Warehouse Management System

Database Management

CMPT 308N-113

Bezos' Indentured Servants

Marist College
School of Computer Science and Mathematics

Submitted To:
Dr. Reza Sadeghi

9/6/2023

Table of Contents

Table of Figures

I. Project Description of *Bezos' Indentured Servants*

Team Name

Name of the Team --*Bezos' Indentured Servants*

Github Link

https://github.com/gabrielle-knapp1/Project-1-Database-Man-

Team Members

1. Ethan Morton -- ethan.morton1@marist.edu (Team Head)

2. Gabrielle Knapp-----------------------------------gabrielle.knapp1@marist.edu (Team Member)

Description of Team Members

1. ***Ethan Morton***

   I am a sophomore and I'm majoring in computer science with a concentration in software development. I am minoring in mathematics, information technology, information systems, and cybersecurity. I know C#, C++, Java, Python, HTML, CSS, JavaScript, and SQL. I am from Granby, Connecticut and I enjoy playing tennis, hanging out with my friends, and playing games. Some clubs I'm involved in are campus ministry, computer society, games society, and club tennis. Together we chose me to be the Team Head because of my ability to manage communications.

2. ***Gabrielle Knapp***

   Gabrielle Knapp is a sophomore at Marist College majoring in Computer Science with minors in Spanish and Economics. She comes from Carlisle, Pennsylvania and has been enjoying her time adjusting to college life. Activities she currently is participating in include intramural badminton and the games society. In her free time, she loves reading, going on long walks, and playing board games. She is excited to see how she learns and grows her programming skills throughout this class and her entire time at Marist. Gabrielle is excited to bring her hardworking, can-do attitude to her group in this project for her Database Management class. She chose to work with Ethan on this project because she has enjoyed working with him in other classes, including working together on their project in Dr. Sadeghi’s Intro to Programming class. Together, as a pair, they chose Ethan to be the Team Head because of his excellent ability to manage communications.

II. Project Objectives

Summary

We have selected project sample 4, the warehouse management system. The warehouse management system (WMS) provides an organized way of storing different products and elements in a warehouse. You can consider a library as a warehouse, which maintains books' details and user libraries. A general WMS stores details of name and identification number of products, their store time, the required storage condition, price, weight, height, etc. Following this, this system allows guest users to search for different content and request to borrow/buy them. Our WMS will store the data of different user types in distinct SQL tables.

Modules

- Admin Roles
  - Admin has login (username/password) and ability to change that login
  - Admin can remove users from WMS
  - Admin has ability to add a guest user with a login (guest has limited abilities)
    - Guest user cannot define/remove other users
  - Admin can add, delete, and edit items to WMS with various details
  - Admin can view, accept, and reject the list of borrowing requests
- User Roles
  - Users can search through items in WMS depending on various item details
  - Users have ability to save favorite items
  - Users can request to borrow/buy specific items at specific times
  - Users have ability to view the history of borrowed/bought items
- WMS should be user-friendly software

    - Welcome page
    - Menu with all functions
    - All functions in a tabular format
    - Well-organized list of requested items
    - Exit function with friendly goodbye
    - Should show warnings IF:
        - The Admin user tries to add a new item to the library with an existing ID
        - If a guest user tries to borrow more than 3 items
        - A user search request returns null items
- WMS should protect User's information/data
    - WMS passwords & recorded info should be Ciphered using Caesar Cipher Method

III. Review the Related Work

1) *BR Williams Warehouse Management System* ***link***

a) *Positive Aspects*

i) Inventory history and transaction logs. Makes it easy for users to work with the system.

b) *Negative Aspects*

i) Uses barcodes to scan which automatically update the database with the correct information.

2) *Koha Management System* ***link***

a) *Positive Aspects*

i) Intuitive navigation for users. This also has very good GUI and user graphics.

b) *Negative Aspects*

i) Built for library management rather than warehouse management.

3) *ShipHero Warehouse Management System* ***link***

a) *Positive Aspects*

i) Users can return items. It also is very well organized.

b) *Negative Aspects*

i) Multi-carrier shopping: searching the same items from different stores won’t work in our project.

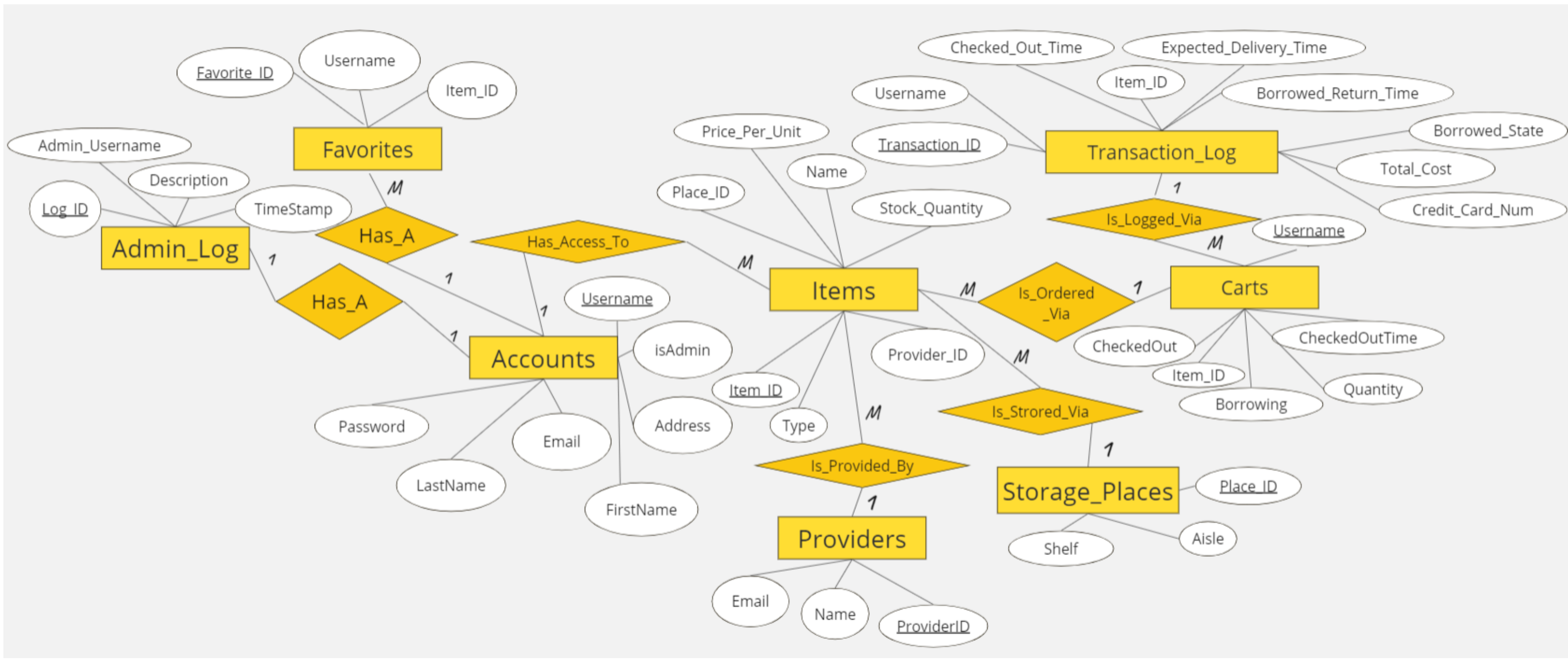
IV. Merits of Project

Merits:

- The WMS will organize all of the items in the warehouse and will allow users to easily access a list of all of these items and their details. This is something that we learned after studying other Warehouse Management Systems and realizing the benefits of including this practice.
- We plan on implementing the positives of other Warehouse Management Systems, including the implementation of detailed inventory logs, allowing for clear details about returning items, and creating an intuitive navigation for users.
- The WMS will clearly differentiate the roles of User and Admin, and allow Admin complete control over the warehouse and the privileges of the users. This is according to the necessary components detailed in the rubric.

V. Entity Relationship Model & Diagram

Figure 1 (ER Diagram):



Overview (ER Model):

- Here, we have a more detailed version of what we might need for the warehouse and what is described in the EER Diagram. We have the Account, which is anyone using the WMS. The Account is either Admin or not admin, and that is controlled by the isAdmin boolean. If they are admin, then the user has access to the Admin abilities, which are listed in the Admin_Log table. Each account also has the ability to have a list of their favorite items, stored in the Favorites table. The items which could be included in this table are all stored in the Items table, which stores all of the items in the warehouse along with all of the important information about those items. The Providers table shows the list of providers who provide the items and their important info. The Items storage information is stored in the Storage_Places table, which lists its location in the warehouse. The items can be requested for purchase or requested to be borrowed by accounts and that is stored in the Carts table. A history of previous orders is stored in the Transaction_Log table.

Description of each entity, attribute, relationship, participation, and cardinality:

- Accounts
  - Username: ID of the user

    - Email & Address: Primary contact information of this user
    - Password: login information
    - FirstName & LastName: name of account holder
    - isAdmin: type of user
- Providers
    - ProviderID: ID of provider
    - Email & Name: Contact Info
- Admin_Log
    - Log_ID: Admin ID of administrator
    - Admin_Username: Username of this administrator
    - Description: Action taken
    - Timestamp: When this action was taken
- Favorites
    - FavoriteID: ID of this favorite Item
    - Username: ID of user who has this as their favorite
    - Item_ID: ID of this item
- Items
    - PlaceID: ID of where item is stored
    - ItemID: ID of this particular item
    - Price_Per_Unit: Price of 1 item
    - StockQuantity: How much item is stored in warehouse
    - Name: Title of item
    - ProviderID: ID of provider
    - Type: Category of Item
- Storage_Places
    - PlaceID: ID of the storage location of this item
    - Aisle & Shelf: Specific location where this item is stored
- Carts
    - AcceptedYN: Boolean, Whether this order has been accepted
    - ItemID: ID of the item being requested
    - Username: ID of the user
    - CheckedOut: Whether the item is checked out
    - Borrowing: Whether the item is being borrowed
    - CheckedOutTime: timestamp of request
    - Quantity: quantity of items requested
- Transaction_Log
    - Total_Cost Total of price
    - ItemID: ID of the item being ordered
    - Username: IDs of the user who placed these orders
    - TransactionID: ID of the transaction log (PK)

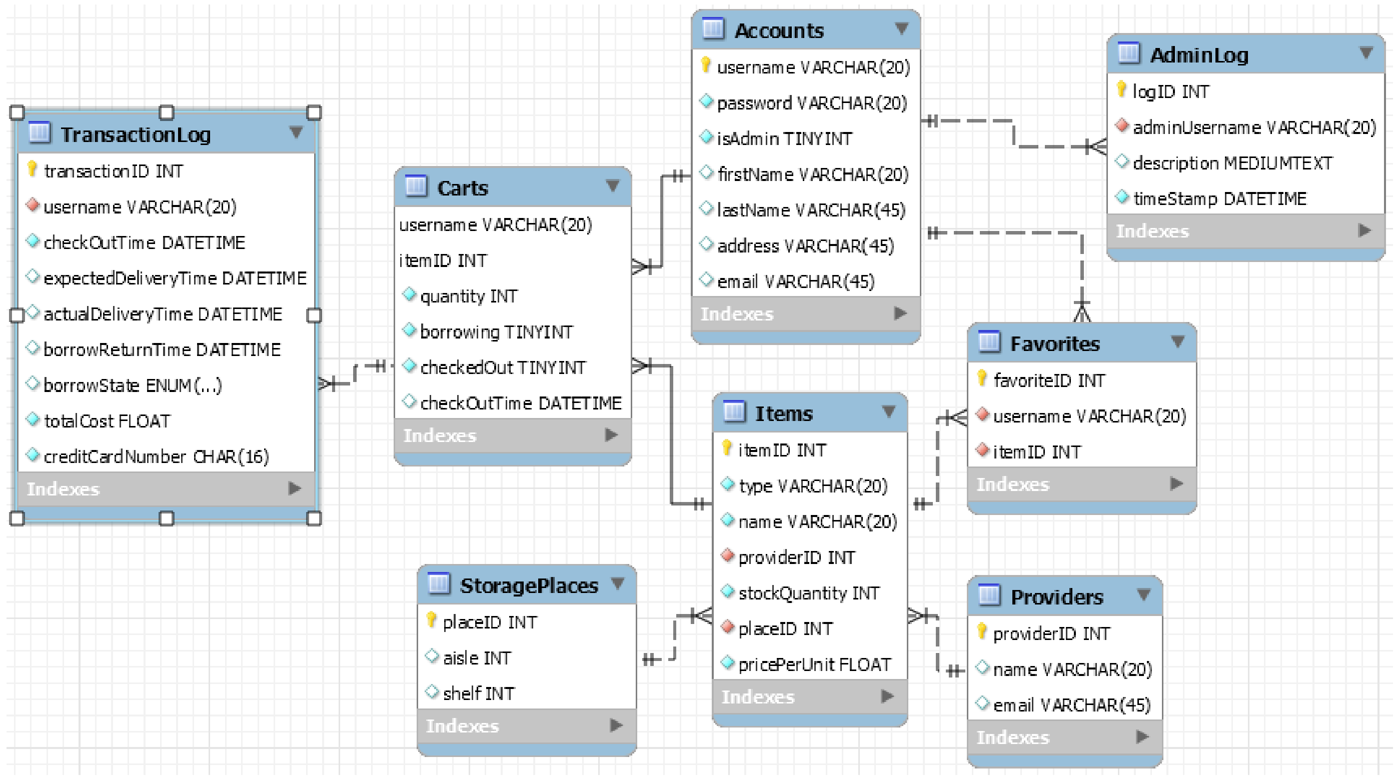
    - CreditCardNum: Credit Card Number of the account
    -
- Item → Has A → Provider
    - Each item has a provider
- Account → Has A → AdminLog
    - Each user has a value declaring whether it has access to Admin powers
- Account → Has A → FavoriteItem
    - Each user has access to a list of favorite items
- Account → Has Access to → Items
    - Each uers has access to the table of items
- Item → Is Stored Via → Storage_Places
    - Each item has a PlaceID with info on where it is stored
- Item → Is OrderedVia→ Carts
    - Each Item can be ordered via an Order Request in the Cart
- Carts → Are logged Via →Transaction Log
    - Each Order Request ever requested can be viewed in the TransactionLog table

Detailed Description of Creation Process

In creating this mini-world for a warehouse management system, I carefully selected and designed the entities, attributes, relationships, participations, and cardinalities to represent the key elements of the system. I chose entities like "Accounts," "Providers," "Items," and "Carts" to reflect the fundamental components within the warehouse environment. Within each entity, I defined attributes such as "Username," "Email," and "Price_Per_Unit" to capture essential properties and information. I established relationships like "Item → Has A → Provider" and "Account → Has A → AdminLog" to connect these entities, mirroring the real-world associations between items and providers, as well as between user accounts and administrative logs. I carefully considered the participation of entities in relationships and defined cardinalities (e.g., one-to-many or many-to-many) to model the interactions and dependencies in the system effectively. This conceptual model serves as the foundation for building a database that can manage inventory, user accounts, orders, and more within a warehouse environment.

VI. Enhanced Entity Relationship Model (EER Model)

Figure 2 (EER Diagram):



Description about keys & relationships:

- Each Account:
  - has a unique username
  - has a password
  - is either an admin or guest
  - has a first and last name
  - has an address to ship the items to
  - has an email to contact the user
- Each Item:
  - has a unique item ID
  - has a type
  - has a name
  - has a provider
  - has a quantity of that item in the warehouse to prevent redundancy

    - has a place it’s stored in the warehouse
    - has a price per unit
- Each Transaction Log:
    - has a unique transaction ID
    - references all the items in each cart via a username
    - has the time of checkout
    - has an expected delivery time
    - has an actual delivery time
    - has a time when borrowed items should be returned
    - has a state for items being borrowed
    - has the total cost
    - has the credit card number used for the transaction
- Each Favorite:
    - has a unique favorite ID
    - holds the username of the account that has favorited the item
    - has the item being favorited
- Each Storage Place:
    - has a unique ID
    - has an aisle number
    - has a shelf number
- Each Provider:
    - has a unique ID
    - has a name
    - has an email
- Each Admin Log:
    - has a unique ID
    - has the username of the admin who did an action
    - has a description of the action
    - has a timestamp
- Each Cart:
    - has a unique username-itemID pair
    - has a checkout time
    - has the quantity of that item
    - stores whether the item is being borrowed

    - stores whether the cart has been checked out

Description of implementation of these features:

- Account
    - Logging in will give you access to the website and only show your personal data.
    - Admins may remove specific accounts.
    - Admins may approve/reject borrow requests
- Admin Log
    - Admins may approve or deny borrow requests.
    - Admins may ban users from using the WMS.
    - All admin-related actions are logged here.
- Items
    - All items in the warehouse will be displayed.
    - Users can search for items and sort the list.
    - Admins can edit the list of items.
- Transaction Log
    - Users will be able to view their past transactions.
    - Users will be able to make transactions by checking out the cart.
    - Admins can view all transactions.
- Favorites
    - Each user can view the items they favorited to access them quickly.
- Carts
    - Users can have 1 current cart which stores the items they'd like to buy but have not paid for.
    - When users pay for the items in their cart, the cart is saved to the database and they are given a new empty cart.
    - Carts that have checked out are referenced by the transaction log so users can view their previous purchases.

VII. Database Development

SQL Code:

```
create database Warehouse;
use Warehouse;

create table Accounts(
        username varchar(20) primary key,
   password varchar(20) not null,
   isAdmin bool not null default false,
   firstName varchar(20),
   lastName varchar(20),
   address varchar(45),
   email varchar(45)
);

create table AdminLog(
        logID int primary key,
   adminUsername varchar(20),
   description text,
   timeStamp datetime not null,
   foreign key (adminUsername) references Accounts(username)
);

create table Carts(
        username varchar(20) primary key,
   itemID int primary key,
   quantity int not null default 0,
   borrowing bool not null default false,
   checkedOut bool not null default false,
   foreign key (username) references Accounts(username),
   foreign key (itemID) references Items(itemID)
);

create table Favorites(
        favoriteID int primary key,
    username varchar(20),
    itemID int,
    foreign key (username) references Accounts(username),
    foreign key (itemID) references Items(itemID)
);

create table Items(
        itemID int primary key,
    type varchar(20) not null,
    name varchar(20) not null,
    providerID int,
    stockQuantity int not null default 0,
    placeID int,
    pricePerUnit float not null default 0.00,
    foreign key (providerID) references Providers(providerID),
    foreign key (placeID) references StoragePlaces(placeID)
);

create table Providers(
        providerID int primary key,
    name varchar(20),
    email varchar(45)
);

create table StoragePlaces(
        placeID int primary key,
    aisle int,
    shelf int
);

create table TransactionLog(
        transactionID int primary key,
    username varchar(20),
    checkoutTime datetime not null,
    expectedDeliveryTime datetime,
    actualDeliveryTime datetime,
    borrowReturnTime datetime,
    borrowState enum("pending", "accepted", "rejected"),
    totalCost float not null default 0.00,
    creditCardNum char(16) not null,
    foreign key (username) references Carts(username)
);
```

Tabular Description of ER Diagram + SQL Code

ER Entity	**SQL Table**	**Attributes**	**Relationships**
Accounts	Accounts	username (PK), password, isAdmin, firstName, LastName, Address, Email	
AdminLog	AdminLog	logID (PK), adminUsername, description, timeStamp	adminUsername is a foreign key referencing Accounts(username)
Carts	Carts	username (PK), itemID (PK), quantity, borrowing, checkedOut	username is a foreign key referencing Accounts(username) itemID is a foreign key referencing Items(itemID)
Favorites	Favorites	favoriteID (PK), username, itemID	username is a foreign key referencing Accounts(username) itemID is a foreign key referencing Items(itemID)
Items	Items	itemID (PK), type, name,	providerID is a foreign key referencing Providers(providerID)

		providerID, stockQuantity, placeID, pricePerUnit	placeID is a foreign key referencing StoragePlaces(placeID)
Providers	Providers	providerID (PK), name, email	
Storage Places	Storage Places	placeID (PK), aisle, shelf	
Transaction Log	Transaction Log	transactionID (PK), username, checkoutTime, expectedDeliveryTime, actualDeliveryTime, borrowReturnTime, borrowState, totalCost, CreditCardNum	username is a foreign key referencing Carts(username)

VIII. References

- ***BR Williams Warehouse Management System*** ***link*** — page 8
- ***Koha Management System*** ***link*** — page 8
- ***ShipHero Warehouse Management System*** ***link*** — page 8