1. Use case name : sell-orders
2. Actors : employee
3. Brief description : to enable employee to write the orders of customers and give them a copy and save another copy into the DB of the market
4. Flow of events :
	1. Main flow : write the order
		1. Employee choose ‘ sell –orders ‘ option
		2. System display a empty sell order form , with a new order number
		3. Employee enters the id of customer , good id ,and the quantity for each good item , and set the discount on the order , note that the system set the date of the order implicitly then employee choose ‘add to sell’ option .
		4. The system validate the data and ensure that: good id is entered as an integer, and the quantity and discount entered as a number not letter neither character, The system check if the good’s is exist in market, by checking it’s id ,.
		5. / The system show the unit price for each good item ,the total price for a good item , and the final value of order before and after discount in a proper field in the form .
		6. The system send the order to printer to give customer a hard copy , and save the order into DB , finally show message say “done , operation success ”.
	2. Alternative flow : improper data form
		1. After step 4.1.4 (validating data ) , and the system discovers a wrong data was enterd ,display an error message , and telling the employee to re-write the wrong data .// should we write : what then happend
	3. Alternative flow: a good that written in the market is not exist in the market OR wrong good id:
		1. After step 4.1.4 , and the system find that at least one good doesn’t exist in the market display an error message saying “ no such good ”
5. Entry condition : the employee must be logged into market system
6. Exit condition : the system display a success message ,// or display a error message
7. Quality requirements : NONE .

1. Use case name : buy – order.
2. Actors : employee.
3. Brief description : : to enable employee to write the orders of customers and give them a copy and save another copy into the DB of the market
4. Flow of events :
	1. Main flow : : write the order
		1. Employee choose ‘ buy –orders ‘ option
		2. System display a empty buy order form , with a new order number
		3. Employee enters the id of company , good id , good name ,the quantity for each good item , the price of each item ,and set the discount on the order , note that the system set the date of the order implicitly ,then employee choose ‘add to buy’ option
		4. The system validate the data and ensure that: good id is entered as an integer, and the quantity and discount entered as a number not letter neither character.
		5. The system show the total price for a good item , and the final value of order before and after discount in a proper field in the form .
		6. / the system checks if there is a new good entered and save it
		7. The system send the order to printer to give the company a hard copy , and save the order into DB , finally show message say “done , operation success ”.
	2. Main flow : Delete a buy- Order
		1. employee selects “delete purchase order.”
		2. System ask the employee to enter a buy order id(include search use case )
		3. The employee enters a buy order id.
		4. System checks if the order is exist , then show the order .
		5. System display a message to ensure deleting this order
		6. Employee choose ‘ensure ’ .
		7. System delete the order.
		8. System display a message saying “order deleted ”.
	3. Alternative flow : improper data form
		1. After step (validating data ) , the system discovers a wrong data was entered ,then display an error message , and telling the employee to re-write the wrong data .// should we write : what then happened.
	4. Alternative flow : The actor selects the exit button :
		1. . If the actor choose the ‘add to buy’ option the system state is unchanged and the use case ends, else the hall operation will canceld.
	5. Alternative flow : no such id found in 4.2.4 :
		1. Sysem display message saying “no such order id found” .
	6. Alternative flow : employee choose ‘no’ in 4.2.5
		1. System cancel the operation and keep the order.
5. Entry condition : the employee must be logged into market system
6. Exit condition : the system display a success message to the employee ,// or display a error message
7. Quality requirements : NONE .
8. Use case name : customers payments
9. Actors : employee
10. Brief description : this use case used for saving and registration the payments of customer .
11. Flow of events :
	1. Main flow : add new payments for a customer
		1. Employee choose ‘customer payments’ option from main work space.
		2. Systems display a frame for payment .
		3. Employee fill the frame with data that include : customer id and the value of payment, note that the system set the date of the payment implicitly .
		4. System validate the data enterd and ensure that : the coustomer id is an integer number and value of payments is a number only ,also System ensure that the number of customer is already exist(or known) in the market, furthermore the system ensure that the value of the payment is less than or equal to the debt of the coustomer .
		5. System save the payment for coustomer into DB and display message “done”.
	2. Alternative flow : invalid data :
		1. After step 4.1.4 if there are a wrong in the data that input , the system display a massage say “ wrong data input ” , and ask the user to correct the data
	3. Alternative flow : wrong customer id
		1. After step 4.1.4 if the customer id isn’t found in DB system display a message say “no such customer ” .
	4. Alternative flow : the payment is larger than the debt of the coustomer
		1. After step 4.1.4 if the system find the above , system=m display a message saying “the payment’s exceed the value of order’s” .
12. Entry condition : the employee must log in .
13. Exit condition : the system display a success message to the employee .
14. Quality requirements : NONE .
15. Use case name : company payments.
16. Brief description : this use case used for saving and registration the payments of company .
17. Actors : employee.
18. Flow of events :
	1. Main flow : add new payments for a company.
		1. Employee choose ‘company payments’ option from main work space.
		2. Systems display a frame for payment.
		3. Employee fill the frame with data that include : customer id and the value of payment, note that the system set the date of the payment implicitly.
		4. System validate the data enterd and ensure that : the company id is an integer number and value of payments is a number only ,also System ensure that the number of company is already exist(or known) in the market, furthermore the system ensure that the value of the payment is less than or equal to the debt of the market to company .
		5. System save the payment for company into DB and display message “done”.
	2. Alternative flow invalid data :
		1. After step 4.1.4 if there are a wrong in the data that input , the system display a massage say “ wrong data input ” , and ask the user to correct the data.
	3. Alternative flow : wrong company id
		1. After step 4.1.4 if the company id isn’t found in DB , system display a message say “no such company ”.
	4. Alternative flow : the payment is larger than the debt of market for the company.
		1. After step 4.1.4 if the system find the above , system=m display a message saying “the payment’s exceed the value of order’s”.