POINT OF SALE SYSTEM

SE452 – Object Oriented Enterprise Application Development

Sample project

Contents

Overview

Develop a simple point of sale system that stores menu items, including allergy information, ingredients, and price that can be updated and changed. My system takes a session object that stores the employee that is using the system as well as what items are available to order, what orders are currently being worked on and makes sure that all information is backed up on the database. The program also pulls information out of the database for the user to be able to manipulate it without making duplicate entries in the database. The program checks to make sure that the employee using certain pages has the access level to do so.

Requirements

Use Case

User → client → server → db software → server → client → user

Description of problem

User logs into client application to enter in information needed for the transactions. The client communicates with server, which accesses the db software to pull up menus, as well as employee information such as access credentials to update data. The client application retrieves the current information from the database and communicates with the server to keep a record of the transactions processed on the client application to be able to update any changes made by the client application to the data stored (like menu updates, ingredients, prices, etc.). The client application will interact with the user by a web interface. The program needs to be able to travel from one html page to another depending on data manipulation and methods to complete specific tasks.

Design

Sequence of major functionality

Web UI (Common case)

The web interface starts off on a login page that leads into a screen that displays table buttons that automatically assign a transaction a table menu so that is ready to hold an order at that specific table. When you click on the table button, it goes into a screen that displays the current information for the transaction that is being manipulated. Underneath the current transaction information, there is a series of checkboxes that allows you to click on an item, or multiple items, that can be ordered. After clicking "order" the system, updates the information on the display as well as the information on the database. The transaction is added to a session arraylist that keeps track of the current orders, as well as the order that is being manipulated at the current time. The transaction screen allows you to close the transaction, entering the payment type and uploading the final information to the database and return to the table screen. Going back to the table screen, there is also the option to go into administration screens that can update employee information, create a new employee or the information and do the same for menu items. To be able to enter the admin pages, the program makes sure that you have the proper security level. After making updates/changes you are redirected back to the table screen. You are also able to log out from the table screen, which will invalidate the session id and redirect you to the login page.

Table layout

Employee Table

firstName	lastName	Id	Pay	payType	secLevel	current
varchar	Varchar	Integer	numeric	Varchar	Integer	Varchar

Menu Table

ID	Name	Type	subtype	Price	Ingredients	Allergies	Current
MenuItem ID	Varchar	Varchar	Varchar	Numeric	Varchar	Varchar	Varchar

Transaction Table

ID = transaction ID
eID = employee ID

ID	eID	Ingredients	Allergies	NetTotal	GrossTotal	pmType
Integer	Integer	String[]	String[]	Numeric	Numeric	varchar

Deployment

Discussion of how your design met the requirements

My design meets the requirements because my program interacts with a database for retrieval and modification while keeping the information persistent within the requirements of my application. I used NoSQL because I did not need to have the information be exactly correct immediately. I integrated the database with a web user interface that utilizes many different pages and moves between them while passing information between them and being able to manipulate the data. My program, in addition to the login page, has multiple utilities. You can update or create an employee or menu item if you have the correct security clearance. You can view the items associated with the current order, add to the items in the current order and have them displayed on the screen as you order them. The program integrates with the data and models as well as keeps the information current for the session you are on and does not allow the session to continue when the user is not using the system.

Discussion of lessons learned

I have learned many lessons. I have learned that it is a good idea to map out the program by hand to really make sure that you are covering everything that your program to do. It will also help with doing things in pieces like we were supposed to be doing in this assignment. I try to just use linux and notepad++ to do my programming stuff. This makes things take longer, while I think that it does me good to really learn the programming, it really does impede my progress. I do not normally use IDE's for my work because I do not understand how things are found or used and I think that I am not really understanding things. I also had an issue with wondering why things were so confused with all the stuff that we went through that we really did no know. From going through all the things for this assignment, I understand how things need to be understood before you jump to the more advanced procedures. I also learned that having a full understanding of the platforms being used is really useful. How different programs and application deal with things and what they allow is very useful. From all this, I learned that I code too much and there are a lot of things that can be done without writing specific methods for specific tasks, but need to allocate more time.

I learned that using Spring makes all of the other persistence, web.xml and all of that not needed. I learned that the ID's for MongoDB need to explicitly named, or the items will be repeated in the DB. I learned that not all data types are supported by all platforms. I learned that you need to have either the server or the client keep track of the data objects, not just passing them by the model. You can make different things happen by mapping the html pages to different methods. You need to have special control features in html to handle different data types.

Decision Log

Problem	**What was decided**	**Alternatives considered**	**Rationale**
Warnings for jdk-9 timeStampUtils	Used MongoDB	Different drivers, update drivers, different DBM	there is a problem in java from my research, will look into it
Mongodb adding multiple items	Explicitly add a plain id field	Trying to access the automatic id field	I could not access the _id field from the annotated bean and I kept on getting multiple entries in my db
Not being able to access ModelAttributes	Add attributes to session	Cookies, sending multiple objects through form	I could not find a way to pass things around efficiently
Using ArrayList, but MongoDB does not support	Change to arrays	Trying to manipulate to make it work	I do not like having excess null attributes
Needed to have multiple buttons that do different things	Use params	Using different methods	I needed more options than just two.
Objects auto-filling in	Passed new object	Using reset buttons	So I wouldn't have multiple entries

options			
Displaying Array Contents	Using <ul> <li>	Using literals	I need to have more options than what I can think of at the moment
Some forms wouldn’t work with objects	Used strings	Passing Custom Objects	Seems like the forms only accept certain things