Items

Inventory SKU:	**can be edited**
Warehouse ID:	**can be edited**
Item Name:	**can be edited
Item Quantity:	**add**
Item Location:	**can be edited**
Item Description:	**can be edited**
Cost Per Unit:	$$$$
Selling Price Per Unit:	$$$$
Supplier Name:	**can be edited**
Supplier Contact:	**can be edited**
Expiration Date:	**can be edited**

**Highlighted fields are MANDATORY!!

☐ Raw Materials
☐ Semifinished Goods
☐ Finished Goods
☐ Trading Goods

ADD SUCCESS!

You have successfully added the inventory item!

Warehouse ID: 000000-02
Inventory SKU: 000-00000-0000-777
Item Name: Men Shoes Size 10
Item Quantity: 50
Item Location: Shoe Dept - Level 3 - Aisle N2
Item Description: Colors: Black/Blue
Running Shoes
Soft Foam Comfort
Cost Per Unit: $15.11
Selling Price Per Unit: $68.00
Supplier Name: www.CheapShoes.com
Supplier Contact: (800)455-6785

Inventory Type: Finished Goods

	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
✓	000-00000-0000-000	000000-00	Example0	100	Kids Dept - Level 2 - Aisle X9
☐	000-00000-0000-001	000000-01	Example1	50	Cold Room - Level 2 - Shelf B7
☐	000-00000-0000-002	000000-02	Example2	1000	Junior Dept - Level 1 - Aisle J2
☐	000-00000-0000-003	000000-03	Example3	715	East Wing - Level 1 - Section KK4

CONFIRM DELETE ITEM?

Are you sure you wish to delete this item?

Warehouse ID: 000000-00
Inventory SKU: 000-00000-0000-000
Item Name: Example0
Item Quantity: 100
Item Location: Kids Dept - Level 2 - Aisle X9

DELETE SUCCESS!

You have successfully deleted the inventory item!

~~**Warehouse ID:** 000000-00~~
~~**Inventory SKU:** 000-00000-0000-000~~
~~**Item Name:** Example0~~
~~**Item Quantity:** 100~~
~~**Item Location:** Kids Dept - Level 2 - Aisle X9~~

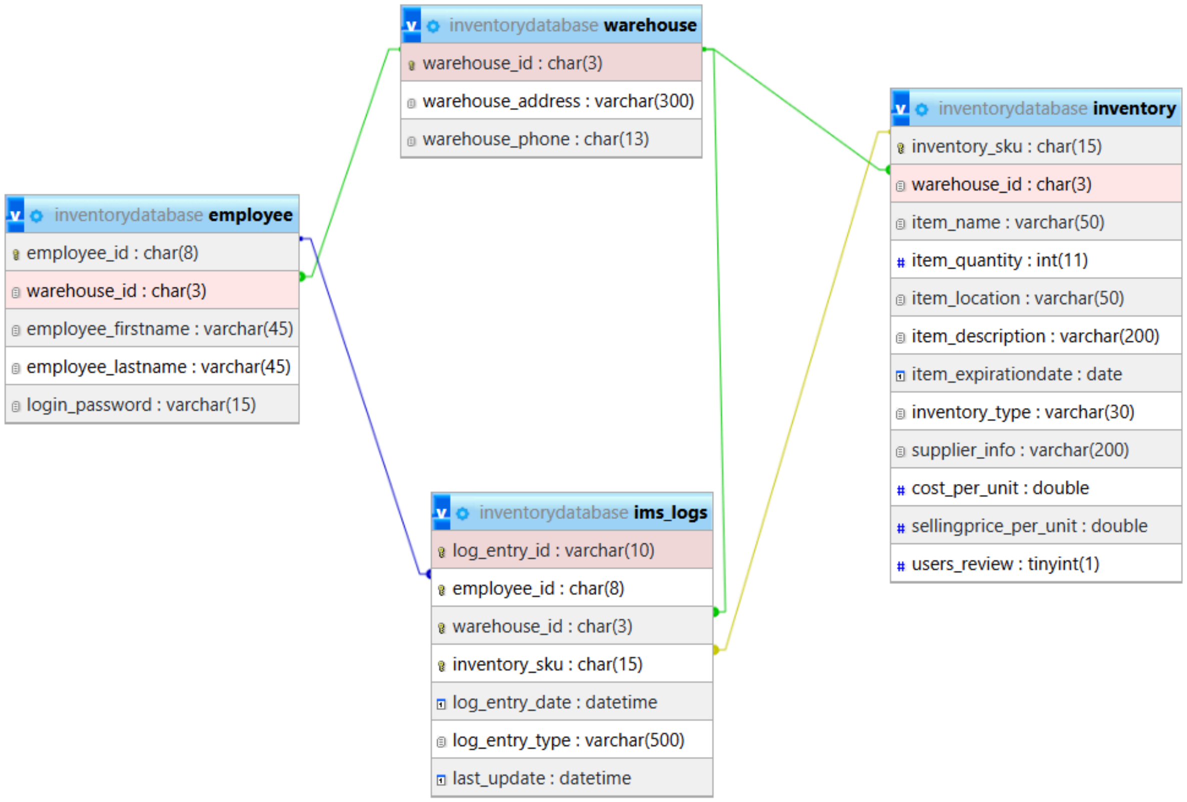
	Inventory SKU:	Warehouse ID:	Item Name:	Item Quantity:	Item Location:
☐	000-00000-0000-000	000000-00	Example0	100	Kids Dept - Level 2 - Aisle X9
☐	000-00000-0000-001	000000-01	Example1	50	Cold Room - Level 2 - Shelf B7
✓	000-00000-0000-002	000000-02	Example2	1000	Junior Dept - Level 1 - Aisle J2
☐	000-00000-0000-003	000000-03	Example3	715	East Wing - Level 1 - Section KK4

Update Item Information

✓ Example0
✓ Confirm Update?

Inventory SKU:	**can be edited**
Warehouse ID:	**can be edited**
Item Name:	Baby Blankets - Pink
Item Quantity:	72
Item Location:	Kids Dept - Level 2 - Aisle I1
Item Description:	Magical fairies & butterflies 60 inches by 50 inches
Supplier Name:	**can be edited**
Supplier Contact:	**can be edited**
Expiration Date:	**can be edited**
Users' Reviews:	★★★☆☆

phpMyAdmin Physical ERD



inventorydatabase **warehouse**
- warehouse_id : char(3)
- warehouse_address : varchar(300)
- warehouse_phone : char(13)

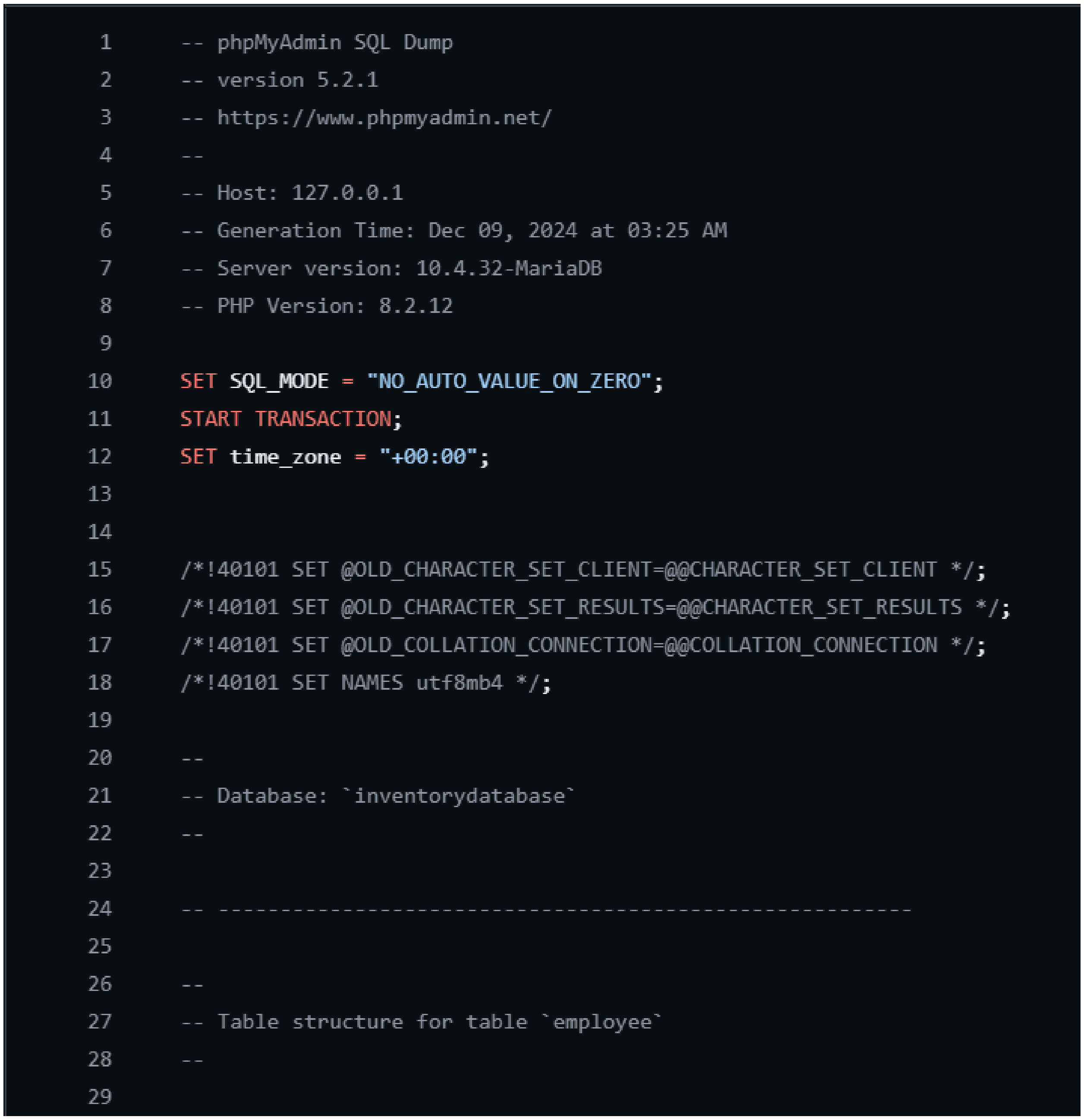
inventorydatabase **inventory**
- inventory_sku : char(15)
- warehouse_id : char(3)
- item_name : varchar(50)
- item_quantity : int(11)
- item_location : varchar(50)
- item_description : varchar(200)
- item_expirationdate : date
- inventory_type : varchar(30)
- supplier_info : varchar(200)
- cost_per_unit : double
- sellingprice_per_unit : double
- users_review : tinyint(1)

inventorydatabase **employee**
- employee_id : char(8)
- warehouse_id : char(3)
- employee_firstname : varchar(45)
- employee_lastname : varchar(45)
- login_password : varchar(15)

inventorydatabase **ims_logs**
- log_entry_id : varchar(10)
- employee_id : char(8)
- warehouse_id : char(3)
- inventory_sku : char(15)
- log_entry_date : datetime
- log_entry_type : varchar(500)
- last_update : datetime

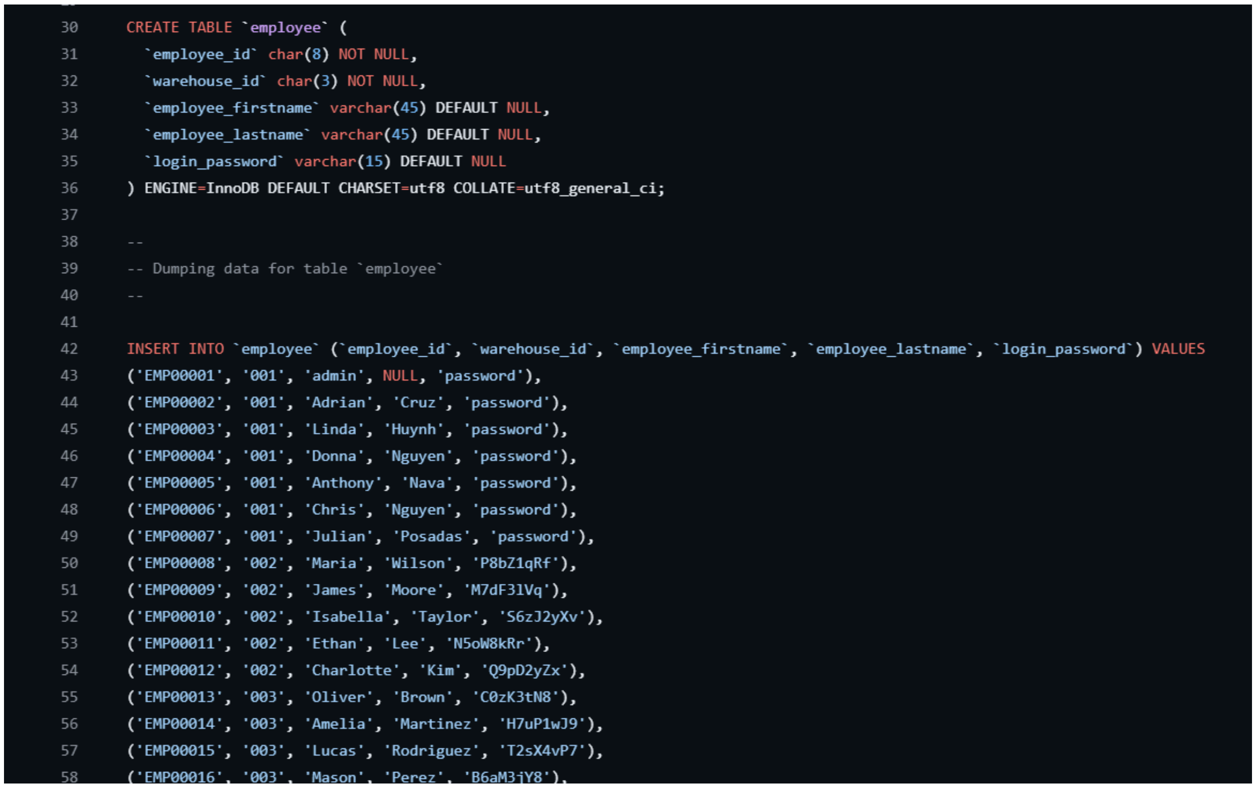
SQL Script

Initial Config



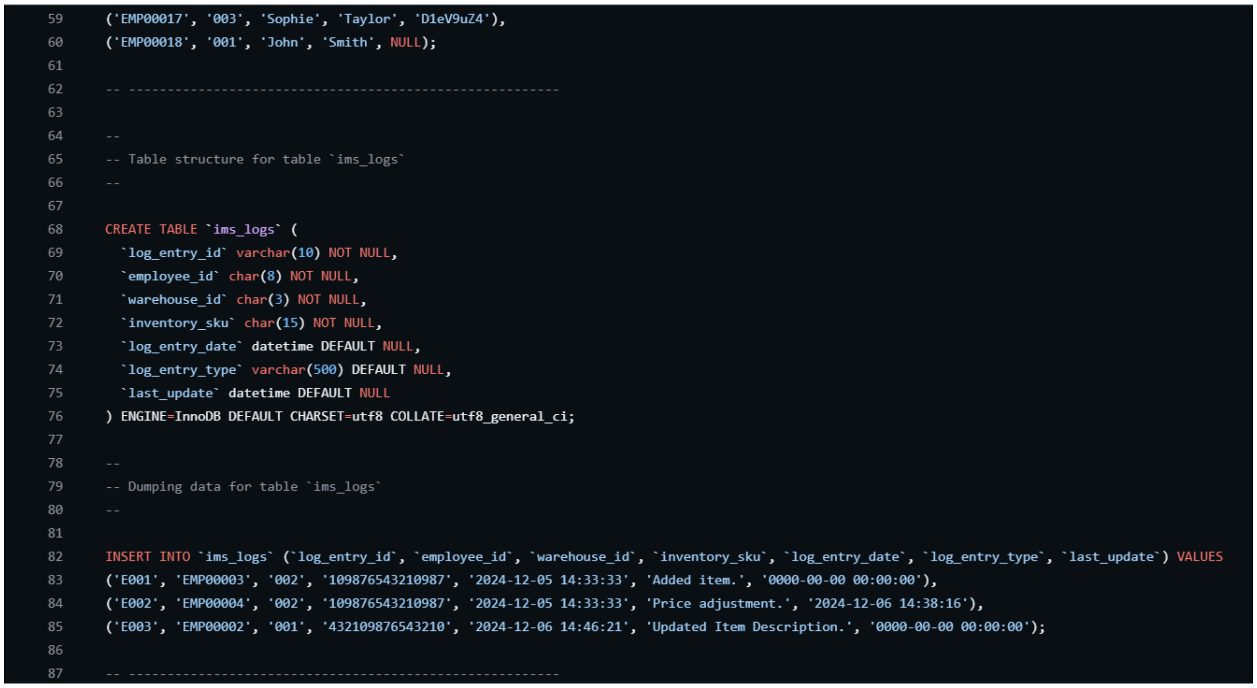
```sql
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Dec 09, 2024 at 03:25 AM
-- Server version: 10.4.32-MariaDB
-- PHP Version: 8.2.12

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `inventorydatabase`
--

-- --------------------------------------------------------

--
-- Table structure for table `employee`
--

```

Create Employee Table and Insert Data



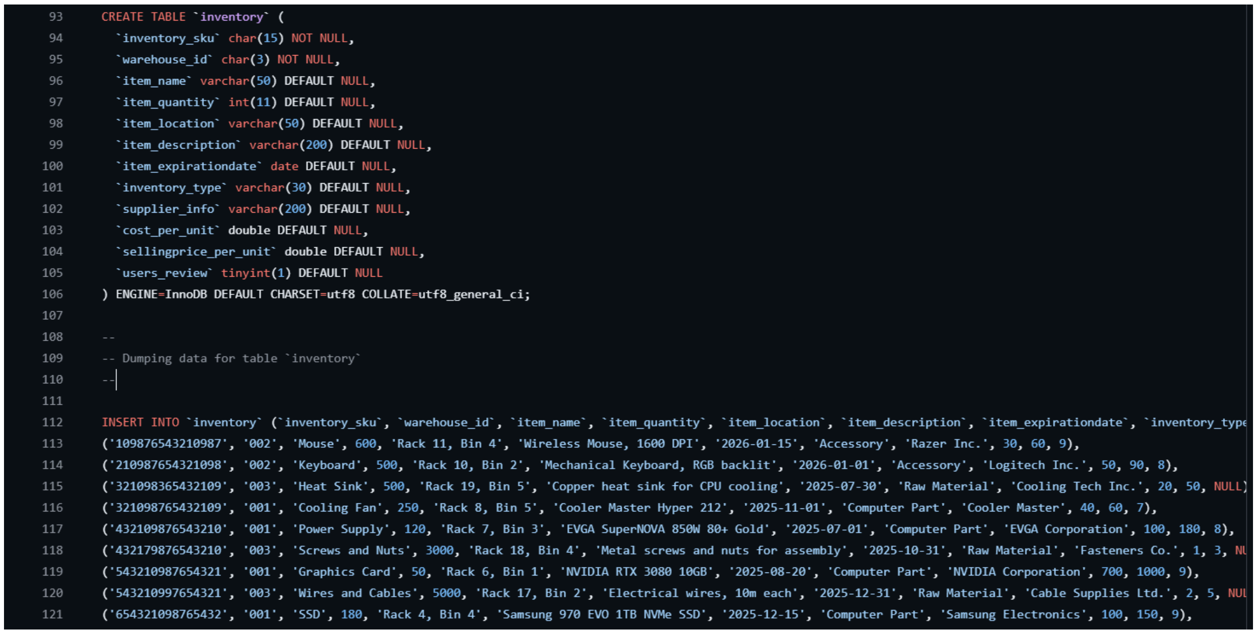
```sql
CREATE TABLE `employee` (
  `employee_id` char(8) NOT NULL,
  `warehouse_id` char(3) NOT NULL,
  `employee_firstname` varchar(45) DEFAULT NULL,
  `employee_lastname` varchar(45) DEFAULT NULL,
  `login_password` varchar(15) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

--
-- Dumping data for table `employee`
--

INSERT INTO `employee` (`employee_id`, `warehouse_id`, `employee_firstname`, `employee_lastname`, `login_password`) VALUES
('EMP00001', '001', 'admin', NULL, 'password'),
('EMP00002', '001', 'Adrian', 'Cruz', 'password'),
('EMP00003', '001', 'Linda', 'Huynh', 'password'),
('EMP00004', '001', 'Donna', 'Nguyen', 'password'),
('EMP00005', '001', 'Anthony', 'Nava', 'password'),
('EMP00006', '001', 'Chris', 'Nguyen', 'password'),
('EMP00007', '001', 'Julian', 'Posadas', 'password'),
('EMP00008', '002', 'Maria', 'Wilson', 'P8bZ1qRf'),
('EMP00009', '002', 'James', 'Moore', 'M7dF31Vq'),
('EMP00010', '002', 'Isabella', 'Taylor', 'S6zJ2yXv'),
('EMP00011', '002', 'Ethan', 'Lee', 'N5oW8kRr'),
('EMP00012', '002', 'Charlotte', 'Kim', 'Q9pD2yZx'),
('EMP00013', '003', 'Oliver', 'Brown', 'C0zK3tN8'),
('EMP00014', '003', 'Amelia', 'Martinez', 'H7uP1wJ9'),
('EMP00015', '003', 'Lucas', 'Rodriguez', 'T2sX4vP7'),
('EMP00016', '003', 'Mason', 'Perez', 'B6aM3jY8'),
```

Create ims_logs Table and Insert Data



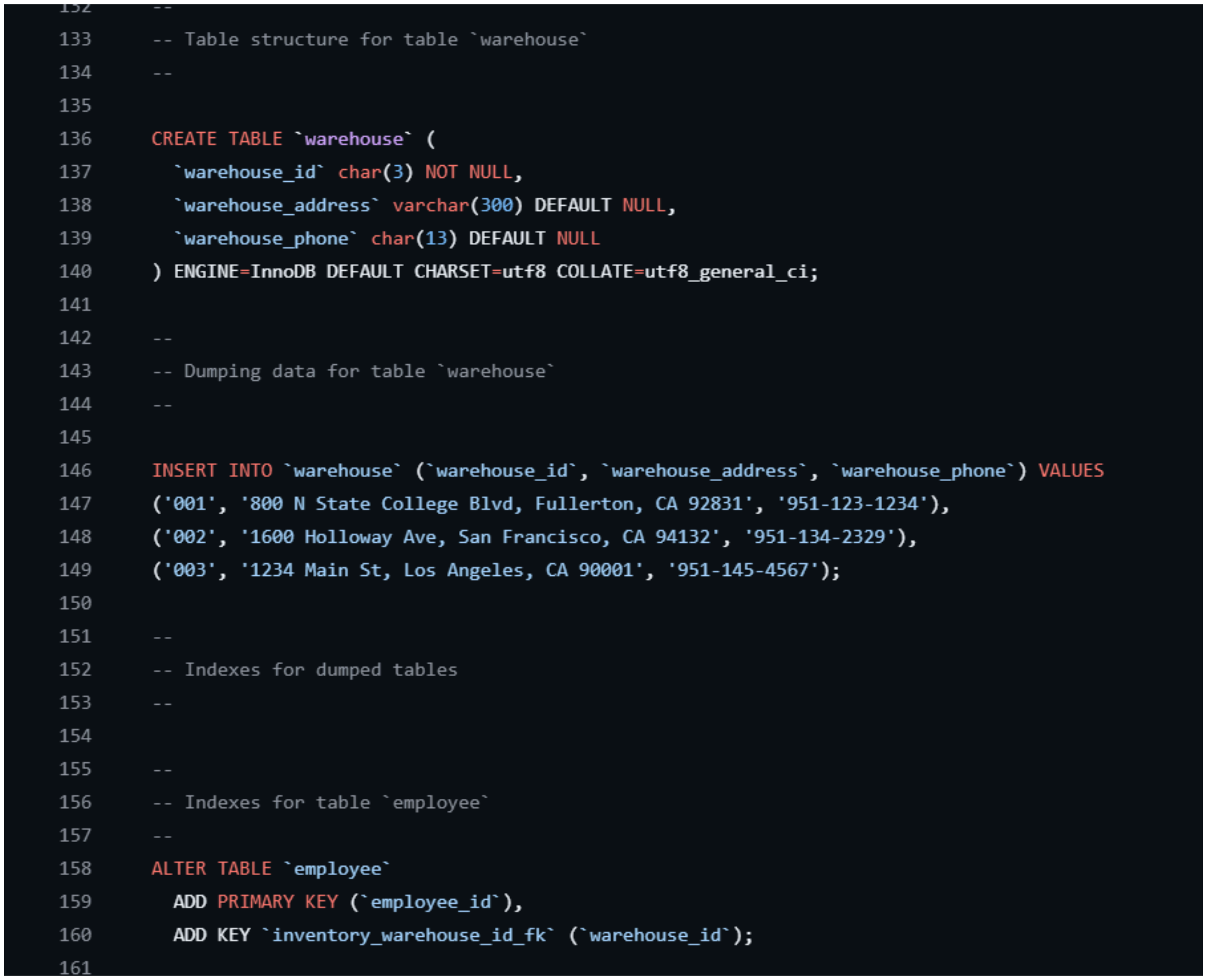
```sql
('EMP00017', '003', 'Sophie', 'Taylor', 'D1eV9uZ4'),
('EMP00018', '001', 'John', 'Smith', NULL);

-- --------------------------------------------------------

--
-- Table structure for table `ims_logs`
--

CREATE TABLE `ims_logs` (
  `log_entry_id` varchar(10) NOT NULL,
  `employee_id` char(8) NOT NULL,
  `warehouse_id` char(3) NOT NULL,
  `inventory_sku` char(15) NOT NULL,
  `log_entry_date` datetime DEFAULT NULL,
  `log_entry_type` varchar(500) DEFAULT NULL,
  `last_update` datetime DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

--
-- Dumping data for table `ims_logs`
--

INSERT INTO `ims_logs` (`log_entry_id`, `employee_id`, `warehouse_id`, `inventory_sku`, `log_entry_date`, `log_entry_type`, `last_update`) VALUES
('E001', 'EMP00003', '002', '109876543210987', '2024-12-05 14:33:33', 'Added item.', '0000-00-00 00:00:00'),
('E002', 'EMP00004', '002', '109876543210987', '2024-12-05 14:33:33', 'Price adjustment.', '2024-12-06 14:38:16'),
('E003', 'EMP00002', '001', '432109876543210', '2024-12-06 14:46:21', 'Updated Item Description.', '0000-00-00 00:00:00');

-- --------------------------------------------------------
```

Create Inventory Table and Insert Data



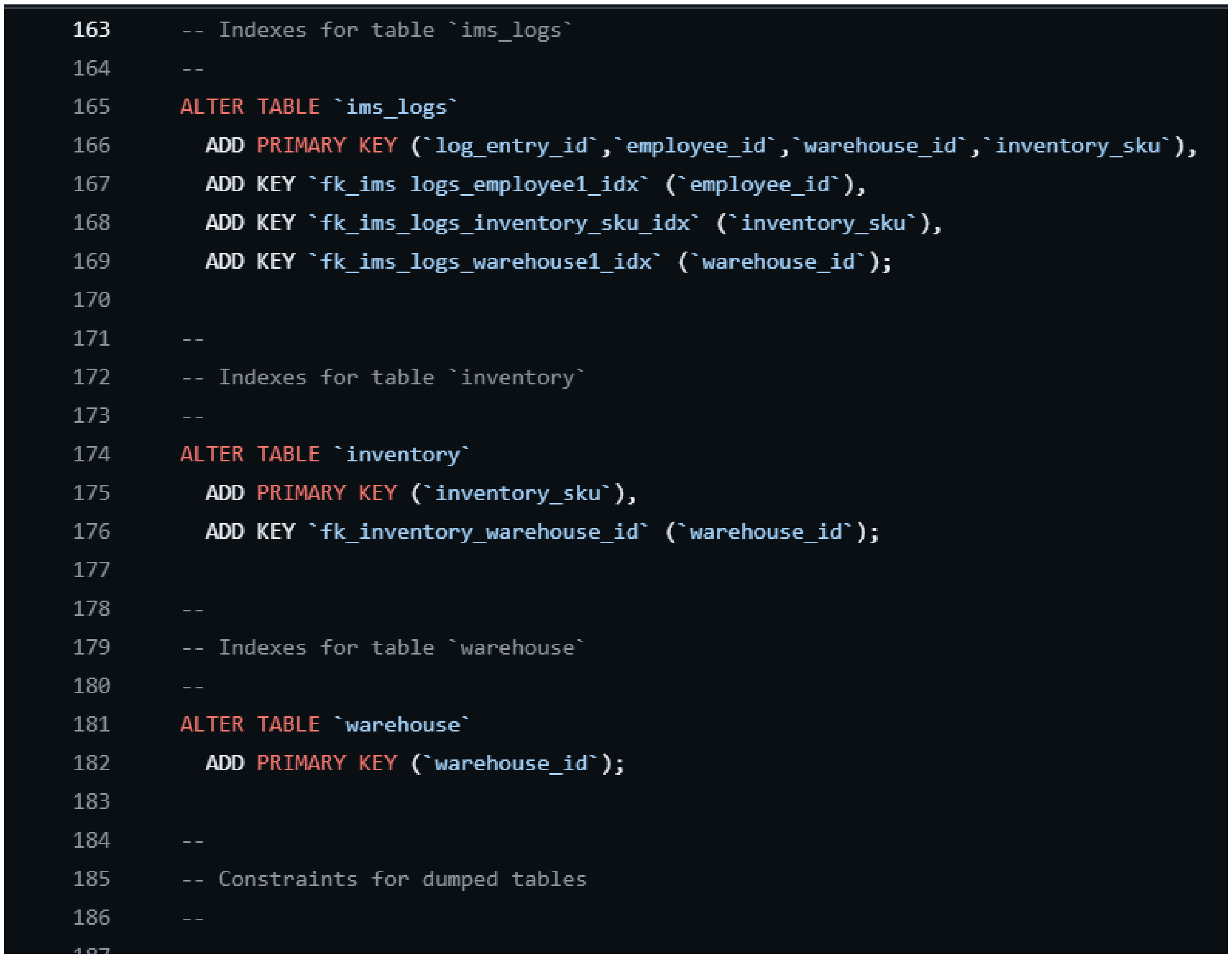
```sql
CREATE TABLE `inventory` (
  `inventory_sku` char(15) NOT NULL,
  `warehouse_id` char(3) NOT NULL,
  `item_name` varchar(50) DEFAULT NULL,
  `item_quantity` int(11) DEFAULT NULL,
  `item_location` varchar(50) DEFAULT NULL,
  `item_description` varchar(200) DEFAULT NULL,
  `item_expirationdate` date DEFAULT NULL,
  `inventory_type` varchar(30) DEFAULT NULL,
  `supplier_info` varchar(200) DEFAULT NULL,
  `cost_per_unit` double DEFAULT NULL,
  `sellingprice_per_unit` double DEFAULT NULL,
  `users_review` tinyint(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

--
-- Dumping data for table `inventory`
--

INSERT INTO `inventory` (`inventory_sku`, `warehouse_id`, `item_name`, `item_quantity`, `item_location`, `item_description`, `item_expirationdate`, `inventory_type
('109876543210987', '002', 'Mouse', 600, 'Rack 11, Bin 4', 'Wireless Mouse, 1600 DPI', '2026-01-15', 'Accessory', 'Razer Inc.', 30, 60, 9),
('210987654321098', '002', 'Keyboard', 500, 'Rack 10, Bin 2', 'Mechanical Keyboard, RGB backlit', '2026-01-01', 'Accessory', 'Logitech Inc.', 50, 90, 8),
('321098365432109', '003', 'Heat Sink', 500, 'Rack 19, Bin 5', 'Copper heat sink for CPU cooling', '2025-07-30', 'Raw Material', 'Cooling Tech Inc.', 20, 50, NULL)
('321098765432109', '001', 'Cooling Fan', 250, 'Rack 8, Bin 5', 'Cooler Master Hyper 212', '2025-11-01', 'Computer Part', 'Cooler Master', 40, 60, 7),
('432109876543210', '001', 'Power Supply', 120, 'Rack 7, Bin 3', 'EVGA SuperNOVA 850W 80+ Gold', '2025-07-01', 'Computer Part', 'EVGA Corporation', 100, 180, 8),
('432179876543210', '003', 'Screws and Nuts', 3000, 'Rack 18, Bin 4', 'Metal screws and nuts for assembly', '2025-10-31', 'Raw Material', 'Fasteners Co.', 1, 3, NU
('543210987654321', '001', 'Graphics Card', 50, 'Rack 6, Bin 1', 'NVIDIA RTX 3080 10GB', '2025-08-20', 'Computer Part', 'NVIDIA Corporation', 700, 1000, 9),
('543210997654321', '003', 'Wires and Cables', 5000, 'Rack 17, Bin 2', 'Electrical wires, 10m each', '2025-12-31', 'Raw Material', 'Cable Supplies Ltd.', 2, 5, NUL
('654321098765432', '001', 'SSD', 180, 'Rack 4, Bin 4', 'Samsung 970 EVO 1TB NVMe SSD', '2025-12-15', 'Computer Part', 'Samsung Electronics', 100, 150, 9),
```

Create Warehouse Table and Insert Data



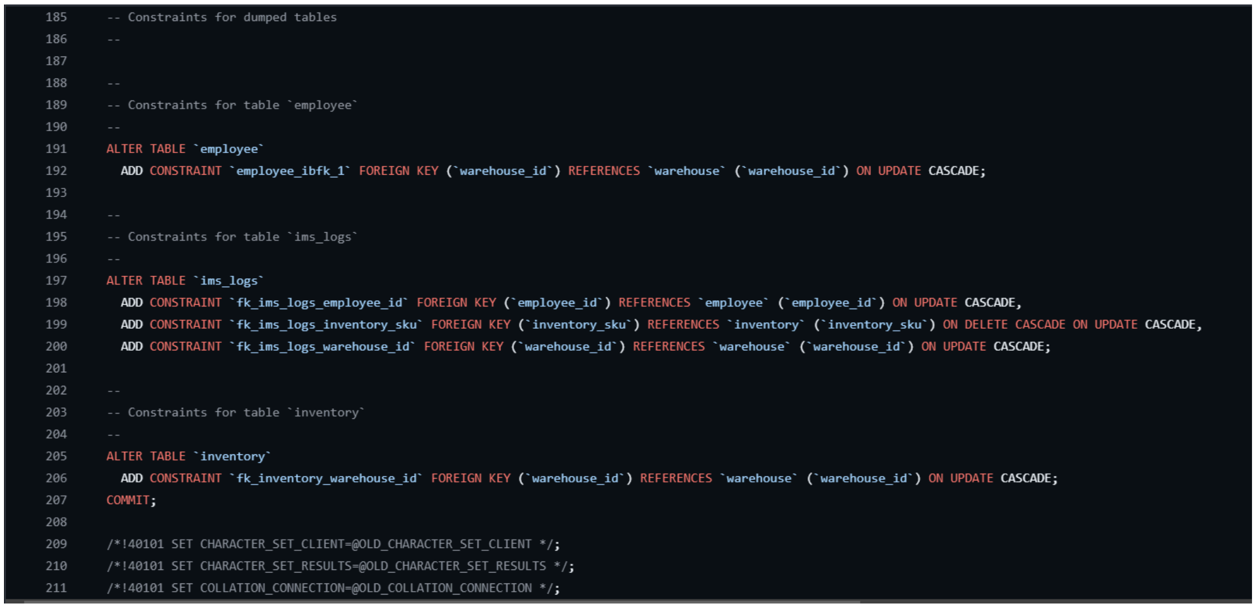
```
--
-- Table structure for table `warehouse`
--

CREATE TABLE `warehouse` (
  `warehouse_id` char(3) NOT NULL,
  `warehouse_address` varchar(300) DEFAULT NULL,
  `warehouse_phone` char(13) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

--
-- Dumping data for table `warehouse`
--

INSERT INTO `warehouse` (`warehouse_id`, `warehouse_address`, `warehouse_phone`) VALUES
('001', '800 N State College Blvd, Fullerton, CA 92831', '951-123-1234'),
('002', '1600 Holloway Ave, San Francisco, CA 94132', '951-134-2329'),
('003', '1234 Main St, Los Angeles, CA 90001', '951-145-4567');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `employee`
--
ALTER TABLE `employee`
  ADD PRIMARY KEY (`employee_id`),
  ADD KEY `inventory_warehouse_id_fk` (`warehouse_id`);

```

Indexes



```
-- Indexes for table `ims_logs`
--
ALTER TABLE `ims_logs`
  ADD PRIMARY KEY (`log_entry_id`,`employee_id`,`warehouse_id`,`inventory_sku`),
  ADD KEY `fk_ims logs_employee1_idx` (`employee_id`),
  ADD KEY `fk_ims_logs_inventory_sku_idx` (`inventory_sku`),
  ADD KEY `fk_ims_logs_warehouse1_idx` (`warehouse_id`);

--
-- Indexes for table `inventory`
--
ALTER TABLE `inventory`
  ADD PRIMARY KEY (`inventory_sku`),
  ADD KEY `fk_inventory_warehouse_id` (`warehouse_id`);

--
-- Indexes for table `warehouse`
--
ALTER TABLE `warehouse`
  ADD PRIMARY KEY (`warehouse_id`);

--
-- Constraints for dumped tables
--
```

Constraints



```
-- Constraints for dumped tables
--

--
-- Constraints for table `employee`
--
ALTER TABLE `employee`
  ADD CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`warehouse_id`) REFERENCES `warehouse` (`warehouse_id`) ON UPDATE CASCADE;

--
-- Constraints for table `ims_logs`
--
ALTER TABLE `ims_logs`
  ADD CONSTRAINT `fk_ims_logs_employee_id` FOREIGN KEY (`employee_id`) REFERENCES `employee` (`employee_id`) ON UPDATE CASCADE,
  ADD CONSTRAINT `fk_ims_logs_inventory_sku` FOREIGN KEY (`inventory_sku`) REFERENCES `inventory` (`inventory_sku`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `fk_ims_logs_warehouse_id` FOREIGN KEY (`warehouse_id`) REFERENCES `warehouse` (`warehouse_id`) ON UPDATE CASCADE;

--
-- Constraints for table `inventory`
--
ALTER TABLE `inventory`
  ADD CONSTRAINT `fk_inventory_warehouse_id` FOREIGN KEY (`warehouse_id`) REFERENCES `warehouse` (`warehouse_id`) ON UPDATE CASCADE;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

SQL Script Summary

This SQL Script was written based on the design of our physical ERD's relationships and data types. This SQL "dump" provides the complete source script to implement the "inventorydatabase" database locally. It contains the create and insert statements for each table. It also declares the primary and foreign keys to create the relationships between our 4 tables.

System Development Summary

Group 7 has a team volunteer leader, Adrian Cruz, and our team consists of six total members with different levels of knowledge and backgrounds upon entering this class. We decided and agreed to create and implement an Inventory Management System.

First, we gathered information to determine our system's Functional and Non-Functional Requirements. Once we determined the required functions, we created use cases describing how the inventory specialists would interact with the system.

We then began creating Level 0 and Level 1 Data Flow Diagrams which included entities and data stores within the system to describe the data flows between the processes within the system. Each use case represents an entity within the DFD, and the data stores represent the tables in the database.
We developed an Entity Relationship Diagram to determine the tables and data types for the fields in those tables. We considered the type of information stored when selecting each field's data type and character length. To design our Physical ERD, we utilized Draw.IO.

The data dictionary was created from all the attributes we used in the tables of our Physical ERD that represent the rules each field will follow, such as data type and character length. This provides a standard reference for the structure our database follows when storing our data for the Inventory Management System.

We created the storyboard prototype with MoqUps and the Interface Structure Diagram with Draw.IO. We shared and collaborated via Google Drive and stored all our work there for easy access among team members to refer to. Anthony Nava mostly maintains our presentation through the Prezi website.

Julian, Chris, and Adrian wrote the source code to implement the system. Donna helped with gathering, video editing, testing the system, and editing all the documents. Linda researched backup systems like Wix Studio since some members don't have experience with Visual Studio and MySQL. Some members were more proficient at certain things than other members, which was a significant advantage for the outcome. Overall, everyone worked diligently to develop a functional IMS system. A disadvantage is that some members couldn't do well in the Visual Studio, SQL Server, and programming part of the implementation.

The tools used by the team included Google Docs, Google Slides, Google Drive, Prezi, MoqUps, Draw.IO, GitHub, Microsoft Visual Studio, Xampp, MySQL Workbench, PhpMyAdmin, Discord, Zoom, QuickTime, as well as our laptops to develop.