Requirement engineering Exercise – the POS System solution

- **Problem Description**

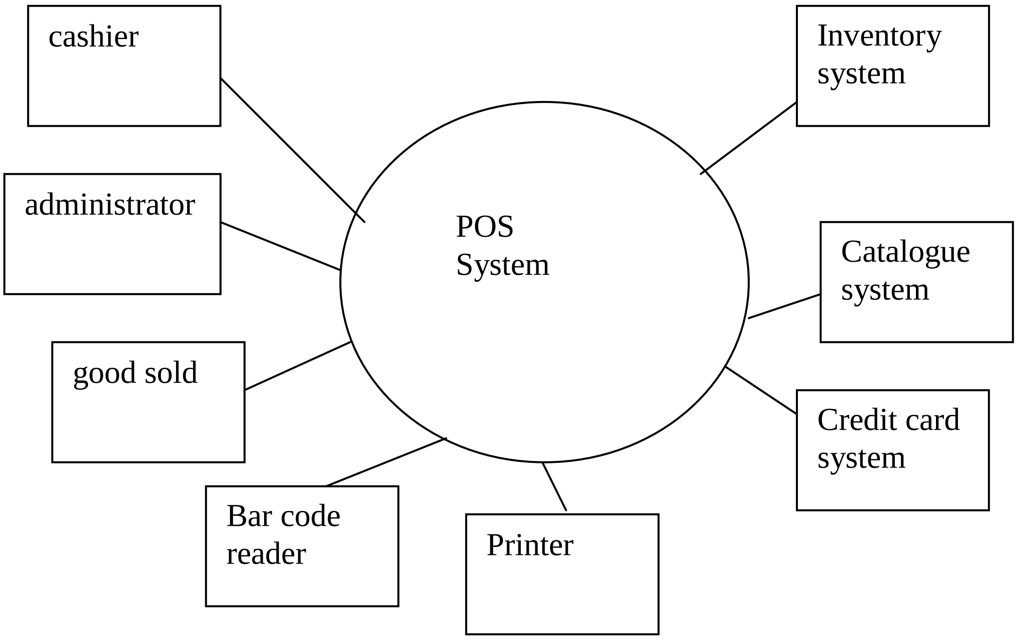
A POS (Point-Of-Sale) system is a computer system typically used to manage the sales in retail stores. It includes hardware components such as a computer, a bar code scanner, a printer and also software to manage the operation of the store.
The most basic function of a POS system is to handle sales. When a customer arrives at a POS counter with goods to purchase, the cashier will start a new sale transaction. When the barcode of a good is read by the POS system, it will retrieve the name and price of this good from the backend catalog system and interact with inventory system to deduce the stock amount of this good. When the sale transaction is over, the customer can pay in cash, credit card or even check. After the payment is successful, a receipt will be printed. Note that for promotion, the store frequently issue gift coupons. The customer can use the coupons for a better price when purchasing goods.

Another function of a POS system is to handle returns…

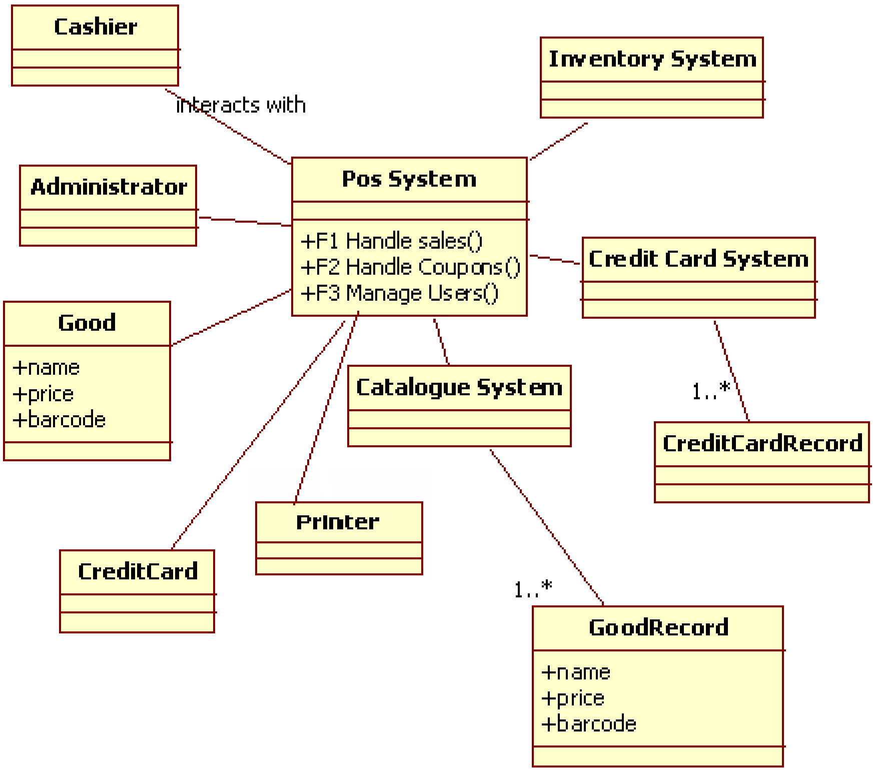
A user must log in to use the POS. The users of a POS system are the employees of the store including cashiers and the administrator. The administrator can access the system management functions of the POS system including user management and security configuration that cashiers can't do.

1. Context diagrams and interfaces.

1-a Define the context diagram of the application



Context diagram as UML class diagram

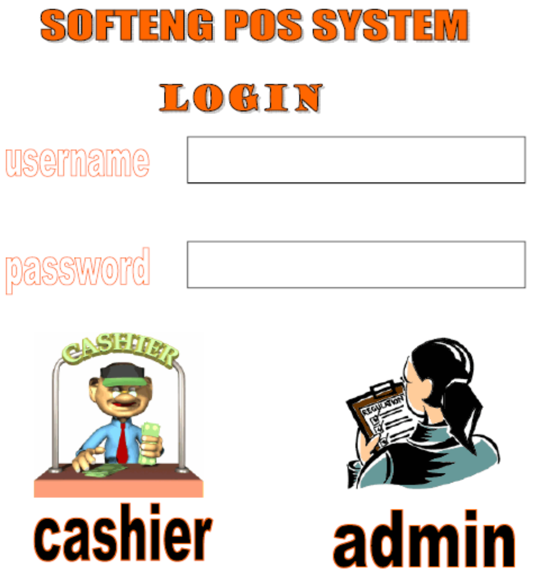


1-b Describe the interfaces of the application (to other systems/devices)

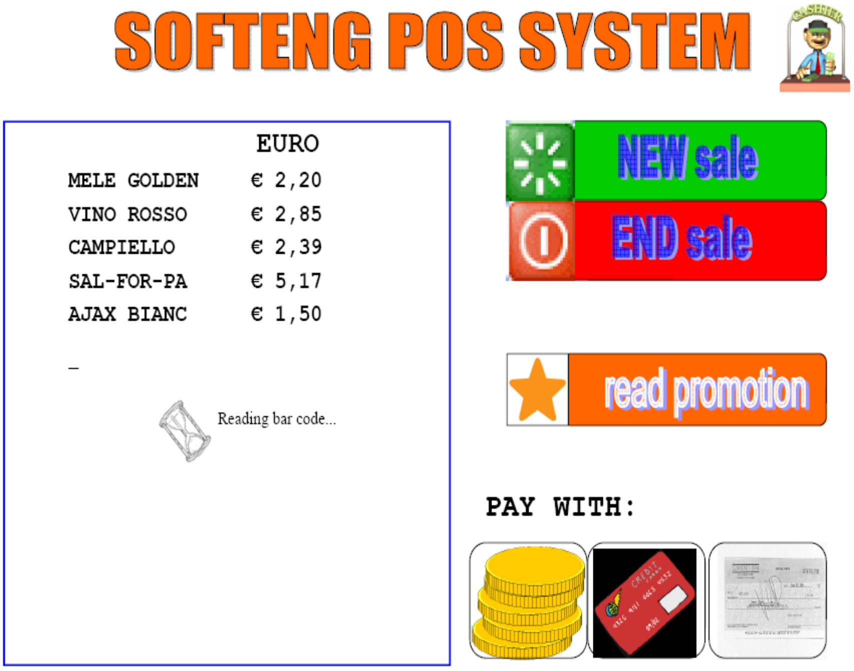
	Physical	**Procedural**	**Data**
Printer	USB 2.0		Format of receipt
Credit card system	Internet connection	URL with web service (SOAP – http + xml), and SSL	Format of credit card data sent, error descriptions
Catalogue system		RPC retrieveProduct(String barcode)	barcode : String
Inventory system		RPC deduceAmount(String barcode, int amount)	Result: boolean

1-c Describe the interfaces of the application (to users)

- Screenshot login



- Screenshot sales



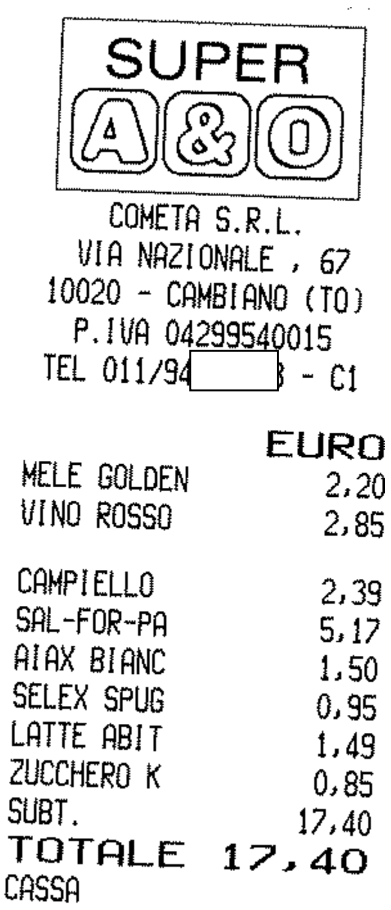
SOFTENG POS SYSTEM

	EURO
MELE GOLDEN	€ 2,20
VINO ROSSO	€ 2,85
CAMPIELLO	€ 2,39
SAL-FOR-PA	€ 5,17
AJAX BIANC	€ 1,50

_

Reading bar code...

- Paper receipt

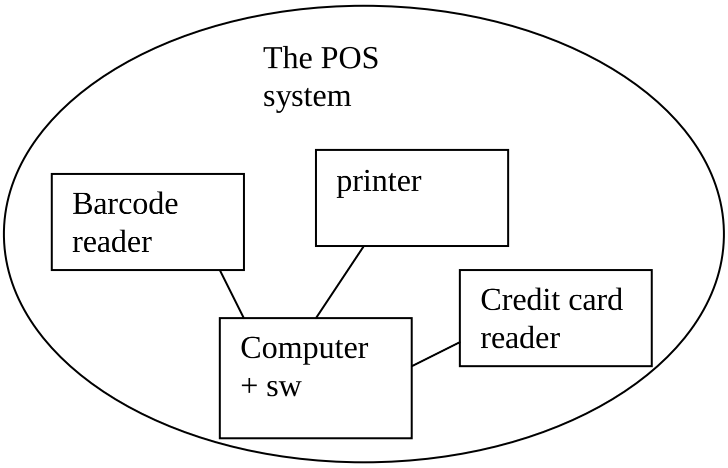


SUPER
A & O

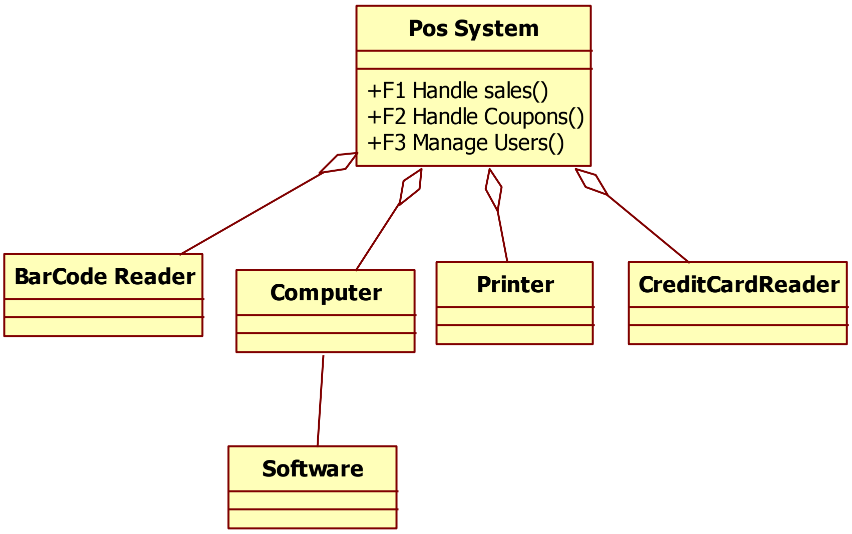
COMETA S.R.L.
VIA NAZIONALE , 67
10020 - CAMBIANO (TO)
P.IVA 04299540015
TEL 011/94[redacted] - C1

	EURO
MELE GOLDEN	2,20
VINO ROSSO	2,85
CAMPIELLO	2,39
SAL-FOR-PA	5,17
AIAX BIANC	1,50
SELEX SPUG	0,95
LATTE ABIT	1,49
ZUCCHERO K	0,85
SUBT.	17,40
TOTALE	17,40
CASSA	

1-d System design
System design as free skecth



System design as UML class diagram



2. User requirements.

2-a Define the user requirements, notably using a table with functional and non functional requirements.

Requirement ID	**Description**
F1	Handle sales
F1.1	Retrieve name and price of good
F1.2	Handle payment
F1.2.1	Handle payment cash
F1.2.2	Handle payment check
F1.2.3	Handle payment credit card
F1.3	Print receipt
F1.4	Read bar code of good
F1.5	Deduce stock amount of good
F1.6	Compute total amount
F2	Handle coupons
F2.1	Issue coupon
F2.2	Make discount to coupon owner
F3	Manage Users
F3.1	Handle login
F3.2	Handle logout
F3.3	Define user, define user rights
	Needs more analysis
F4	Handle returns
	Needs more analysis
NF	Each function less than ½ sec
	Secure payments (F1.2 to F1.2.3)
	Warranty access to functions only to authorized users → f3

2-b Define the user requirements. As an alternative to the technique above described each requirement with the following form (from 03_requirements slides)

Name	F1.1 retrieve name and price of good
Description	
Input	Barcode
Output	Name and price of product
Action	Access backend catalogue system and given barcode find and retrieve product description
Precondition	Valid barcode (consistent with standard + corresponding product exists)
Postcondition	Price and name of product available

Name	F1.2.1 Handle payment cash
Description	
Input	Amount to be paid, cash received
Output	Change to be given
Action	If result ok, Add amount to be paid to total amount of cash
Pre condition	Cash received >= amount to be paid
Post condition	Amount of cash (after payment) = amount of cash (before payment) + amount to be paid

Name	F1.5 deduce stock amount
Description	
Input	Barcode
Output	Stock amount after this sale
Action	Deduce one from stock amount of product
Precondition	Valid code, at least one product in stock
Postcondition	Product.stockAmount_after == Product.stockAmount_before - 1

Name	F2 – Handle coupon
Description	
Input	Coupon (id of coupon, id of promotion, id of product)
Output	Discounted price for product
Action	Read bar code on coupon , find id of coupon, retrieve name of promotion and id of product, verify that promotion is still valid, verify that product is available, retrieve discount rate

2-c Define scenarios of use with the following template (from heating control system)

Scenario name: Sale n product S0	**General description** Use case: F1 Handle sales Sale N products, payment cash	
Step	**Description**	**Requirement ID**
1	Start sales transaction	F1
2	Read bar code	F1.1
3	Retrieve name and price given barcode	F1.4
	Repeat 2 and 3 for all products	
4	Compute total	F1.6
5	Manage payment cash	F1.2.2
6	Deduce stock amount of product	F1.5
7	Print receipt	F1.3
8	Close transaction	F1.4

Scenario name: S1	**General description** Use case: F1 Handle sales Sale 1 product	
Step	**Description**	**Requirement ID**
1	Start sales transaction	F1
2	Read bar code	F1.1
3	Retrieve name and price given barcode	F1.4
4	Compute total	F1.6
5	Manage payment cash	F1.2.2
6	Deduce stock amount of product	F1.5
7	Print receipt	F1.3
8	Close transaction	F1.4

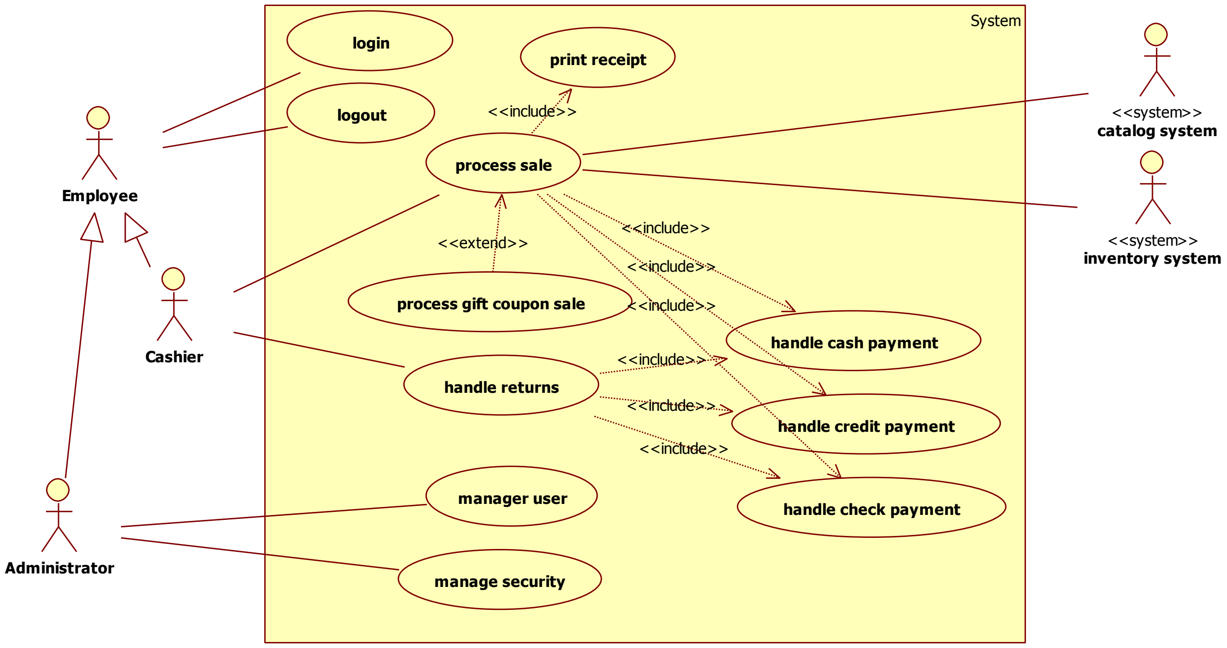
Scenario name: S2	**General description** Use case: F1 Handle sales Handle sale , no credit, abort sale	
Step	**Description**	**Requirement ID**
1	Start sale	F1
2	Read bar code	F1.1
3	Retrieve name and price given barcode	F1.4
4	Compute total	F1.6
5	Manage payment credit card	F1.2.3
7	No credit, abort sale	F4

Scenario name S3	**General description** Use case: F1 Handle sales S3: Sale of 3 goods, payment cash	
Step	**Description**	**Requirement ID**
0	Start new sale transaction	F1.7
1	Read bar code of good	F1.4
2	Retrieve name and price of good	F1.1
3	Read bar code of good	
4	Retrieve name and price of good	
5	Read bar code of good	F111
6	Retrieve name and price of good	
7	Compute total amount of sale	F1.6
8	Handle payment cash	F1.2.1
9	Print receipt	F1.3
10	Deduce stock amount of good	F1.5

Scenario name S4	**General description** Use case: F1 Handle sales S4: Sale of 2 goods, payment credit card	
Step	**Description**	**Requirement ID**
0	Start new sale transaction	F1.7
1	Read bar code of good	
2	Retrieve name and price of good	
3	Read bar code of good	
4	Retrieve name and price of good	
5	Compute total amount of sale	F1.6
6	Handle payment credit card	F1.2.3
7	Print receipt	F1.3
8	Deduce stock amount of good	F1.5

Scenario name S5	**General description** Use case: F1 Handle sales S5: Sale of one good, bar code unreadable	
Step	**Description**	**Requirement ID**
0	Start new sale transaction	F1.7
1	Read bar code of good	
2	Bar code not readable	
3	Input bar code manually	F1.x (to be added)
4	Retrieve name and price of good	
5	Compute total amount of sale	F1.6
6	Handle payment credit card	F1.2.3
7	Print receipt	F1.3
8	Deduce stock amount of good	F1.5

2-d Define the use case diagrams

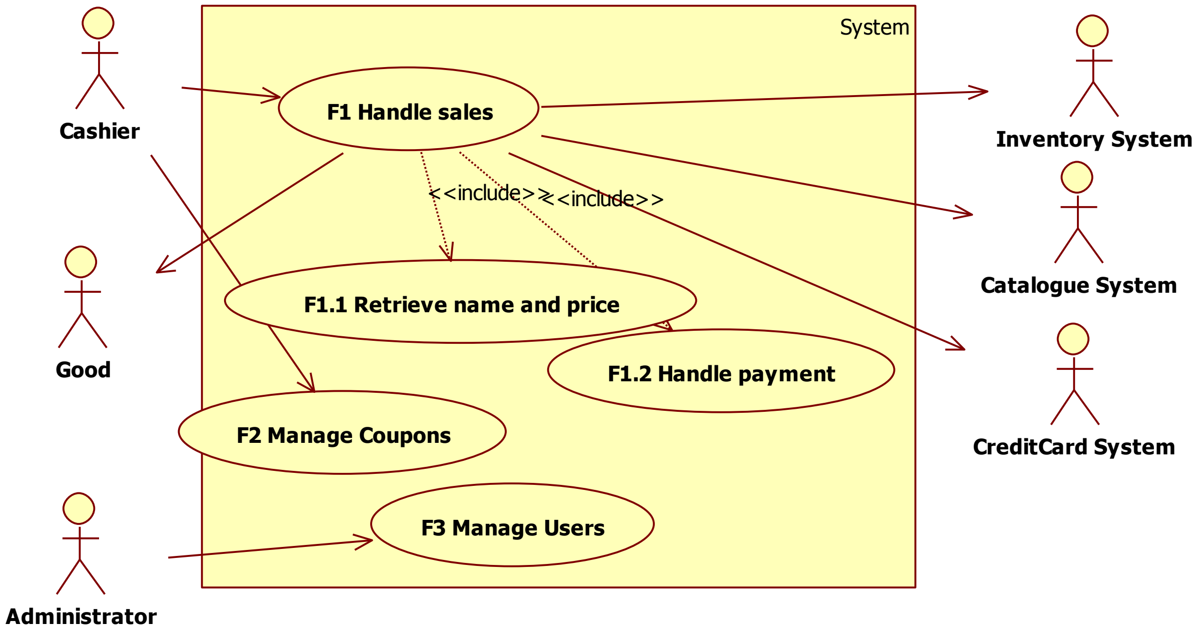


USE CASE	Process Sale	
Primary Actor	Cashier	
Preconditions	Cashier is identified and authenticated	
Success End Condition	Sale is saved. Receipt is printed. Stock data updated. Payment authorization approvals are recorded.	
Basic Flow	**Step**	**Action**
	1	Customer arrives at POS checkout with goods to purchase.
	2	Cashier starts a new sale.
	3	Cashier enters item identifier.
	4	System retrieve item information from the catalog system and, records sale line item and presents item description, price, and running total. Cashier repeats steps 3-4 until indicates done.
	5	System calculates and presents total price.
	6	Cashier tells Customer the total, and asks for payment.
	7	Customer pays and System handles payment.
	8	System records completed sale and sends sale information to the external Inventory system for stock update.
	9	System presents receipt.
	10	Customer leaves with receipt and goods.
EXTENSIONS	**Step**	**Branching Action**
	*	At any time, System fails: To support recovery and correct accounting, ensure all transaction sensitive state and events can be recovered from any step of the scenario. 1. Cashier restarts System, logs in, and requests recovery of prior state. 2. System reconstructs prior state. 2a. System detects anomalies preventing recovery: 1. System signals error to the Cashier, records the error, and enters a clean state.

		2. Cashier starts a new sale.
	3a	Invalid identifier: 1. System signals error and rejects entry.
	3b	There are multiple of same item: 1. Cashier can enter item category identifier and the quantity.
	3-6a	Customer asks Cashier to remove an item from the purchase: 1. Cashier enters item identifier for removal from sale. 2. System displays updated running total.
	3-6b	Customer tells Cashier to cancel sale: 1. Cashier cancels sale on System.
	3-6c	Cashier suspends the sale: 1. System records sale so that it is available for retrieval on any POS terminal.
	4a	The item’s price is not the customer wanted (e.g., Customer complained that the item is offered at a lower price): 1. Cashier enters override price. 2. System presents new price.
	6a	Customer says they intended to pay by cash but don’t have enough cash: 1a. Customer uses an alternate payment method. 1b. Customer tells Cashier to cancel sale. Cashier cancels sale on System.
	7a	Paying by cash: (UC Handle Cash Payment)
	7b	Paying by credit: (UC Handle Credit Payment)
	7c	Paying by check: (UC Handle Check Payment)
Non-Functional Requirements		
	1	Touch screen Ul on a large flat panel monitor. Text must be visible from 1 meter.
	2	Credit authorization response within 30 seconds 90% of the time.
	3	Somehow, we want robust recovery when access to remote services such the invent tory system is failing.
	4	Language internationalization on the text displayed.

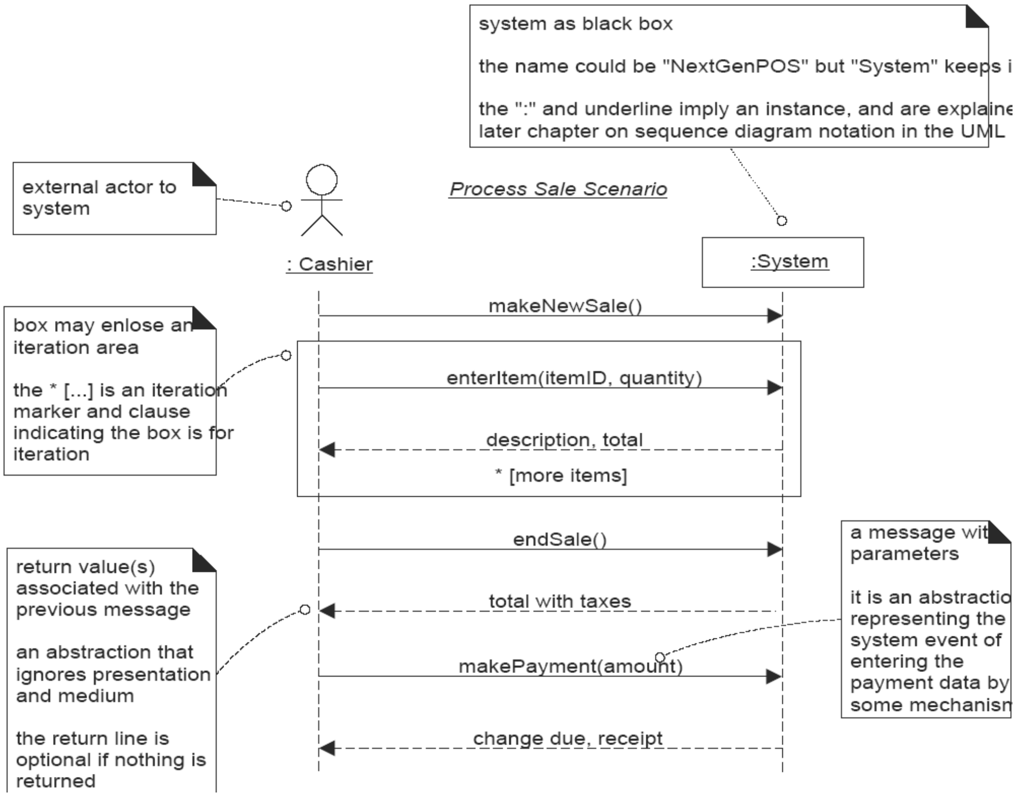
USE CASE	Process Gift Coupon Sale (extends UC Process Sale)	
EXTENSIONS	**Step**	**Branching Action**
	7a	Customer presents coupons: 1. Before handling payment, Cashier records each coupon and System reduces price as appropriate. System records the used coupons for accounting reasons. 1a. Coupon entered is not for any purchased item: 1. System signals error to Cashier.

A simpler and more efficient variant:

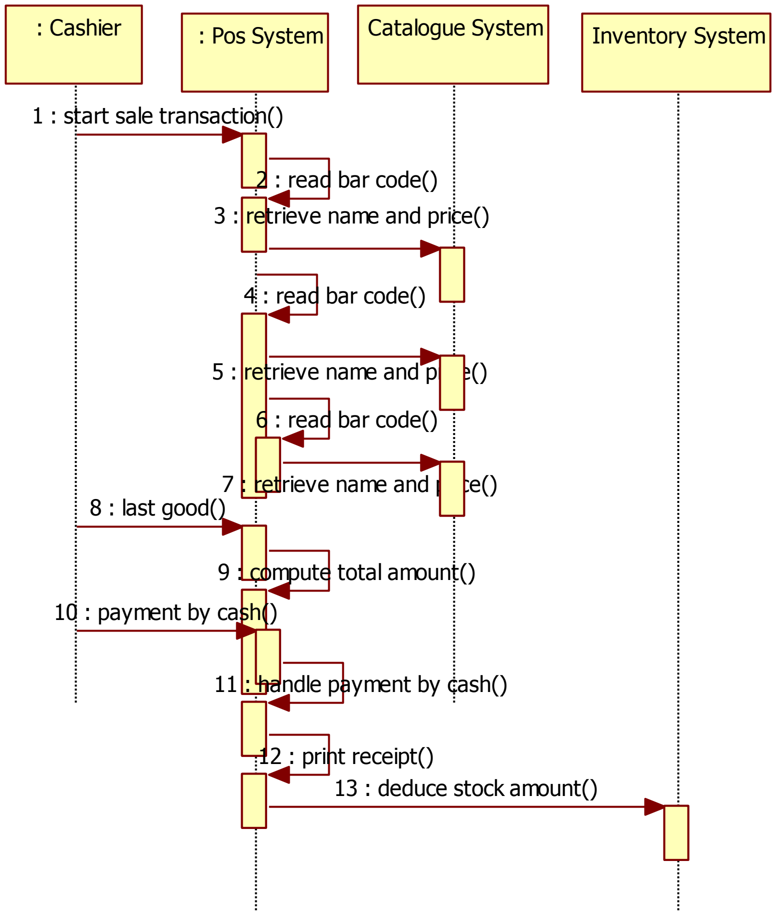

System
Cashier
F1 Handle sales
Inventory System
<<include>>
<<include>>
Catalogue System
Good
F1.1 Retrieve name and price
F1.2 Handle payment
CreditCard System
F2 Manage Coupons
F3 Manage Users
Administrator

2-e Define the sequence diagrams for some specific scenarios

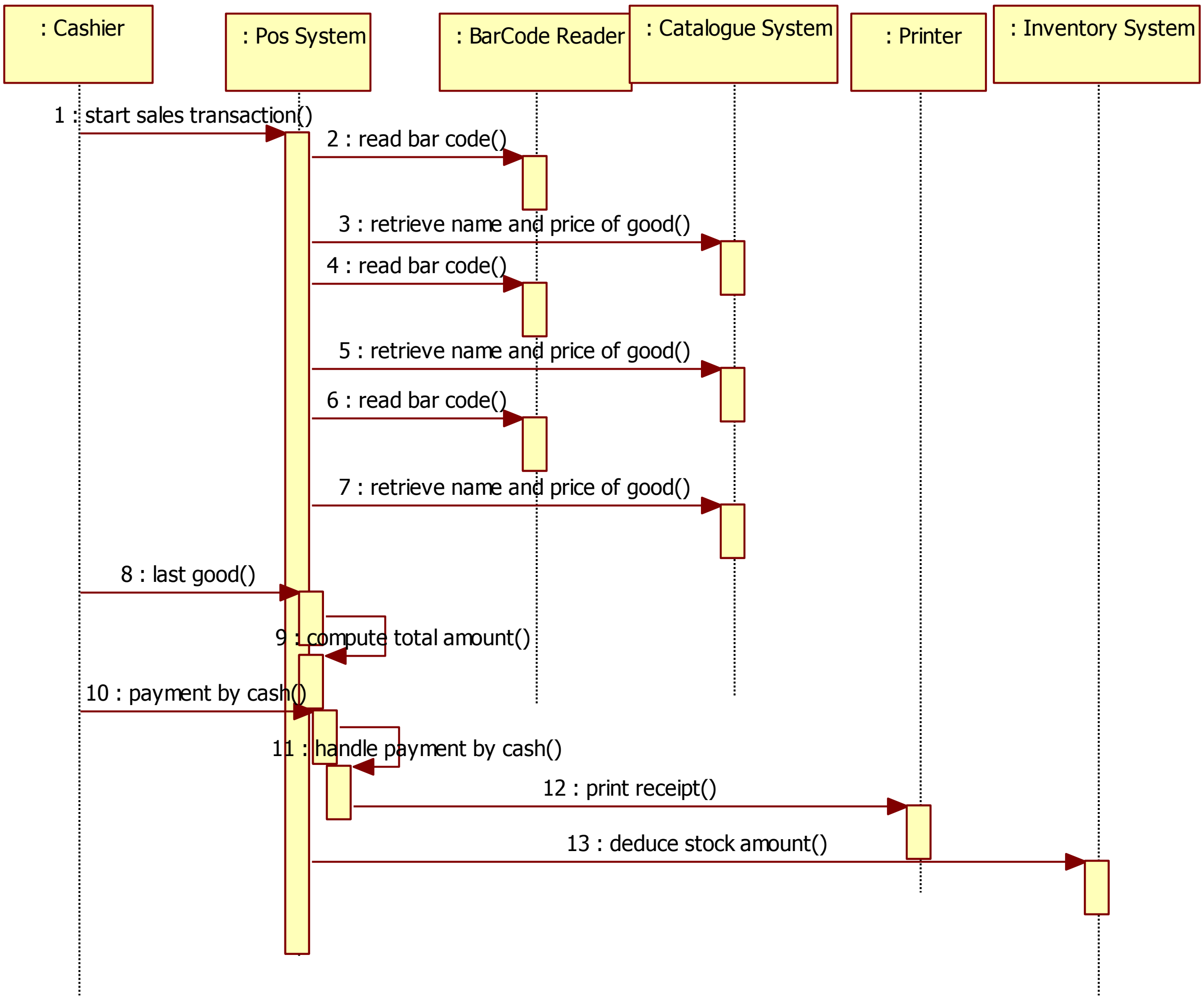
Sequence diagram for S0


system as black box
the name could be "NextGenPOS" but "System" keeps i
the ":" and underline imply an instance, and are explaine
later chapter on sequence diagram notation in the UML
external actor to system
Process Sale Scenario
: Cashier
:System
makeNewSale()
box may enlose an iteration area
the * [...] is an iteration marker and clause indicating the box is for iteration
enterItem(itemID, quantity)
description, total
* [more items]
endSale()
a message wit parameters
it is an abstractio representing the system event of entering the payment data by some mechanism
return value(s) associated with the previous message
an abstraction that ignores presentation and medium
the return line is optional if nothing is returned
total with taxes
makePayment(amount)
change due, receipt

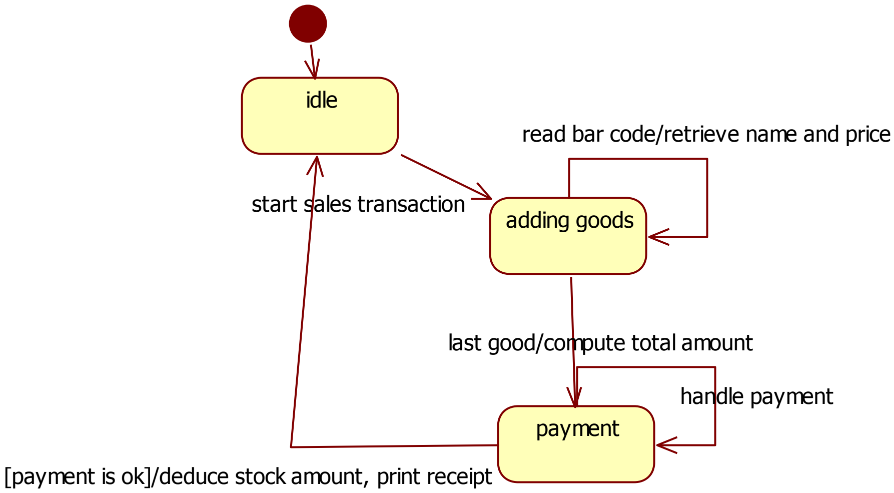
Sequence diagram for S3, no internals of POS System


: Cashier
: Pos System
Catalogue System
Inventory System
1 : start sale transaction()
2 : read bar code()
3 : retrieve name and price()
4 : read bar code()
5 : retrieve name and p e()
6 : read bar code()
7 : retrieve name and p ce()
8 : last good()
9 : compute total amount()
10 : payment by cash()
11 : handle payment by cash()
12 : print receipt()
13 : deduce stock amount()

Sequence diagram for S1, with internals of POS System as described by System design diagram


: Cashier
: Pos System
: BarCode Reader
: Catalogue System
: Printer
: Inventory System
1 : start sales transaction()
2 : read bar code()
3 : retrieve name and price of good()
4 : read bar code()
5 : retrieve name and price of good()
6 : read bar code()
7 : retrieve name and price of good()
8 : last good()
9 : compute total amount()
10 : payment by cash()
11 : handle payment by cash()
12 : print receipt()
13 : deduce stock amount()

2f - Statechart of POS System class


idle
read bar code/retrieve name and price
start sales transaction
adding goods
last good/compute total amount
handle payment
payment
[payment is ok]/deduce stock amount, print receipt