Spring Boot Final Project

Inventory Management System

URL to GitHub Repository: https://github.com/Yesica-MP/Inventory.git

URL to video: https://www.youtube.com/watch?v=ialaZ644Edc

Objectives

Develop a REST API to manage products of a liquor store, utilizing a Spring Boot project for a seamless web API experience. The inventory will be organized by category, brand, and supplier.

Requirements

Database Design: Develop a database structure comprising a minimum of three entities and three tables.

CRUD Operations: Implement all CRUD operations (Create, Read, Update & Delete).

Each entity should feature at least one CRUD operation. At least one entity must support all four CRUD operations.

Relationships:

Include a one-to-many relationship.

Feature a many-to-many relationship, with one or more CRUD operations on this relationship.

REST Web API Server Testing: Ensure testing through Swagger, Postman, AdvancedRestClient (ARC), or a front-end client.

REST API Inventory Management System Description:

Entities

***1. Brand*:**

ID: Unique brand identifier.

Name: Brand name.

Description: Brief description of the brand.

Price: Liquor price from this brand.

Category: Liquor category.

Supplier: Supplier providing this liquor brand.

2. Supplier:

ID: Unique supplier identifier.

Name: Supplier name.

Phone: Contact phone number for the supplier.

Address: Supplier's address.

Brands: List of liquor brands supplied by this supplier.

Categories: List of categories supplied by this supplier.

3.Category:

ID: Unique category identifier.

Category Name: Name of the liquor category.

Brands: List of brands belonging to this category.

Suppliers: List of suppliers providing liquor in this category.

4.SupplierCategory:

ID: Unique identifier for the supplier-category association.

SupplierID: ID of the supplier associated with the category.

CategoryID: ID of the category associated with the supplier.

Entity Relationships

- A Brand belongs to a single Supplier and a single Category.
- A Supplier has a one-to-many relationship with Brand and a many-to-many relationship with Category.
- A Category has a one-to-many relationship with Brand and a many-to-many relationship with Supplier.

Error Handling

In this project, I have implemented an error handling mechanism to handle exceptions and provide meaningful responses.

Exception Handling Methods

handleNoSuchElementException:

Handles NoSuchElementException and returns a custom ExceptionMessage with a 404 status code.

handleDuplicateKeyException:

Handles DuplicateKeyException and returns a custom ExceptionMessage with a 409 status code.

It is imperative to follow the specified order of creation. Please note the following steps:

1. Supplier Creation:

Before proceeding with any other steps, it is essential to create a supplier. The application's foundation relies on the existence of suppliers.

2. Brand Creation:

Following the establishment of a supplier, the next step is to create a brand. A supplier must be in place before initiating the brand creation process.

3.Category Association:

To associate a category with a supplier, it is mandatory to have a brand created. This ensures a structured and accurate categorization process.

To initiate the project, follow these steps:

- Clone the repository.
- Set up your database and configure the application properties.
- Build and run the project.