Lab 6 : Class diagram

In this assignment you will continue designing the station box for the check-out system. You can continue with the use cases you made in lab 1 or you can base it on the use cases and requirements below. Starting from the employee (Check-out assistant) use-cases, design a class diagram. A good starting point is to think of the nouns (item, receipt etc.) as classes and the verbs (edit, register etc.) as methods of the classes. Also consider the interfaces (display, numpad, database, bank etc.) as adapter classes.

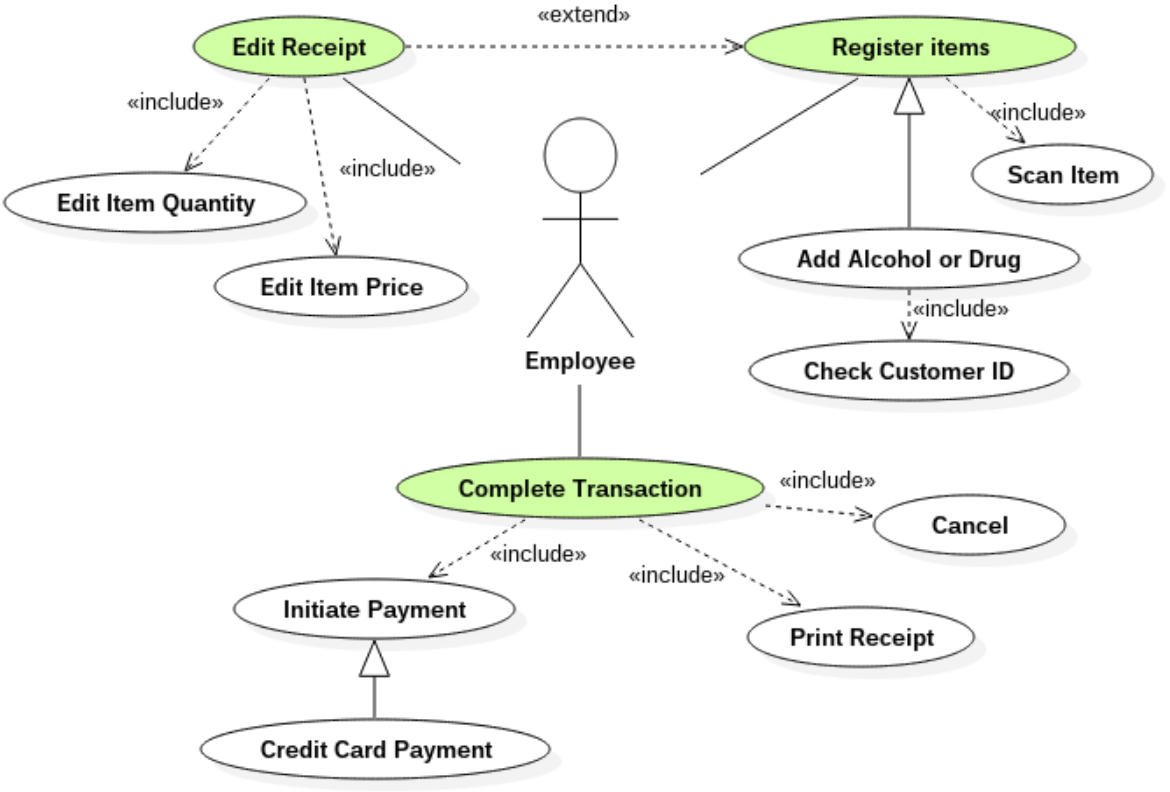

Figure 1 - Use-case diagram for the employee

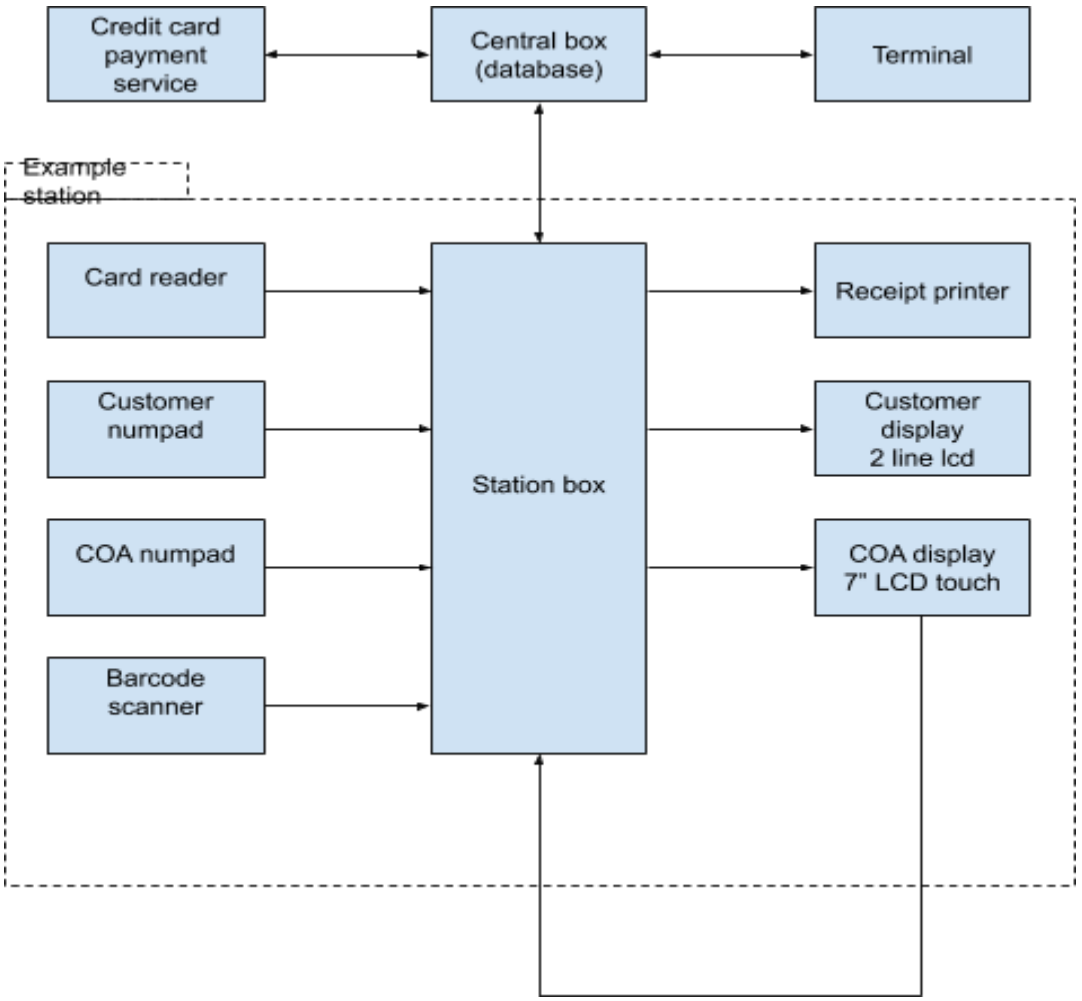

Figure 2 - Block diagram of the station box interfaces

Use case:	**Register items**
Actors:	Employee
Purpose:	To register items on a receipt
Overview:	The Employee registers the items to generate a receipt
Scope:	Global
Level:	System
Preconditions:	System is running. Customer ready to check-out.
Postconditions:	The receipt is ready for payment
Special requirements:	None

Actor action	**System response**
1. Employee scans item	
	2. Retrieves item info from database
	3. Displays info on customer display
	4. Adds item to receipt. Return to 1.

Alternative flow of events

Line 1 : If item is restricted the employee checks customer ID
Line 1 : If item has no readable barcode, item number is entered on keypad
Line 1 : If identical items, employee types quantity + ‘x’ on the keypad before scanning
Line 2 : If database returns error. Sound warning and return to 1.
Line 3 : If customer disagrees, complete ‘Edit Receipt’ use-case before continuing
Line 4 : If first item, request receipt number and create new receipt.
Line 4 : If last item, continue to Complete Transaction use-case.

Use case:	**Edit receipt**
Actors:	Employee
Purpose:	To edit a receipt
Overview:	The employee edits price or quantity of one or more lines in the receipt
Scope:	Global
Level:	Subsystem
Preconditions:	System running. Receipt is open
Postconditions:	The receipt is edited
Special requirements:	None

Actor action	**System response**
1. Employee presses ‘edit receipt’ and selects the line to be edited.	
	2. Displays the line on the employee display
3. Employee selects quantity, price or delete line.	
	4. Updates the receipt

Alternative flow of events

Use case:	**Complete Transaction**
Actors:	Employee
Purpose:	To pay a receipt
Overview:	The employee receives payment for the receipt from the customer
Scope:	Global
Level:	System
Preconditions:	System running. Register items completed
Postconditions:	The receipt is payed
Special requirements:	None

Actor action	**System response**
1. Employee presses total	
	2. Calculates total for the receipt and displays total on customer display.
3. Employee enters cash amount	
	4. Calculates change and displays change on customer display and employee display. Marks receipt as payed and prints receipt.
5. Employee returns change and hands out the printed receipt.	

Alternative flow of events

Line 3 : If customer pays with credit card, finish ‘Credit Card Payment’ use-case before continuing this use-case

Line 3 : If the customer is unable to pay, the receipt is cancelled.

Use case:	**Credit card payment**
Actors:	Customer, bank
Purpose:	Make a credit card payment
Overview:	The customer swipes his card, inputs PIN and approves amount.
Scope:	Complete transaction
Level:	Subsystem
Preconditions:	System is running. Amount received from check-out system.
Postconditions:	The amount has been paid
Special requirements:	If the transaction fails, the customer must be prompted to start over.

Actor action	**System response**
1. Customer swipes the credit card.	
	2. Card number is verified and buffered. Customer is prompted to input PIN
3. Customer inputs PIN	
	4. PIN is buffered. Customer is prompted to approve amount.
5. Customer approves amount	
	6. Card number, PIN and amount are encrypted and send to the bank
	7. Bank approves the transaction.

Alternative flow of events

Line 2 : Invalid card number. Indicate error. Ask user to swipe again. Return to step 1.
Line 5 : Customer does not approve the amount. Go to ‘edit receipt’ use-case.
Line 6 : Wrong PIN returned from bank. Return to step 1.
Line 7 : Bank rejects the transaction. Ask customer to pay with different method.

Requirements specification

FR = Functional requirements (what does the system do?)
OR = Operational requirements (how does the system interact with its environment?)
QR = Quality of service requirements (how much?)
PR = Parametric requirements (non-functional and non-operational)
DR = Design requirements

The system must be able to:

1. display information on the customer display (FR)
    1.1. RS-232 protocol interface (OR)
    1.2. Display protocol (OR)
2. print receipts on the receipt printer (FR)
    2.1. USB interface (OR)
    2.2. ESC/POS command system (OR)
3. display information on the employee display (FR)
    3.1. HDMI interface (OR)
4. take touch input from the employee display (FR)
    4.1. USB interface (OR)
5. communicate with the credit card payment service (FR)
    5.1. TCP/IP interface (OR)
    5.2. Credit card payment service interface (OR)
6. take input from the card reader (FR)
    6.1. USB interface (OR)
7. take input from the customer numpad (FR)
    7.1. GPIO interface (OR)
8. take input from the employee numpad (FR)
    8.1. GPIO interface (OR)
9. take input from the barcode scanner (FR)
    9.1. USB interface (OR)
10. play sound with buzzer (FR)
    10.1. GPIO interface (OR)
    10.2. 700Hz, 45 dB (QR)
11. register items on a receipt (FR)
    11.1. Create a new receipt (FR)
        11.1.1. Receipt numbers must be unique (PR)
    11.2. Notify employee that an item is restricted (FR)
        11.2.1. Sound notification (PR)
        11.2.2. Notification on employee display (PR)
    11.3. Notify employee that database returned error (FR)
        11.3.1. Sound notification (PR)
        11.3.2. Notification on employee display (PR)
    11.4. Take item number input from numpad (FR)
    11.5. Take quantity input from numpad (FR)
    11.6. Consistently scan 2 items per second (QR)
12. facilitate paying the receipt (FR)
    12.1. Pay by cash (PR)

12.2. Pay by credit card (PR)
13. retrieve item data from the central database (FR)
13.1. TCP/IP interface (OR)
13.2. Database protocol (OR)
13.3. Maximum 100ms latency (QR)
14. let the employee edit a currently open receipt (FR)
14.1. Select receipt line on employee display (FR)
14.1.1. Touch input (PR)
14.2. Edit quantity on employee display (FR)
14.2.1. Touch input (PR)
14.3. Edit price on employee display (FR)
14.3.1. Touch input (PR)
14.4. Delete line on employee display (FR)
14.4.1. Touch input (PR)
15. calculate receipt total (FR)
16. calculate change (FR)
17. mark receipt as paid in the central database (FR)
18. verify and buffer card number (FR)
18.1. Luhn algorithm (OR)
19. buffer PIN (FR)
20. encrypt card number, PIN and amount (FR)
20.1. SHA-256 encryption (OR)
21. System must be scalable up to 100 stations (DR)
22. Unnoticeable latencies (DR)
22.1. Maximum 100ms latency on any operation (QR)
23. Robust system design (DR)
23.1. Down time less than 0.001% (QR)

Database Interface

The communication to the database is being done through a socket. At the station's side of communication, the station box should use the operations from **DatabaseInterface**. These operations request from the database server the required data. Then the data is returned as structs (**Item, Receipt**) as seen in the following diagram.