Supermarket Inventory Management System (SIMS) Proposal CPSC 471 - Team T02-2

Group Members: Adhira John (10193007) | Leonie Mertens (30298707) | Taha Ghumman (30176505)

Executive Summary

SIMS (Supermarket Inventory Management Systems) is a one-stop inventory management system for supermarket companies with one or more branches. It offers features like organizing items by overlapping types, as well as the ability to easily order new items when supply gets low. With the use of a web-application, supermarket chains can manage and organize their inventory effectively. SIMS also offers alerts when stock is low, and information about designated suppliers for specific items. Users are employees of the supermarket chain and can be categorized as either a Branch Manager (BM) or a Cashier. After logging in with unique user details, a BM's can order new stock directly with a supplier once inventory runs low. Once an item has been scanned by a Cashier to bill a customer, the stock of that item automatically decreases both globally within the supermarket company but it is also locally decremented within the branch's inventory quantities. If quantities dips below a certain threshold, SIMS alerts the BM of low stock so that new stock can be ordered.

Users: Branch Manager (access to all), Cashier (access to Items)

ER Diagram

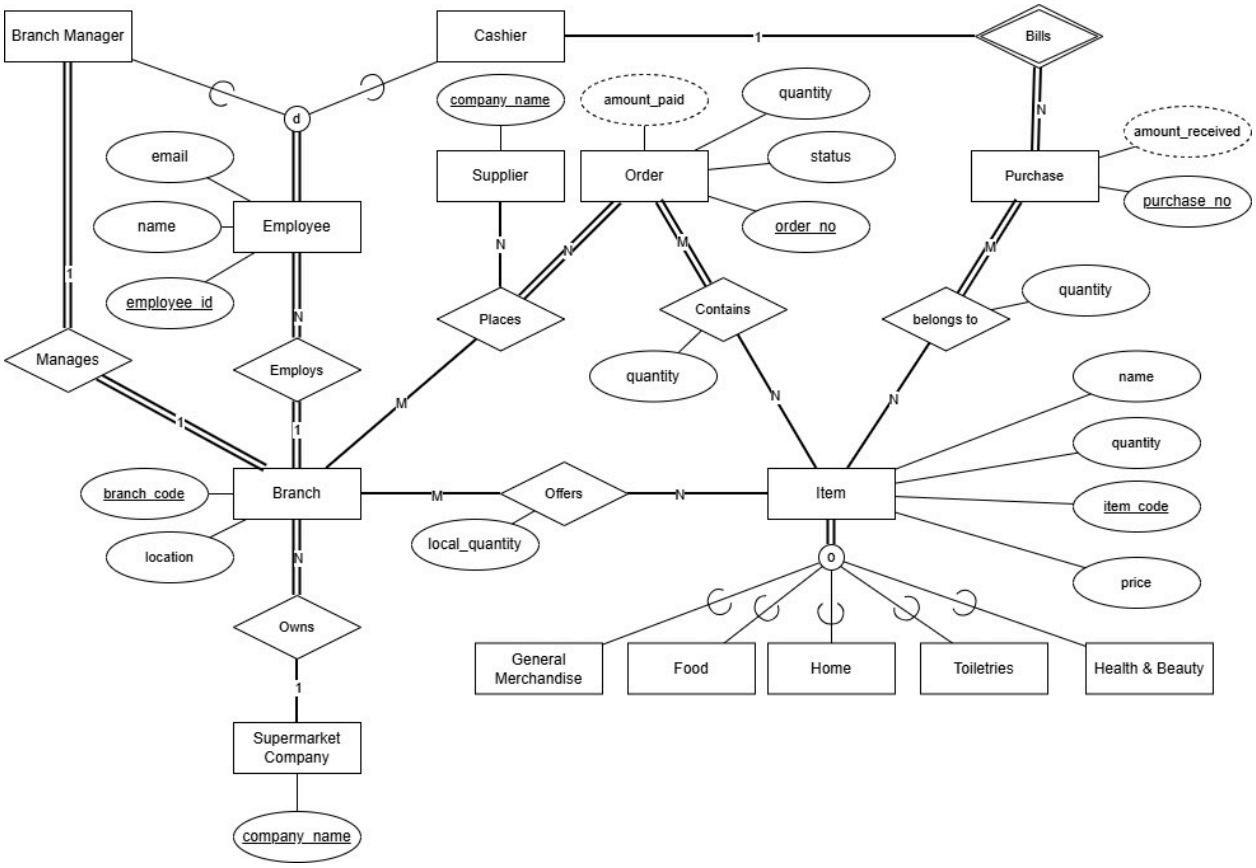

Diagram Details

We've designed an ER diagram that encompasses a generic supermarket store branch detailing the interactions that affect inventory of a branch on a day-to-day basis. Monetary totals, like the amount received in a transaction, are treated as derived attributes as the focus for our application is inventory, not accounting. The customer is represented by the Purchase entity, which is identified by its purchase number. The customer does not play a larger role in the system. This means the system searches by transaction number to find how many items were bought as well as the quantity of each item. This kind of information can provide analytics including identifying the most popularly sold items. These analytics can be leveraged by a BM to make appropriate orders of stock in the future.

System Specifications

Items can fit into overlapping specializations made of five categories: General Merchandise, Home, Food, Toiletries and Health & Beauty. Employees relevant to SIMS fit disjointly into being either a Branch Manager or a Cashier. Only Items can be purchased, and only Cashiers can bill them. This limits the scope of SIMS such that the system's focus is inventory, not staff details, customers relationships, or other aspects of a supermarket. Of course, the system can be broadened depending on the supermarket, so that the Items category has further sub-categories, but for the purpose of the proposal, our diagram features generic categories found in many supermarkets. Each item is identified by its code and quantity, this allows the system to track when a supplier order may be required depending on the unique id.

Employee information is kept for verification purposes. If an item sells, the employee's id is saved alongside the item to know who executed the sale. This information could be leveraged to identify the cashier with the most sales.

Reference Sources

The Flask API [1] is great for seeing all methods within Flask alongside their signatures for their arguments. It's useful to see a brief description on usage as well as the example usages for some methods to effectively use Flask for certain use-cases.

This Flask Web App Tutorial [2] by geeksforgeeks detailing how to build an application is great to get an initial set up for a first application. It is more specific than general documentation or tutorials and provides a good framework for our web application.

This page of Flask Tutorials [3] by real Python links many different tutorials for building Flask applications from scratch, including instructions for using SQLAlchemy and integrating a database into Flask. It is particularly useful for our project because it explains step-by-step how to connect Flask with a relational database, which is essential for building our inventory management system.

References:

[1] Flask Documentation, "API," Pallets Projects, [Online]. Available: https://flask.palletsprojects.com/en/stable/api/. [Accessed: 27-Oct-2025].

[2] GeeksforGeeks, "Flask – Creating First Simple Application," [Online]. Available: https://www.geeksforgeeks.org/python/flask-creating-first-simple-application/. [Accessed: 27-Oct-2025].

[3] Real Python, "Flask Tutorials," [Online]. Available: https://realpython.com/tutorials/flask/. [Accessed: 27-Oct-2025].