Lab 1 - Embed

Actors

- Customer
- Cashier
- Manager

Use cases

Requirements

Id	Description	Evaluation	Analysis	Test
	Able to register items	Input test		
	Optional receipt for customer	Input test		
	Cashier needs to immediately verify eligibility after first restricted item	Functional inspection		
	Cashier must be able to modify registered item and quantity	Input test		
	Query Item IDs, Prices from central box	Functional inspection		
	Communication with central box using Ethernet zeroMQ	Integration test		
	Customer Display - Item scanned now, quantity, price total	Functional test		
	Cashier Display - Line items, Total Cost, Transaction ID	Functional test		
	Scalable up to 50 stations	Design inspection		
	System can handle payment using credit card	Input test		
	System can handle payment using cash	Input test		
	Cashier can cancel transaction	Functional test		
	Customer is able to return items	Functional test		

Table 1: caption

*

Lecture 6 - Requirements as Presented in the Class

Requirements as shown to the class

1. Scan 2 items per second.
2. Scaleable up to 50 stations.
3. Pay by cash or card.
4. Transaction cancellation.
5. zmq use for communication (port and ip). No the exact server, merely the interface.
6. Info for customer on customer display.
    - Total at the end.
    - Current total.
    - Current item.
7. Cash tracking (number fo sales).
8. optional receipt printing

- A simple item is a item representation with only the most necessary information. (id,name,barcode).
- We should be able to make a new receipt (`newReceipt()`).
- Complete transaction by printing the receipt `completeTransaction()` .
- We need to be able to access the receipt in different contexts `getRecipt()`.
- `getPaymentStatus()` show us the status, either it is `payed()` or `canceled()`, these returns strings verifying the information from the database.

9. Use data base interface.
10. Employee screen.
    - Active receipt.
    - Current/final total.
11. Possibility of returning item/items. Policy/Protocol:
    - 30 day limitation.
    - Requires receipt.
    - Time stamp in receipt.
12. Make it user friendly (When you can scan one item, a multiplier can be added in order to not scan i.e 100 bananas). We should add shortcuts for tedious tasks in the system.
13. Edit receipt.
14. USER FRIENDLY!
15. Special item alert - Notifying when special items are being scanned by the system.
16. Acceptable wrong info: Name, price. We assume that the barcode always is correct. The barcode is ground truth.
17. Query item data. Meaning the customer wants the price of some product, maybe it is not shown on the shelf.
18. Luha algorithm for credit card validation.
19. Reboot station functionality for simple maintenance, done by the employee.
20. Any receipt size, which regards to the number of items in a transaction - a must.
21. USB for barcode scanner, card swiper, customer display.
22. A visual representation of the system in form of diagrams.

Amount of stock is not up for us.

Return item Protocol:

1. Customer provides receipt
2. The employee should input the receipt id.
3. The system checks the id to verify the time stamp etc. This is done on the station box.
4. How to represent the returning phenomenon in the system, a negative buy? Return items class as mentioned by Aske.