JAVA Shopping Cart

By Rabih El Khatib, Gerrell Bones, Zachary Bundarin, Solan Degefa

Group 20

COP 4331-001

Delivery #2 Design

Contents

Introduction

Project Title: JAVA Shopping Cart

Functional Specification:

The JAVA Shopping Cart application is created to manage all processes of online shopping. The user can create a new account and log in as either a customer or a seller. After logging in, the customer can see a list of items and their cart information, which starts out empty. The user can directly select an item to add to their invoice, and a tally of the total price is kept on the invoice section. Each item has an info button that the user can click on to view details about that specific item, name, description, price, available quantity, and discount if the item has one. Additionally, items can also come in bundles which are a set of different items.

At any point, the invoice section shows the whole list of items chosen, and the customer can proceed to checkout. At the checkout window the customer can confirm the list of purchases or go back and review. The customer is the customer will be prompted to enter their credit card information. After confirming the information. The purchase is complete.

A seller can log in by changing a toggle at the login screen. The seller menu shows the current inventory. The seller can view the inventory information including internal product ID, type, quantity, invoice price, selling price and discounts for items that have ones. On the Profits tab, the seller can view costs, revenue, and profits. The seller can add a product or update an existing one through the new product option, simply specifies the product name, invoice price, sell price, and quantity. If the product already exists, it will be updated. Otherwise, a new item will be added. The seller can also add a new bundle through the bundle option and can also remove any item from the inventory.

Use Cases

List of Use Cases:

1. Customer Log in.
2. Seller Log in.
3. Create new account.
4. Customer adds item to invoice.
5. Customer removes item from invoice.
6. Customer reviews product information.
7. Customer completes payment.
8. Seller adds new item to inventory.
9. Seller adds new bundle to inventory.
10. Seller removes item from inventory.
11. Seller checks revenue, costs, and profit.

Use Cases 1: Customer Log in.

1. User clicks customer radio button.
2. User enters username and password.
3. System verifies login information.
4. System displays Customer menu..

Variation #1: Invalid login.

1. In step 2, user enters invalid username or password.
2. System displays label message “invalid login”.

Use Cases 2: Seller Log in.

1. User clicks seller radio button.
2. User enters username and password.
3. System verifies login information..
4. System displays Seller menu

Use Cases 3: Create new account.

1. User clicks “Create Account”.
2. System displays Sign up menu.
3. User enters username, password, and confirm password.
4. System verifies signup information.
5. System adds new account to users list.
6. System display login menu.

Variation #1: Account already exists.

1. In step 3, user enters existing username.
2. System displays label message “Account Already Exist”.

Variation #2: Password doesn’t match.

1. In step 3, user enters different password and password confirm.
2. System displays label message “password mismatch”.

Use Case 4: Customer adds item to invoice.

1. User carries out **Customer Log in.**
2. User selects item from list.
3. User clicks add item.
4. System adds selected item to invoice.
5. System displays invoice.

Variation #1: Out of stock.

1. In step 4, Systems display the message “item out of stock”.

Use Case 5: Customer removes item from invoice.

1. User carries out **Customer adds item to invoice.**
2. User clicks remove item.
3. System removes item from invoice.

Use Case 6: Customer reviews product information.

1. User carries out **Customer Log in.**
2. User selects item from list.
3. User clicks info button.
4. System displays item name, description, quantity, price and discount.

Use Case 7: Customer completes payment

1. User carries out **Customer adds item to invoice.**
2. User clicks Check out button.
3. System displays credit card menu.
4. User adds credit card information.
5. User clicks pay.
6. System verifies credit card information.
7. System adds revenue.
8. System displays message “Thank you for purchasing”.

Use Case 8: Seller adds new item to inventory.

1. User carries out **Seller Log in.**
2. User clicks new item.
3. System displays new item menu.
4. User adds new item details.
5. User clicks add.
6. System verifies new item details.
7. System adds new item to inventory.
8. System updates costs.

Variation #1: Item already in inventory

1. In step 7, System updates details of item in inventory.
2. System updates costs if item quantity is higher than before.

Use Case 9: Seller adds new bundle to inventory.

1. User carries out **Seller Log in.**
2. User clicks new bundle.
3. System displays new bundle menu.
4. User adds new bundle details.
5. User clicks add.
6. System verifies new bundle details.
7. System adds new bundle to inventory.

Use Case 10: Seller removes item from inventory.

1. User carries out **Seller Log in.**
2. User selects item from list.
3. User clicks remove.
4. System removes item from inventory.

Use Case 11: Seller checks revenue, costs, and profit.

1. User carries out **Seller Log in.**
2. User clicks profit.
3. System displays menu showing revenue, costs, and profit.

CRC Cards

Profit	
- Manages revenue. - Manages costs. - Manages revenue.	- Inventory

Inventory	
- Manages product, discounted item, and bundle details. - Stores new LineItems.	- LineItem

UserModel	
- Manages username and password. - Store user accounts.	

UserController	
- Manages user login and signup.	

UserUI	
- Displays user login and signup menus.	

CustomerController	
- Manages customer invoice - Manages customer payment - Shows item information	- Invoice - LineItem

CustomerUI	
- Display customer invoice - Display available inventory - Display payment menu - Display pop up messages	

SellerController	
- Manages seller option - Adds items to inventory - Removes items from inventory - Shows profit data	- Inventory - LineItem

SellerUI	
- Display seller menu - Display profit data	

Invoice	
- Manages invoice details	- InvoiceFormatter

<<Interface>> **InvoiceFormatter**	
- Manages invoice formatting	

SimpleFormatter implements **InvoiceFormatter**	
- Formats invoice to display title, items, and total price	

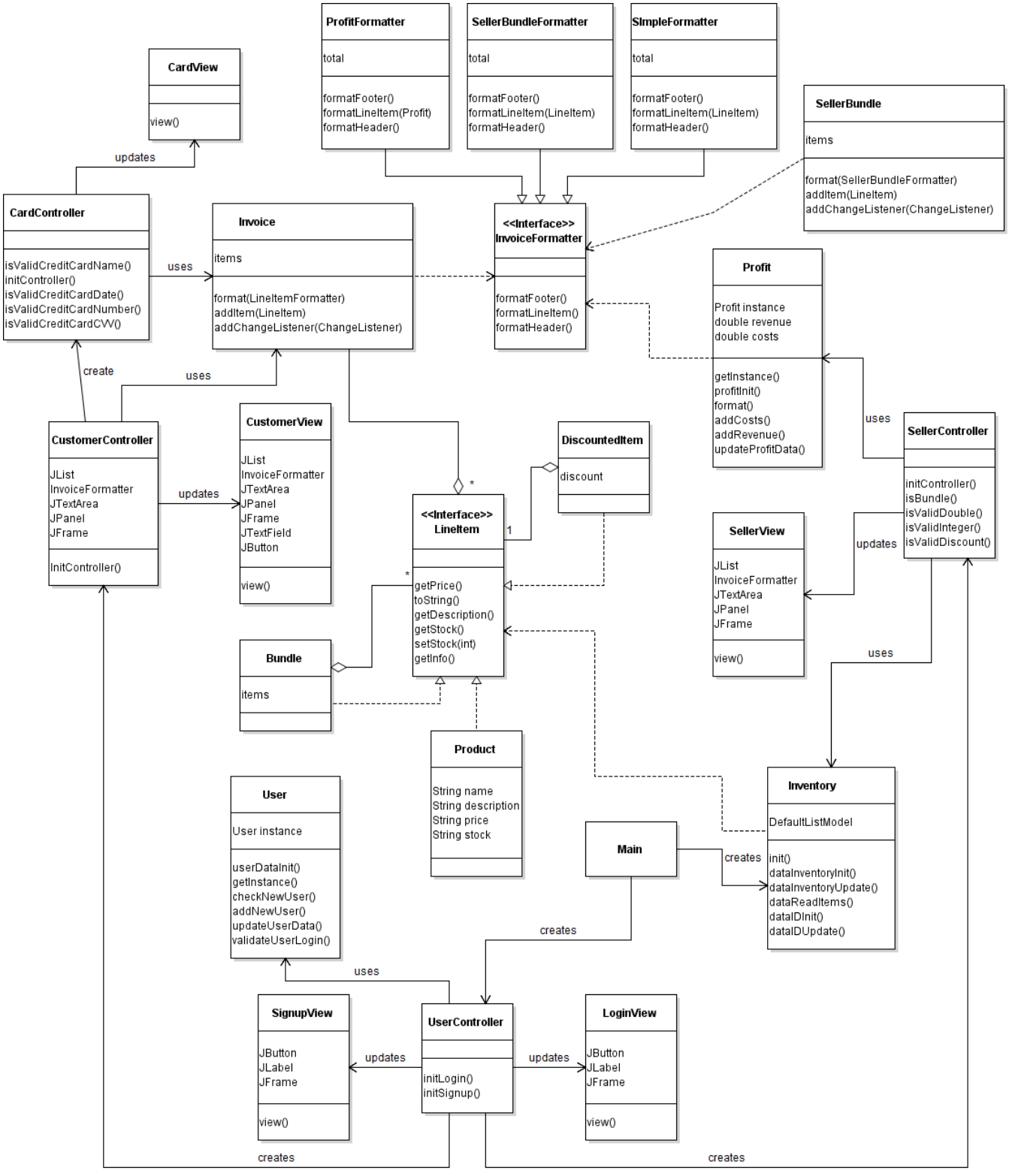
<<Interface>> **LineItem**	
- Creates a LineItem	

Product implements **LineItem**	
- Creates a Product	

DiscountedItem implements **LineItem**	
- Creates a discounted Product	

Bundle implements **LineItem**	
- Creates a bundle of Products	

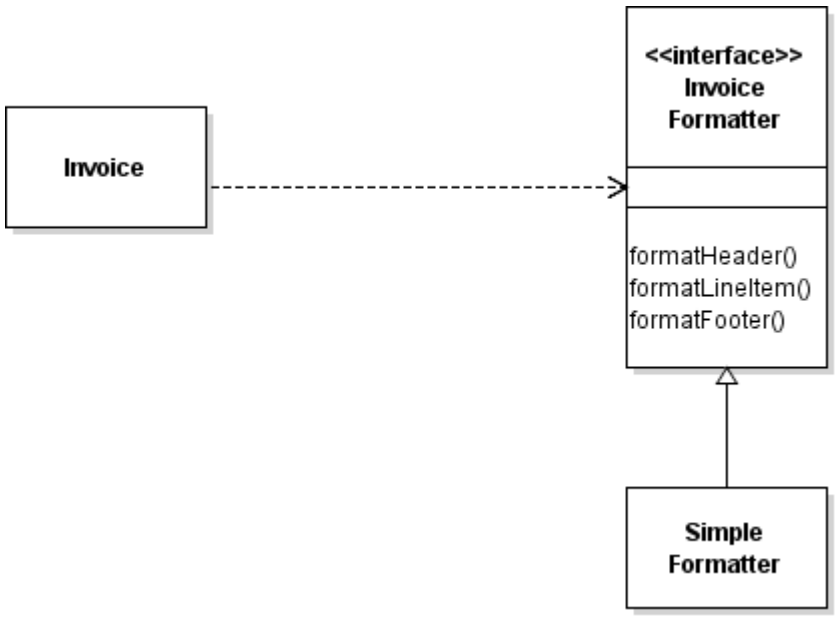
Class Diagram



*Class Diagram was updated based on a partially completed version of the program

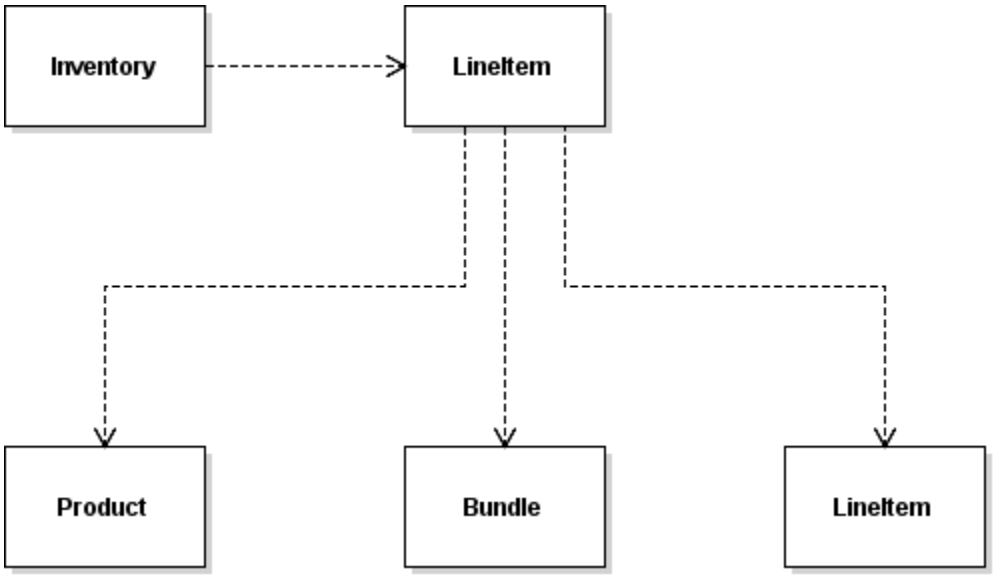
Design Patterns

Strategy Pattern



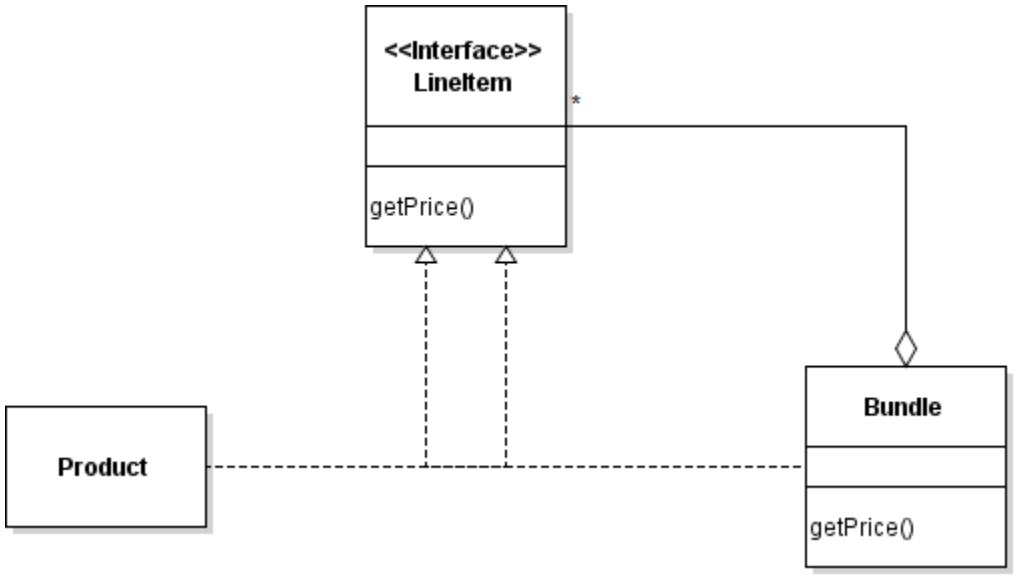
Name in Design Pattern	Actual Name
Context	Invoice
Strategy	InvoiceFormatter
ConcreteStrategy	SimpleFormatter
doWork()	formatHeader() formatLineItem() formatFooter()

Facade Pattern



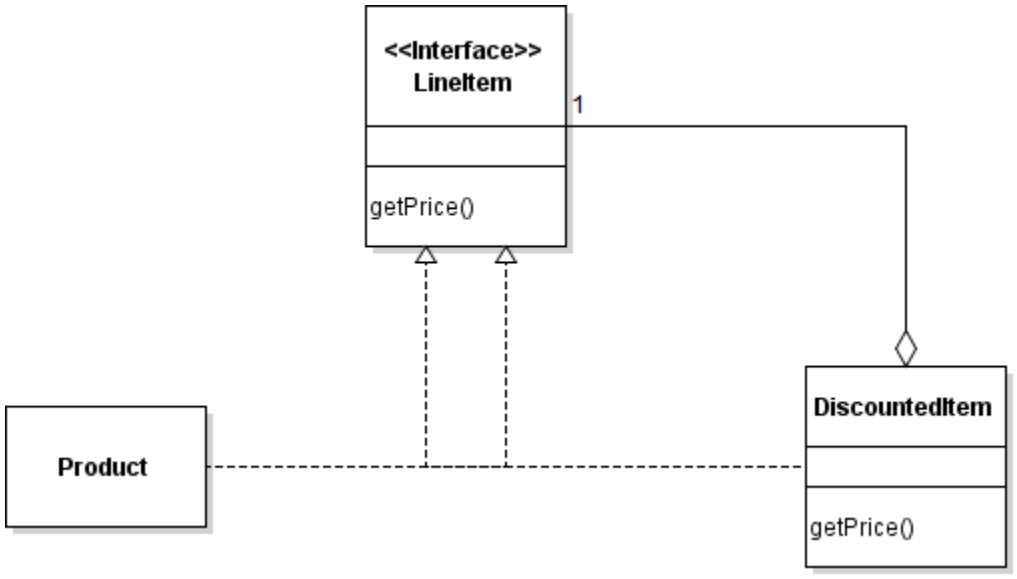
Name in Design Pattern	Actual Name
Client	Inventory
Facade	LineItem
SubSystemClass 1	Product
SubSystemClass 2	Bundle
SubSystemClass 3	DiscountedItem

Composite Pattern



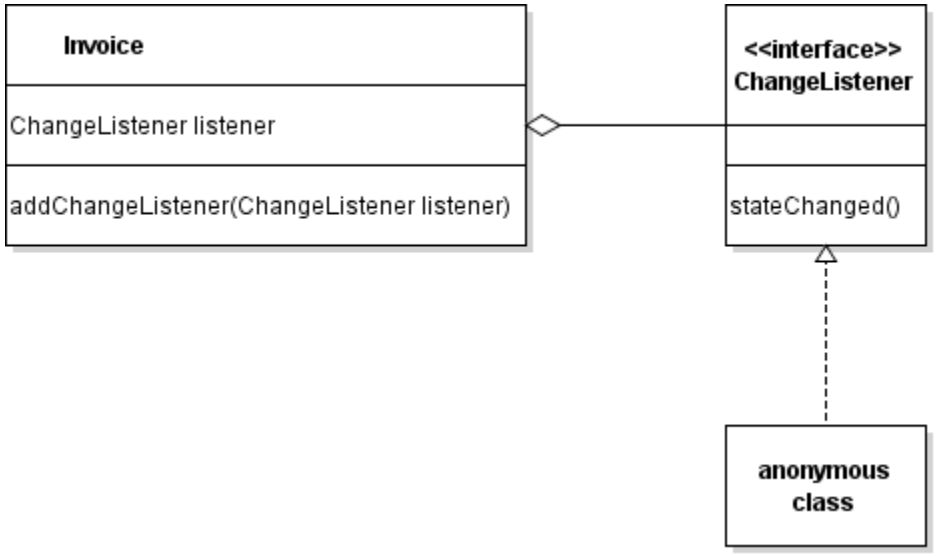
Name in Design Pattern	Actual Name
Primitive	LineItem
Composite	Bundle
Leaf	Product
method	getPrice()

Decorator Pattern



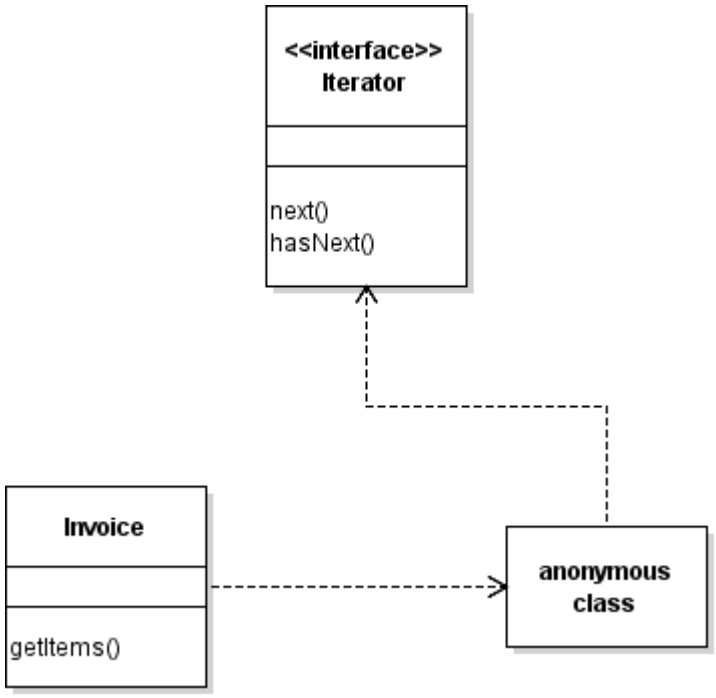
Name in Design Pattern	Actual Name
Component	LineItem
ConcreteComponent	Product
Decorator	DiscountedItem
Method()	getPrice()

Observer Pattern



Name in Design Pattern	Actual Name
Subject	Invoice
Observer	ChangeListener
ConcreteObserver	anonymous class
attach()	addChangeListener()
notify()	stateChanged()

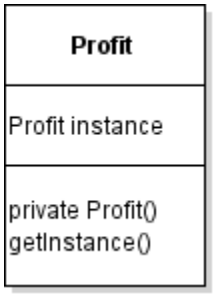
Iterator Pattern



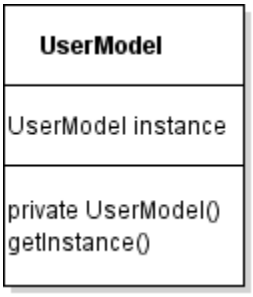
Name in Design Pattern	Actual Name
Aggregate	Invoice
Iterator	Iterator
createIterator()	anonymous class
next()	next()
isDone()	hasNext()
currentItem()	get()

Singleton Pattern

#1

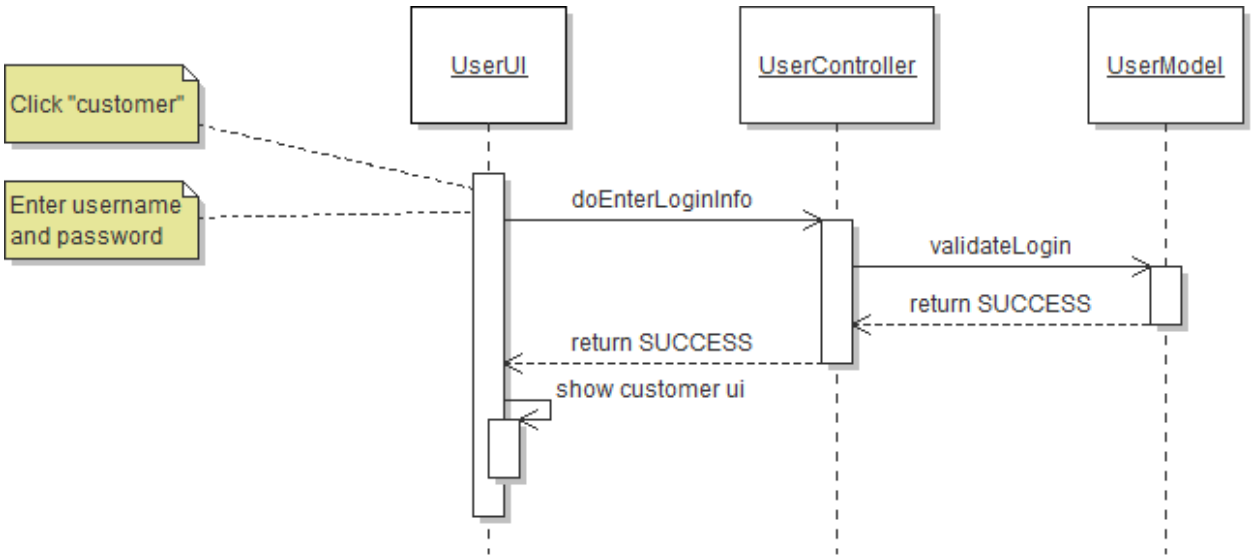


#2

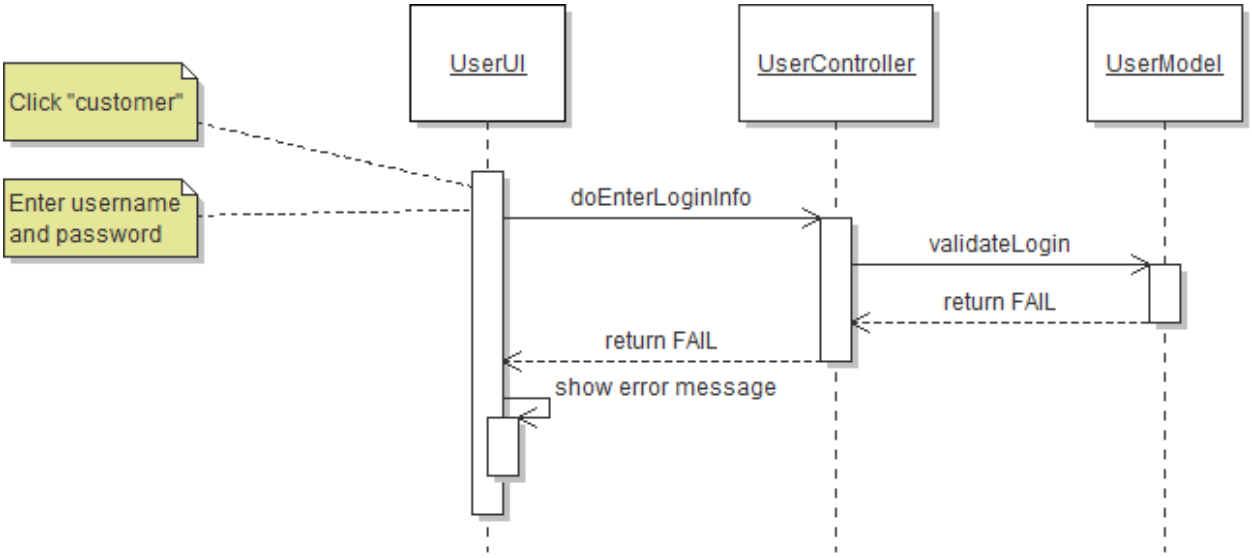


Sequence Diagrams

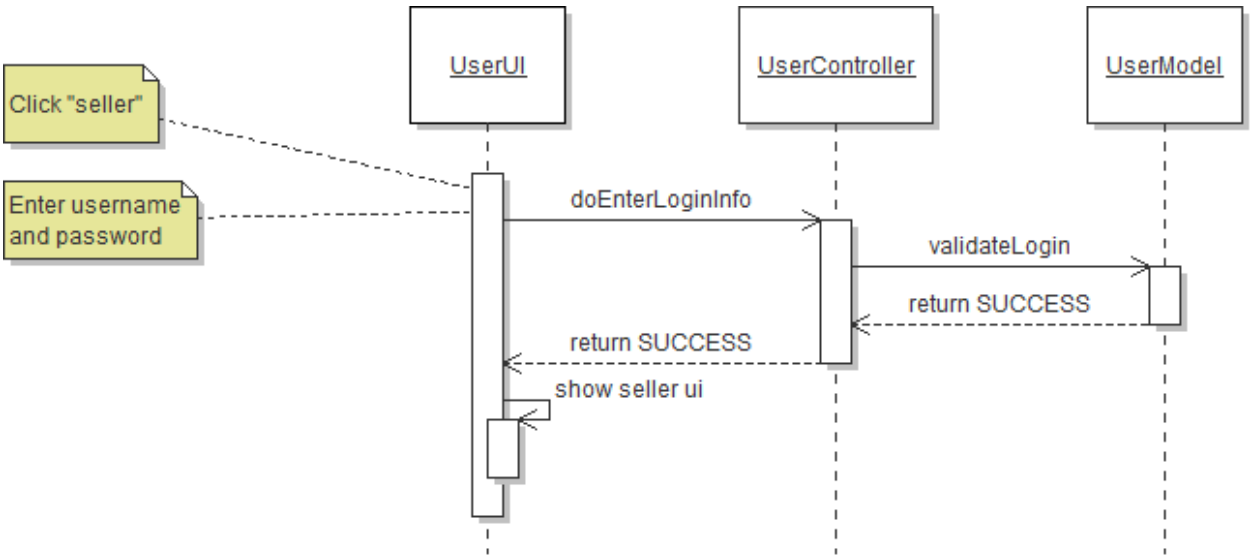
Use Case 1: Customer Log in.



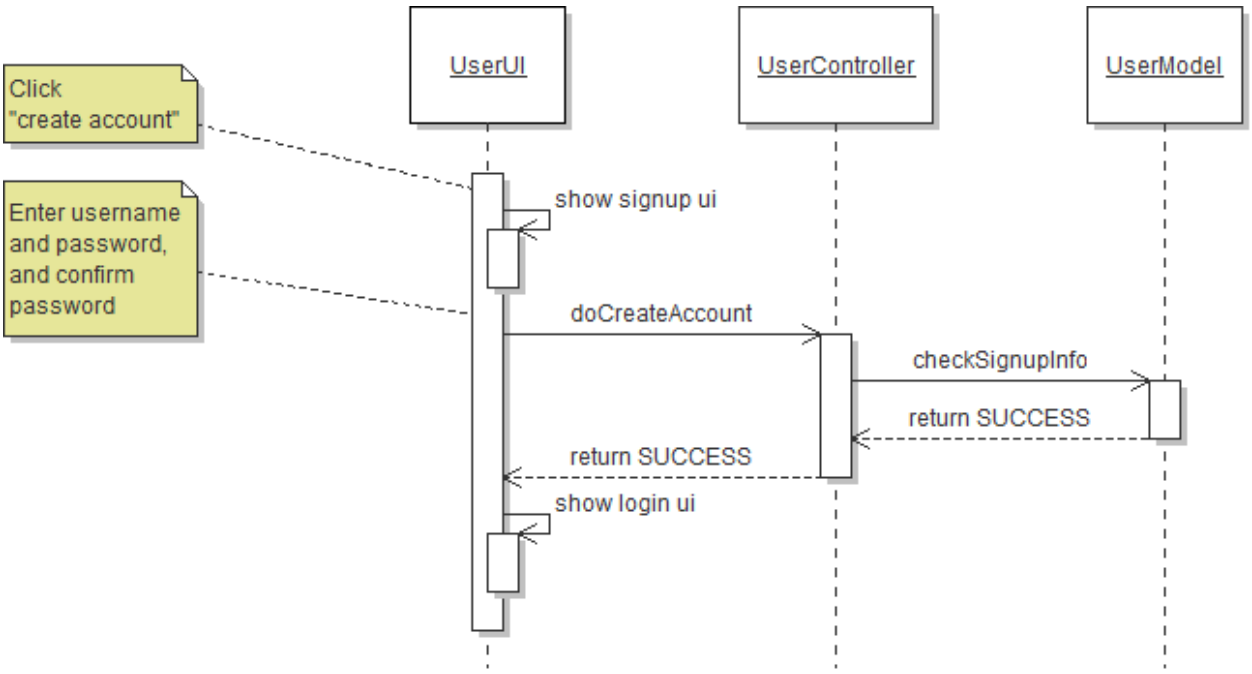
Variation #1: Invalid login.



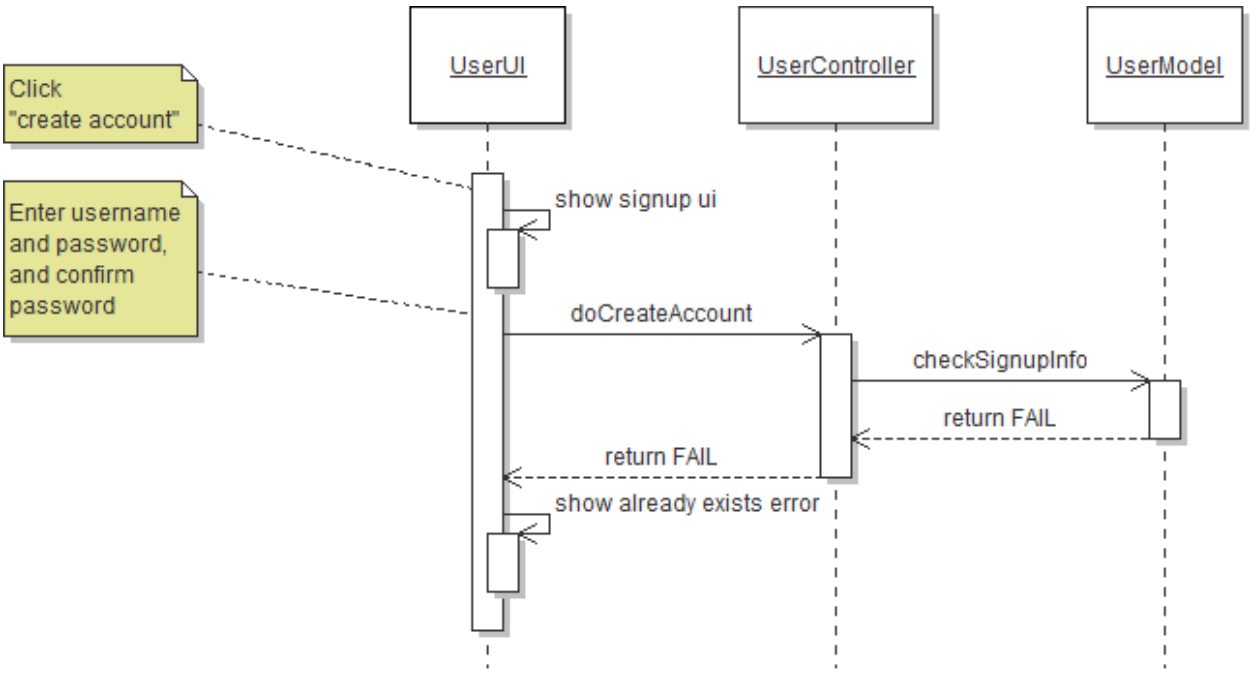
Use Case 2: Seller Log in.



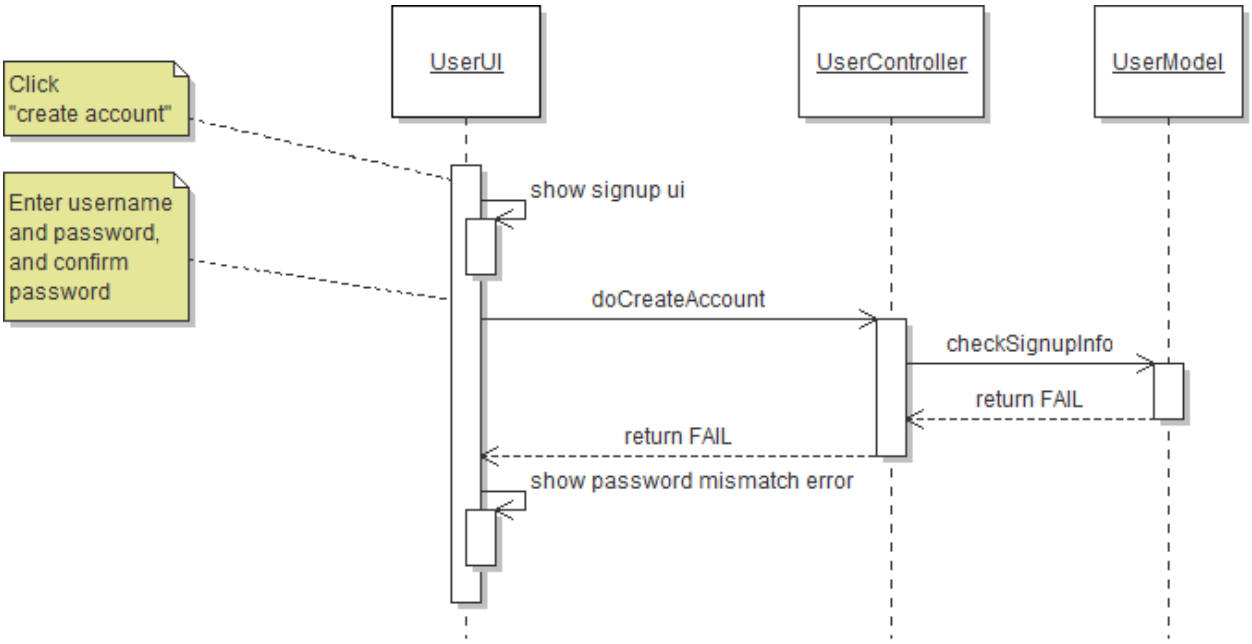
Use Case 3: Create new account.



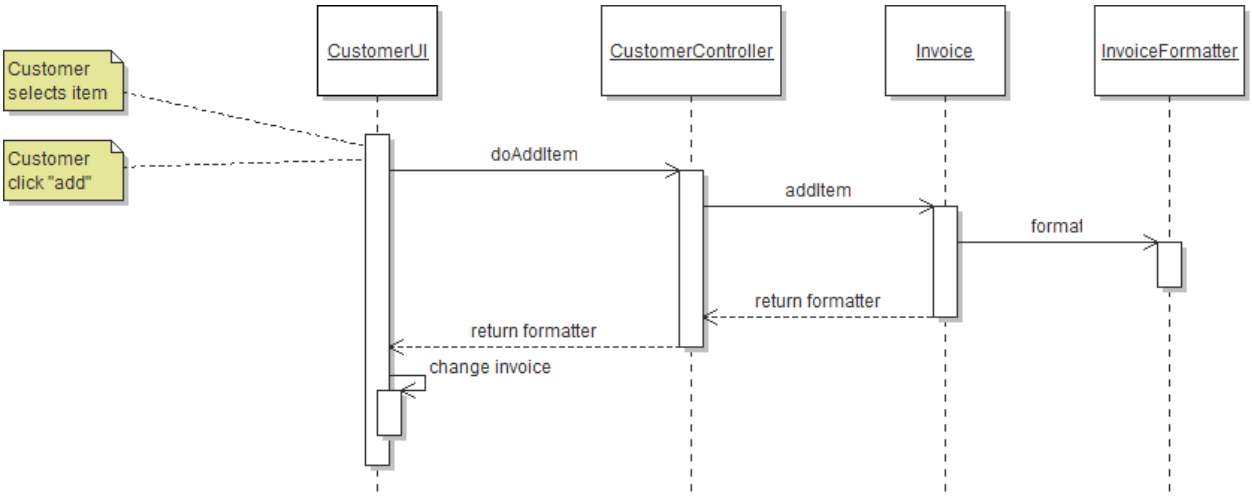
Variation #1: Account already exists.



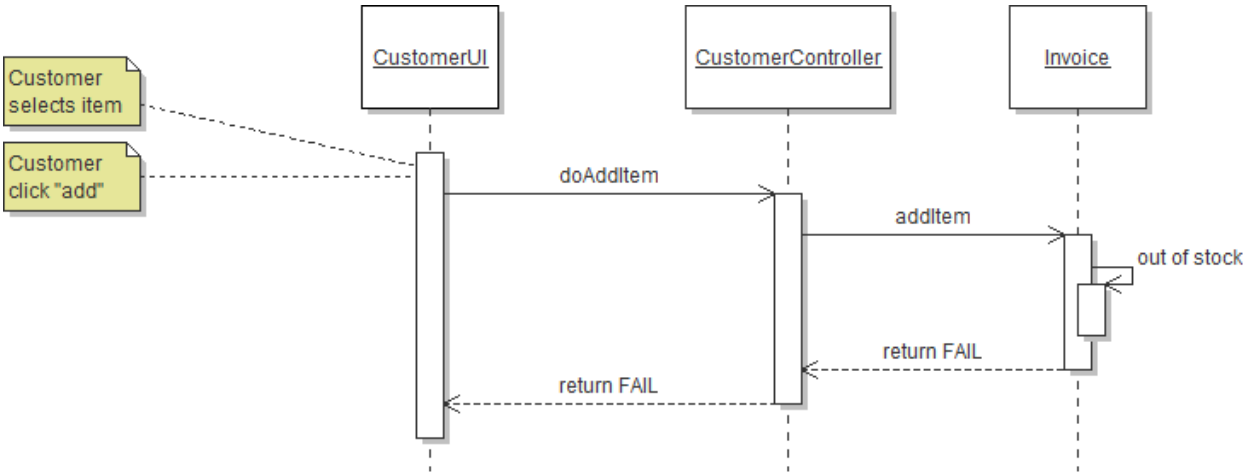
Variation #2: Password doesn't match.



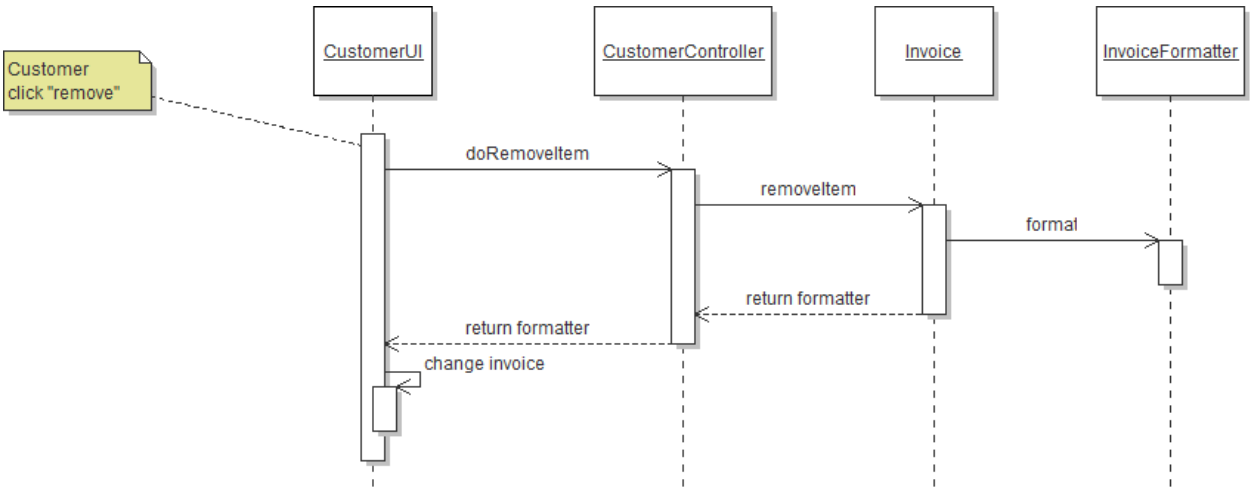
Use Case 4: Customer adds item to invoice.



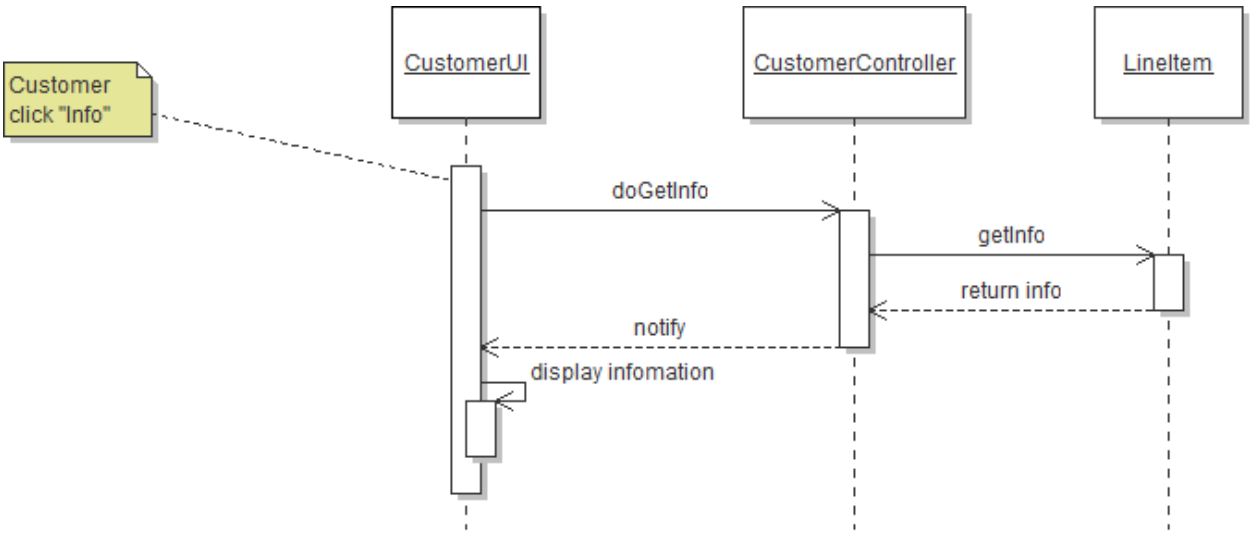
Variation #1: Out of stock.



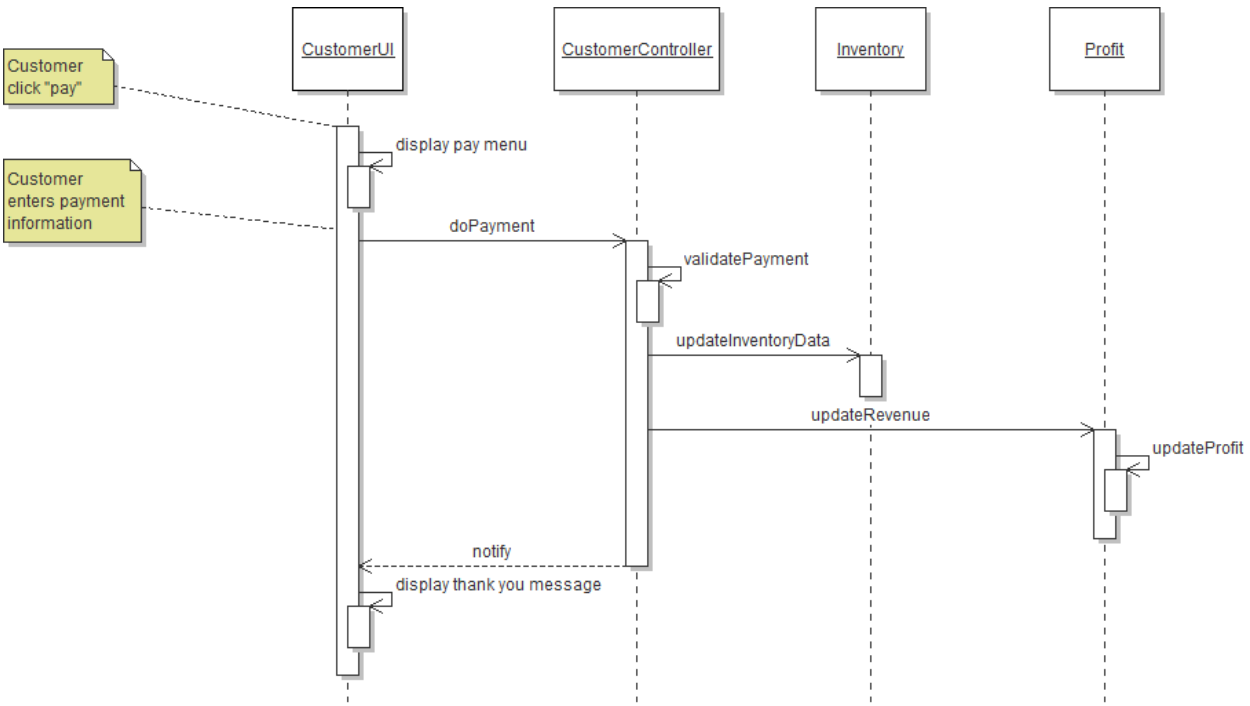
Use Case 5: Customer removes item from invoice.



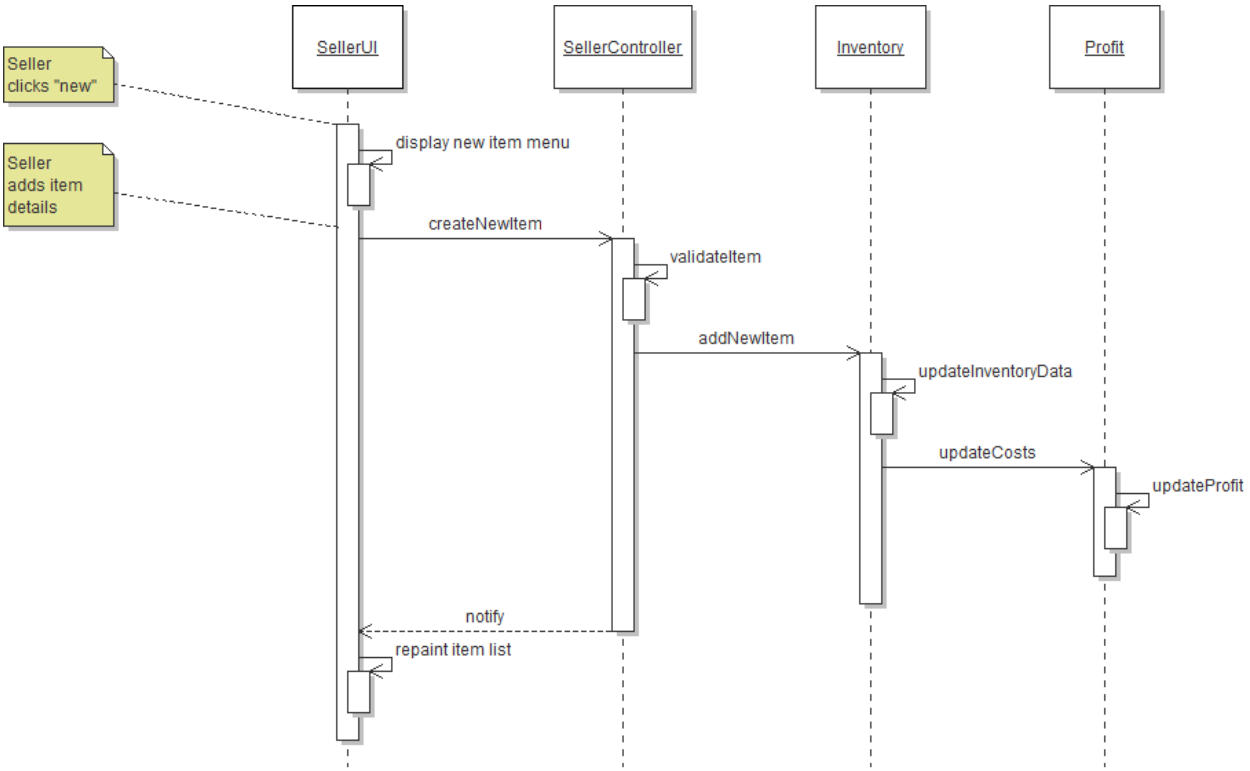
Use Case 6: Customer reviews product information.



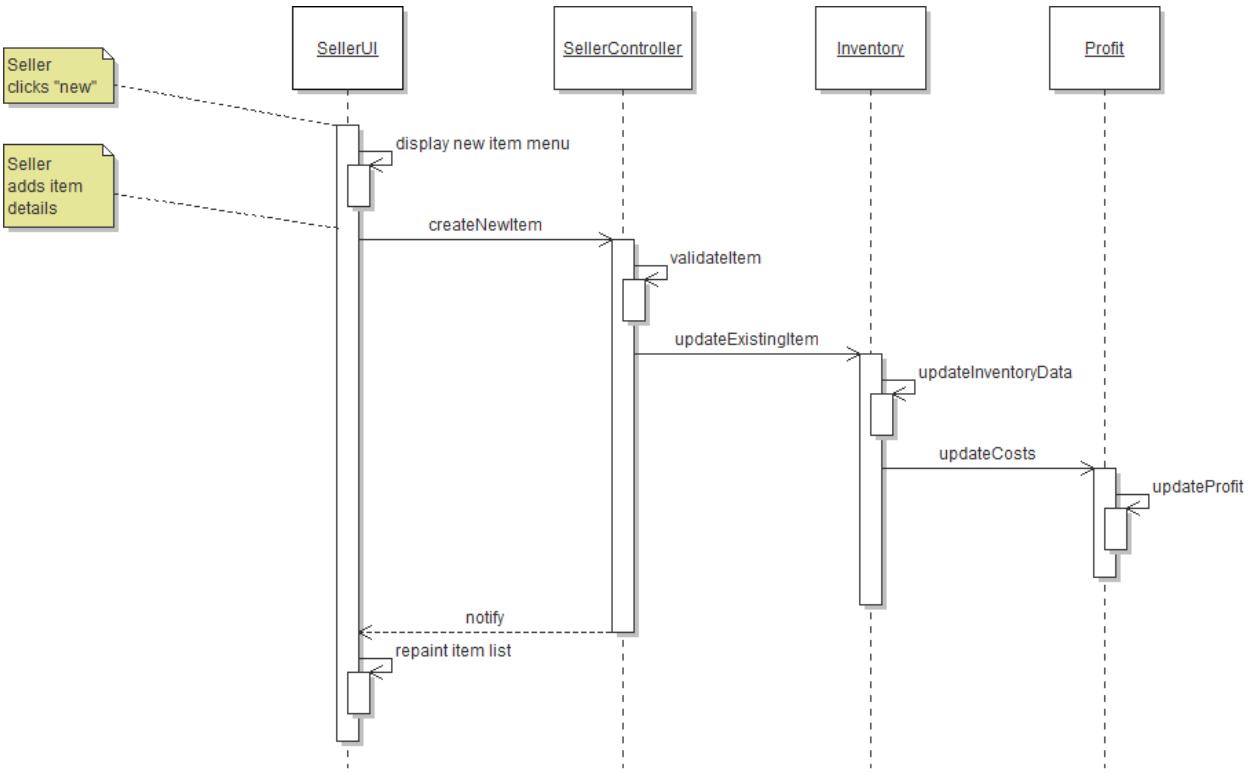
Use Case 7: Customer completes payment.



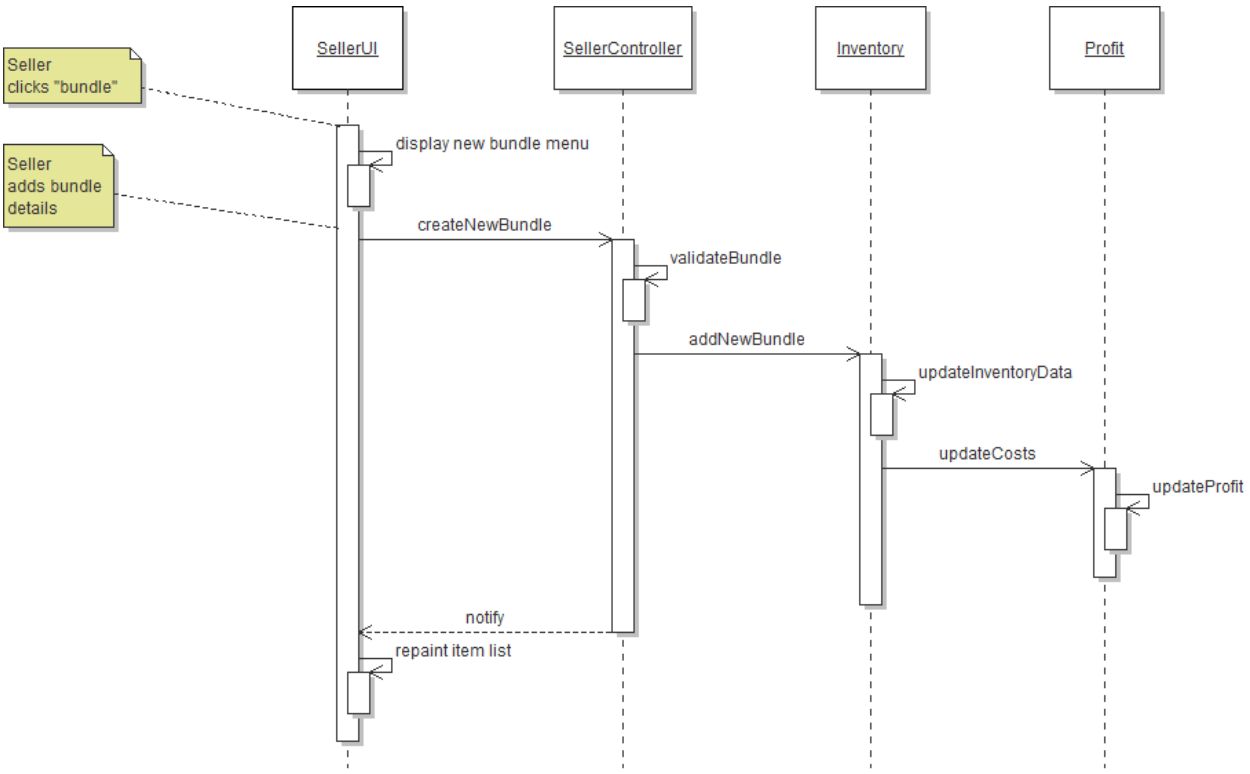
Use Case 8: Seller adds new item to inventory.



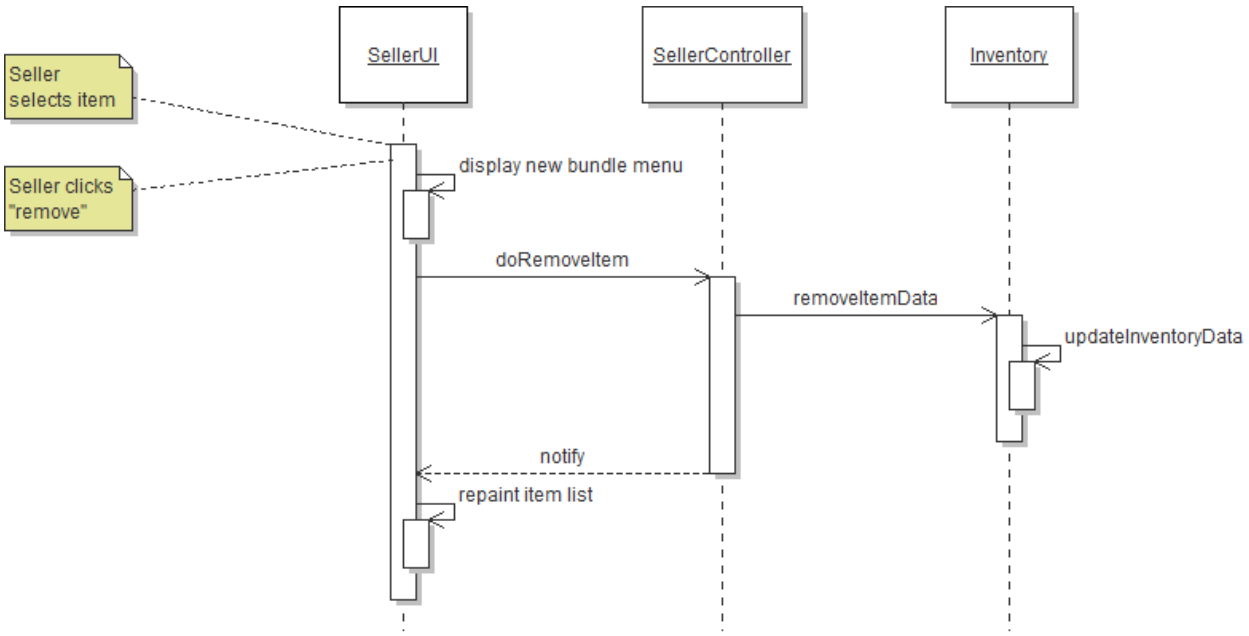
Variation #1: Item already in inventory



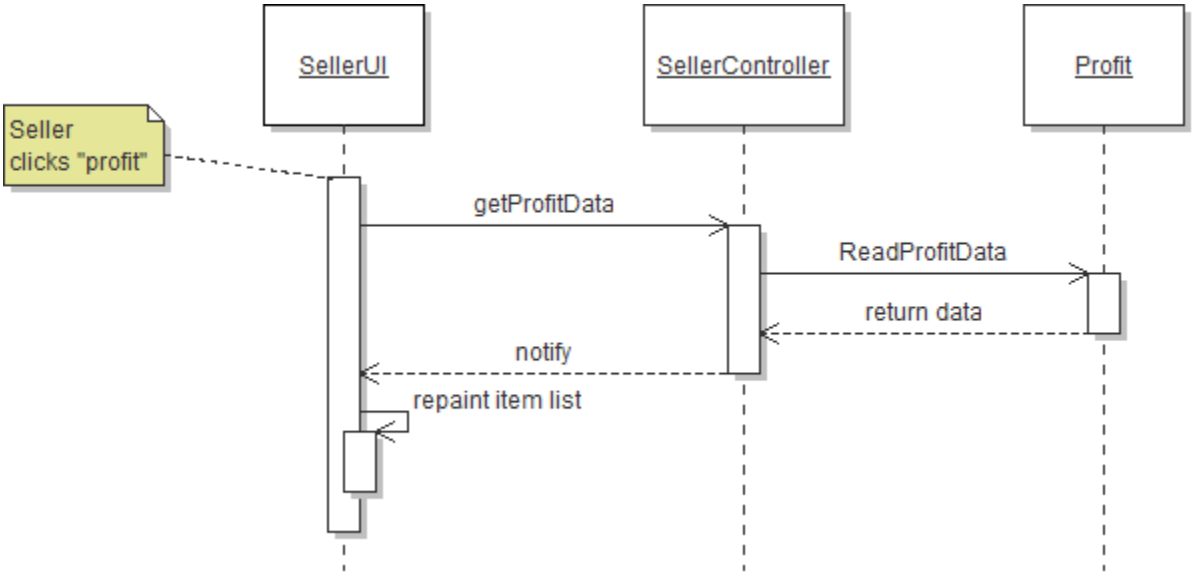
Use Case 9: Seller adds new bundle to inventory.



Use Case 10: Seller removes item from inventory.



Use Case 11: Seller checks revenue, costs, and profit.



State Diagrams