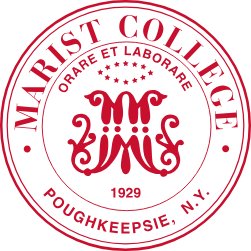
**Warehouse Management System – Phase 2**

**Database Management Systems**

**MSCS 542L-256**

**Embrace Database**



Marist College

School of Computer Science and Mathematics

Submitted To:

Dr. Reza Sadeghi

9/27/2023

**Project Report of Warehouse Management System – Phase 2**

**Team Name**

Embrace Database

**Team Members**

1. Reem Ooka (Team Head) [Reem.Ooka1@marist.edu](mailto:Reem.Ooka1@marist.edu)

2. Frank Seelmann (Team Member) [Frank.Seelmann1@marist.edu](mailto:frank.seelmann1@marist.edu)

**Description of Team Members**

1. Reem Ooka

Hailing from the Incredible country of India, I proudly call Mumbai my home - a vibrant metropolitan city renowned as the economic and entertainment heartbeat of our nation. I graduated in May 2022 with a bachelor's degree in computer science. During my professional career, I gained expertise in developing websites tailored for citizen-centric solutions while working at an e-governance company as a Full Stack Developer. Presently, I am furthering my academic pursuits as a graduate student at Marist, where I am specializing in Cloud Computing.

Frank voted Reem to be the team lead since he already has a job, so putting “Team Lead” on his resume will not be as useful for him.

2. Frank Seelmann

Hudson Valley native, graduated from SUNY New Paltz in May 2022 with a Bachelor of Computer Engineering. Works at IBM in Poughkeepsie as a Verification Engineer. Has some experience using database management systems, specifically MySQL and MongoDB, but in both cases the DBMS was supplemental to the project, not its focus.

**Table of Contents**

[1. Project Objective 4](#_Toc1855530273)

[2. Review the Related Work 5](#_Toc969752311)

[3. The Merits of Our Project 6](#_Toc1437996341)

[4. GitHub Repository 6](#_Toc558610680)

[5. Entity Relationship Model 6](#_Toc358827755)

[5.a ER Diagram 6](#_Toc478951371)

[Figure 5.a - Entity Relationship Diagram 7](#_Toc1395234353)

[5.b ER Model 8](#_Toc1343436103)

[Figure 5.b - Entity Relationship Model 9](#_Toc1682895482)

[6. Enhanced Entity Relationship Model 9](#_Toc675937407)

[Figure 6 – Enhanced Entity Relationship Model 10](#_Toc2117402060)

[Works Cited 10](#_Toc351156969)

**Table of Figures**

[1. Project Objective 4](#_Toc805014347)

[2. Review the Related Work 5](#_Toc215042955)

[3. The Merits of Our Project 6](#_Toc941531926)

[4. GitHub Repository 6](#_Toc360308193)

[5. Entity Relationship Model 6](#_Toc120291976)

[5.a ER Diagram 6](#_Toc338795510)

[Figure 5.a - Entity Relationship Diagram 7](#_Toc1365113691)

[5.b ER Model 8](#_Toc491734337)

[Figure 5.b - Entity Relationship Model 9](#_Toc430587628)

[6. Enhanced Entity Relationship Model 9](#_Toc835529983)

[Figure 6 – Enhanced Entity Relationship Model 10](#_Toc411824058)

[Works Cited 10](#_Toc1662178320)

# 1. Project Objective

The core objective of implementing the General Warehouse Management System (WMS) is to effectively oversee and streamline product information management. This encompasses attributes such as product name, identification, storage specifications, pricing, weight, height, and other relevant data points. The WMS aims to enable users to efficiently search for items and initiate requests for either borrowing or purchasing. It strives to achieve accurate product tracking, foster seamless user interactions, offer a user-friendly interface for inventory management and transactions, all while upholding data integrity and security.

For this class, our WMS has four minimum criteria:

1. An admin user who has several exclusive capabilities, such as adding a guest user, removing users, adding/deleting/editing items from the warehouse, and accepting/rejecting borrowing requests.
2. Users must be able search the warehouse, request to borrow/buy items for a specific time and view the history of borrowed items.
3. The WMS should have several user-friendly features, such as a welcome page, a menu of all functions available to a user, tabularized reports, an exit function, and providing warning/error messages.
4. The WMS should protect user information, particularly passwords.

The WMS will use MySQL as the DBMS, and Java to handle the front-end.

# 2. Review the Related Work

Our project is a prototype of a real-world WMS. To better understand what is expected of WMS, and what might make one better than another, we examined other existing solutions.

Using Capterra [1], a website for comparing software solutions, we were able to compare several WMS providers and see some of the key features that set them apart. Capterra has user ratings for 10 popular WMS providers, as well as listing seven key features for comparison. These features are Barcoding/RFID, Forecasting, Inventory Management, Purchasing & Receiving, Receiving/Put away Management, Returns Management, and Shipping Management.

The WMS with the most reviews, 1335, is NetSuite from Oracle, which has all 7 features. This product is also very expensive, with a pricing range of $1,299 to $9,999 a month, depending on the subscription package [2]. Users critique it for its high cost and complexity.

The second most reviewed WMS, with 300 reviews, is “Inventory Cloud.” Users praise its short learning curve but criticize its lack of customization.

“Carton Cloud” has the highest rating at 4.8 stars. It lacks 2 of the 7 features, Forecasting and Returns Shipment. Users praise it for its ease of use and affordability (pricing starts at $99/week).

# 3. The Merits of Our Project

The best reason to choose our WMS is that we offer a cost-effective solution. Having a small team (2 developers) means we can keep costs low. We expect to provide 5 of the 7 features considered by Capterra: Inventory Management, Purchasing & Receiving, Receiving/Put away Management, Returns Management, and Shipping Management.

From reading the reviews, a “nice-to-have" for our WMS would be adding customizability for the user. This could take the form of having the reports show up in different ways, such as list view vs grid view, which would be remembered by the systems as the preferred viewing option for the user.

# 4. GitHub Repository

To provide version control for our project, we are using GitHub:

<https://github.com/ReemPatel13/MSCS-542L-256_WarehouseManagementSystem_EmbraceDatabase>

# 5. Entity Relationship Model

To illustrate how our database will be constructed, we created an Entity Relationship Diagram and an Entity Relationship Model.

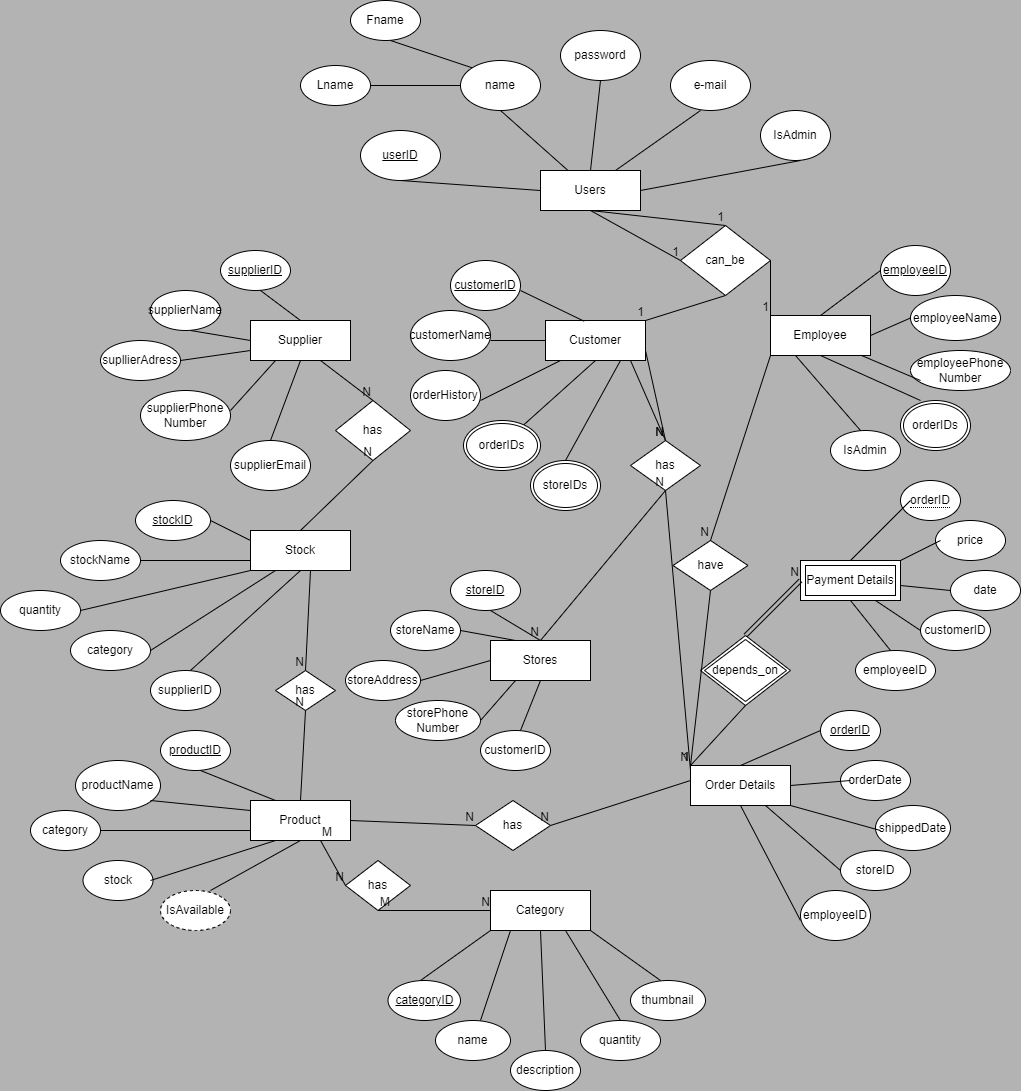
## 5.a ER Diagram

To create this mini world, we used [draw.io](http://draw.io) and the entities, attributes, the relationships between them, the participations and cardinality was mutually discussed adhering to the pointers given.

The following is a short description about each entity and its attributes:

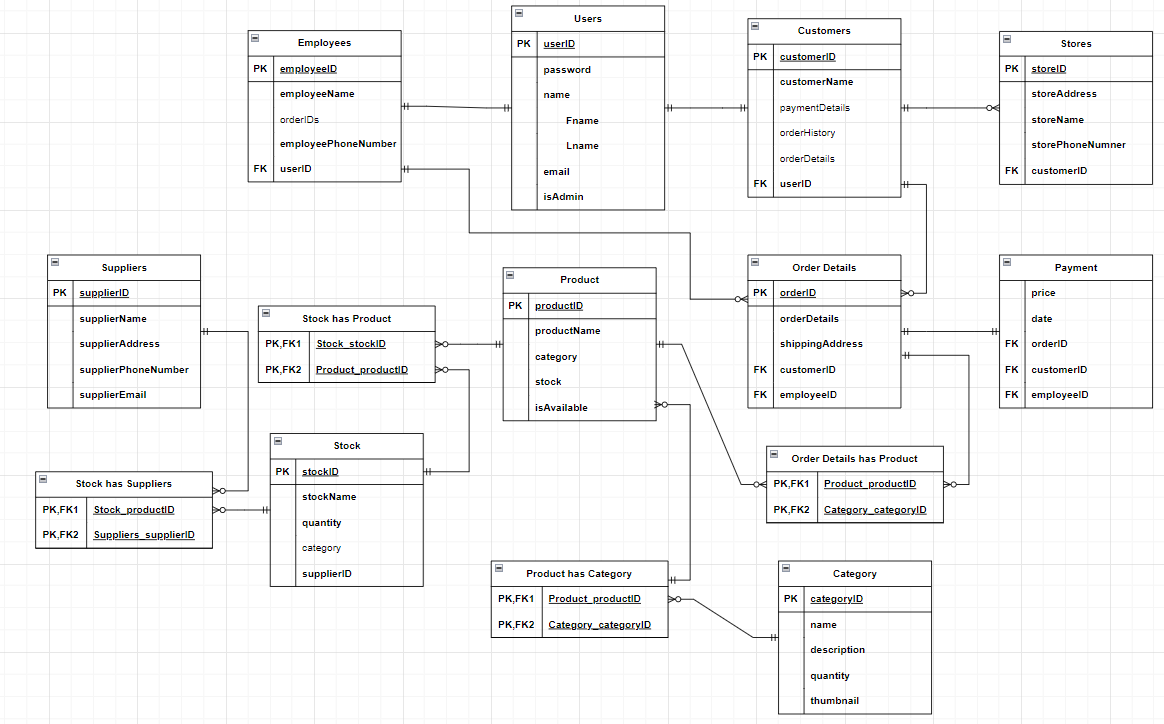
1. Users: Our Warehouse Management System will have users who can be either employees or customers. An employee could also be an admin. It has a Primary key called userID, a composite attribute such as name composing of Fname and Lname, other attributes like password and e-mail as well as a boolean attribute called IsAdmin. The user entity has 1 to 1 relationship with the employee entity and customer entity.
2. Employee: An employee is responsible for handling orders and could also be the admin. Here, the primary key is employeeID. It has attributes like employeeName, employeePhoneNumber, a multivalued attribute like the orderID and a boolean attribute like the IsAdmin. The employee entity has 1 to 1 relationship with the user entity and 1 to Many relationship with the order details entity.
3. Customer: A customer is basically the store owner who is our final client. The customer entity has a Primary key called customerID. Other attributes include customerName and orderHistory. Two multivalued attributes here are orderID and storeID as a customer can have multiple orders and stores. This entity has 1 to 1 relationship with the user entity. It has 1 to Many relationship with the store entity. It has 1 to Many relationship with the orderDetails entity.
4. Supplier: A supplier supplies products to the warehouse. It has a supplierID, supplierName, supplierAddress, supplierPhoneNumber and supplierEmail. It has 1 to many relationship with the stock entity.
5. Stock: The stock keeps a records of all available products. It has a stockID, stockName, quantity, category and a supplierID. It has 1 to Many relationship with the product entity.
6. Product: The product entity has a productID, productName, category, stock and a derived attribute called IsAvailable. It has Many to Many relationship with the category entity.
7. Category: A category will have a categoryID, name, description, quantity and thumbnail. It has a Many to Many relationship with the product entity.
8. Stores: The stores entity has a storeID, storeName, storeAddress, storePhoneNumber and customerID. It has a Many to 1 relationship with the customer entity.
9. OrderDetails: This entity has attributes like orderID, orderDate, shippedDate, storeID and employeeID. It has Many to 1 relationships with the employee and customer entities. It also has a 1 to 1 relationship with the weak payment entity.
10. WEAK PaymentDetails: This is a weak entity which does not have its own primary key and depends on the primary key of the orderDetails entity taking its orderID. It has attributes like price, date, customerID and employeeID.

### Figure 5.a - Entity Relationship Diagram



## 5.b ER Model

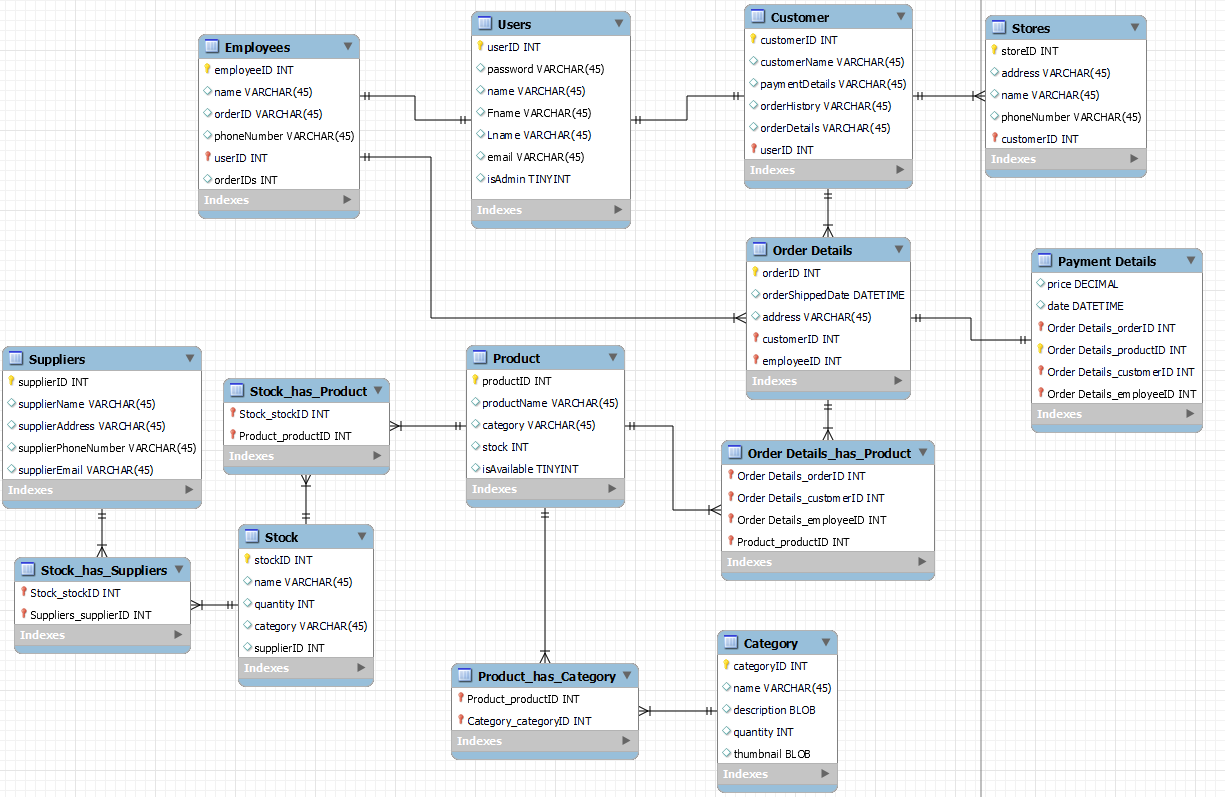
The Entity Relationship model was created using [draw.io](https://app.diagrams.net/#G1aTKjF4hdOrcInVLw2XpZXm5cGy09vSpO), as it is a free online CAD tool with the necessary symbols for creating an ER model. Figure 5.b shows the ER model.



### Figure 5.b - Entity Relationship Model

# 6. Enhanced Entity Relationship Model

For this project we are using the MySQL Workbench. One feature of the Workbench is it allows the creation of Enhanced Entity Relationship Models (EER) models. Figure 6 shows the results of using the tool to illustrate our entity, attribute, and relationship information.



### Figure 6 – Enhanced Entity Relationship Model

Keys in a relational database are attributes (or sets of attributes) are used to uniquely identify records in a table. A primary key is used to uniquely identify a record in the table itself, while a foreign key is used to establish a relationship with the primary key of another table. One nice feature of the EER tool is that it will automatically add foreign keys when you create relationships. These are denoted by the orange pin-like symbol to the left of an attribute name. One example is the userID foreign key in the Employee entity.

Relationships in a relational database refer to associations between tables based on keys. Relationships can be one-to-one, one-to-many, and many-to-many. Examples of these relationships would be Order Details to Payment, Customer to Order Details, and Product to Category, respectively. As a second useful feature, the tool will automatically create an associate entity when appropriate, such as when defining a relationship as many-to-many. The Product to Category relationship is an example of many-to-many since a product could have multiple categories, and a category may have multiple products.

# Works Cited

[1] “Capterra: Find & evaluate top software & business services,” Warehouse Management Software, <https://www.capterra.com/sem-compare/warehouse-management-software/?utm_source=ps-google&utm_medium=ppc&utm_campaign=:1:CAP:2:COM:3:All:4:US:5:BAU:6:SOF:7:Desktop:8:BR:9:Warehouse_Management&network=g&gclid=EAIaIQobChMI3o6cj5umgQMVmklyCh310AN7EAAYAyAAEgLcOfD_BwE> (accessed Sep. 12, 2023).

[2] “Capterra: Find & evaluate top software & business services,” Warehouse Management Software, <https://www.capterra.com/> (accessed Sep. 12, 2023)