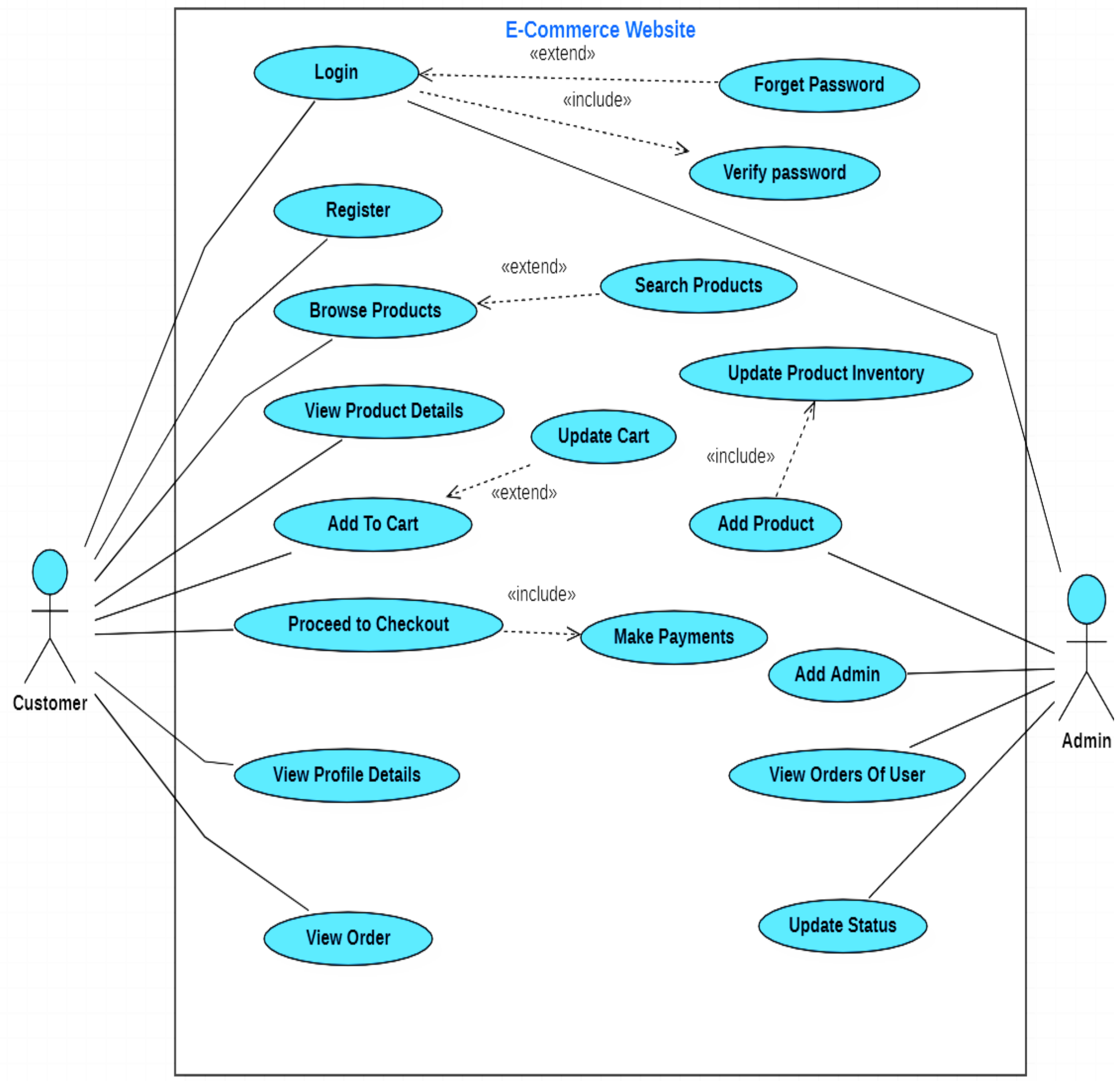
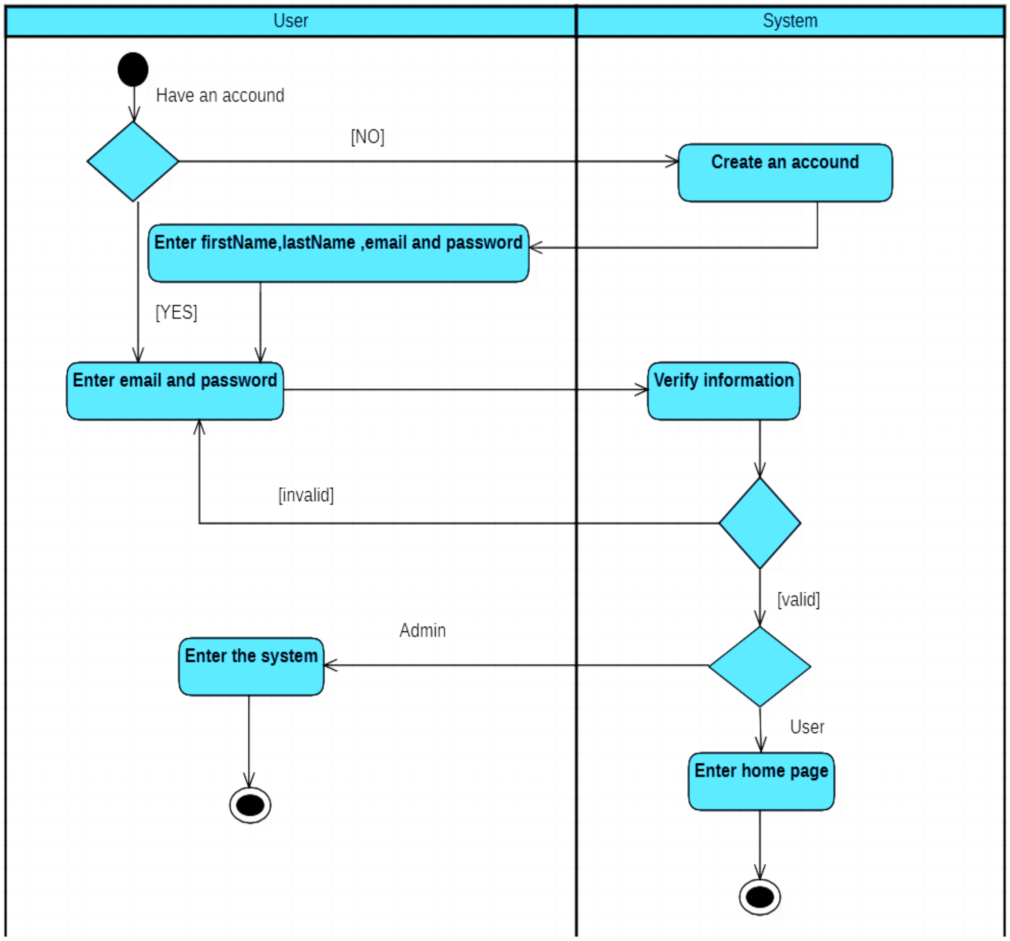
Use Case Diagram

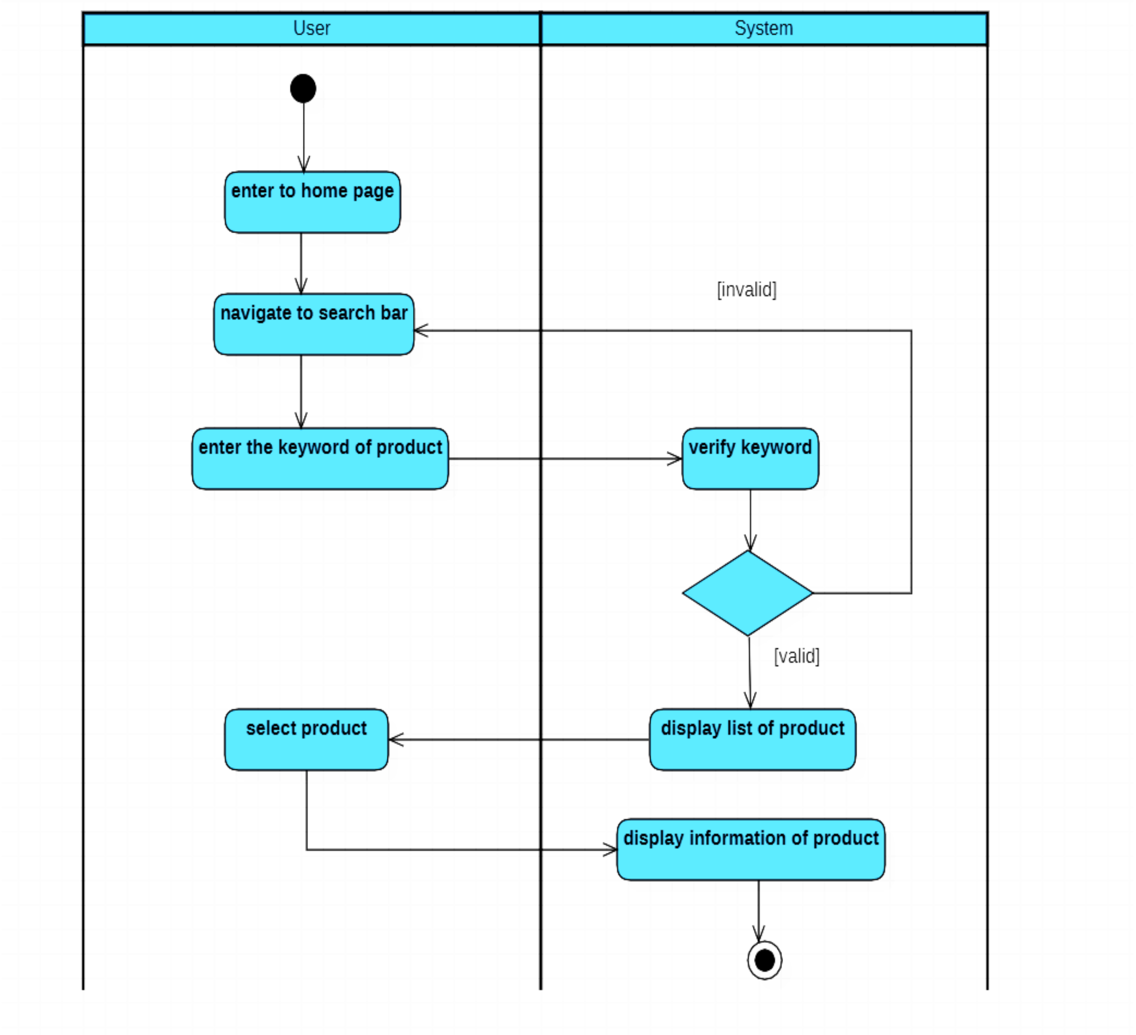


Activity Diagram

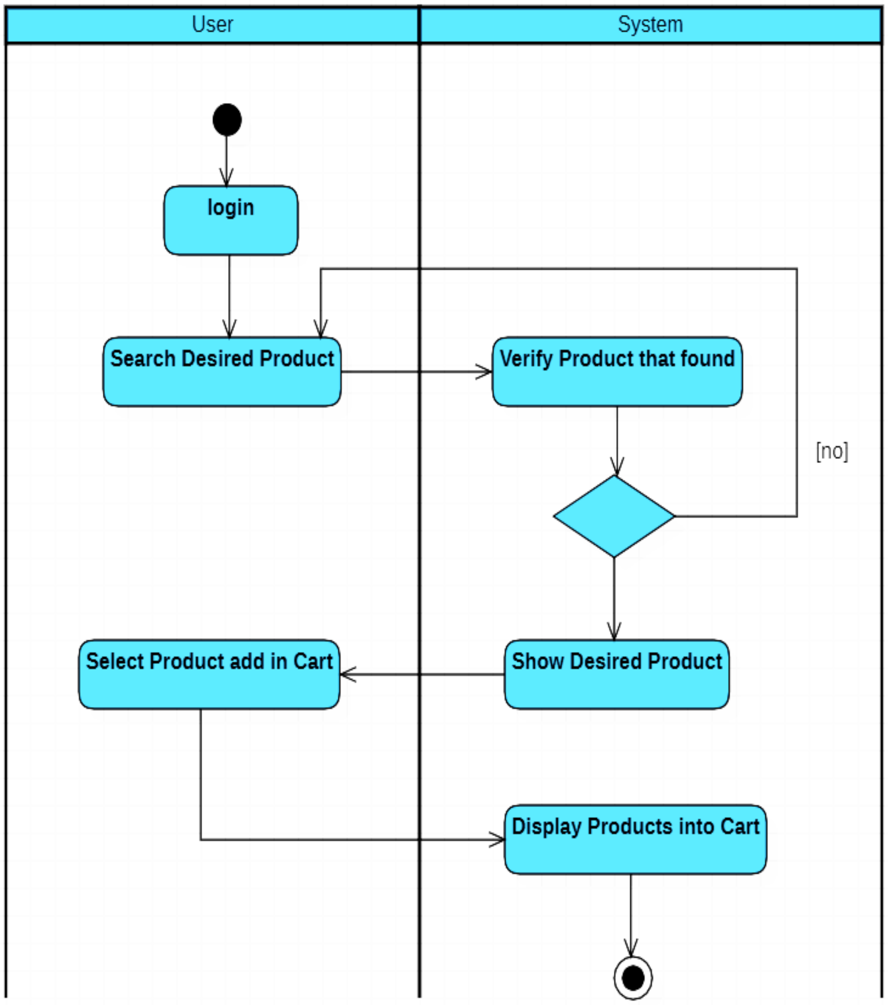
Login:



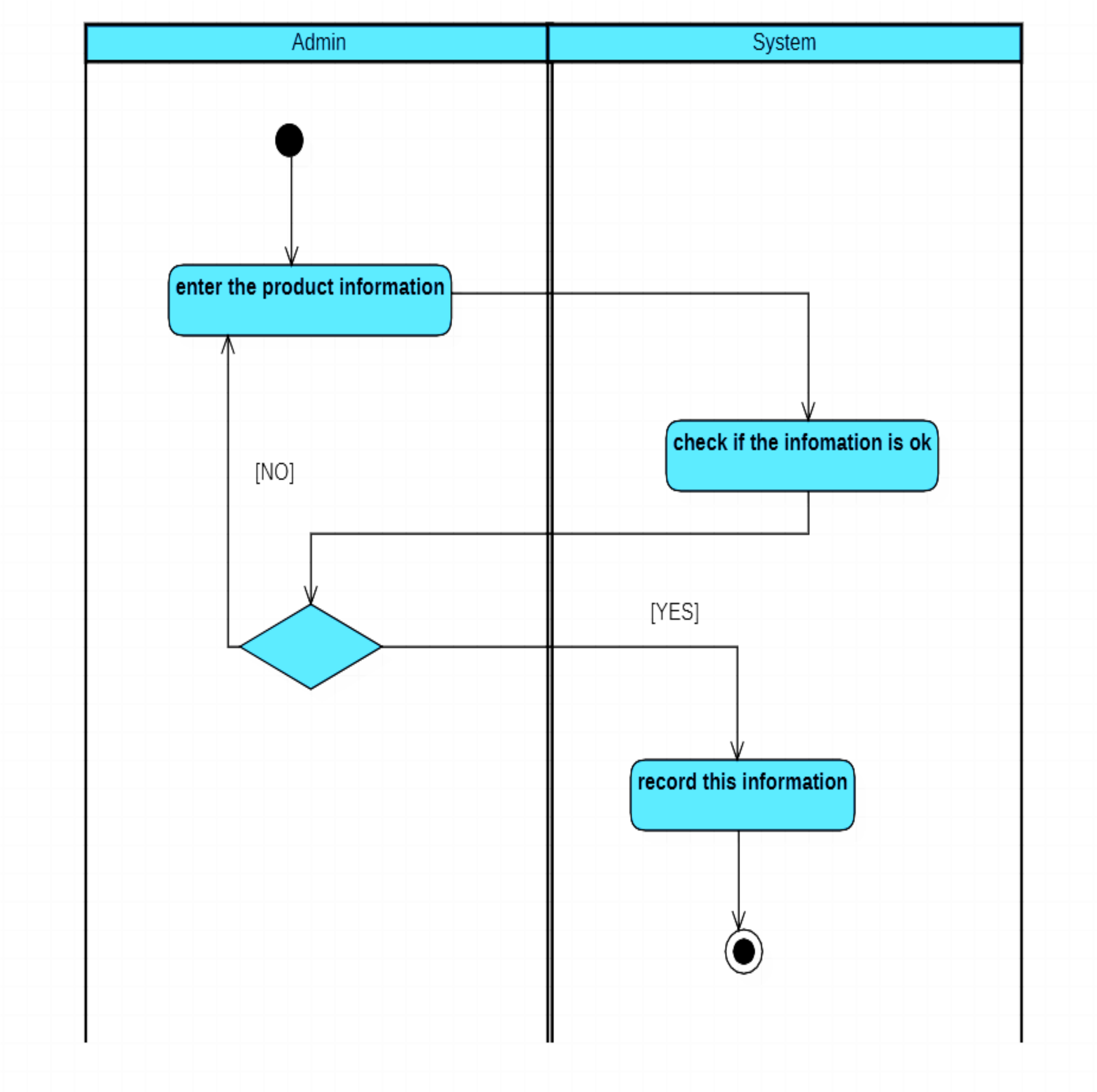
Search:


User
System
enter to home page
navigate to search bar
[invalid]
enter the keyword of product
verify keyword
[valid]
select product
display list of product
display information of product

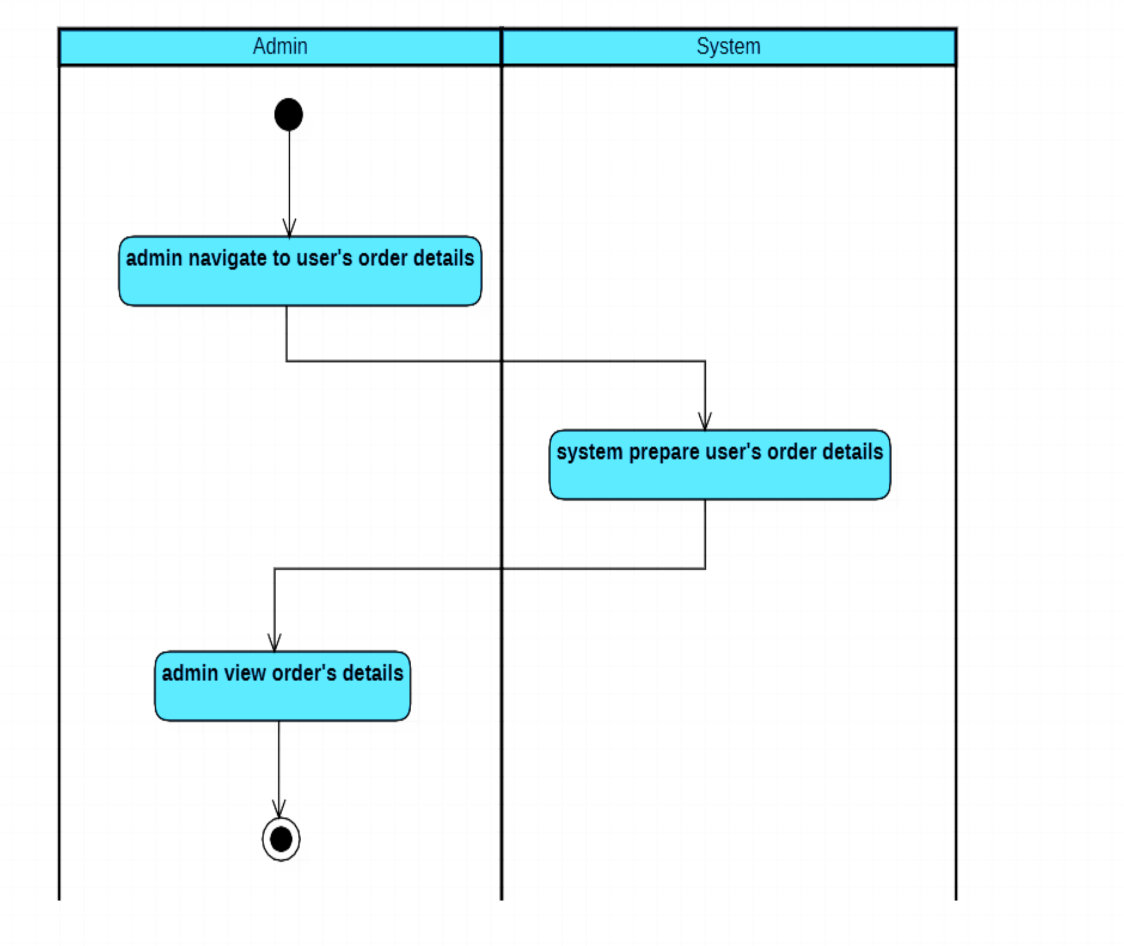
Add To Cart:


User
System
login
Search Desired Product
Verify Product that found
[no]
Select Product add in Cart
Show Desired Product
Display Products into Cart

Add Product:

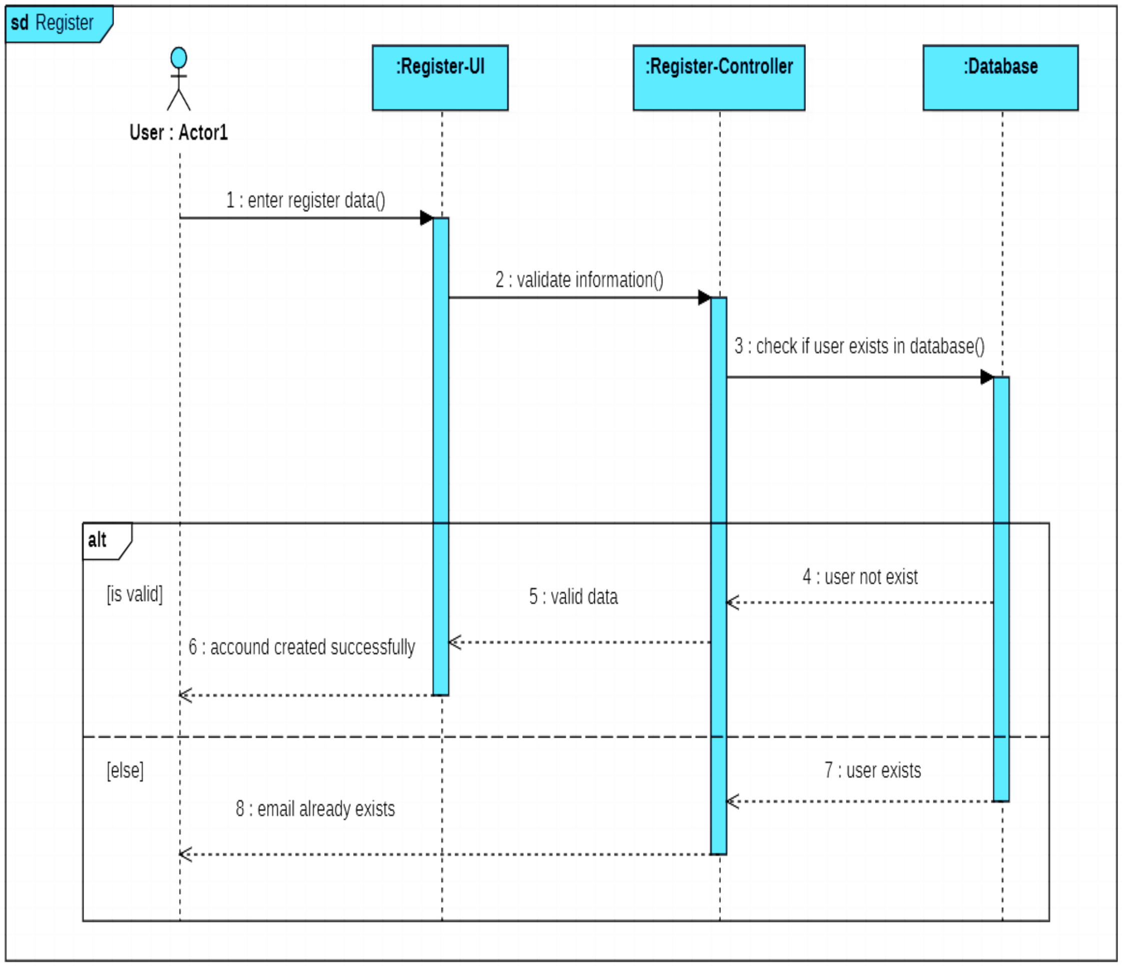

Admin
System
enter the product information
check if the infomation is ok
[NO]
[YES]
record this information

View Order’s User:

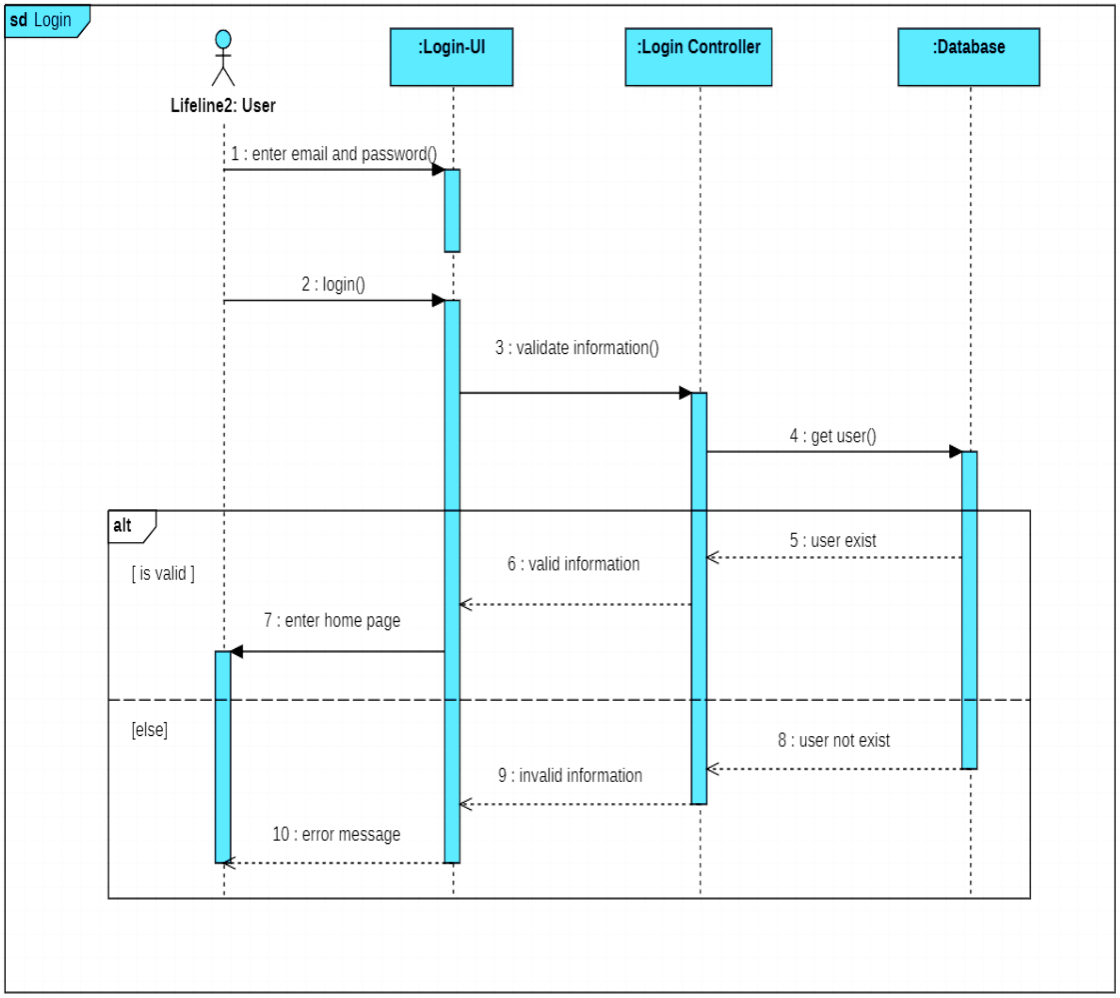

Admin
System
admin navigate to user's order details
system prepare user's order details
admin view order's details

Sequence Diagram

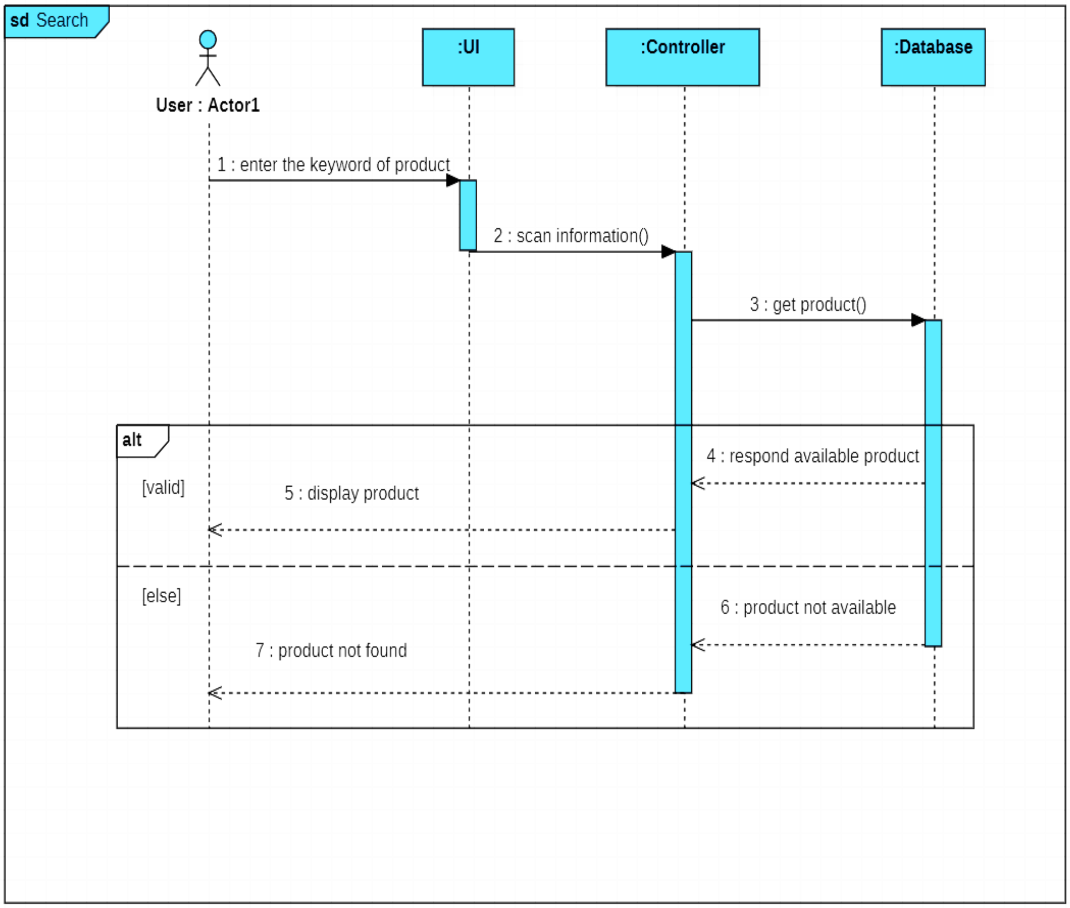
Register:



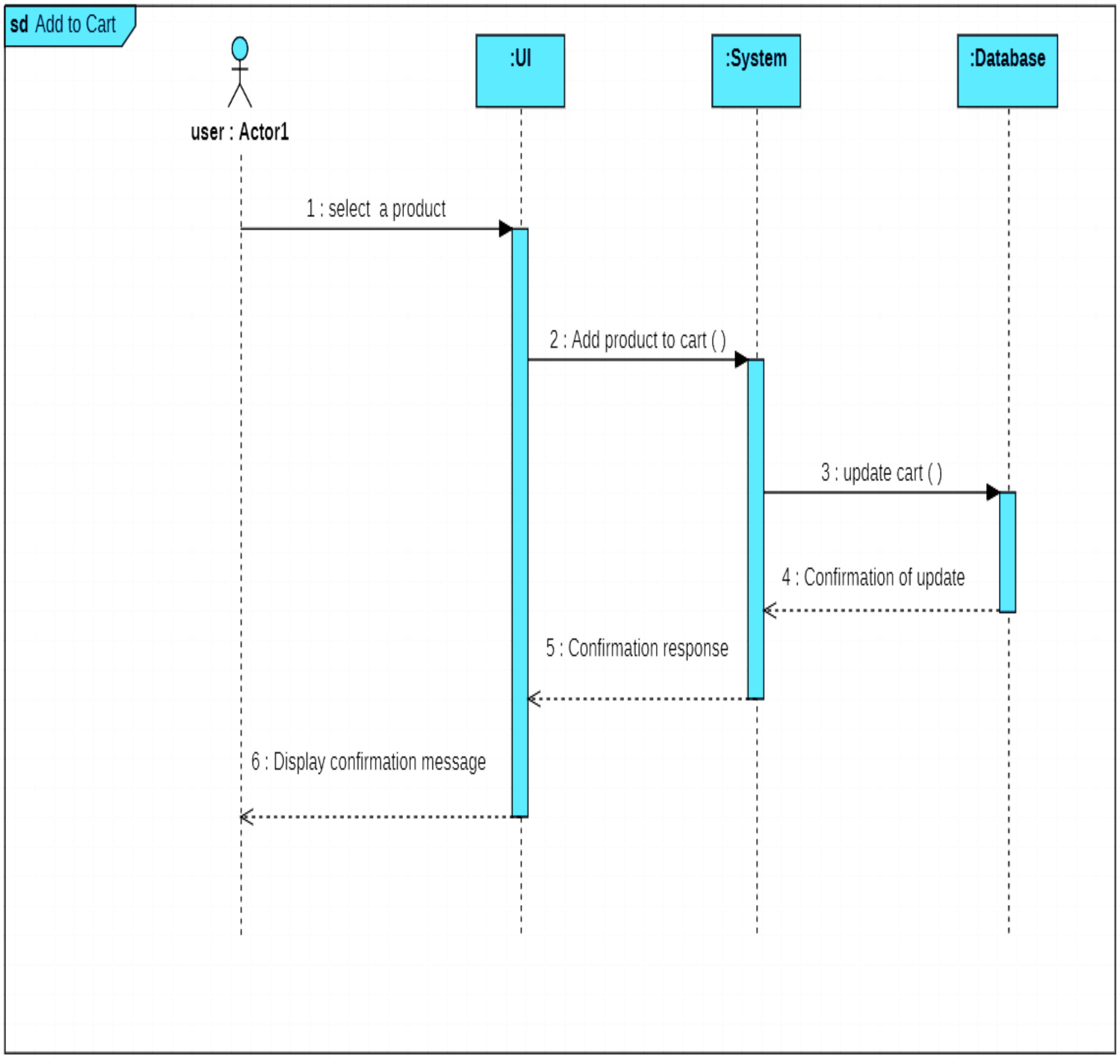
Login:


sd Login
Lifeline2: User
:Login-UI
:Login Controller
:Database
1 : enter email and password()
2 : login()
3 : validate information()
4 : get user()
alt
[is valid]
5 : user exist
6 : valid information
7 : enter home page
[else]
8 : user not exist
9 : invalid information
10 : error message

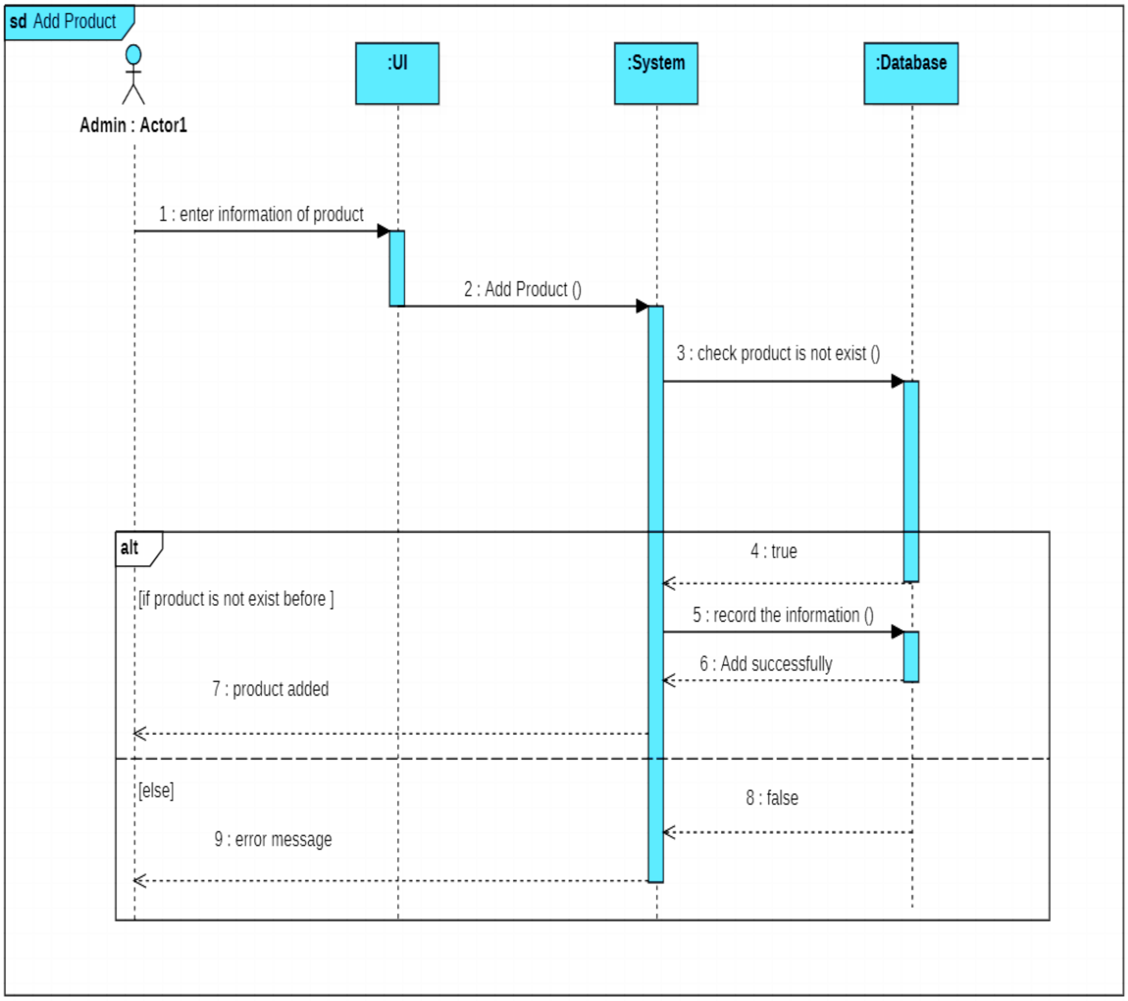
Search:


sd Search
User : Actor1
:UI
:Controller
:Database
1 : enter the keyword of product
2 : scan information()
3 : get product()
alt
[valid]
4 : respond available product
5 : display product
[else]
6 : product not available
7 : product not found

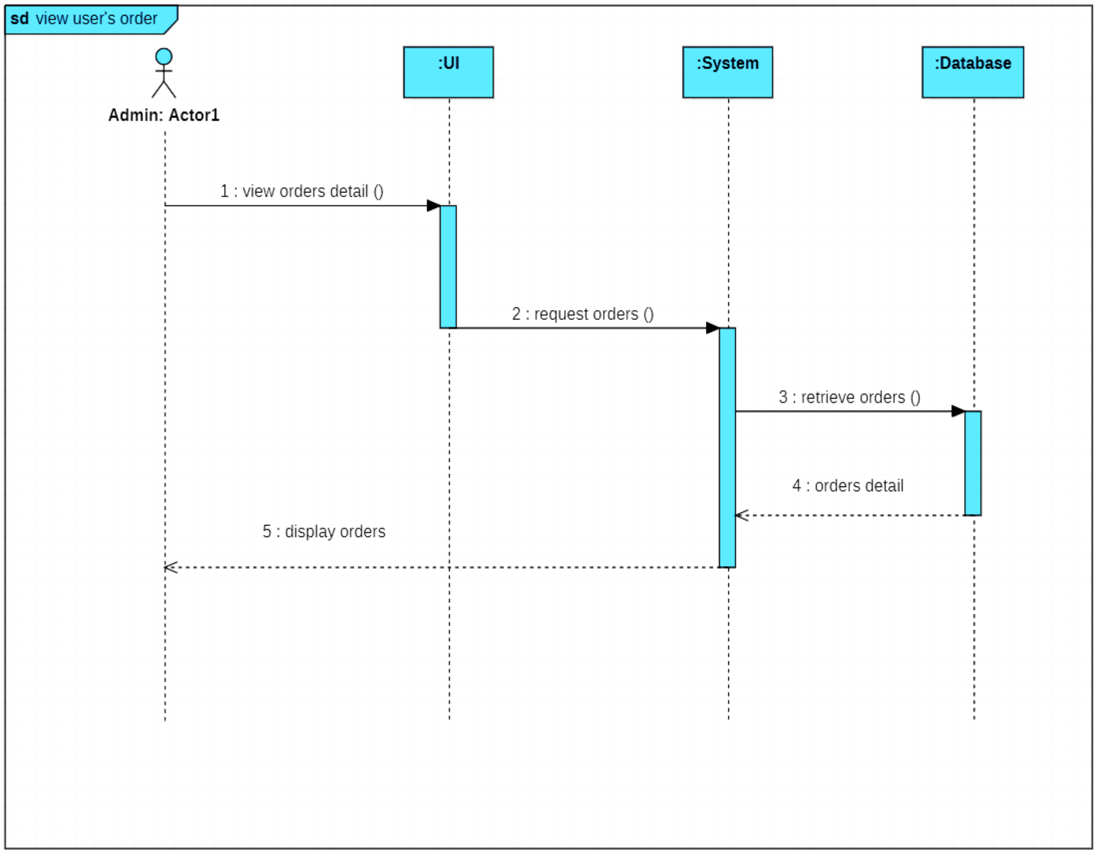
Add To Cart:


sd Add to Cart
user : Actor1
:UI
:System
:Database
1 : select a product
2 : Add product to cart ( )
3 : update cart ( )
4 : Confirmation of update
5 : Confirmation response
6 : Display confirmation message

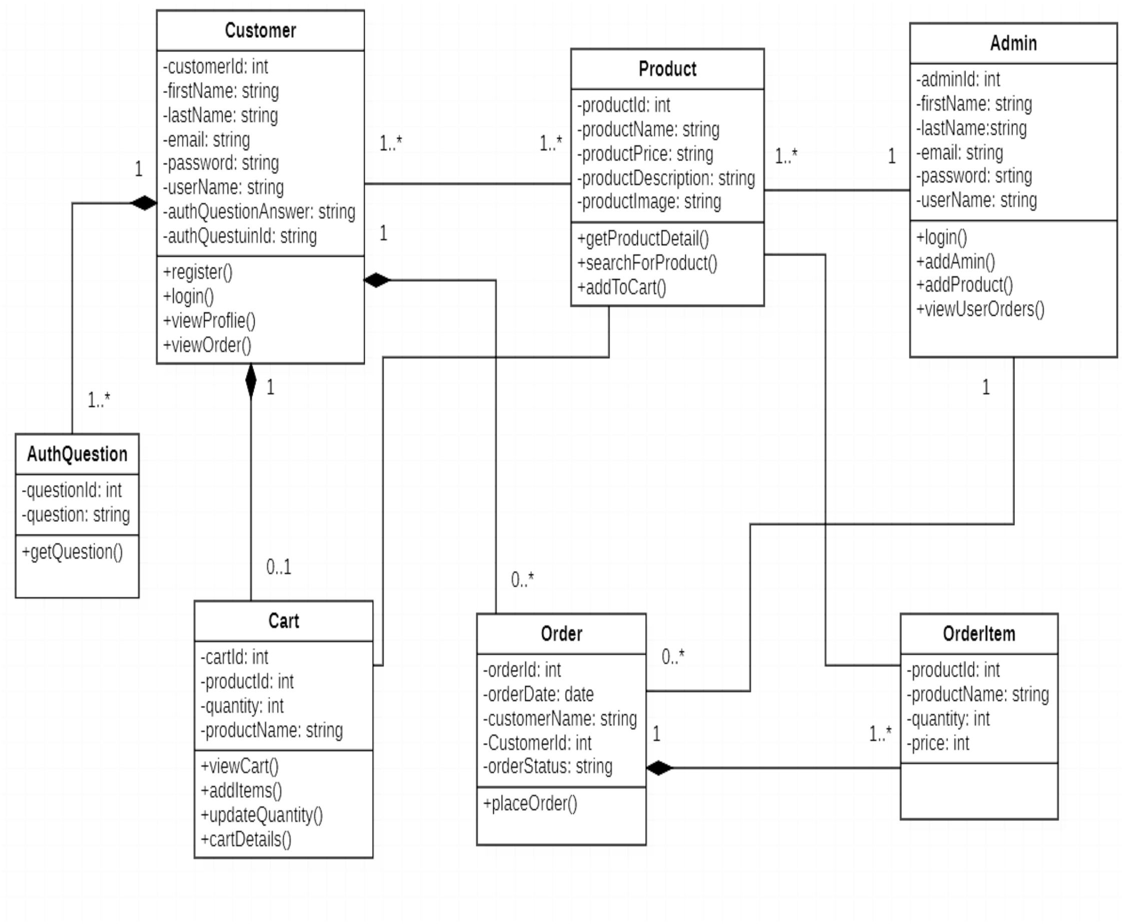
Add Product:


sd Add Product
Admin : Actor1
:UI
:System
:Database
1 : enter information of product
2 : Add Product ()
3 : check product is not exist ()
alt
4 : true
[if product is not exist before]
5 : record the information ()
6 : Add successfully
7 : product added
[else]
8 : false
9 : error message

View Order’s User:

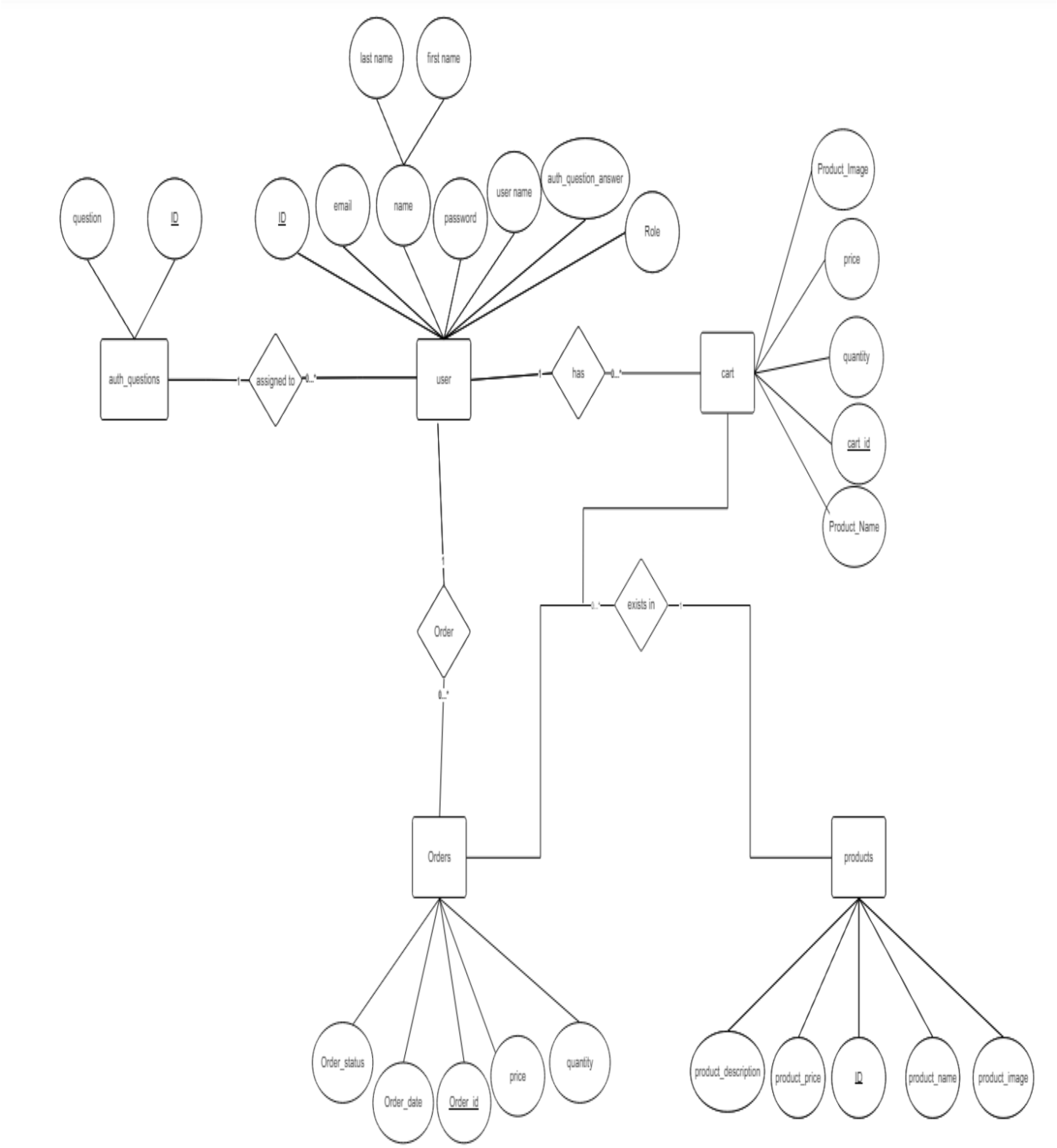


Class Diagram

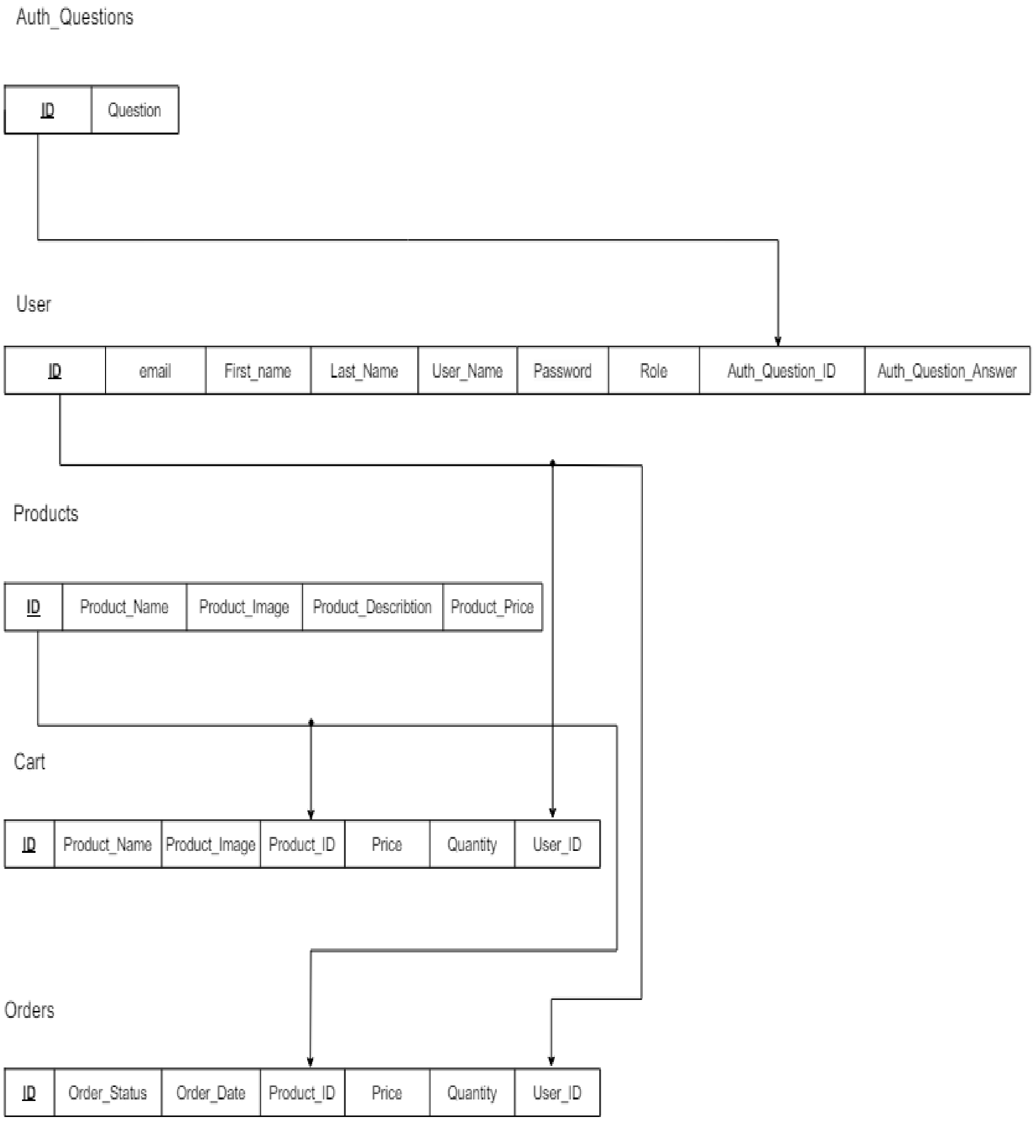


ERD

Conceptual ERD:



Physical ERD:



Auth_Questions

ID	Question

User

ID	email	First_name	Last_Name	User_Name	Password	Role	Auth_Question_ID	Auth_Question_Answer

Products

ID	Product_Name	Product_Image	Product_Describtion	Product_Price

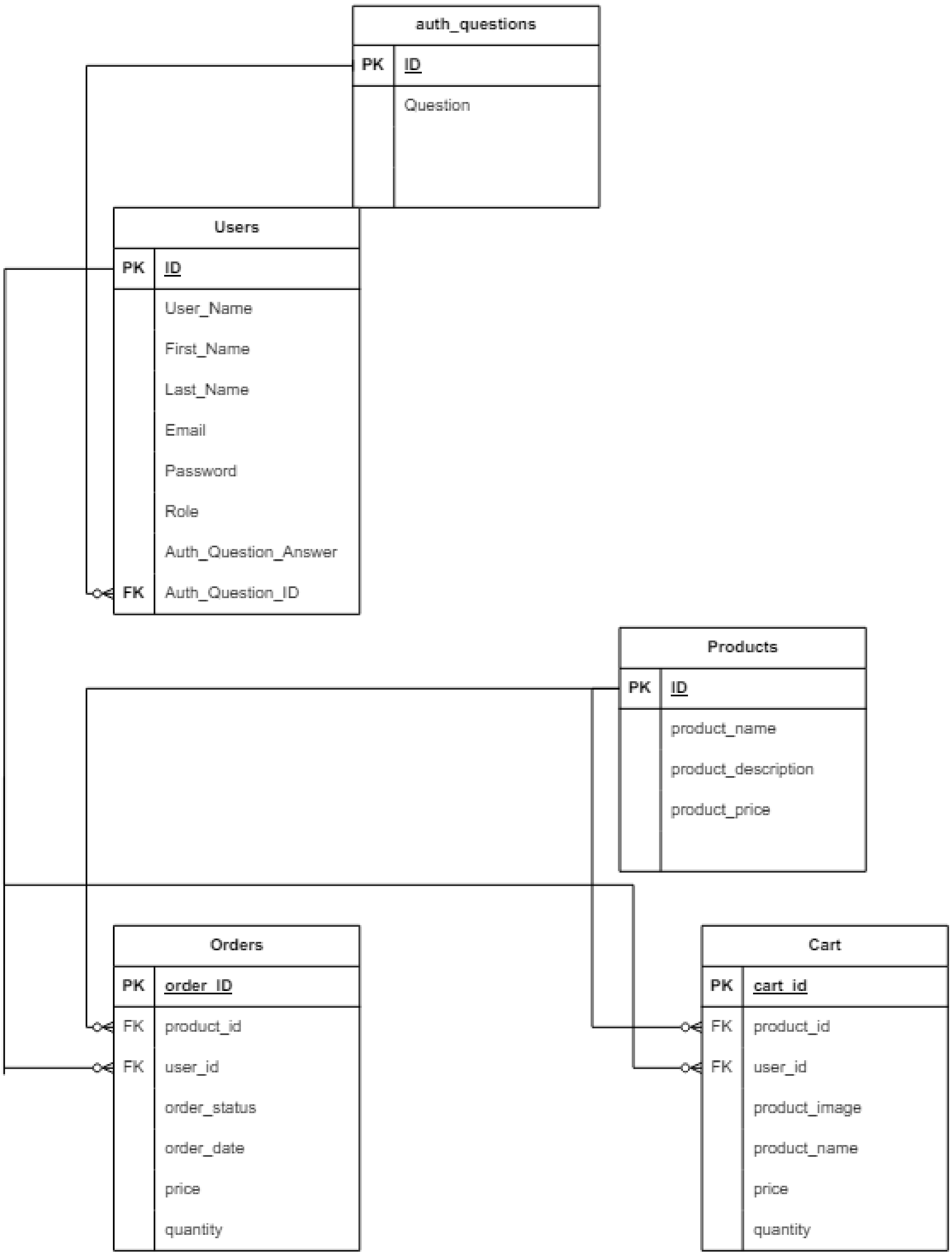
Cart

ID	Product_Name	Product_Image	Product_ID	Price	Quantity	User_ID

Orders

ID	Order_Status	Order_Date	Product_ID	Price	Quantity	User_ID

Database Schema:



auth_questions	
PK	ID
	Question

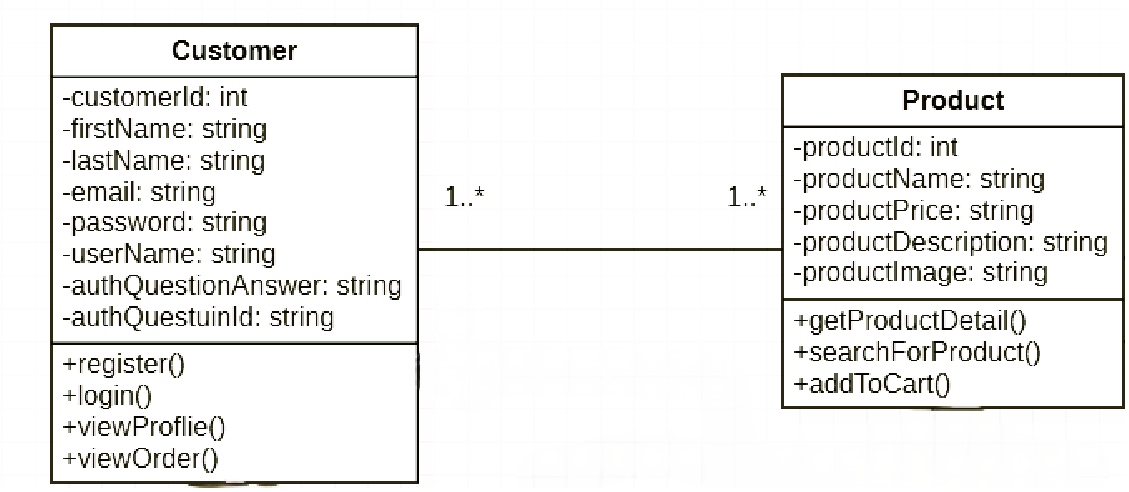
Users	
PK	ID
	User_Name
	First_Name
	Last_Name
	Email
	Password
	Role
	Auth_Question_Answer
FK	Auth_Question_ID

Products	
PK	ID
	product_name
	product_description
	product_price

Orders	
PK	order_ID
FK	product_id
FK	user_id
	order_status
	order_date
	price
	quantity

Cart	
PK	cart_id
FK	product_id
FK	user_id
	product_image
	product_name
	price
	quantity

OCL



1. context Customer
   invariant: self.customerId <> null.

2. context Customer
   invariant: self.firstName->size()>=3 and self.lastName->size()>=3 and self.userName->size()>=3.

3. context Customer
   invariant: self.email.matches('[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}\.com').

4. context Customer
   invariant: self.password->size()>=4 and self.password->size()<=10.

5. context Customer::login()
   pre loginpre: self.register()

6. context Customer::viewProfile()
   pre viewpre: self.login()

7. context Customer::homepage()
   post homepost: self.login()

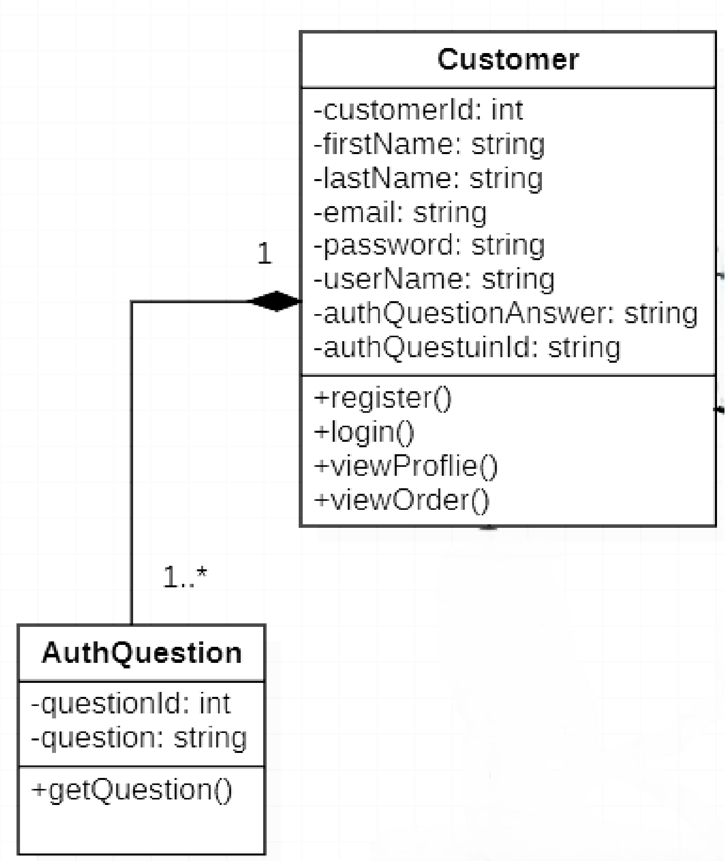
8. context Customer
   invariant:
   self.allInstances()->forAll(p | p.getProductDetail->notEmpty()).

9. context Product
   invariant: self.productId <> null.

10. context Product
    invariant: self.productPrice >=0.

11. context Product
    invariant: self.productName <> null and not self.productName.isEmpty() and self.productDescription <> null and not self.productDescription.isEmpty().

12. context Product ::getProductDetail()
    pre: Customer.login().



1. context AuthQuestion
   invariant: self.questionId <> null and self.questionId > 0.

2. context AuthQuestion
   invariant: self.question <> null and not self.question.isEmpty()
   and self.question ->size()>=3 and self.question ->size()<=20 .

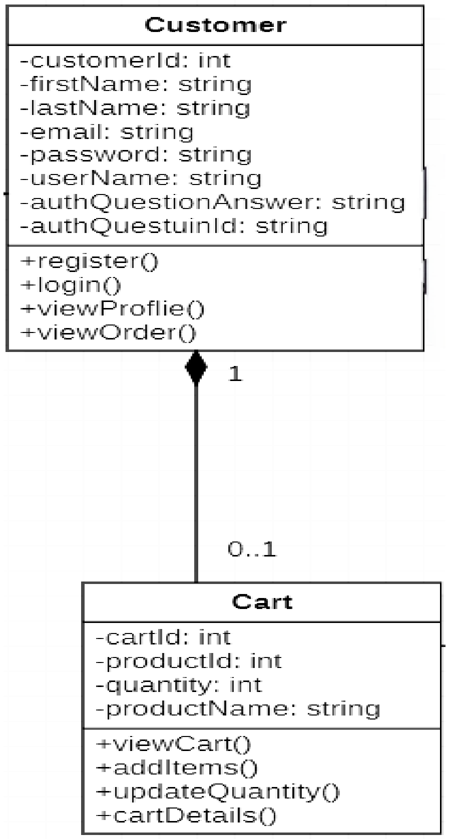
3. context Customer::register()
   pre: self.authQuestionAnswer->notEmpty()

4. context Customer
   invariant: self. authQuestionAnswer <> null and
   self.authQuestionId<>null.

5. context Customer::getAuthQuestion(customerId: int) : AuthQuestion

   pre: self. authQuestionAnswers->exists(q | q.customerId=customerId)

   post: result <> null.



1. context Cart
   invariant: self.cartId <> null.

2. context Cart
   Invariant:self.quantity >=0.

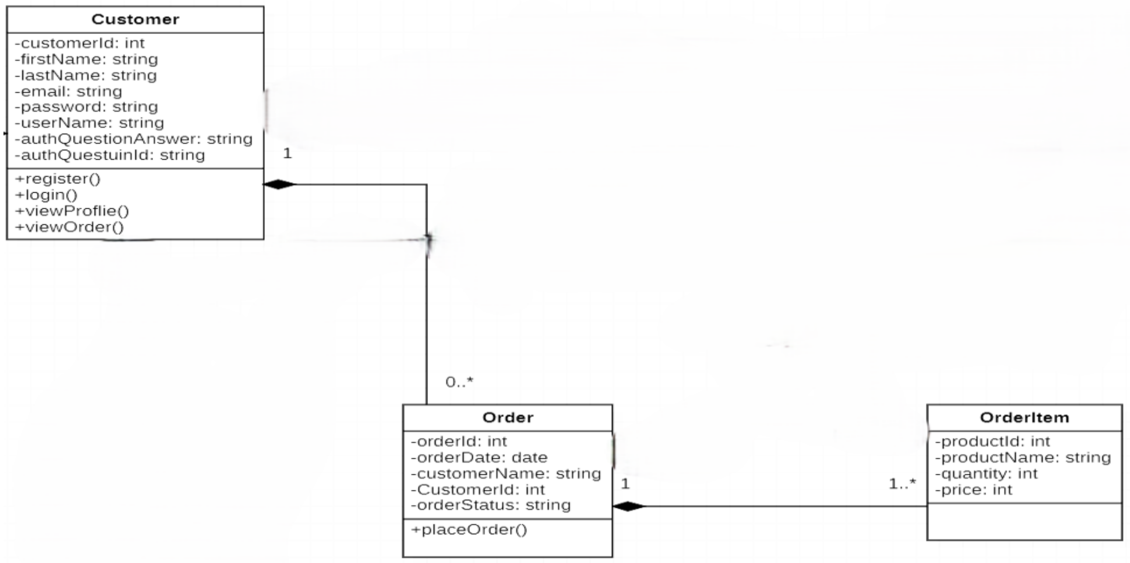
3. context Customer
   invariant: self.allInstances()->forAll(c | c. activeCart <> null and c.carts->size() = 1).

4. context Customer
   invariant: shoppingCarts->exists(cart | cart.productName->notEmpty()).

5. context Customer
   invariant: shoppingCarts->forAll(cart | cart.updateQuantity()).

6. context Customer
   invariant:
   shoppingCarts()->forAll(cart | cart.viewCart()->notEmpty()).

7. context Cart::addItems()
   pre: self.cartId <> null



1. context Customer
   invariant:
   orders->forAll(order | order.orderItems.productName->notEmpty()).

2. context Customer
   inv: orders->forAll(order | order.customer = self)

3. context Order
   invariant: self.orderId <> null.

4. contextOrder
   invariant: self.orderDate <> null.

5. context Order
   invariant: self.orderDate <= Date.today()

6. context Order
   invariant:
   self.orderStatus <> null and (self.orderStatus = 'pending' or
   self. orderStatus = 'shipped' or self. orderStatus = 'delivered').

7. context Order
   invariant:
   self.allInstances()->forAll(order | order.OrderStatus <> 'checkout'
   implies (order.customerId = null and order.customerName = null).

8. context Order::placeOrder()
   pre: orderItems.productName->notEmpty().