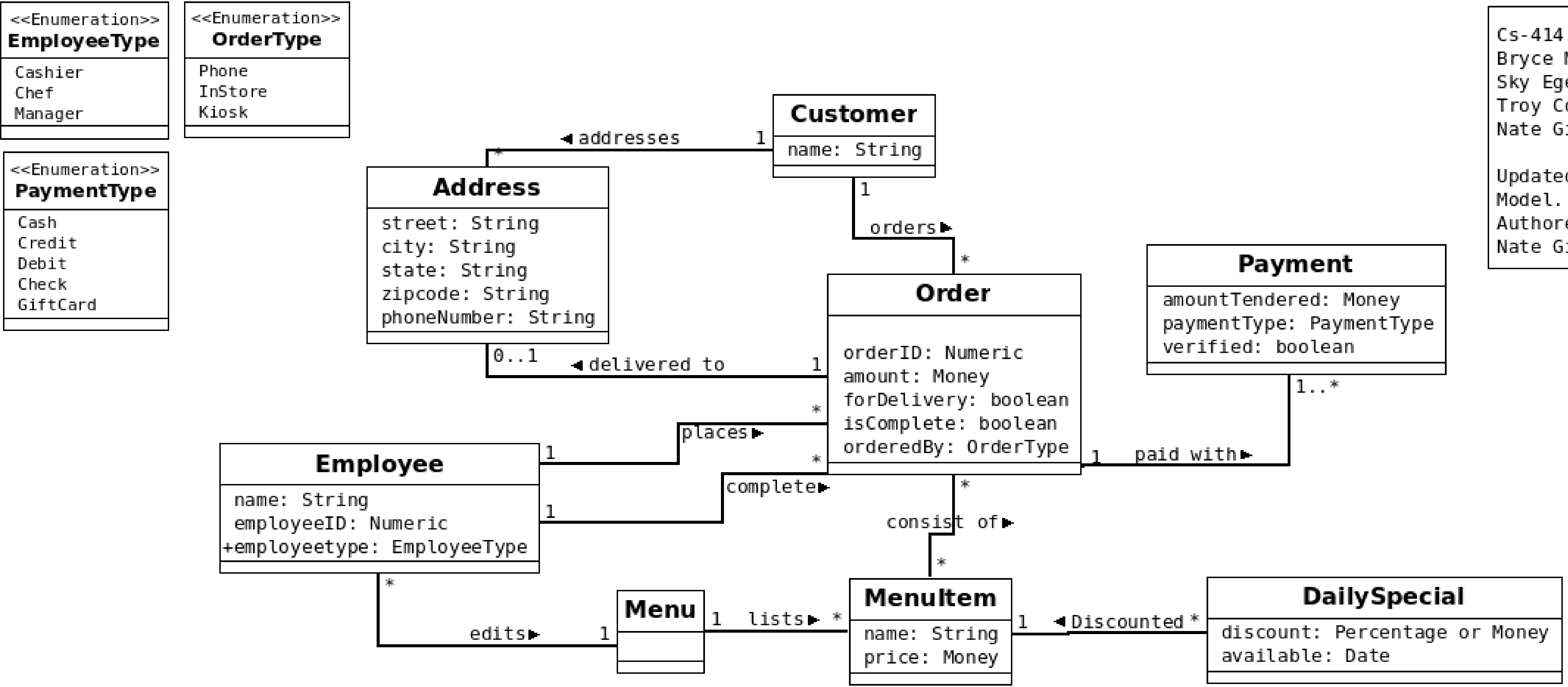
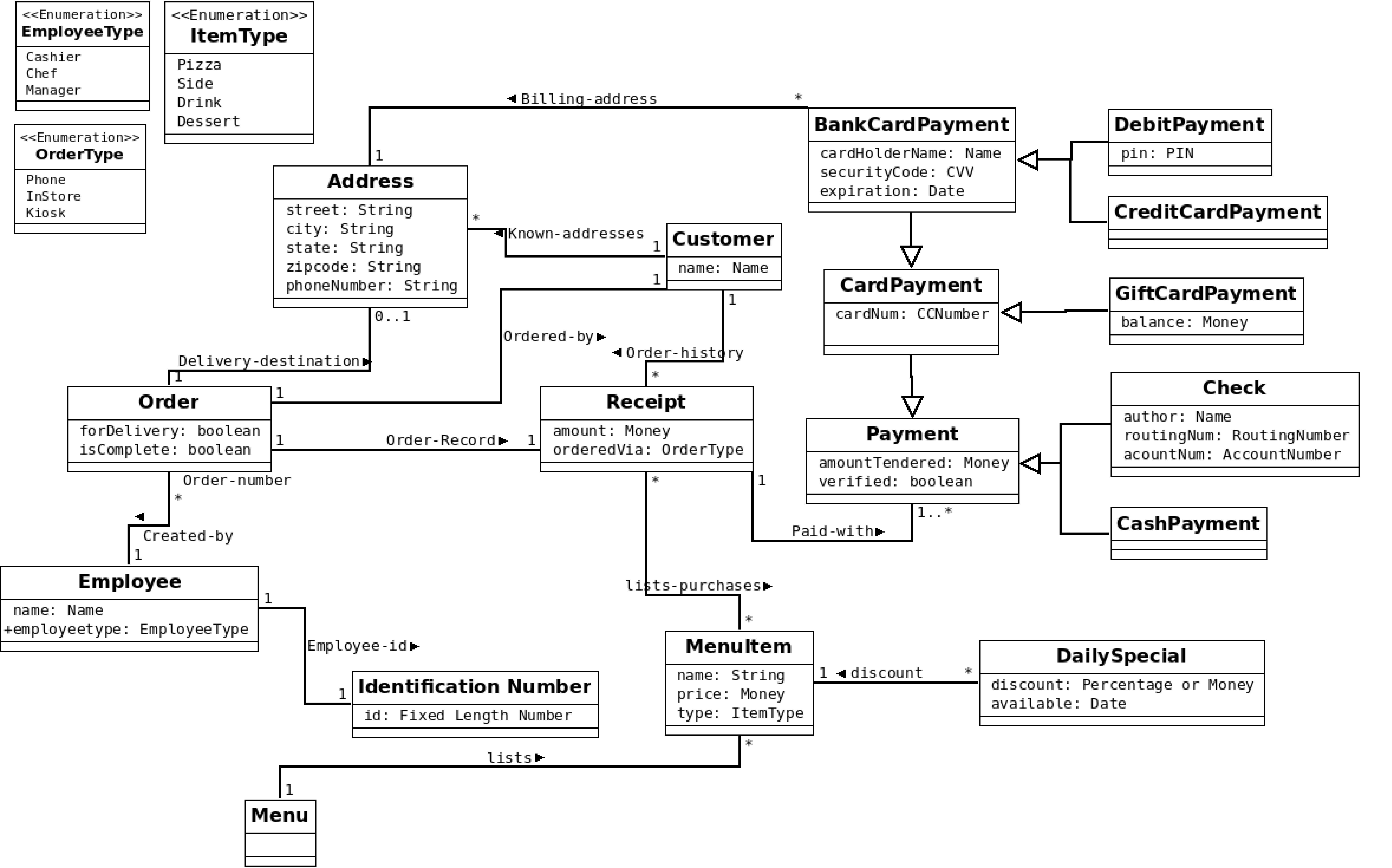


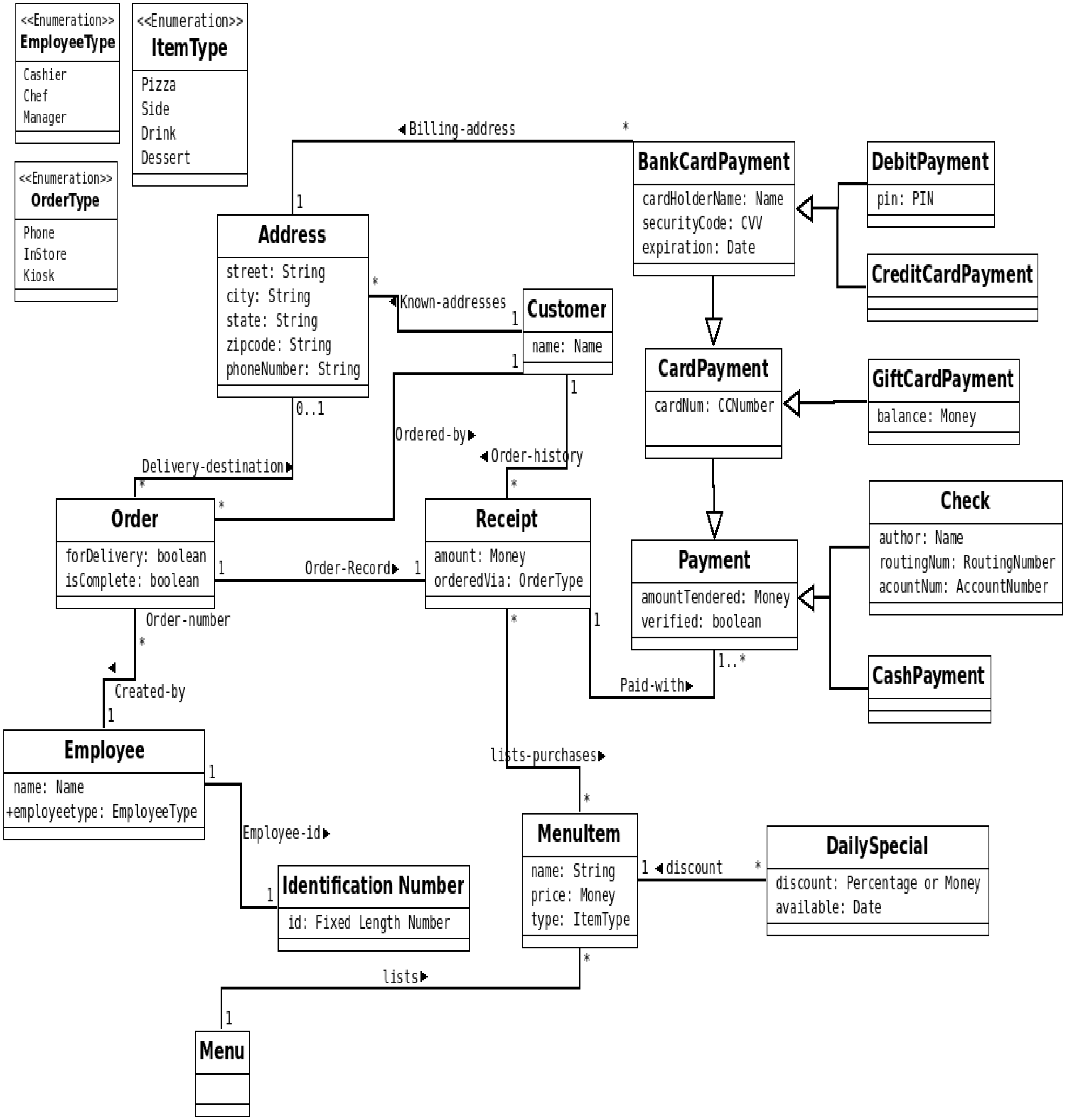
Cs-414 Group H
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard

Updated Domain
Model. 10/22/14
Authored by
Nate Gillard



Cs-414 Group H
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard

Updated Domain
Model. 10/22/14
Authored by
Nate Gillard



Cs-414 Group H
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard

Updated Domain
Model. 10/22/14
Authored by
Nate Gillard

## Glossary:

Customer: Represents a person purchasing items at the restaurant.  Associated with an address for delivery.

Address: A location record for delivery.  Locations are associated with customers and BankCardPayments.

Receipt: A record of the order transaction.  Associated to an Order, a number of MenuItems. and a Payment.

Order: Represents the customer's purchase and status.  Associated to an Employee a Customer and a Receipt.

Employee:  Represents an employee of the restaurant; may be a cashier, chef, or manager.

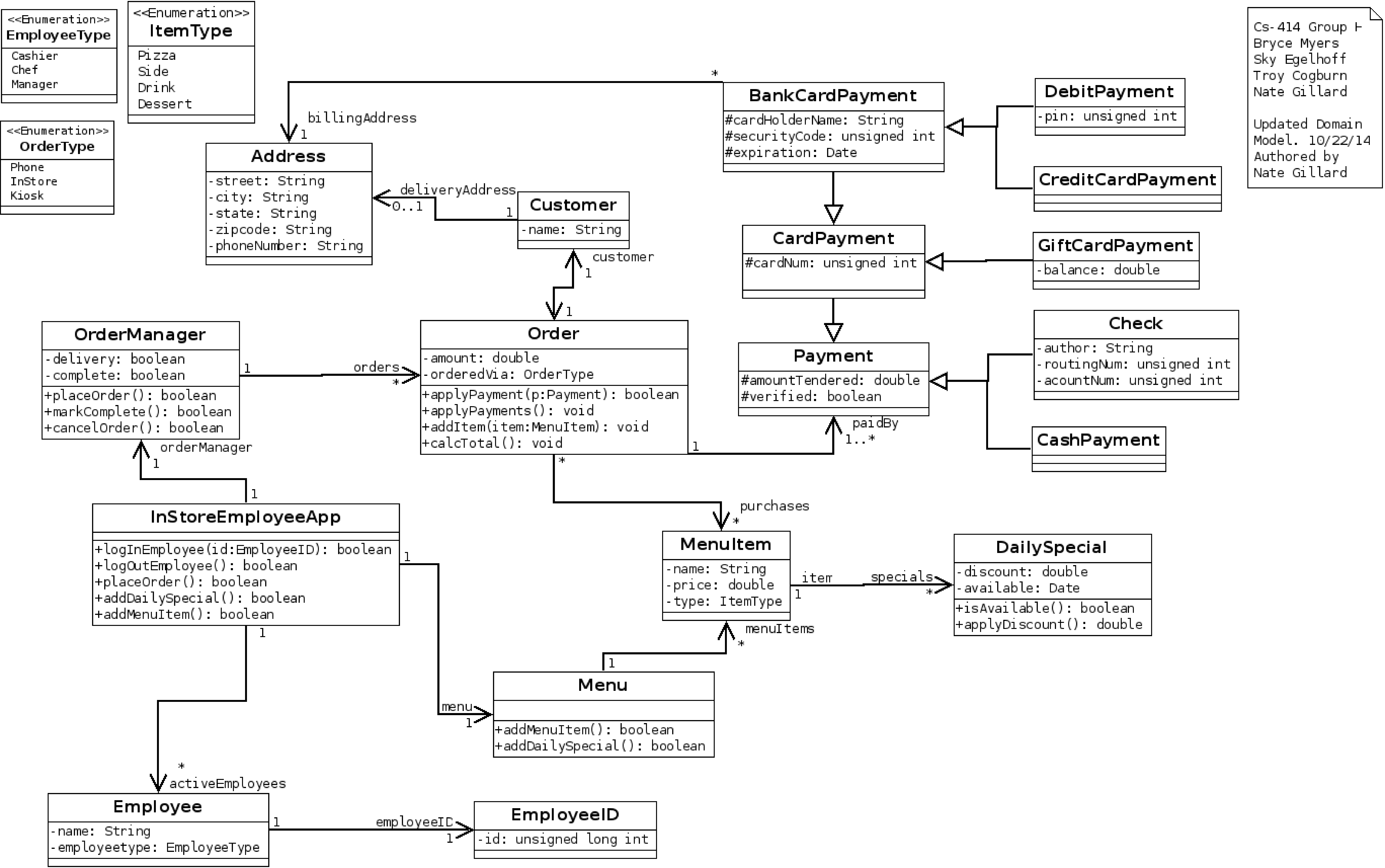
Employee Identification: A number to identify an employee.

Menu: a list of foods and drinks offered at the restaurant.

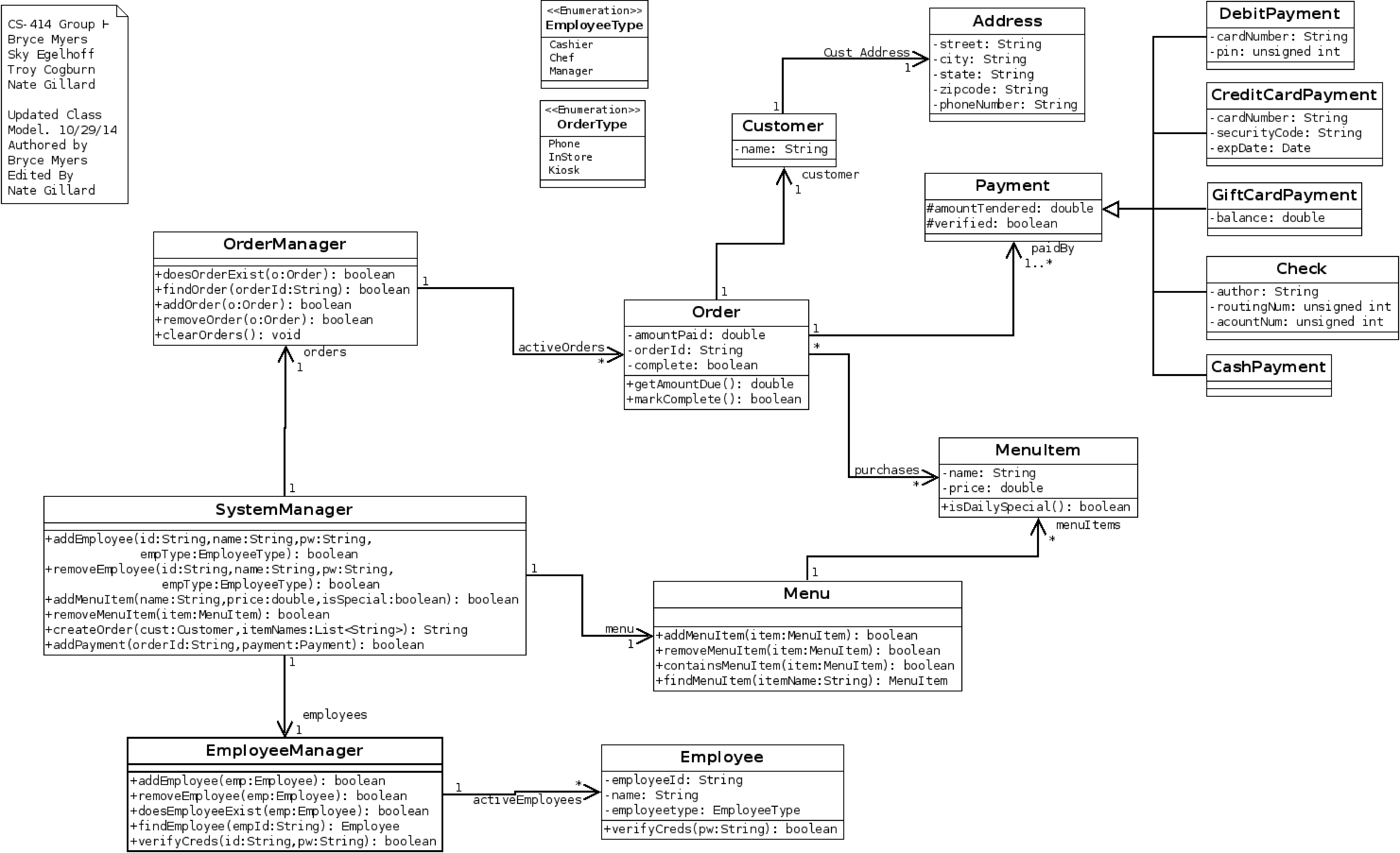
MenuItem: A single item a offered on the menu; may be a pizza, a side, a drink, or a dessert.

DailySpecial:  A discount on a menu item.

Payment:  An amount of money in different forms, used by customer to purchase items.  Payment forms require different information for processing in various scenarios.  Associated with a Receipt.



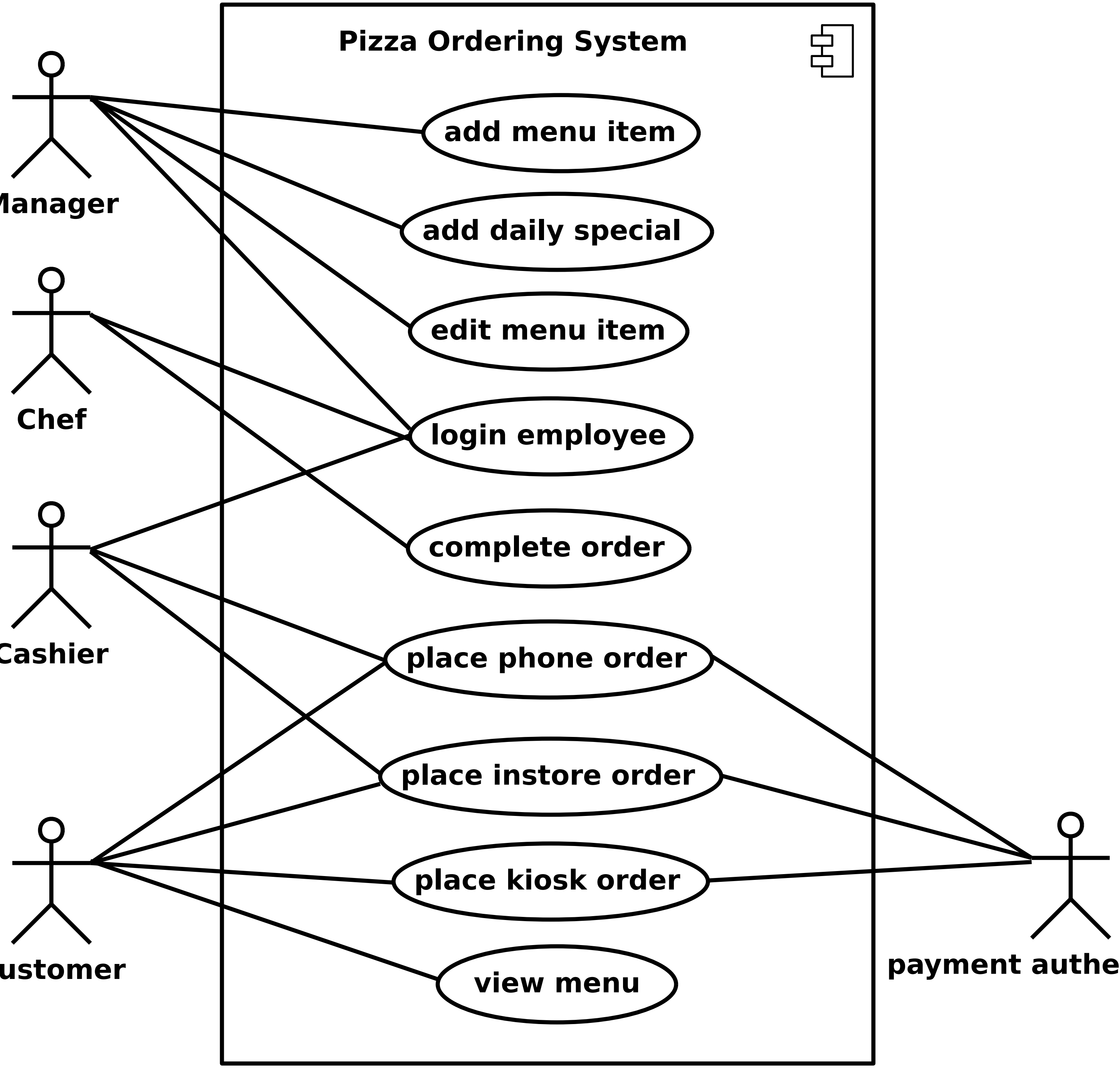
<<Enumeration>>
EmployeeType
Cashier
Chef
Manager
<<Enumeration>>
ItemType
Pizza
Side
Drink
Dessert
<<Enumeration>>
OrderType
Phone
InStore
Kiosk
Cs-414 Group H
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard
Updated Domain
Model. 10/22/14
Authored by
Nate Gillard
*
billingAddress
1
Address
-street: String
-city: String
-state: String
-zipcode: String
-phoneNumber: String
deliveryAddress
0..1
1
Customer
-name: String
customer
1
1
BankCardPayment
#cardHolderName: String
#securityCode: unsigned int
#expiration: Date
DebitPayment
-pin: unsigned int
CreditCardPayment
CardPayment
#cardNum: unsigned int
GiftCardPayment
-balance: double
Check
-author: String
-routingNum: unsigned int
-acountNum: unsigned int
Payment
#amountTendered: double
#verified: boolean
paidBy
1..*
CashPayment
OrderManager
-delivery: boolean
-complete: boolean
+placeOrder(): boolean
+markComplete(): boolean
+cancelOrder(): boolean
1
orders
*
Order
-amount: double
-orderedVia: OrderType
+applyPayment(p:Payment): boolean
+applyPayments(): void
+addItem(item:MenuItem): void
+calcTotal(): void
1
*
orderManager
1
1
purchases
*
InStoreEmployeeApp
+logInEmployee(id:EmployeeID): boolean
+logOutEmployee(): boolean
+placeOrder(): boolean
+addDailySpecial(): boolean
+addMenuItem(): boolean
1
1
MenuItem
-name: String
-price: double
-type: ItemType
item
1
specials
*
DailySpecial
-discount: double
-available: Date
+isAvailable(): boolean
+applyDiscount(): double
menuItems
*
1
Menu
+addMenuItem(): boolean
+addDailySpecial(): boolean
menu
1
*
activeEmployees
Employee
-name: String
-employeetype: EmployeeType
1
employeeID
1
EmployeeID
-id: unsigned long int

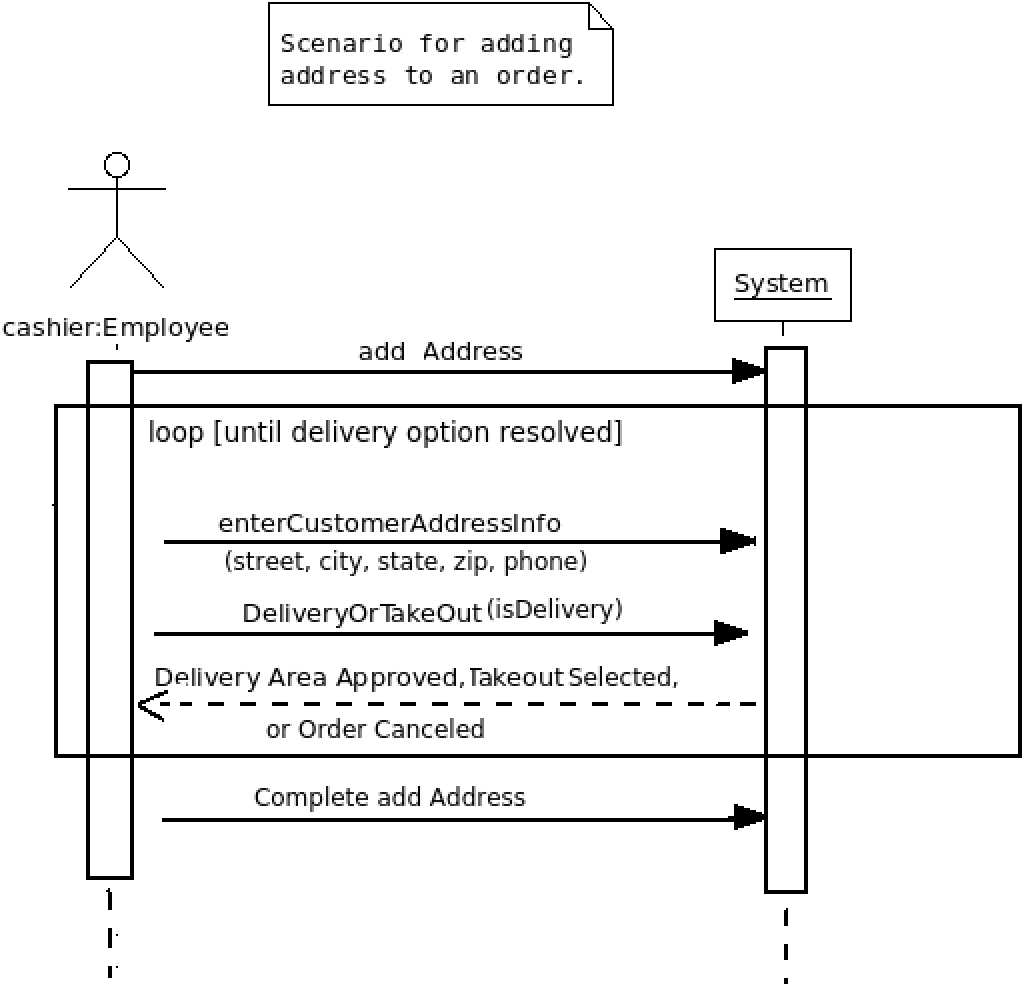


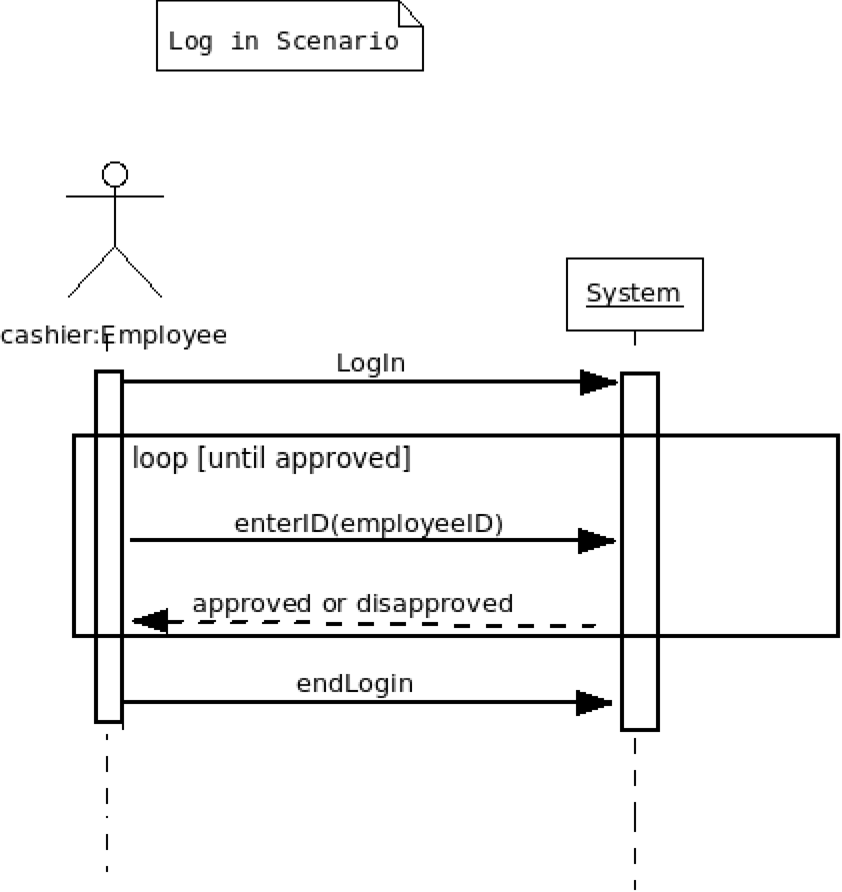
CS-414 Group H
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard

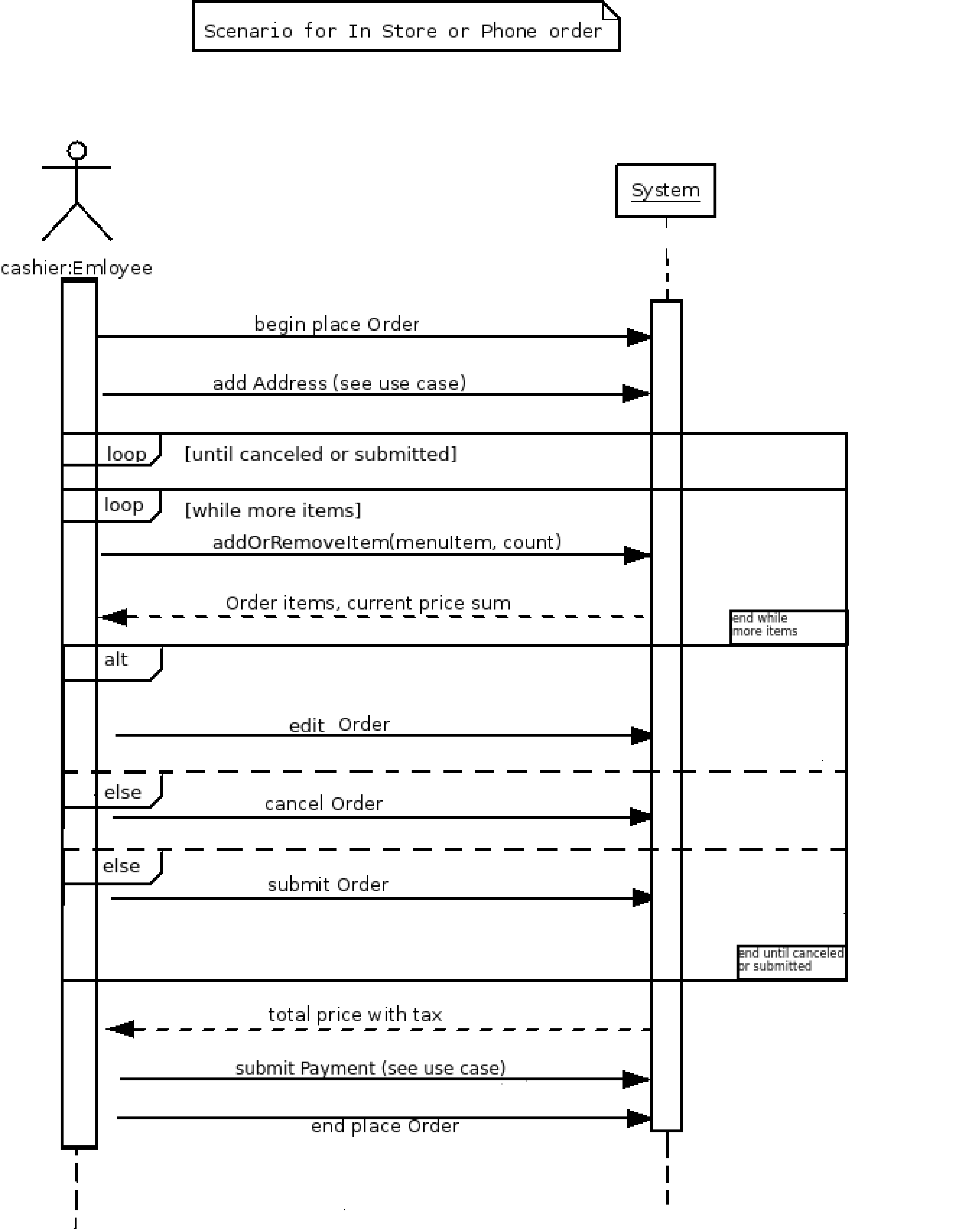
Updated Class Model. 10/29/14
Authored by
Bryce Myers
Edited By
Nate Gillard

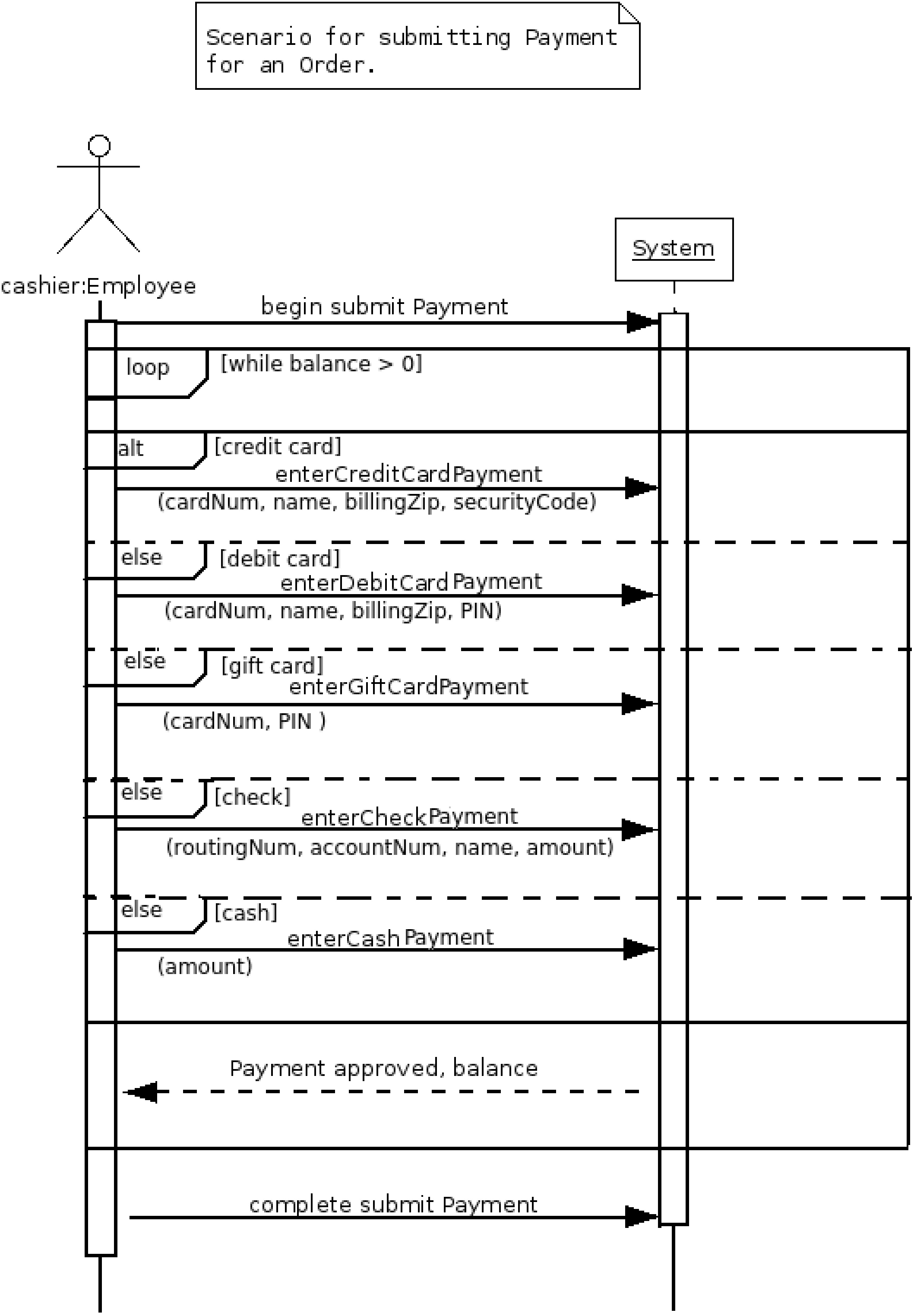
<<Enumeration>>
EmployeeType
Cashier
Chef
Manager
<<Enumeration>>
OrderType
Phone
InStore
Kiosk
Address
-street: String
-city: String
-state: String
-zipcode: String
-phoneNumber: String
Cust Address
1
1
Customer
-name: String
customer
1
DebitPayment
-cardNumber: String
-pin: unsigned int
CreditCardPayment
-cardNumber: String
-securityCode: String
-expDate: Date
Payment
#amountTendered: double
#verified: boolean
GiftCardPayment
-balance: double
Check
-author: String
-routingNum: unsigned int
-acountNum: unsigned int
CashPayment
paidBy
1..*
OrderManager
+doesOrderExist(o:Order): boolean
+findOrder(orderId:String): boolean
+addOrder(o:Order): boolean
+removeOrder(o:Order): boolean
+clearOrders(): void
1
activeOrders
*
orders
1
1
Order
-amountPaid: double
-orderId: String
-complete: boolean
+getAmountDue(): double
+markComplete(): boolean
1
*
purchases
*
MenuItem
-name: String
-price: double
+isDailySpecial(): boolean
menuItems
*
1
SystemManager
+addEmployee(id:String,name:String,pw:String,
empType:EmployeeType): boolean
+removeEmployee(id:String,name:String,pw:String,
empType:EmployeeType): boolean
+addMenuItem(name:String,price:double,isSpecial:boolean): boolean
+removeMenuItem(item:MenuItem): boolean
+createOrder(cust:Customer,itemNames:List<String>): String
+addPayment(orderId:String,payment:Payment): boolean
1
menu
1
1
Menu
+addMenuItem(item:MenuItem): boolean
+removeMenuItem(item:MenuItem): boolean
+containsMenuItem(item:MenuItem): boolean
+findMenuItem(itemName:String): MenuItem
1
employees
1
EmployeeManager
+addEmployee(emp:Employee): boolean
+removeEmployee(emp:Employee): boolean
+doesEmployeeExist(emp:Employee): boolean
+findEmployee(empId:String): Employee
+verifyCreds(id:String,pw:String): boolean
1
activeEmployees
*
Employee
-employeeId: String
-name: String
-employeetype: EmployeeType
+verifyCreds(pw:String): boolean


Pizza Ordering System
Manager
Chef
Cashier
ustomer
payment authe
add menu item
add daily special
edit menu item
login employee
complete order
place phone order
place instore order
place kiosk order
view menu


Scenario for adding address to an order.
cashier:Employee
System
add Address
loop [until delivery option resolved]
enterCustomerAddressInfo
(street, city, state, zip, phone)
DeliveryOrTakeOut (isDelivery)
Delivery Area Approved, Takeout Selected,
or Order Canceled
Complete add Address


Log in Scenario
cashier:Employee
System
LogIn
loop [until approved]
enterID(employeeID)
approved or disapproved
endLogin

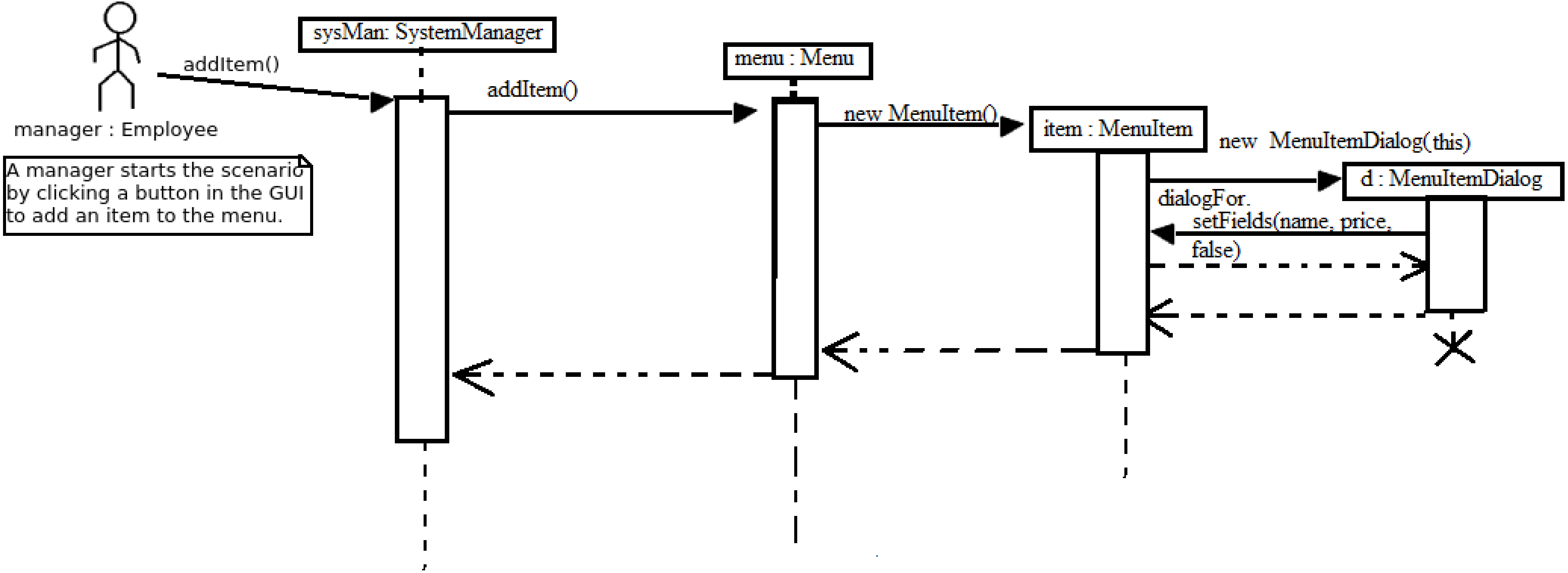

Scenario for In Store or Phone order
System
cashier:Emloyee
begin place Order
add Address (see use case)
loop
[until canceled or submitted]
loop
[while more items]
addOrRemoveItem(menuItem, count)
Order items, current price sum
end while
more items
alt
edit  Order
else
cancel Order
else
submit Order
end until canceled
or submitted
total price with tax
submit Payment (see use case)
end place Order



Scenario for submitting Payment for an Order.

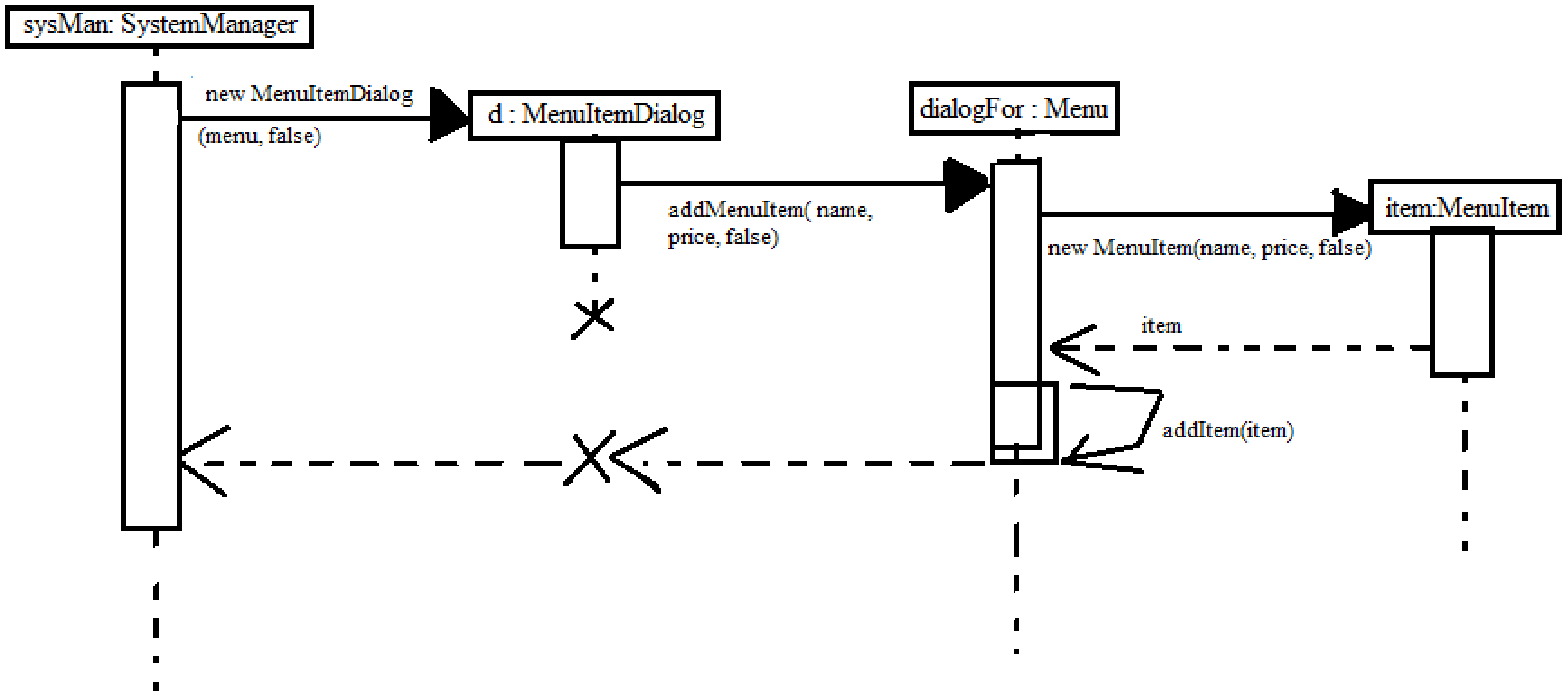
# Add MenuItem Sequence Diagram.

Cs-414 Group H
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard



A manager starts the scenario by clicking a button in the GUI to add an item to the menu.

## Current Implementation of add MenuItem



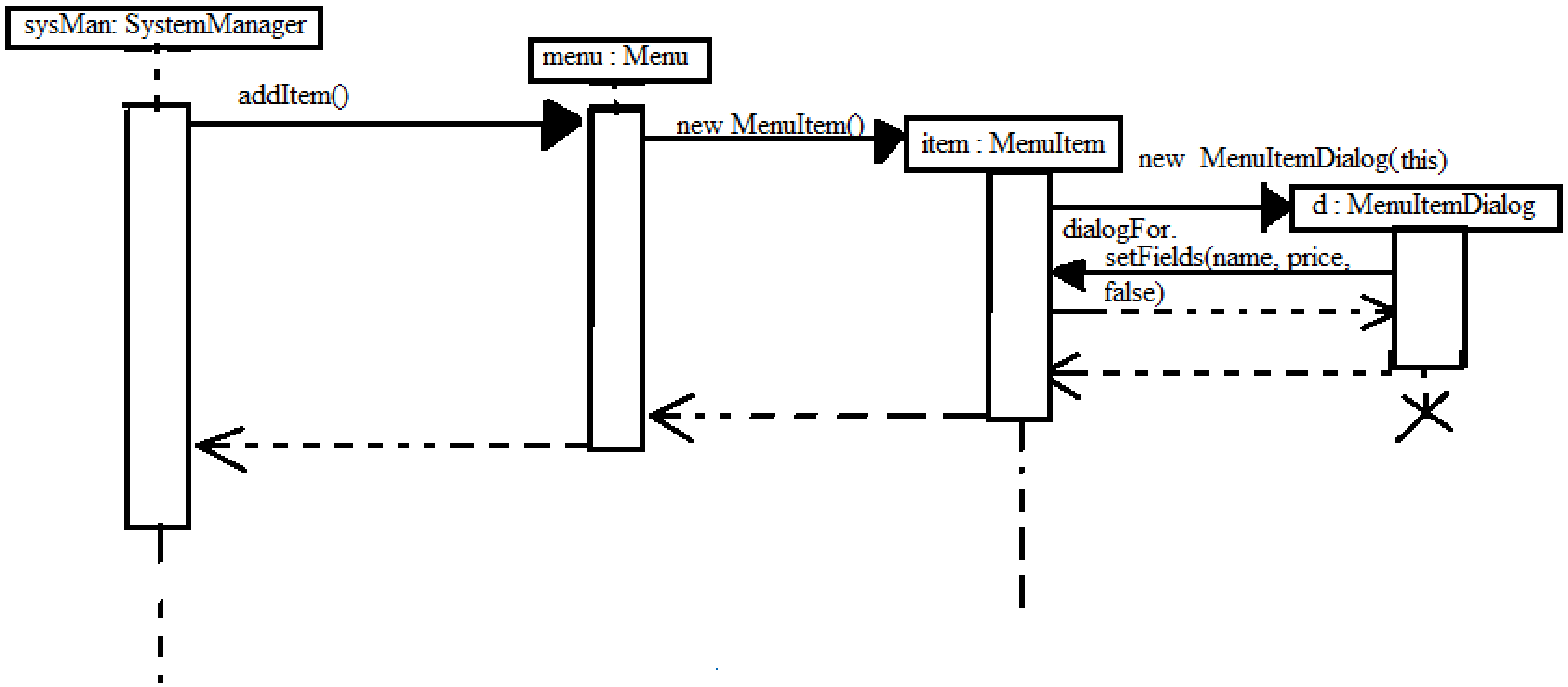
Issues:

1. parameter passing. What if we need to add an item description? At least two calls need to change maybe more.

2. What happens when dialogFor.addMenuItem() call returns to non exisitant item?

---

## Suggested Implmentation:

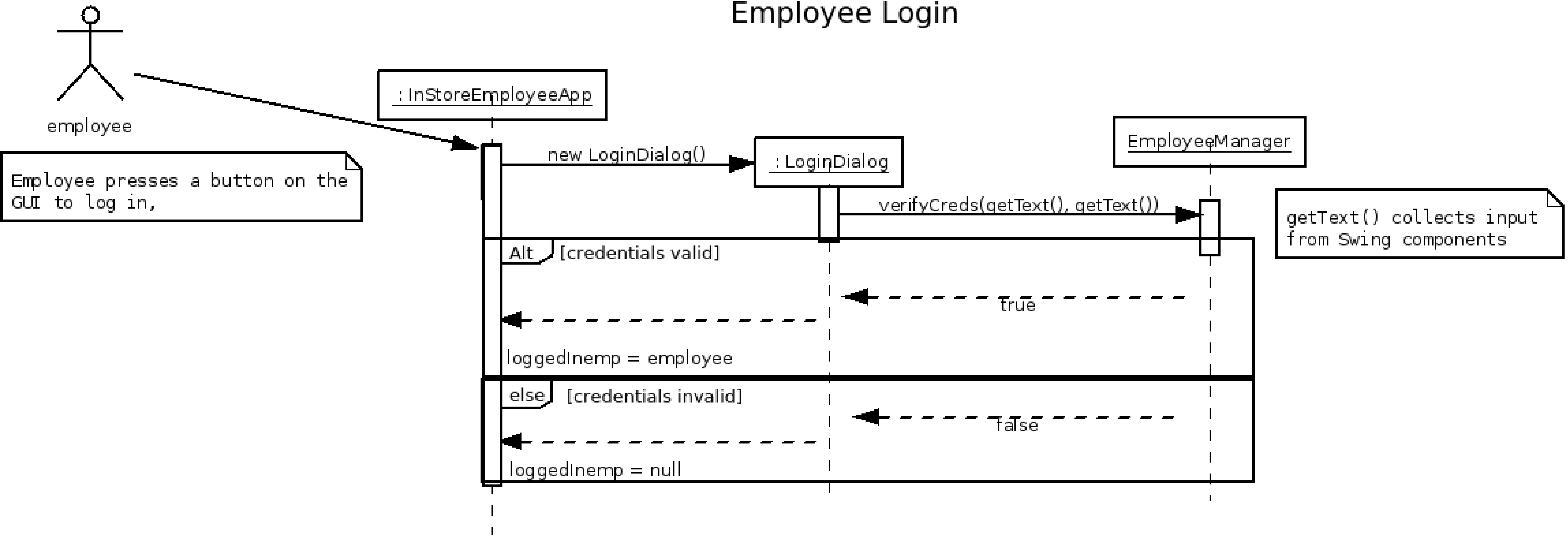


Notes:

1. One one instance of pramater passing. this one can possibly be eliminated if 'd' is assigned to a variable in MenuItem constructor

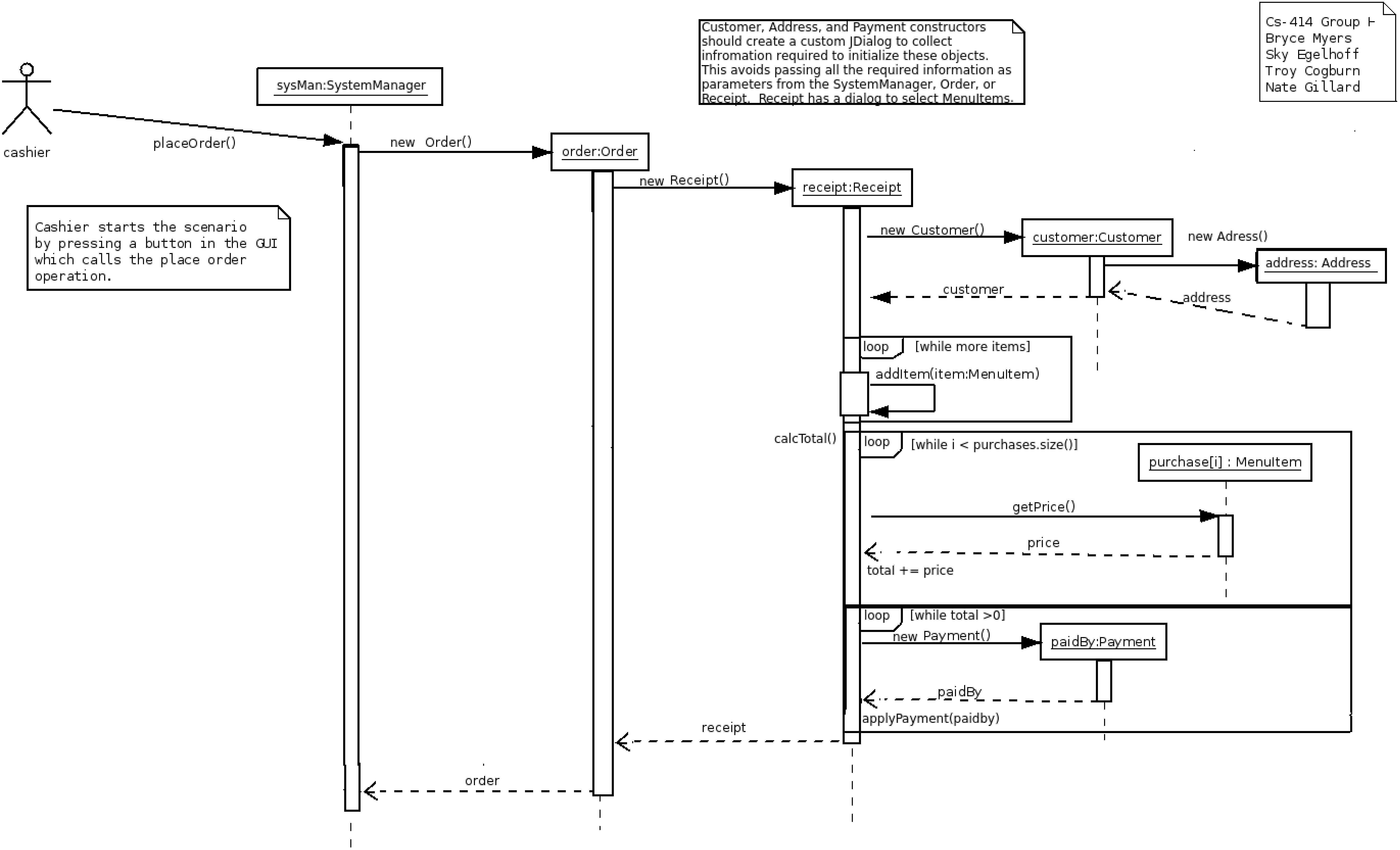
2. No unpreidctable return to non-existant objects.

# Employee Login



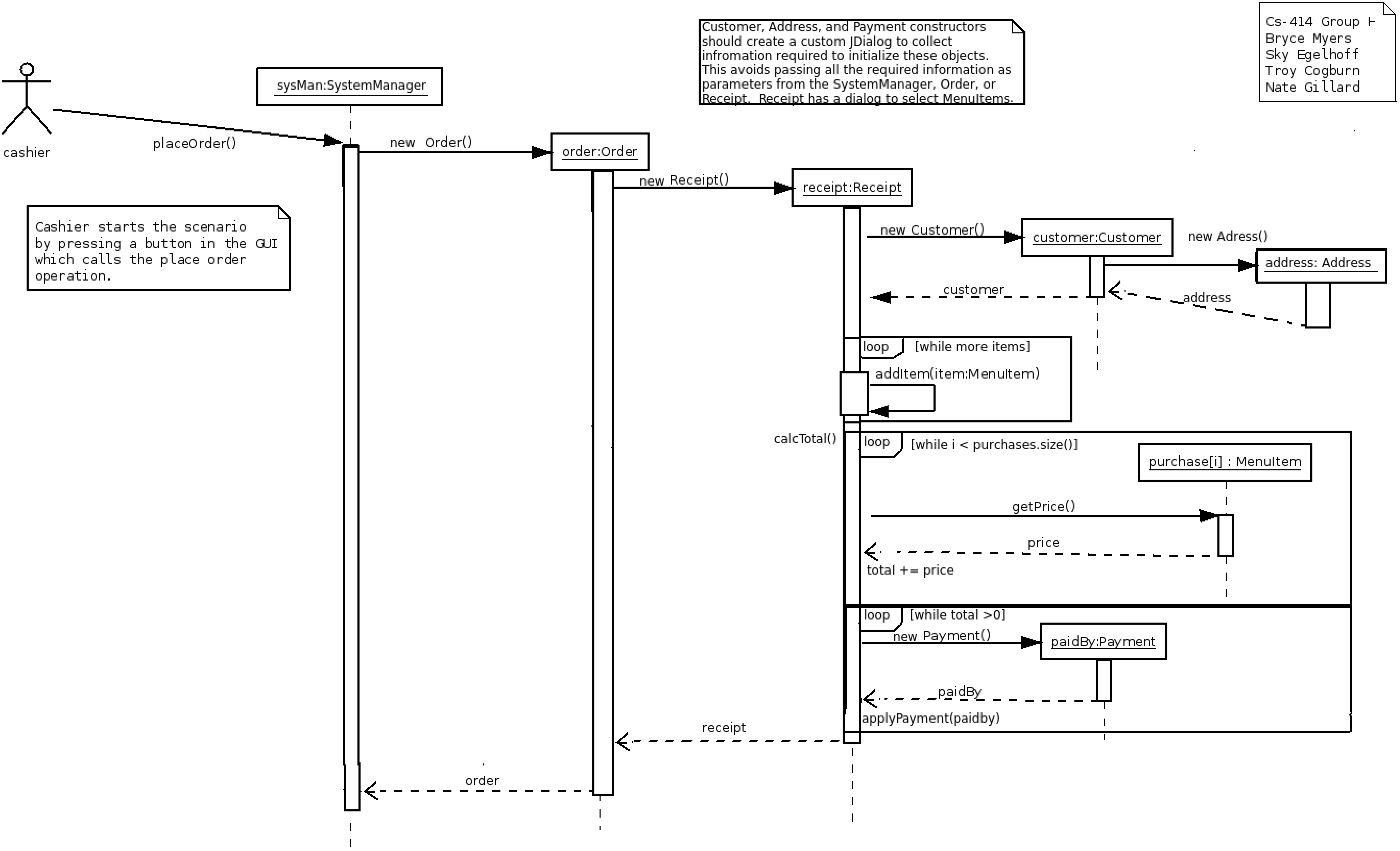
Employee presses a button on the GUI to log in,

getText() collects input from Swing components


Cs-414 Group F
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard

Cashier starts the scenario by pressing a button in the GUI which calls the place order operation.

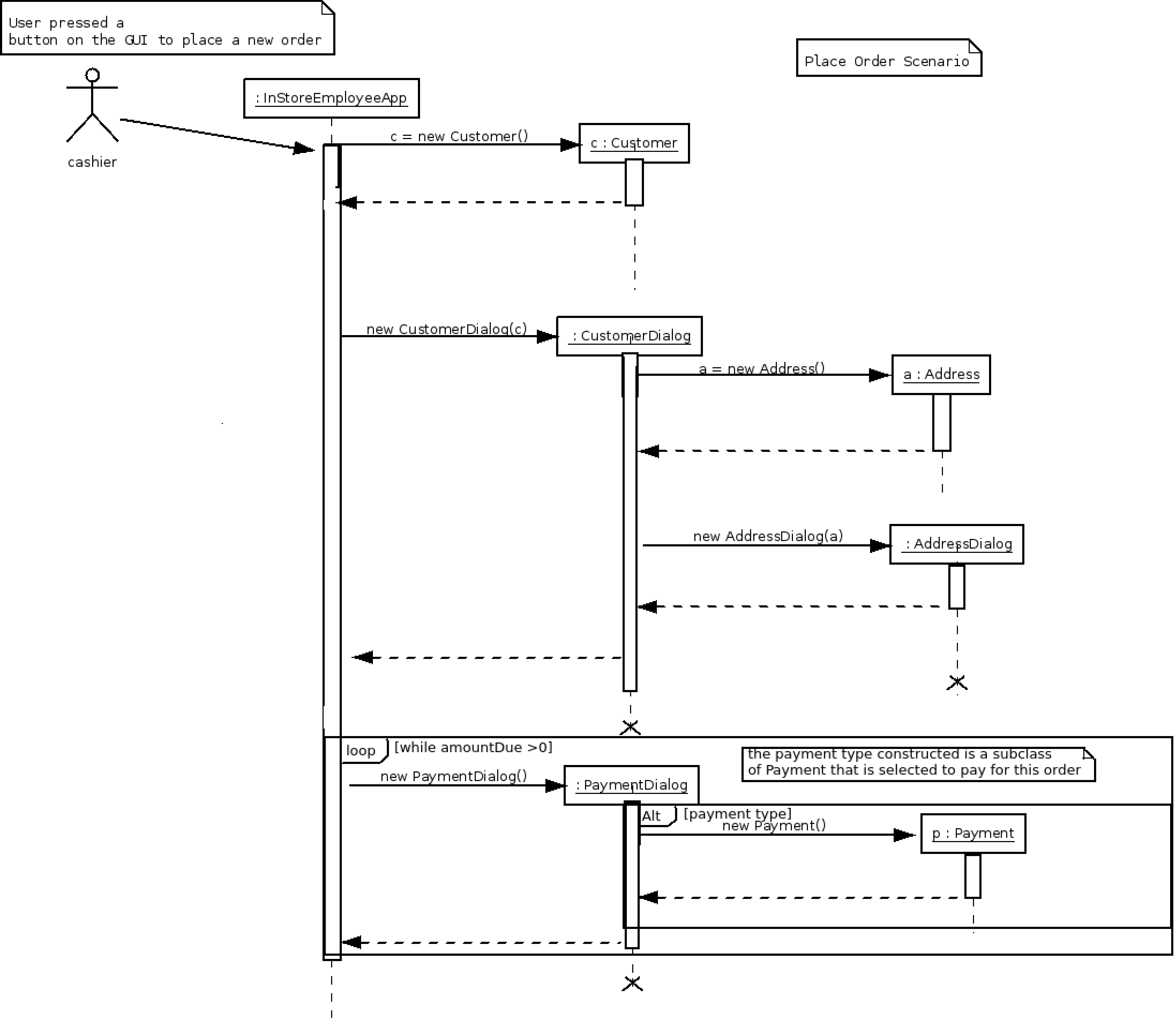
Customer, Address, and Payment constructors should create a custom JDialog to collect infromation required to initialize these objects. This avoids passing all the required information as parameters from the SystemManager, Order, or Receipt. Receipt has a dialog to select MenuItems.



Cs- 414 Group F
Bryce Myers
Sky Egelhoff
Troy Cogburn
Nate Gillard

Customer, Address, and Payment constructors should create a custom JDialog to collect infromation required to initialize these objects. This avoids passing all the required information as parameters from the SystemManager, Order, or Receipt.  Receipt has a dialog to select MenuItems.

Cashier starts the scenario by pressing a button in the GUI which calls the place order operation.


User pressed a
button on the GUI to place a new order
Place Order Scenario
cashier
: InStoreEmployeeApp
c = new Customer()
c : Customer
new CustomerDialog(c)
: CustomerDialog
a = new Address()
a : Address
new AddressDialog(a)
: AddressDialog
loop
[while amountDue >0]
the payment type constructed is a subclass
of Payment that is selected to pay for this order
new PaymentDialog()
: PaymentDialog
Alt
[payment type]
new Payment()
p : Payment