Automated Smart Shopping Cart

Group Assignment

We are getting spoiled by online shopping. We add things or remove them from our cart, and at any time we can see what we have in the cart. And checkout is easy too: we can often just hit the one-click checkout button, and we are done.

Physical shopping isn't so convenient. We have to wait in line, and then unload our shopping cart to scan everything, and then put it back in the cart. And then we must endure the indignity (in some stores) of the security person checking our cart to make sure we aren't trying to make off with the family jewels.

Your company is creating an automatic checkout system. It senses when items are put in the cart, and keeps a running total. It senses when things are removed. At any time, you can see a running total of the stuff in your cart, as well as a list of what is in the cart. Best of all, there is no waiting at the checkout stand. When you walk out, you just confirm the charge to your debit or credit card, and off you go!

Rough Specs:

- Detects when an item is put in the cart
- Detects when an item is removed from the cart
- Displays a running total to an app on your phone. (If someone doesn't have a phone, there will be a device attached to the cart that will do everything the phone app does. See below.)
- Displays a list of stuff in the cart
- The cart may display a list of recommended/suggested items that go well with your list (you usually buy them, others usually buy them together with your items, promotions, etc.)
- When you leave, you are asked to confirm the purchase.
    - If you leave and don't confirm the purchase, a store alarm goes off
    - Confirmation should include entering a PIN or other security measure
- When someone enters the store and takes a cart, the cart somehow detects the person's phone (assuming they have the app), and the two are linked together. One reason is that the person may have coupons on their phone, which give member discounts on certain items.
- When someone leaves, if the person pushing the cart (or very near the cart) isn't the original person, the cart will have to detect the new person (and maybe recalculate the bill)
- Payment is via the customer's favorite way: credit/debit card, Venmo, etc. That is probably best setup at the time the app is downloaded and initialized.
- If the customer doesn't have a phone or the app, the cart itself has a device to show the running total, and it has a slot for a debit/credit card. (No cash!)
- Customer memberships include discounts and sales, and electronic coupons.
- The store has the ability to set prices, put items on sale, etc.
- There is an item lookup feature: the system can tell which aisle an item is in. (Note: does the user select from a menu of items, or does the user input the item name? We haven't decided yet, but the method you pick may have architectural implications.)
- Future thought: this could probably eventually also handle store inventory, or be merged with the inventory management system. For now, assume they can access the same database.

The company has a team of hardware folks who are working on the hardware to detect items in the cart, and the cart hardware unit for display and payment (see above). They are also working on the store exit detector and alarm. You may assume they will get these working, but they will ask you about what the messages should be.

Note: As you design the system, think about binding times. There are lots of different binding times in this system. Think about how they affect the architecture (you don't have to write about binding times specifically, but it might be useful to mention them where they are important.)

Do the following and turn in the usual:

1. Design the architecture, following a pattern-based approach, using the patterns you know already from the classes (and others you may have found by yourself).
2. Document the architecture using the C4 model and UML diagrams.
    a. Context diagrams
    b. Containers diagrams
    c. Component diagrams
    d. A few excerpts of code for some use cases you may prefer, such as:
        i. Adding an item to the cart
        ii. Walking out of the store (checkout)
3. Provide additional textual documentation to give the big picture and explain how everything will work. Include a general explanation of how everything hangs together. You might also include:
    a. Discussions of important quality attribute requirements, and how they will be addressed.
    b. Describe the architecture patterns used (it should be clear in the diagrams, but sometimes additional text is needed as explanation.)
    c. Describe why you made certain key architectural decisions (rationale).
4. List the quality attributes that are important to this system, and how the architecture handles each

See for example:
https://www.washingtonpost.com/technology/2019/10/25/tired-long-lines-canadian-grocery-chain-debuts-smart-carts-with-self-checkout/