Static Models

«application logic» AuthorizationCenterUl

authorizeCard(cardNumber: long, pin: int, cost: double): int

Authorizes card transactions requested by the transaction manager.

«user interface» CustomerUl

-co: CustomerOrder -cardNumber: Long

-pin: int

#CheckoutOrder(i: CustomerOrder): void

#PromptToScan(): void #ScanItem(i: int): void #UpdateOrderDisplay(): void

#PrintTotal(): void

#PromptForPaymentType(): void #SelectPaymentType(): void #ShowSubtotal(): void #SendAuthorizationRequest()

#NewScreen(): void

#ReportToTransManager(CustomerOrder: List<Item>): void

This class resides on the checkout station itself, and communicates with the CustomerInterface.

uses ►

1 1

«entity» CustomerOrder {Some Properties}

-ClassAttribute: List<InventoryItem>

#getItemAtIndex(i: int): InventoryItem #size(): int

#getSubtotal(): double

This class is a list of type InventoryItem whose contents are used throughout the system for inventory management and receipt printing. Reset for each customer.

«application logic» InventoryManager inventory: List<InventoryItem> «entity» inventory Levels: List < Stocking Level> StockingLevel inventory MessageReport: -id: int List<InventoryMessage> -aty: int PrintDailvReport(): void :msgThresh: int UpdateInventory(): void Keeps track of threshold and qty Keeps track of the items carried of item by the store, how many of each have been sold, and how many of each remain. → includes includes ► 0..* 0..* «entity» «entity» InventoryItem InventoryMessage id: int itemId: int name: String Denotes an item price: float whose stock is discount: float getting low. Describes an item carried by the store.

This class interacts with the system to perform the functions in the Manager class «entity» Printer

#Print(s: string): int

Printer object sends signal to printer to print, and returns success/error code

«user interface» RestockerUI

#PromptForScan(): void #RestockItem(i: int): int #UpdateQuantity(i: int): void

#AddNewItem(i: InventoryItem): void

This class initiates the restock use case associated with the actor

Restocker

This class interacts with the rest of the system to fulfill the restock use case

«control» Timer

#PrintDailyReport(): void #PrintInventoryMessage(): void

At midnight, interacts with InventoryManager and TransactionManager to print inventory message and daily report

«application logic» TransactionManager

-transLog: TransactionLog

#GenerateAndPrintTransactionReport(s: string): int

Generates and prints transaction reports.

object: TransactionLog

id: Date

Daily list of all customer transactions.

has ►

object: Transaction

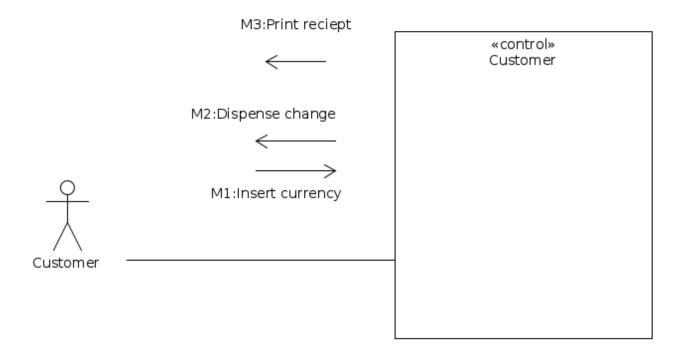
item:int price:float quantity:int

Historic record of a customer transaction, including items purchased, their prices and the quantity purchased.

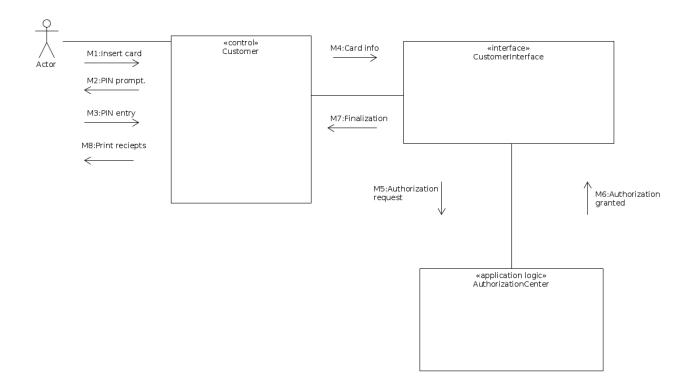
Common Checkout Order 7. Press total 3. Scan item 1. Request check out **Dynamic Models** Customer Interface 2. Prompt items 6. Display item info and prompt more item 4. Request Item Info 5. Return item info 8. Return total price and prompt payment method Inventory Subtotal Customer Interface 7. Press subtotal Customer 8. Return current total price and continue prompting for more item Cancel Checkout 3. Cancel Customer Interface Customer 4. Return to the idle state Cancel Payment 9. Cancel Payment Customer Interface Customer 10. Prompt Payment again Send Inventory Message 3. Scan item Customer Customer 7. Display item info and prompt more item 4. Request Item Info 6. Return item info Inventory Printer 5. Send message 5. Send message

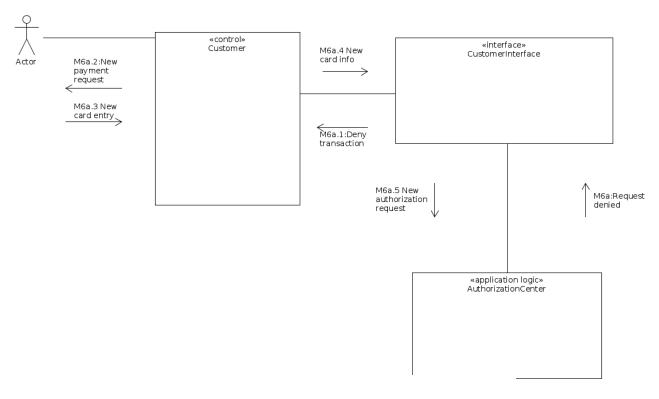
> System Message Buffer

Pay By Cash use case

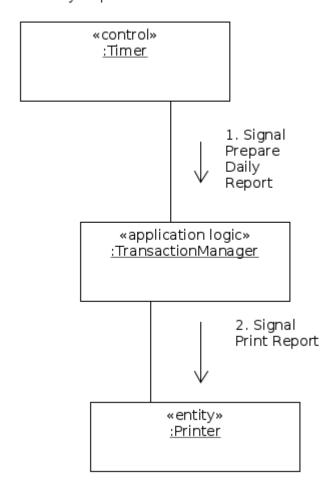


Pay by credit card(Approved) use case.

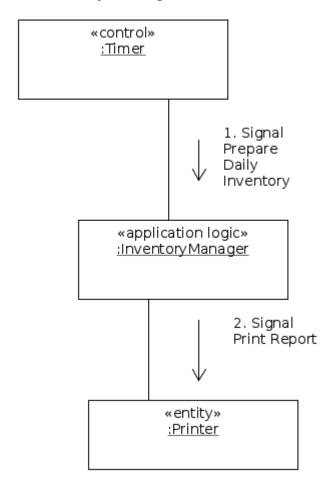




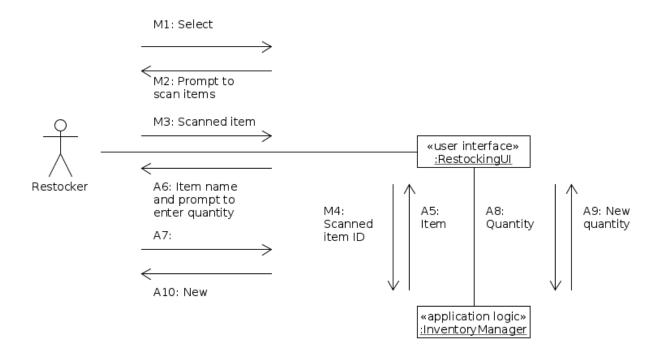
Print Daily Report Use Case



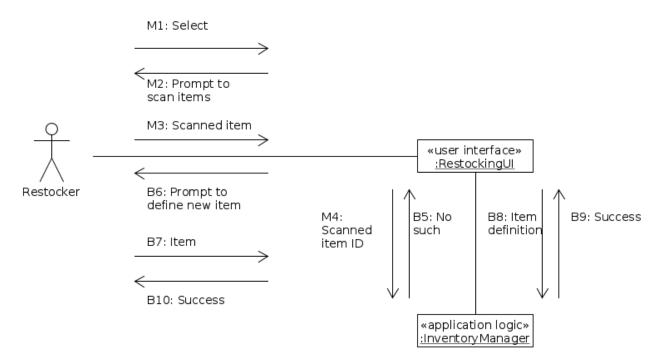
Print Inventory Message Use Case



Restock Inventory use case



Stock New Item use case



View or Update Inventory use case

