# COEN 6312: E-Commerce System

# Deliverable 4

**Submitted To:**

Prof. Abdelwahab Hamou-Lhadj

**Submitted By:**

**Group 10**

Hephzibah Pocharam - 40127128

Nikhil Verma - 40160264

Chirag Jhamb - 40169876

Divyaa Mahalakshmi Guruswamy- 40167923

Jayapriya Muthuramasamy - 40184587

Abdul Rahman Koleilat – 40086025

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# 1. PROBLEM DESCRIPTION

The E–Commerce system is a website which provides a platform to the customers for buying their required products from retailers. The potential users of the E-commerce system are the actors for this system which include the customers, retailers, and the administrator.

The main functions of the customers are to purchase the products from the E-commerce platform. The process of the purchase for the customer is followed which includes adding items in the cart, proceeding to the payment and tracking the delivery. The customer can also rate and provide feedback on the purchased product. If the customer wants to cancel the order due to the delay in delivery or due to product dissatisfaction, a refund is initiated.

While signing in, if the user is identified as a retailer, the privileges are given for adding, modifying or deleting the products in the website. These are the main features of the retailers. Retailers are also responsible for shipping the purchased product and refunding the amount, if requested by the customers.

There is also an administrator who manages the overall system, which includes proper functioning of the website with assured security and ensuring the privacy of the users. The data of the customer is shared only to the required seller. The administrator also categorizes the products provided by the retailer.

# 2. DOMAIN ANALYSIS

## i) REQUIREMENTS

| NO. | FUNCTIONAL REQUIREMENTS |
|---|---|
| 1. | The system should allow users to register by providing their details. |
| 2. | The system should allow users to login to their account using their username and password. |
| 3. | The system should allow users to view and edit their profile. |
| 4. | The system should allow customers to view and browse all the products. |
| 5. | The system should allow the customer to add products to the shopping cart. |

| | |
|---|---|
| 6. | The system should allow the customer to pay for the purchased product. |
| 7. | The system should allow the user to view the delivery details. |
| 8. | The system should allow the customer to track their order and know the status (shipped, in transit, delivered) |
| 9. | The system should allow the customer to rate a purchased product and add feedback. |
| 10. | The system should allow the customer to cancel the order. |
| 11. | The system should allow the customer to request a refund. (due to delay in delivery or dissatisfaction in the product). |
| 12. | The system should allow the retailer to add a new product. |
| 13. | The system should allow the retailer to remove an existing product. |
| 14. | The system should allow the retailer to edit the product details. |
| 15. | The system should allow the retailer to view the customer's address for shipping the purchased product. |
| 16. | The system should allow the retailer to refund the customer. |
| 17. | The system should allow the retailer to receive payment. |
| 18. | The system should allow the administrator to categorize the products provided by the retailers. |

*Table 1 Functional Requirements*

| NO. | NON-FUNCTIONAL REQUIREMENTS |
|---|---|
| 1. | The user's information should not be associated with third party services and their information should be protected. |
| 2. | The website should be available all the time. |
| 3. | The website should be stable and well-maintained. |

*Table 2 Non-Functional Requirements*

## ii) USE CASE DIAGRAM

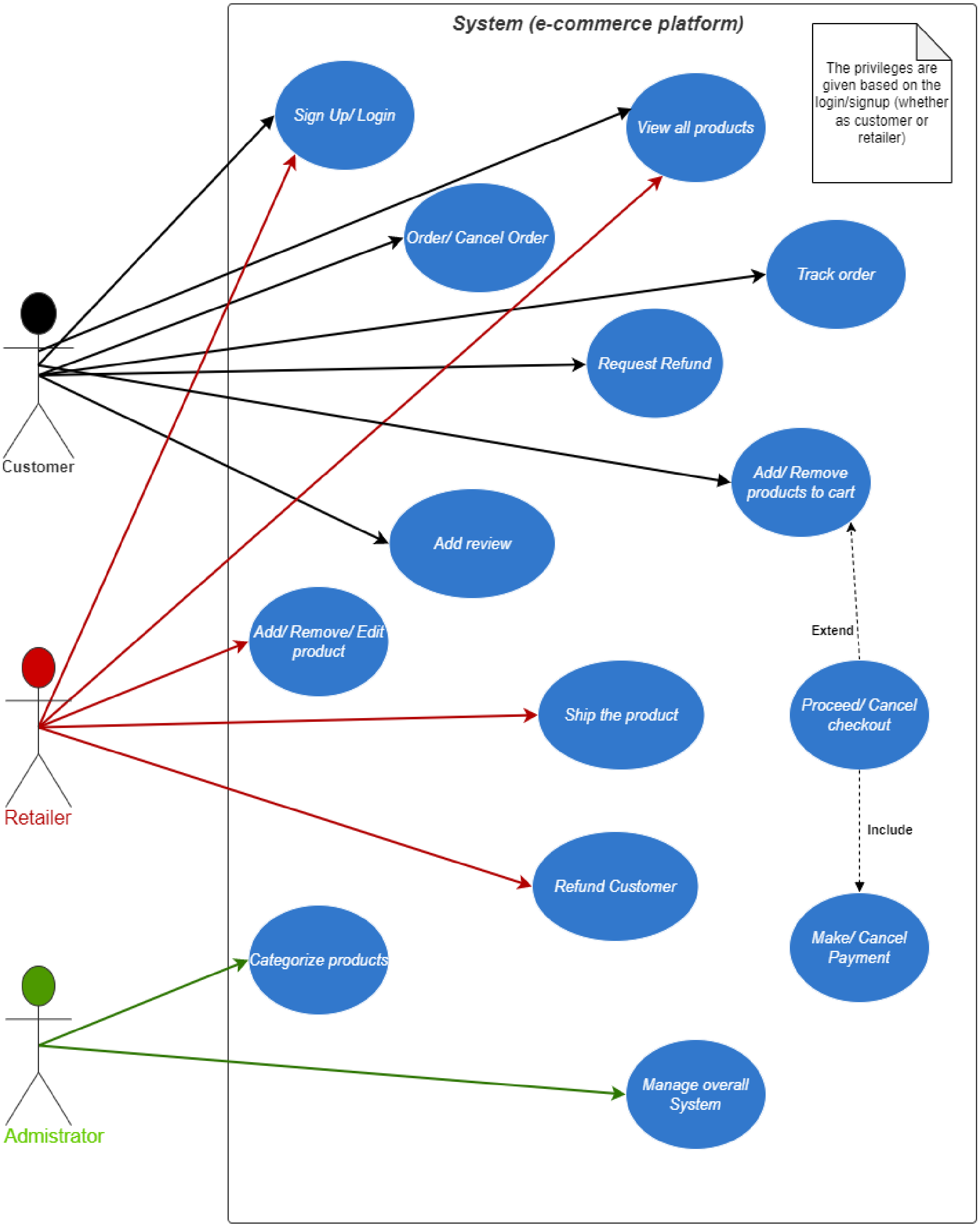


*Figure 1 Use case Diagram of the E-commerce website*

In the Use Case Diagram, there are three main actors which are as follows:

- Customer
- Retailer
- Administrator

**Customer:** The customer performs the following functions -

A) **Sign Up/ Login:** The customer can sign up by giving in their details and create a profile if he is visiting the website for the first time. If there exists a case where the user tries to sign up and create a profile using an email which is already registered. They will not be able to do it and a prompt will be given that the user already exists for that given email.  If the User has already created a profile, then he/she can login to the website by providing the credentials (username and password) used during registration.

B) **View All Products:** The customers can view all the products which are available on the website provided by the retailers.

C) **Add/ Remove Products to Cart:** The customers have an option to either add or remove the products in the cart. They can further proceed to checkout and make payment for the purchased items.

D) **Proceed/ Cancel checkout:** The products added to the cart can be finalized by proceeding to the checkout. The customer also has the option to cancel the purchase in this stage.

E) **Make/ Cancel Payment:** The payment for the purchased product can be made in this case. The customer still has an option to withdraw from the payment to cancel his purchase.

F) **Order/ Cancel Order:** Once the payment is successful the order is created. On the other hand, if the customer wishes to cancel the order due to delay in the delivery the “cancel order option” is provided.

G) **Track order:** Once the order is created the customer will be able to track his order. This will provide the status such as dispatched, in transit or delivered.

H) **Request Refund:** If the customer is not satisfied with the product, they can request for a refund. If the product is delayed the customer can also cancel the order and request for a refund.

I) **Add review:** The customer can add a rating and feedback for the product purchased.

2) **Retailer:** The Retailer performs the following functions –

A) **Sign Up/ Login:** The retailer can sign up by giving in their details and create a profile if he is visiting the website for the first time. By signing in as a retailer the privileges are given for adding or modifying the products. If the retailer has already created a profile, then he/she can login to the website by providing the credentials (username and password) used during registration.

B) **View all Products**: The retailer can also view all the products that are being sold on the website and the products which he has added for sale.

C) **Add/ Remove or Edit products**: The retailer can add a new product and the retailer can also edit / remove the existing products. The retailer can edit the details of the product like its price, and count.

D) **Ship the product:** The retailer is responsible for the shipment of the product to the address provided by the customer. The product is dispatched once the order is created from the customer’s end.

E) **Refund customer**: If the customer is not satisfied with the product or the product delivery gets detailed, the customer can request for a refund, then it is the retailers' responsibility to refund the customer and he can do it using this feature.

**3) Administrator:** The Administrator performs the following functions-

A) **Categorize the products:** The administrator can categorize the products added by the retailers.

B) **Manages the Overall System:** The administrator also manages the overall system and makes sure that the website does not crash.

# 3. DESIGN

## i) CLASS DIAGRAM

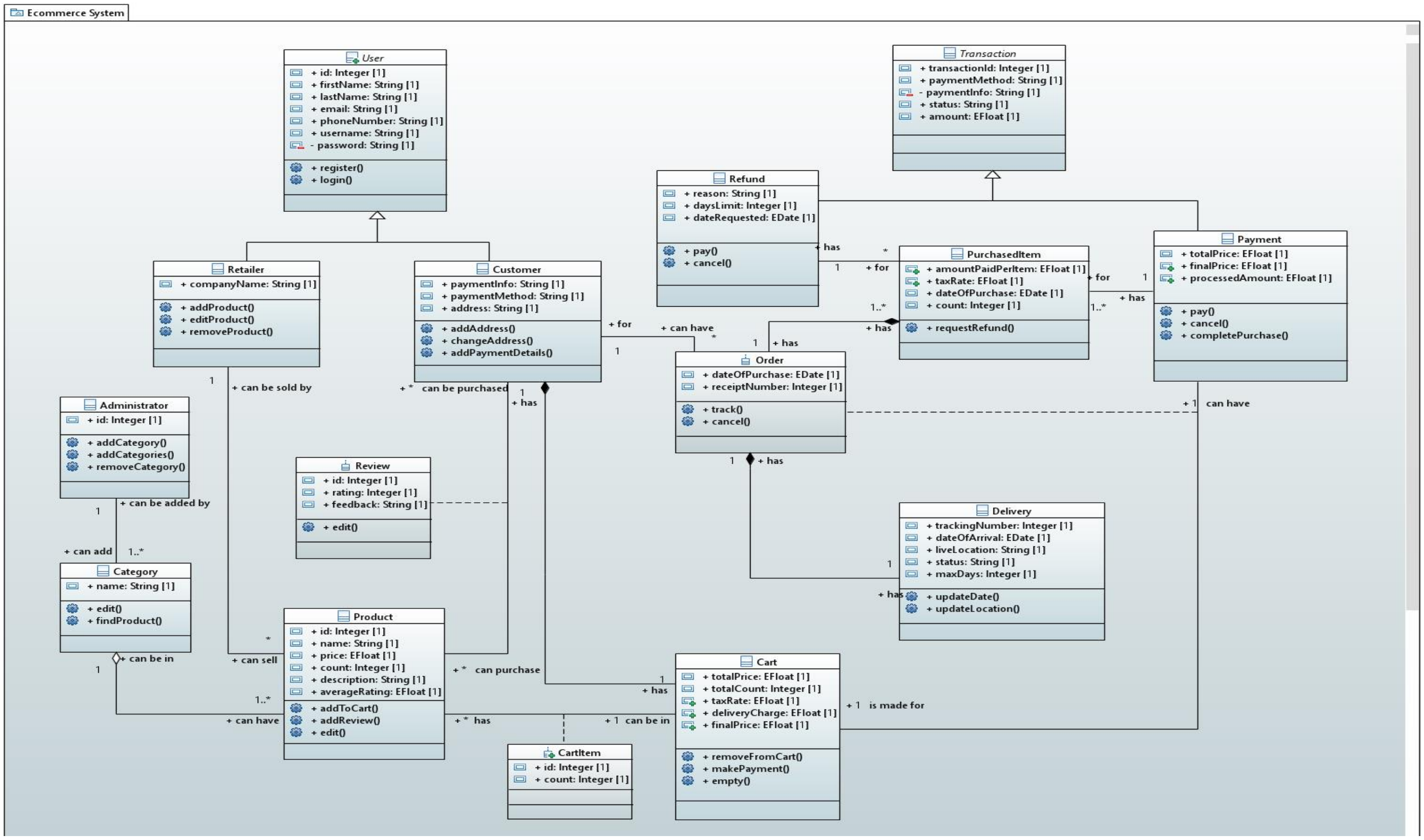


*Figure* 2 *Class Diagram*

The class diagram consists of fifteen classes:

A) The Retailer class and the Customer class is inherited from the **user class** (abstract class). The User class has the following attributes: id, firstName, lastName, email, phoneNumber, username, and password which is private. It has two functions: register() and login().

B) The **Retailer class** has companyName as an attribute and functions such as addProduct(), editProduct() and removeProduct(). It inherits the User class and is associated with the Product class whereby the retailer can sell zero or more products.

C) The **Customer class** has address paymentInfo, paymentMethod, and address as attributes. It has functions such as addAddress(), changeAddress(), and addPaymentDetails(). It inherits the User class and is associated with the Product class and the Cart class. It also has an association class which is the Review class. The customer can purchase zero or many products and has one cart.

D) The **Review class** is an association class that has three attributes: id, rating, and feedback. It has an edit() function. The customer can rate the purchased product between 0 and 5 after their order arrives.

E) The **Product class** has the following attributes: id, name, price, count, description, averageRating. It has the following functions namely addToCart(), addReview(), and edit(). It is associated with four classes: Retailer, Customer, Category and Cart. The product can be sold by one retailer, can be purchased by zero or more customers, can be in one category, and can be in one cart. The product can only be added, edited, deleted, and displayed by its corresponding retailer. The product class is aggregation class of Category which implies that the product class can also exist without the Category class.

F) The **Cart class** has five attributes: totalCount, deliveryCharge, totalPrice, taxRate, and finalPrice. It has three functions: removeFromCart(), makePayment(), and empty(). It is associated with three classes: Product, Customer, and Payment. The cart can have zero or more products, has one customer, and one payment. Cart class is a composition of Customer class.

G) The **CartItem Class** has two attributes namely cart id and count. This class serves as the association class between the product and the cart class.

H) The **Transaction Class** has attributes namely, transactionId, paymentMethod, paymentInfo, status, and amount. This class serves as an abstract class for payment and refund.

I) The **Payment class** has the following attributes: totalPrice, finalPrice, and processedAmount. The functions include pay(), cancel(), and completePurchase(). Payment is made for the cart which has the items purchased by the customer. It is inherited form the transaction Class.

J) The **PurchasedItem Class** has attributes such as amountPaidPerItem, taxRate, dateOfPurchase and count. It has requestRefund() function. This class is connected to Payment and Refund class.

K) The **Order class** is an association class between Cart and Payment. It has the following attributes: dateOfPurchase, and receiptNumber. It has the following functions: track() and cancel(). It is associated with thress classes: Customer, purchasedItem and Delivery. Order has one delivery and can be associated with one customer and is in composition with the purchasedItem class.

L) The **Delivery class** has the following attributes: trackingNumber, dateOfArrival, liveLocation, maxDays and status. It has two functions: updateLocation() and updateDate(). It is an composition of the order Class.

M) The **Refund class** has the following attributes such as reason, daysLimit and dateRequested. It has two functions: pay() and cancel() . It is inherited form the Transaction Class.

N) The **Administrator class** has id as an attribute and addCategory(), addCategories and removeCategory() as functions. It is associated with the Category class. Administrator can add one or more categories.

O) The **Category class** has name as an attribute, name, and functions, edit() and findProduct((). It is associated with two classes: Administrator and Product. Category can be added by one administrator and can have one or more products.

## ii) OCL STATEMENTS

1. Users should have a unique ID, email, and username for authentication.

**Context User**

**Inv:** allInstances()->forAll(u1, u2:User|u1<>u2 implies u1.id<>u2.id AND u1.email<>u2.email AND u1.username<>u2.username)

2. The retailer cannot add the same product again to avoid redundancy.

Context Retailer::addProduct(P:product)

**pre:** self.Product->excludes(P)

**post:** self.Product->includes(P)

3. The customer can initiate the refund process for the product only after receiving it or if the delivery has not arrived on time. The customer can request a refund within 30 days of the product purchase after arrival.

**Context purchased item::requestRefund(r :Refund)**

**pre:** self.Refund -> excludes(r)

(self.order.delivery.status=”arrived” AND (Day(today) – Day(self.dateOfPurchase)) < 30) OR ( (Day(today) - Day(self.Order.Delivery.dateOfArrival)) > 0)

**post:** self.Refund -> includes(r) AND r.status = “Requested”

4. For the customer to give review to the product, he/she should be a verified purchaser.

**Context Customer**

**Inv:** self.product -> includesAll(self.Order.PurchasedItem.Payment.Cart.Product)

5. Two orders cannot have the same receipt number.

**Context Order**

**Inv:** allInstances()->forAll(o1,o2 :Order|o1<>o2 implies o1.receiptNumber<>o2.receiptNumber)

6. The customer will not be able to proceed to checkout if the cart is empty.

**Context Cart**

**Inv:** allInstances() -> forAll(c: Cart | c.state = "Checkout" implies c.CartItem -> isnotEmpty())

7. An item can only be purchased if the payment was successful

**Context Payment**

**Inv:** self.purchaseditem -> isnotEmpty() implies self.state = “Success”

8. Only the customer user can add the product to cart, the cart item count should be equal or less than the product count, and the customer can only add a maximum of 5 of the same product. The total price is then updated.

**Context Product::addToCart(c :CartItem, countTobeAdded :int)**

**pre:** self.User -> forAll(u :user | u.oclIsTypeOf(Customer)) AND self.CartItem -> forAll(c :CartItem, p :Product | c.id = p.id implies p.count >= countToBeAdded AND (countToBeAdded + c.count) <=5)

**post:** self.CartItem -> forAll(c :CartItem, p :Product | c.id = p.id implies p.count >= c.count AND c.count <=5)

totalPrice = totalPrice@pre + c.count * c.Product.price

9. After the customer orders a product, they can add a review. The average rating of the product will be updated.

**Context Product::addReview(r :Review)**

**pre:** self.User -> forAll(u :user | u.oclIsTypeOf(Customer)) AND self.Review -> excludes(r) AND

allInstances() -> includesAll(self.Customer.Order.PurchasedItem.Payment.Cart.Product)

**post:** self.Review -> includes(r) AND

averageRating = (self.Review-> select(r.id = p.id) -> collect(rating) -> sum()) / self.Review-> select(r.id = p.id) -> size()

10. After the customer makes a payment and if payment is successful, their cart will be empty, and the count of the purchased items will be subtracted from the count of the product.

**Context Payment**

**Inv:** allInstances() -> forAll(p: Payment | p.state = “Success” implies self.CartItems -> isEmpty()) AND self.Cart.Product -> forAll(c :CartItem, p :Product | c.id = p.id implies p.count = p.count@pre - c.count)

11. The delivery should be cancelled when it does not arrive on time.

**Context Delivery**

**inv:** self.state = “Cancelled” implies ( (Day(today) - Day(dateOfArrival)) > 0 AND self.state <> “Arrived”)

12. Maximum of 50 items can be added to the cart.

**Context Cart**

**inv:** self.CartItems -> collect(count) -> sum() <= 50

13. A review can only have a rating between 0 and 5.

**Context Review**

inv: self.rating >= 0 AND self.rating <= 5

14. When a cart item is removed from cart, its price will be subtracted from the cart’s total price.

**Context Cart::removeFromCart(c :CartItem, countToBeRemoved :int)**

**pre:** self.CartItem -> includes(c)

**post:** c.count = c.count@pre - countToBeRemoved

totalPrice = totalPrice@pre - c.Product.price * countToBeRemoved

15. If the total price in the cart is less than $50, then there is a $5 delivery charge.

**Context Cart**

**inv: self..deliveryCharge = if(totalPrice < 50) then 5**

else 0

endif

## iii) STATE DIAGRAMS

### Product Class State Diagram

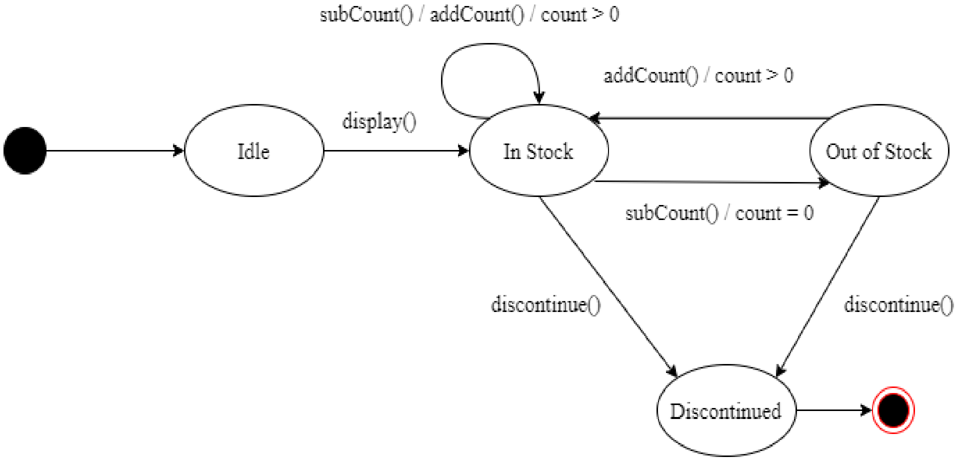


*Figure 3 Product Class State Diagram*

### Delivery Class State Diagram

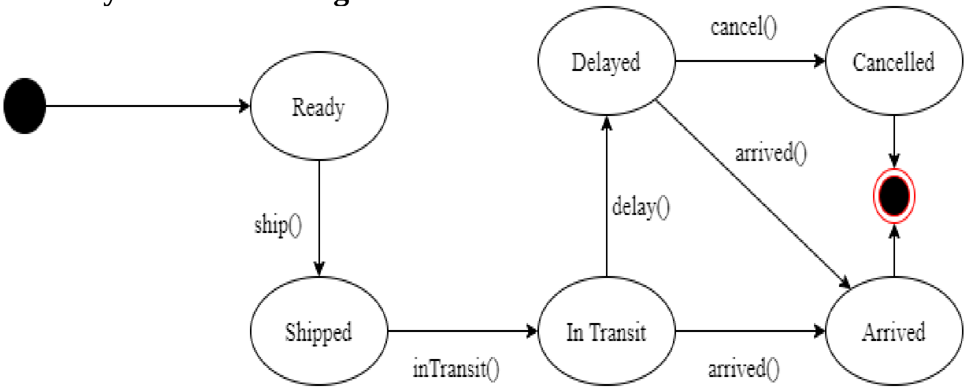


*Figure 4 Delivery Class State Diagram*

## Cart Class State Diagram

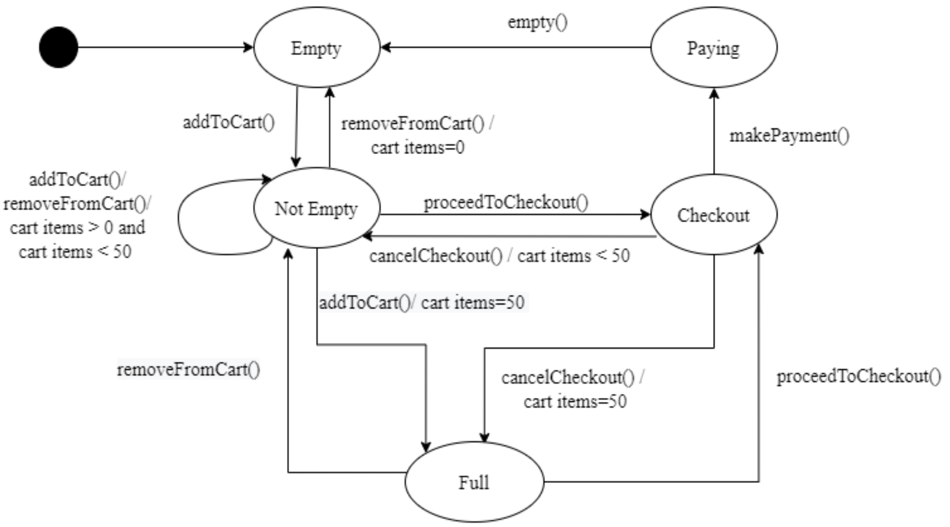

Empty
empty()
Paying
addToCart()
removeFromCart() / cart items=0
makePayment()
addToCart()/ removeFromCart()/ cart items > 0 and cart items < 50
Not Empty
proceedToCheckout()
Checkout
cancelCheckout() / cart items < 50
addToCart()/ cart items=50
removeFromCart()
cancelCheckout() / cart items=50
proceedToCheckout()
Full

*Figure 5 Cart Class State Diagram*

## Payment Class State Diagram

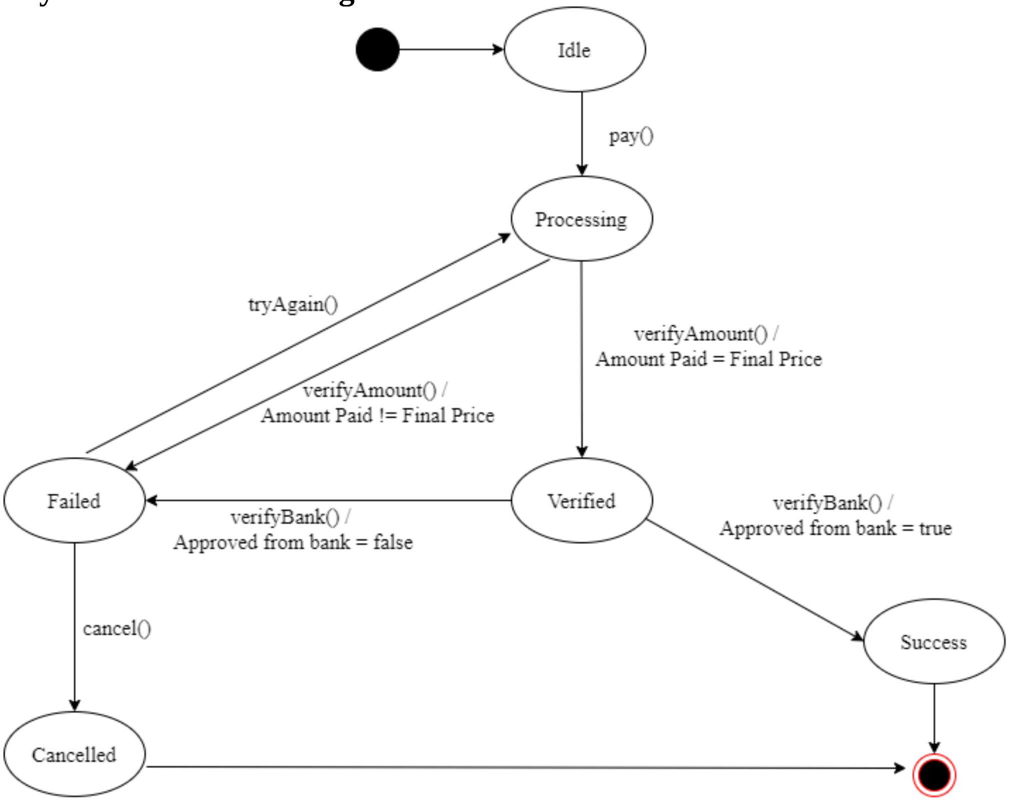
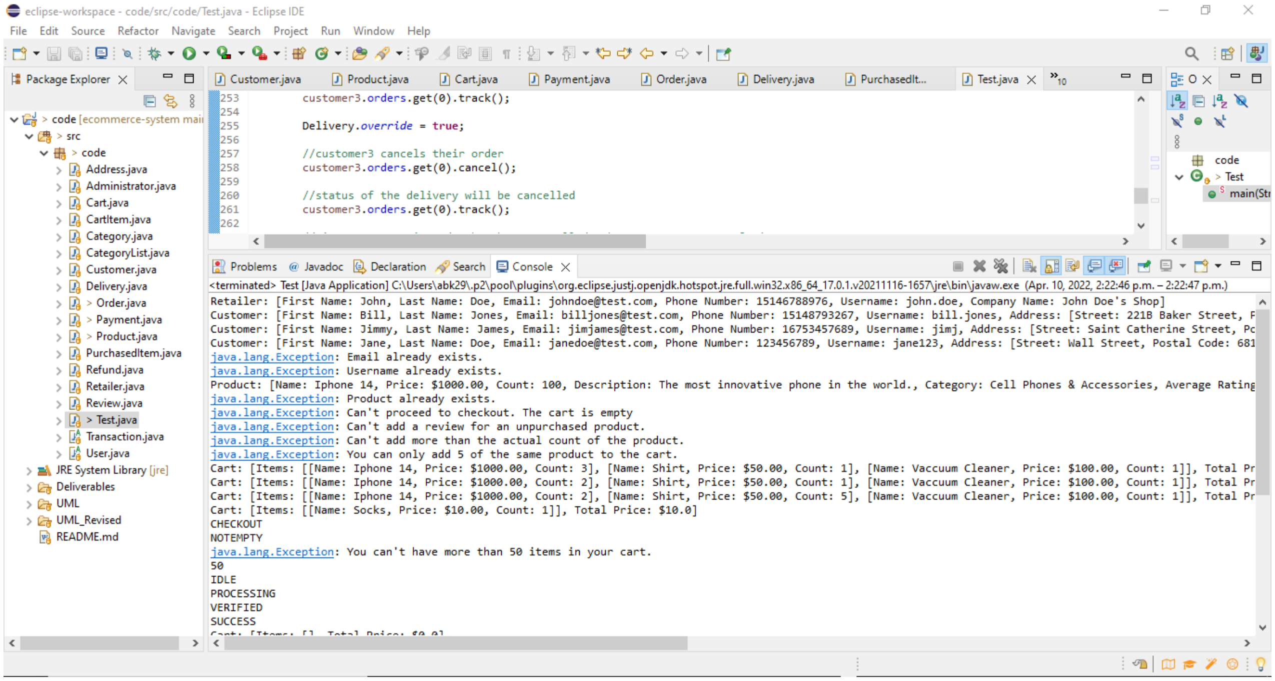

Idle
pay()
Processing
tryAgain()
verifyAmount() / Amount Paid = Final Price
verifyAmount() / Amount Paid != Final Price
Failed
Verified
verifyBank() / Approved from bank = false
verifyBank() / Approved from bank = true
cancel()
Success
Cancelled

*Figure 6 Payment Class State Diagram*

# 4. TESTING

We start the driver by creating an administrator, a retailer, and three customers. The retail and the customers are created using the register() method. Then, we attempted to register a user with an email address that already exists, and an exception is expected to be thrown. Then, we attempted to register a user with a username that already exists, and an exception is expected to be thrown. After that, the categories are added as well as the products of the retailer. We attempted to add a duplicate product and an exception is expected to be thrown. After that, customer 1 attempts to proceed to checkout with an empty cart and an exception is thrown. Customer 1 also attempts to add a review to a product that is not purchased by them, and an exception is thrown. After that, customers 1, 2, and 3 add several products to their carts. After that, customer 1 attempts to add to their cart more than the actual count of a certain product, and an exception is thrown. Then, customer 1 attempts to add to their cart more than 5 items of the same product, which is not permissible, and an exception is thrown. After that, customer 1, removes an item from their cart. Then, all the customers proceed to checkout. Customer 3 cancels their checkout and empties their cart. Customer 3 then tries to add more than 50 items to their cart, which is not permissible, and an exception is thrown. Then, all the customers make a payment, verify the amount, then verify the payments with the bank, and then the purchase is completed. After that, the customers track their orders where it should show the details of the delivery of their order. Customer 1 attempts to cancel their order before the date of arrival, and an exception is thrown. Then, it was made permissible for the customer to cancel their orders and therefore, customer 3 cancelled their order and can request a refund for one of their order's purchased items. The retailers can view the refund request and pay the refund. Customer 1 attempts to add a review with a rating that is not between 0 and 5, and an exception is thrown. Customers 1 and 2 add reviews to the product "Iphone 14". Then, customer 1 attempts to add another review to the same product, and an exception is thrown. Customer 1 attempts to request a refund for a purchased item of an order that hasn't arrived yet, and an exception is thrown. The product "Iphone 14" is printed to show its updated average rating and count as well as the customer reviews.



```
        customer3.orders.get(0).track();

        Delivery.override = true;

        //customer3 cancels their order
        customer3.orders.get(0).cancel();

        //status of the delivery will be cancelled
        customer3.orders.get(0).track();

```

```
<terminated> Test [Java Application] C:\Users\abk29\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (Apr. 10, 2022, 2:22:46 p.m. – 2:22:47 p.m.)
Retailer: [First Name: John, Last Name: Doe, Email: johndoe@test.com, Phone Number: 15146788976, Username: john.doe, Company Name: John Doe's Shop]
Customer: [First Name: Bill, Last Name: Jones, Email: billjones@test.com, Phone Number: 15148793267, Username: bill.jones, Address: [Street: 221B Baker Street, F
Customer: [First Name: Jimmy, Last Name: James, Email: jimjames@test.com, Phone Number: 16753457689, Username: jimj, Address: [Street: Saint Catherine Street, Pc
Customer: [First Name: Jane, Last Name: Doe, Email: janedoe@test.com, Phone Number: 123456789, Username: jane123, Address: [Street: Wall Street, Postal Code: 681
java.lang.Exception: Email already exists.
java.lang.Exception: Username already exists.
Product: [Name: Iphone 14, Price: $1000.00, Count: 100, Description: The most innovative phone in the world., Category: Cell Phones & Accessories, Average Rating
java.lang.Exception: Product already exists.
java.lang.Exception: Can't proceed to checkout. The cart is empty
java.lang.Exception: Can't add a review for an unpurchased product.
java.lang.Exception: Can't add more than the actual count of the product.
java.lang.Exception: You can only add 5 of the same product to the cart.
Cart: [Items: [[Name: Iphone 14, Price: $1000.00, Count: 3], [Name: Shirt, Price: $50.00, Count: 1], [Name: Vaccuum Cleaner, Price: $100.00, Count: 1]], Total Pr
Cart: [Items: [[Name: Iphone 14, Price: $1000.00, Count: 2], [Name: Shirt, Price: $50.00, Count: 1], [Name: Vaccuum Cleaner, Price: $100.00, Count: 1]], Total Pr
Cart: [Items: [[Name: Iphone 14, Price: $1000.00, Count: 2], [Name: Shirt, Price: $50.00, Count: 5], [Name: Vaccuum Cleaner, Price: $100.00, Count: 1]], Total Pr
Cart: [Items: [[Name: Socks, Price: $10.00, Count: 1]], Total Price: $10.0]
CHECKOUT
NOTEMPTY
java.lang.Exception: You can't have more than 50 items in your cart.
50
IDLE
PROCESSING
VERIFIED
SUCCESS
Cart: [Items: [], Total Price: $0.0]
```

*Figure 7 output screen*

```
Retailer: [First Name: John, Last Name: Doe, Email: johndoe@test.com, Phone Number: 15146788976, Username: john.doe, Company Name: John Doe's Shop]
Customer: [First Name: Bill, Last Name: Jones, Email: billjones@test.com, Phone Number: 15148793267, Username: bill.jones, Address: [Street: 221B Baker Street, Postal Code: 1234, City: London, Country: United Kingdom]]
Customer: [First Name: Jimmy, Last Name: James, Email: jimjames@test.com, Phone Number: 16753457689, Username: jimj, Address: [Street: Saint Catherine Street, Postal Code: H3H 289, City: Montreal, Country: Quebec]]
Customer: [First Name: Jane, Last Name: Doe, Email: janedoe@test.com, Phone Number: 123456789, Username: jane123, Address: [Street: Wall Street, Postal Code: 6812, City: New York, Country: USA]]
java.lang.Exception: Email already exists.
java.lang.Exception: Username already exists.
Product: [Name: Iphone 14, Price: $1000.00, Count: 100, Description: The most innovative phone in the world., Category: Cell Phones & Accessories, Average Rating: 0.00, Reviews: []]
java.lang.Exception: Product already exists.
java.lang.Exception: Can't proceed to checkout. The cart is empty
java.lang.Exception: Can't add a review for an unpurchased product.
java.lang.Exception: Can't add more than the actual count of the product.
java.lang.Exception: You can only add 5 of the same product to the cart.
Cart: [Items: [[Name: Iphone 14, Price: $1000.00, Count: 3], [Name: Shirt, Price: $50.00, Count: 1], [Name: Vaccuum Cleaner, Price: $100.00, Count: 1]], Total Price: $3150.0]
```

```
Cart: [Items: [[Name: Iphone 14, Price: $1000.00, Count: 2], [Name: Shirt, Price: $50.00, Count:
1], [Name: Vaccuum Cleaner, Price: $100.00, Count: 1]], Total Price: $2150.0]
Cart: [Items: [[Name: Iphone 14, Price: $1000.00, Count: 2], [Name: Shirt, Price: $50.00, Count:
5], [Name: Vaccuum Cleaner, Price: $100.00, Count: 1]], Total Price: $2350.0]
Cart: [Items: [[Name: Socks, Price: $10.00, Count: 1]], Total Price: $10.0]
CHECKOUT
NOTEMPTY
java.lang.Exception: You can't have more than 50 items in your cart.
50
IDLE
PROCESSING
VERIFIED
SUCCESS
Cart: [Items: [], Total Price: $0.0]
[Order Receipt: [Receipt Number: 1 ,Purchased Items: [[Name: Iphone 14, Price After Tax:
$1150.00, Count: 2], [Name: Shirt, Price After Tax: $57.50, Count: 1], [Name: Vaccuum Cleaner,
Price After Tax: $115.00, Count: 1]], Total Price: $0.0, Added Tax: 5.0%, Delivery Charge: $0.0,
Final Price: $2257.5, Date of Purchase: 2022-04-10T14:22:47.031686]]
Delivery: [Tracking Number: 1, Live Location: NA, Status: READY, Shipping To: Address: [Street:
221B Baker Street, Postal Code: 1234, City: London, Country: United Kingdom], Expected Date of
Arrival: 2022-04-25T14:22:47.035686300]
java.lang.Exception: Order can't be cancelled at this time.
[Order Receipt: [Receipt Number: 2 ,Purchased Items: [[Name: Iphone 14, Price After Tax:
$1150.00, Count: 2], [Name: Shirt, Price After Tax: $57.50, Count: 5], [Name: Vaccuum Cleaner,
Price After Tax: $115.00, Count: 1]], Total Price: $0.0, Added Tax: 5.0%, Delivery Charge: $0.0,
Final Price: $2467.5, Date of Purchase: 2022-04-10T14:22:47.037686900]]
Delivery: [Tracking Number: 2, Live Location: NA, Status: READY, Shipping To: Address: [Street:
Saint Catherine Street, Postal Code: H3H 289, City: Montreal, Country: Quebec], Expected Date
of Arrival: 2022-04-25T14:22:47.037686900]
[Order Receipt: [Receipt Number: 3 ,Purchased Items: [[Name: Toaster, Price After Tax: $57.50,
Count: 1], [Name: Xbox One, Price After Tax: $690.00, Count: 1]], Total Price: $0.0, Added Tax:
5.0%, Delivery Charge: $0.0, Final Price: $682.49994, Date of Purchase: 2022-04-
10T14:22:47.037686900]]
Delivery: [Tracking Number: 3, Live Location: NA, Status: READY, Shipping To: Address: [Street:
Wall Street, Postal Code: 6812, City: New York, Country: USA], Expected Date of Arrival: 2022-
04-25T14:22:47.037686900]
Your order has been cancelled.
Delivery: [Tracking Number: 3, Live Location: NA, Status: CANCELLED, Shipping To: Address:
[Street: Wall Street, Postal Code: 6812, City: New York, Country: USA], Expected Date of Arrival:
2022-04-25T14:22:47.037686900]
```

```
[[Status: Requested, Item: Toaster, Count: 1, Amount: 57.5, Reason: Haven't received this order,
so it was cancelled., Date Requested: 2022-04-10T14:22:47.041688100]]
[[Status: Paid, Item: Toaster, Count: 1, Amount: 57.5, Reason: Haven't received this order, so
it was cancelled., Date Requested: 2022-04-10T14:22:47.041688100]]
java.lang.Exception: Rating should be between 0 and 5.
java.lang.Exception: Review already exists.
java.lang.Exception: You can't request a refund at this time.
Product: [Name: Iphone 14, Price: $1000.00, Count: 96, Description: The most innovative phone
in the world., Category: Cell Phones & Accessories, Average Rating: 3.25, Reviews: [[Rating:
4.50, Feedback: I really liked this phone!], [Rating: 2.00, Feedback: I didn't like this phone.]]]
```

## 5. REFERENCES

[1] Class lectures of COEN 6312 by Prof. Abdelwahab Hamou-Lhadj

[2[ Sommerville, I. (2016). Software engineering