SENG 300 - Group Project, Iteration 3

Members	**UCID**	`git`
Ryan Allard	30085796	`HN67`
Seher Dawar	30087391	`SeherD`
Leinard Grande	30042415	`leinardjg`
Ehab Islam	30105153	`ehabislam`
Lubdhak Juyal	30045497	`monsoonssn`
Christopher Kozbial	30061532	`meebee70`
Suvrat Kumra	30089629	`suvratkumra`
Dongheok Lee	30041784	`HeDoLe`
Htet Myet Aung Naing	30107013	`htetm`
Markus Pistner	30081575	`MarkusP1`
Shahab Salem	30079334	`ShehabSN`
Abhay Sharma	30095410	`Abhay189`
Kyle Wen	30061704	`Kiiii01`
Yuzhe Zhou	30102199	`Zhezhee`

Testing Setup

In order for the testing project to be run as a JUnit test, three dependencies are required, as described in the `.classpath` file.

Namely,

`SelfCheckoutSystem - Hardware - v3.1`,

`SelfCheckoutSystem - Software`,

and JUnit 4

are required dependencies.

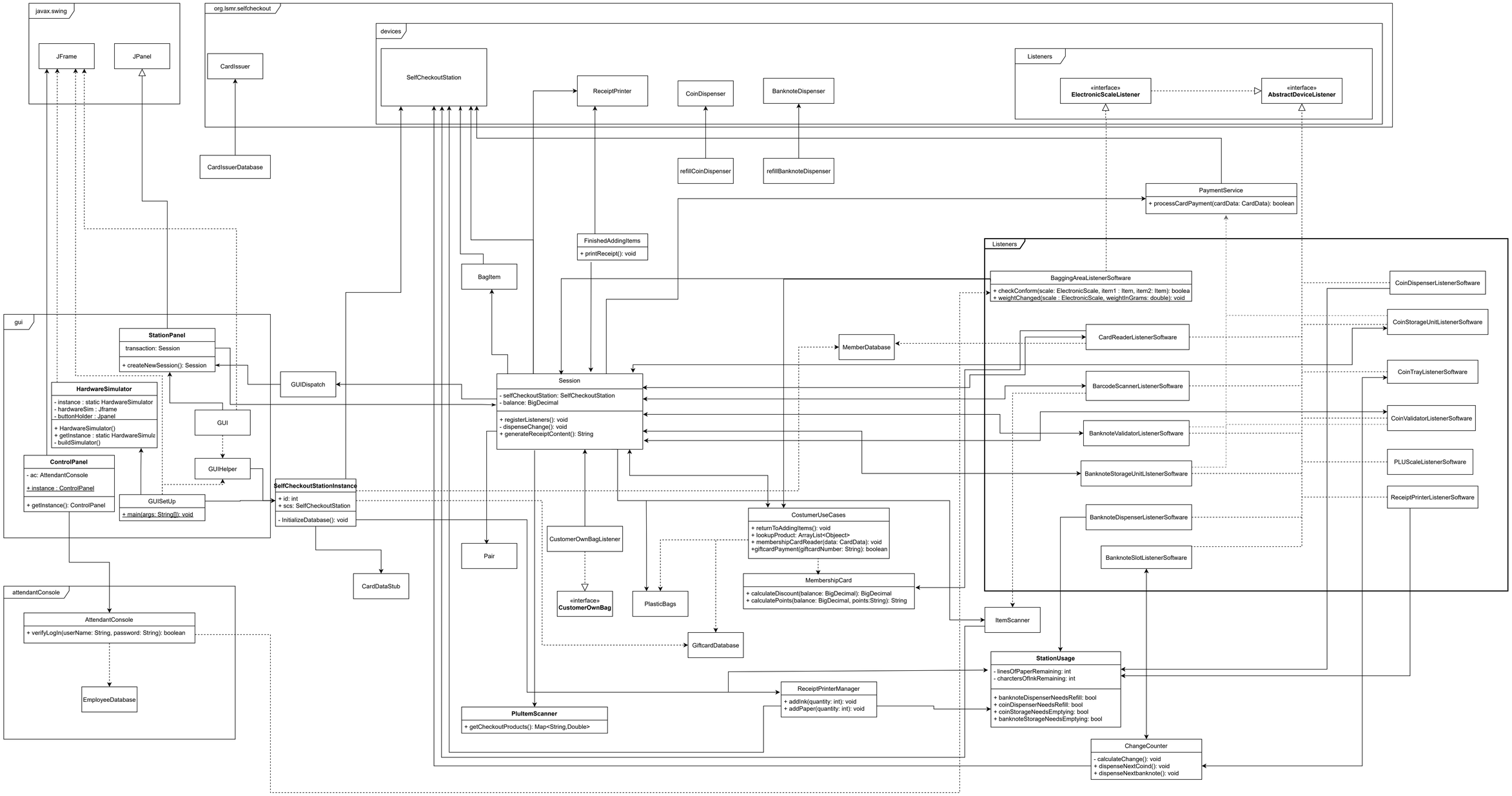
Project Description

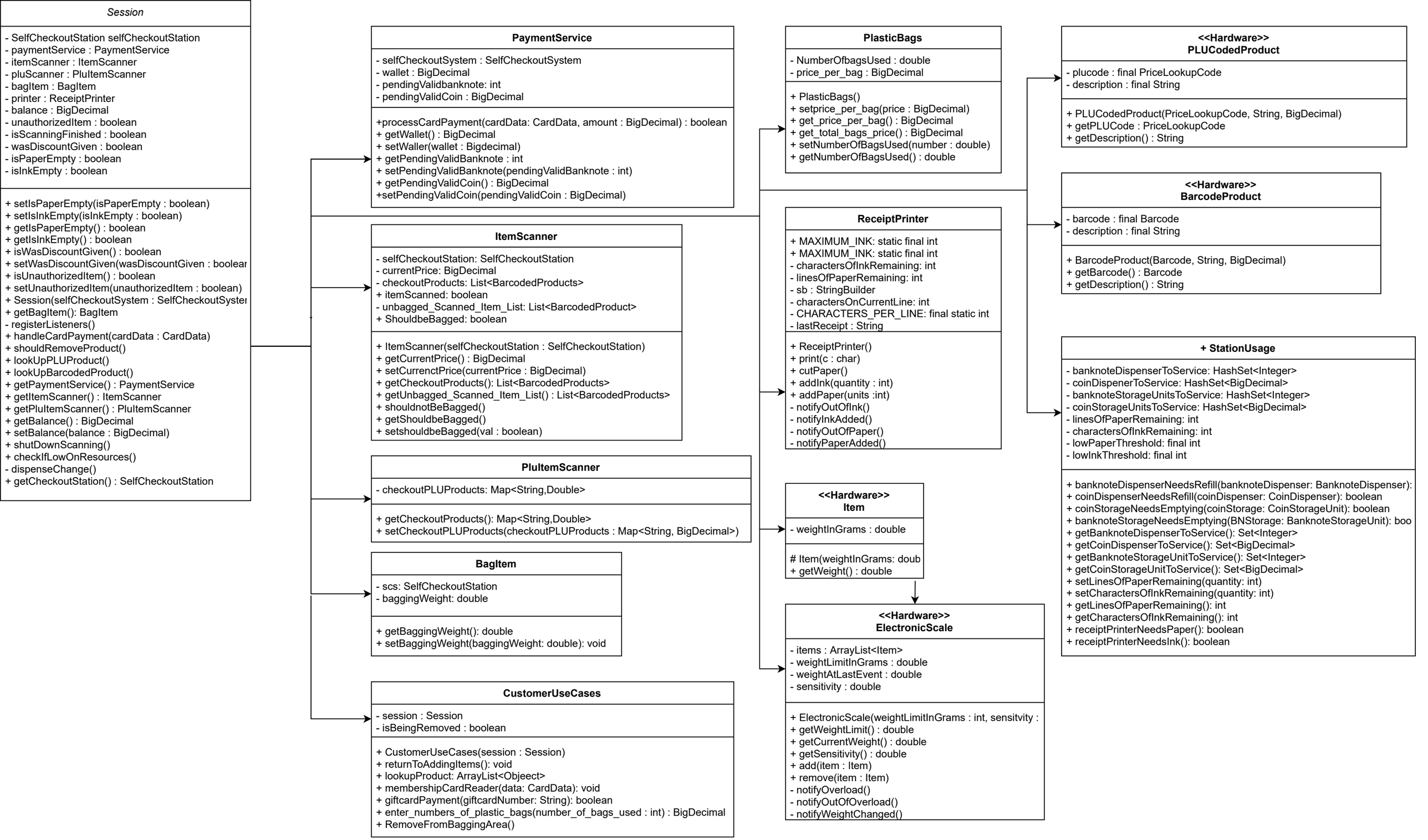
Our simulator has three main sections; an attendant console, station(s), and hardware simulator.

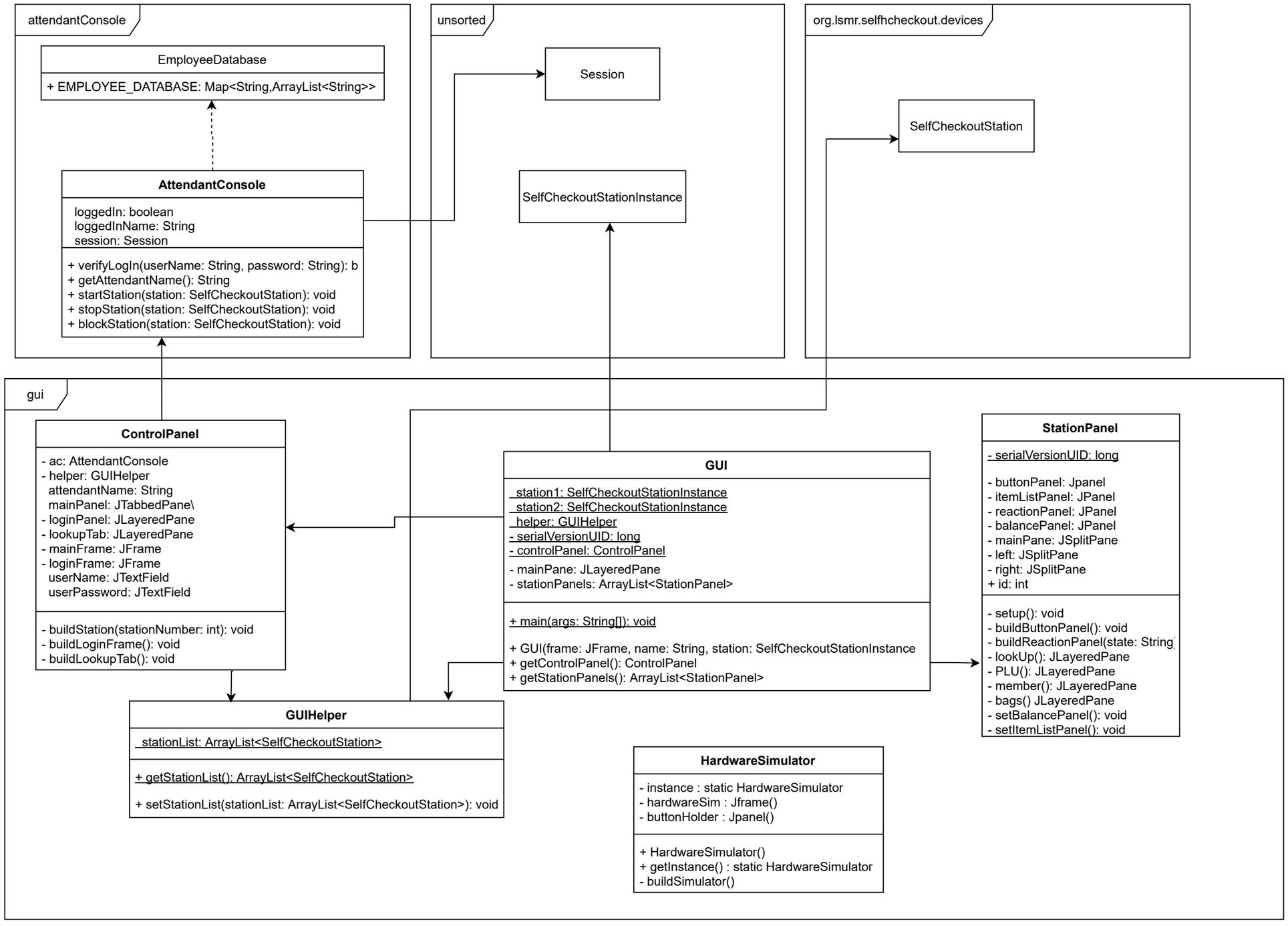
Once a user starts a transaction on a station, they can perform a variety of actions, such as adding their own bags, looking up products, entering a PLU, buying bags, and paging the attendent for help. Most of this functionality is implemented via buttons that toggle between different panels with inputs.

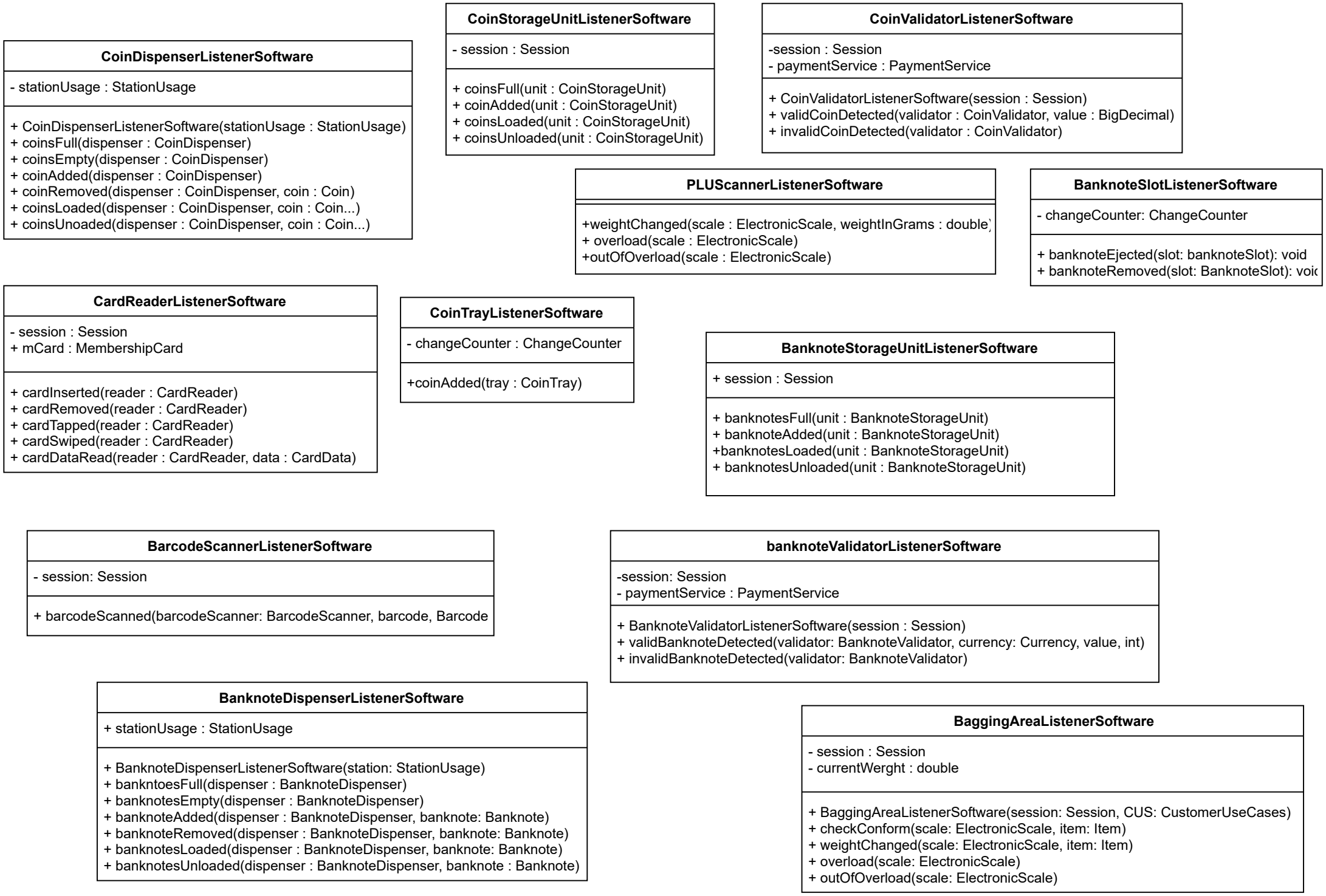
The hardware simulator has functionality to simulate scanning items, adding items to the bagging area, adding coins and banknotes, and inserting a card.

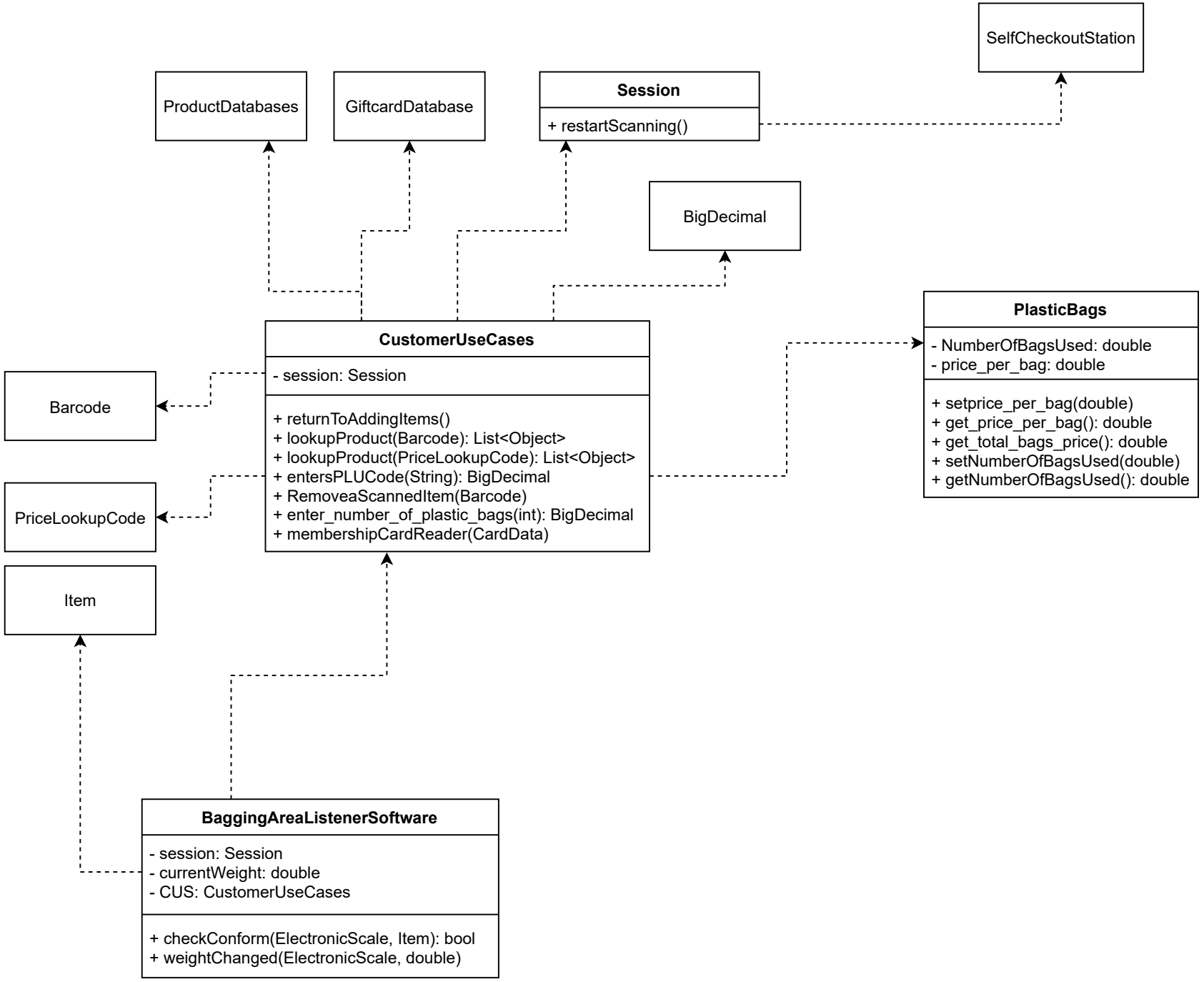
The attendent console can start and shutdown stations, as well as block them.

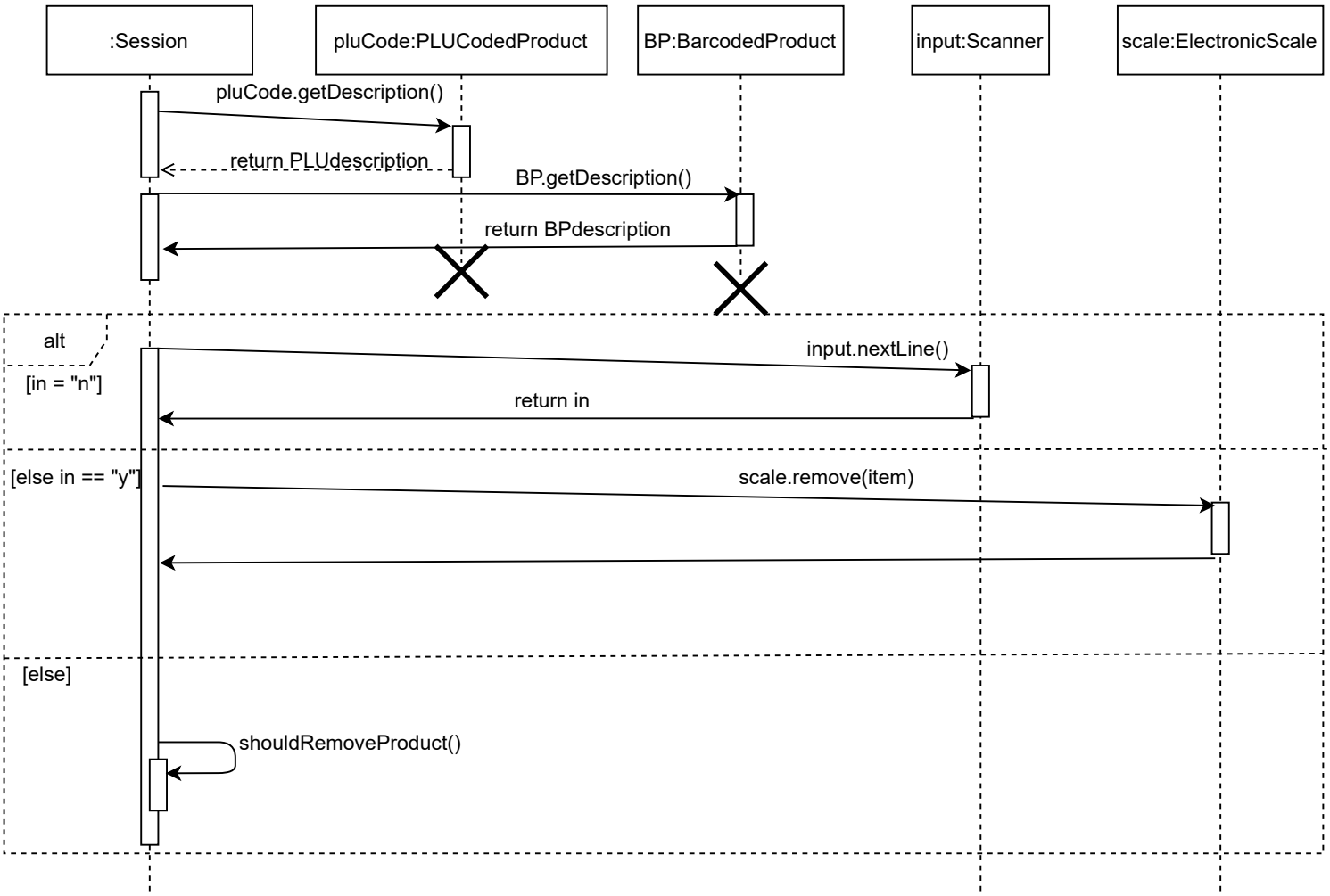

javax.swing
JFrame
JPanel
org.lsmr.selfcheckout
devices
SelfCheckoutStation
ReceiptPrinter
CoinDispenser
BanknoteDispenser
Listeners
«interface»
ElectronicScaleListener
«interface»
AbstractDeviceListener
CardIssuer
CardIssuerDatabase
refillCoinDispenser
refillBanknoteDispenser
PaymentService
+ processCardPayment(cardData: CardData): boolean
FinishedAddingItems
+ printReceipt(): void
BagItem
Listeners
BaggingAreaListenerSoftware
+ checkConform(scale: ElectronicScale, item1 : Item, item2: Item): boolea
+ weightChanged(scale : ElectronicScale, weightInGrams: double): void
CoinDispenserListenerSoftware
CoinStorageUnitListenerSoftware
CoinTrayListenerSoftware
CoinValidatorListenerSoftware
PLUScaleListenerSoftware
ReceiptPrinterListenerSoftware
BanknoteDispenserListenerSoftware
BanknoteSlotListenerSoftware
BanknoteStorageUnitLlstenerSoftware
BanknoteValidatorListenerSoftware
CardReaderListenerSoftware
BarcodeScannerListenerSoftware
gui
StationPanel
transaction: Session
+ createNewSession(): Session
GUIDispatch
MemberDatabase
HardwareSimulator
- instance : static HardwareSimulator
- hardwareSim : Jframe
- buttonHolder : Jpanel
+ HardwareSimulator()
+ getInstance : static HardwareSimula
- buildSimulator()
GUI
GUIHelper
ControlPanel
- ac: AttendantConsole
+ instance : ControlPanel
+ getInstance(): ControlPanel
GUISetUp
+ main(args: String[]): void
SelfCheckoutStationInstance
+ id: int
+ scs: SelfCheckoutStation
- InitializeDatabase(): void
Session
- selfCheckoutStation: SelfCheckoutStation
- balance: BigDecimal
+ registerListeners(): void
- dispenseChange(): void
+ generateReceiptContent(): String
CostumerUseCases
+ returnToAddingItems(): void
+ lookupProduct: ArrayList<Objeect>
+ membershipCardReader(data: CardData): void
+giftcardPayment(giftcardNumber: String): boolean
CustomerOwnBagListener
Pair
«interface»
CustomerOwnBag
PlasticBags
MembershipCard
+ calculateDiscount(balance: BigDecimal): BigDecimal
+ calculatePoints(balance: BigDecimal, points:String): String
ItemScanner
GiftcardDatabase
CardDataStub
attendantConsole
AttendantConsole
+ verifyLogIn(userName: String, password: String): boolean
EmployeeDatabase
StationUsage
- linesOfPaperRemaining: int
- charctersOfInkRemaining: int
+ banknoteDispenserNeedsRefill: bool
+ coinDispenserNeedsRefill: bool
+ coinStorageNeedsEmptying: bool
+ banknoteStorageNeedsEmptying: bool
ReceiptPrinterManager
+ addInk(quantity: int): void
+ addPaper(quantity: int): void
PluItemScanner
+ getCheckoutProducts(): Map<String,Double>
ChangeCounter
- calculateChange(): void
+ dispenseNextCoind(): void
+ dispenseNextbanknote(): void

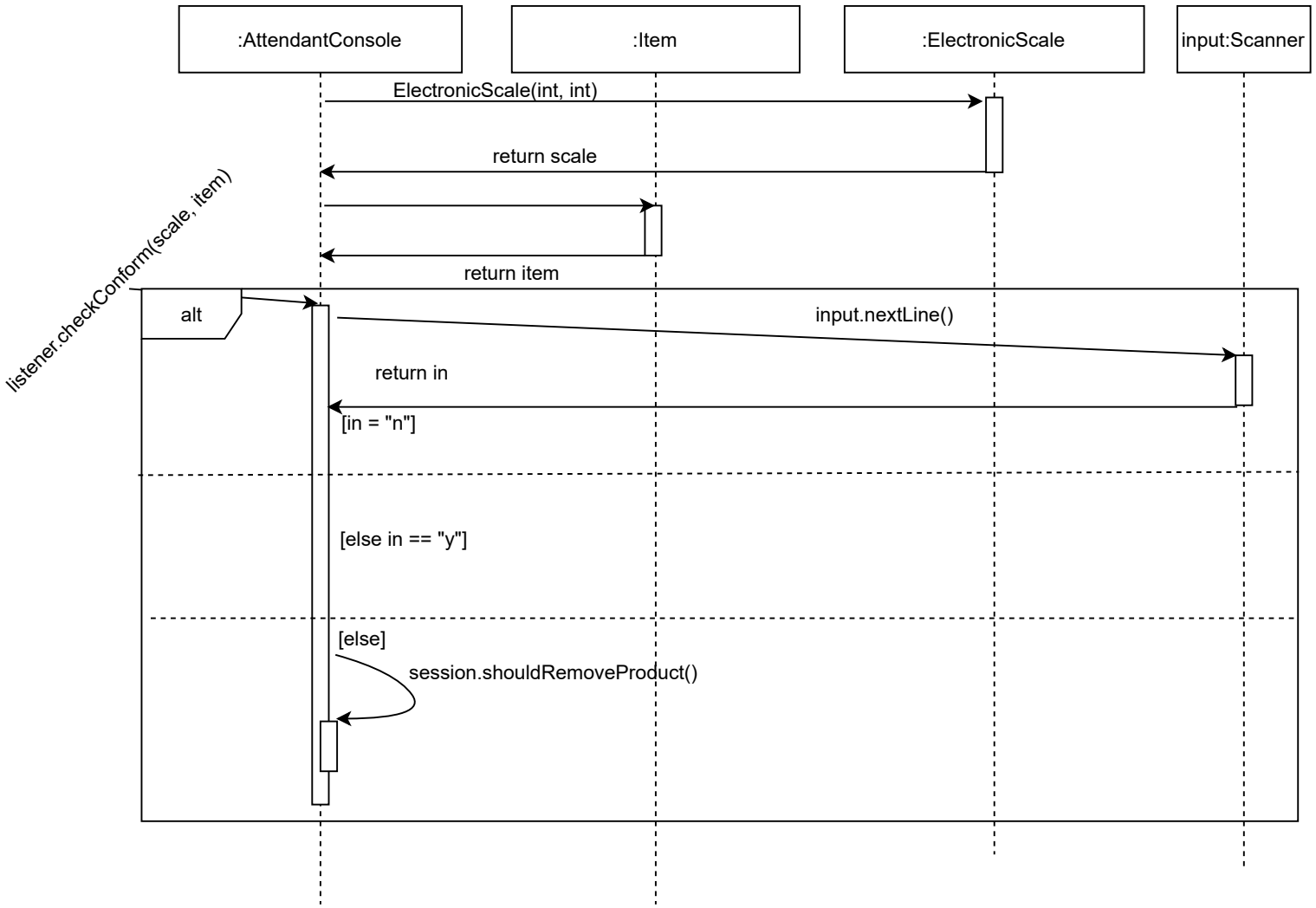

Session
- SelfCheckoutStation selfCheckoutStation
- paymentService : PaymentService
- itemScanner : ItemScanner
- pluScanner : PluItemScanner
- bagItem : BagItem
- printer : ReceiptPrinter
- balance : BigDecimal
- unauthorizedItem : boolean
- isScanningFinished : boolean
- wasDiscountGiven : boolean
- isPaperEmpty : boolean
- isInkEmpty : boolean
+ setIsPaperEmpty(isPaperEmpty : boolean)
+ setIsInkEmpty(isInkEmpty : boolean)
+ getIsPaperEmpty() : boolean
+ getIsInkEmpty() : boolean
+ isWasDiscountGiven() : boolean
+ setWasDiscountGiven(wasDiscountGiven : boolear
+ isUnauthorizedItem() : boolean
+ setUnauthorizedItem(unauthorizedItem : boolean)
+ Session(selfCheckoutSystem : SelfCheckoutSyste
+ getBagItem(): BagItem
- registerListeners()
+ handleCardPayment(cardData : CardData)
+ shouldRemoveProduct()
+ lookUpPLUProduct()
+ lookUpBarcodedProduct()
+ getPaymentService() : PaymentService
+ getItemScanner() : ItemScanner
+ getPluItemScanner() : PluItemScanner
+ getBalance() : BigDecimal
+ setBalance(balance : BigDecimal)
+ shutDownScanning()
+ checkIfLowOnResources()
- dispenseChange()
+ getCheckoutStation() : SelfCheckoutStation
PaymentService
- selfCheckoutSystem : SelfCheckoutSystem
- wallet : BigDecimal
- pendingValidbanknote: int
- pendingValidCoin : BigDecimal
+processCardPayment(cardData: CardData, amount : BigDecimal) : boolean
+ getWallet() : BigDecimal
+ setWaller(wallet : Bigdecimal)
+ getPendingValidBanknote : int
+ setPendingValidBanknote(pendingValidBanknote : int)
+ getPendingValidCoin() : BigDecimal
+setPendingValidCoin(pendingValidCoin : BigDecimal)
ItemScanner
- selfCheckoutStation: SelfCheckoutStation
- currentPrice: BigDecimal
- checkoutProducts: List<BarcodedProducts>
+ itemScanned: boolean
- unbagged_Scanned_Item_List: List<BarcodedProduct>
+ ShouldbeBagged: boolean
+ ItemScanner(selfCheckoutStation : SelfCheckoutStation)
+ getCurrentPrice() : BigDecimal
+ setCurrenctPrice(currentPrice : BigDecimal)
+ getCheckoutProducts(): List<BarcodedProducts>
+ getUnbagged_Scanned_Item_List() : List<BarcodedProducts>
+ shouldnotBeBagged()
+ getShouldbeBagged()
+ setshouldbeBagged(val : boolean)
PluItemScanner
- checkoutPLUProducts: Map<String,Double>
+ getCheckoutProducts(): Map<String,Double>
+ setCheckoutPLUProducts(checkoutPLUProducts : Map<String, BigDecimal>)
BagItem
- scs: SelfCheckoutStation
- baggingWeight: double
+ getBaggingWeight(): double
+ setBaggingWeight(baggingWeight: double): void
CustomerUseCases
- session : Session
- isBeingRemoved : boolean
+ CustomerUseCases(session : Session)
+ returnToAddingItems(): void
+ lookupProduct: ArrayList<Objeect>
+ membershipCardReader(data: CardData): void
+ giftcardPayment(giftcardNumber: String): boolean
+ enter_numbers_of_plastic_bags(number_of_bags_used : int) : BigDecimal
+ RemoveFromBaggingArea()
PlasticBags
- NumberOfbagsUsed : double
- price_per_bag : BigDecimal
+ PlasticBags()
+ setprice_per_bag(price : BigDecimal)
+ get_price_per_bag() : BigDecimal
+ get_total_bags_price() : BigDecimal
+ setNumberOfBagsUsed(number : double)
+ getNumberOfBagsUsed() : double
ReceiptPrinter
+ MAXIMUM_INK: static final int
+ MAXIMUM_INK: static final int
- charactersOfInkRemaining: int
- linesOfPaperRemaining: int
- sb : StringBuilder
- charactersOnCurrentLine: int
- CHARACTERS_PER_LINE: final static int
- lastReceipt : String
+ ReceiptPrinter()
+ print(c : char)
+ cutPaper()
+ addInk(quantity : int)
+ addPaper(units :int)
- notifyOutOfInk()
- notifyInkAdded()
- notifyOutOfPaper()
- notifyPaperAdded()
<<Hardware>>
Item
- weightInGrams : double
Item(weightInGrams: doub
+ getWeight() : double
<<Hardware>>
ElectronicScale
- items : ArrayList<Item>
- weightLimitInGrams : double
- weightAtLastEvent : double
- sensitivity : double
+ ElectronicScale(weightLimitInGrams : int, sensitvity :
+ getWeightLimit() : double
+ getCurrentWeight() : double
+ getSensitivity() : double
+ add(item : Item)
+ remove(item : Item)
- notifyOverload()
- notifyOutOfOverload()
- notifyWeightChanged()
<<Hardware>>
PLUCodedProduct
- plucode : final PriceLookupCode
- description : final String
+ PLUCodedProduct(PriceLookupCode, String, BigDecimal)
+ getPLUCode : PriceLookupCode
+ getDescription() : String
<<Hardware>>
BarcodeProduct
- barcode : final Barcode
- description : final String
+ BarcodeProduct(Barcode, String, BigDecimal)
+ getBarcode() : Barcode
+ getDescription() : String
+ StationUsage
- banknoteDispenserToService: HashSet<Integer>
- coinDispenerToService: HashSet<BigDecimal>
- banknoteStorageUnitsToService: HashSet<Integer>
- coinStorageUnitsToService: HashSet<BigDecimal>
- linesOfPaperRemaining: int
- charactersOfInkRemaining: int
- lowPaperThreshold: final int
- lowInkThreshold: final int
+ banknoteDispenserNeedsRefill(banknoteDispenser: BanknoteDispenser):
+ coinDispenserNeedsRefill(coinDispenser: CoinDispenser): boolean
+ coinStorageNeedsEmptying(coinStorage: CoinStorageUnit): boolean
+ banknoteStorageNeedsEmptying(BNStorage: BanknoteStorageUnit): boo
+ getBanknoteDispenserToService(): Set<Integer>
+ getCoinDispenserToService(): Set<BigDecimal>
+ getBanknoteStorageUnitToService(): Set<Integer>
+ getCoinStorageUnitToService(): Set<BigDecimal>
+ setLinesOfPaperRemaining(quantity: int)
+ setCharactersOfInkRemaining(quantity: int)
+ getLinesOfPaperRemaining(): int
+ getCharactersOfInkRemaining(): int
+ receiptPrinterNeedsPaper(): boolean
+ receiptPrinterNeedsInk(): boolean

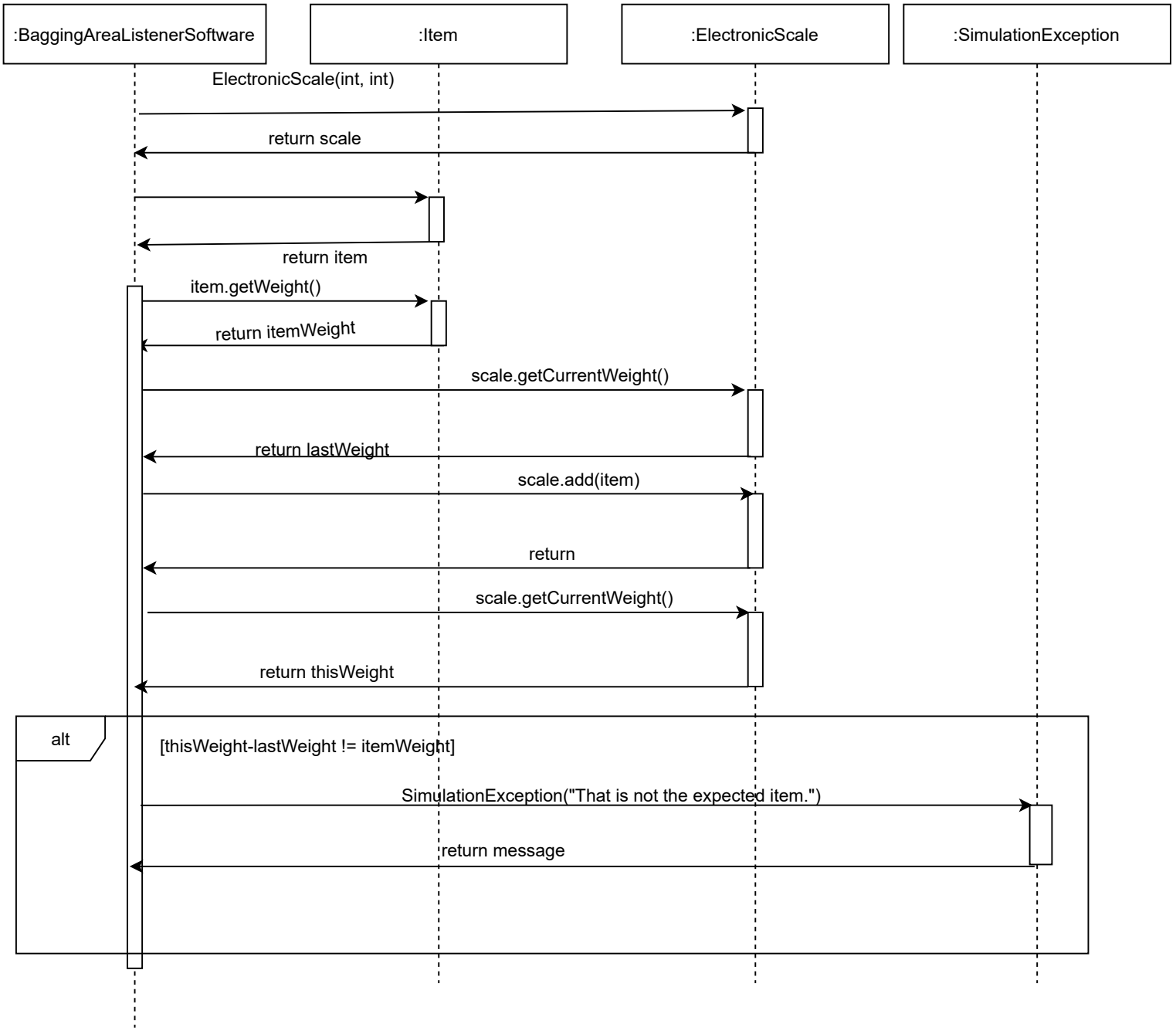

attendantConsole
EmployeeDatabase
+ EMPLOYEE_DATABASE: Map<String,ArrayList<String>>
AttendantConsole
loggedIn: boolean
loggedInName: String
session: Session
+ verifyLogIn(userName: String, password: String): b
+ getAttendantName(): String
+ startStation(station: SelfCheckoutStation): void
+ stopStation(station: SelfCheckoutStation): void
+ blockStation(station: SelfCheckoutStation): void
unsorted
Session
SelfCheckoutStationInstance
org.lsmr.selfhcheckout.devices
SelfCheckoutStation
gui
ControlPanel
- ac: AttendantConsole
- helper: GUIHelper
attendantName: String
mainPanel: JTabbedPane\
- loginPanel: JLayeredPane
- lookupTab: JLayeredPane
- mainFrame: JFrame
- loginFrame: JFrame
userName: JTextField
userPassword: JTextField
- buildStation(stationNumber: int): void
- buildLoginFrame(): void
- buildLookupTab(): void
GUI
station1: SelfCheckoutStationInstance
station2: SelfCheckoutStationInstance
helper: GUIHelper
- serialVersionUID: long
- controlPanel: ControlPanel
- mainPane: JLayeredPane
- stationPanels: ArrayList<StationPanel>
+ main(args: String[]): void
+ GUI(frame: JFrame, name: String, station: SelfCheckoutStationInstance
+ getControlPanel(): ControlPanel
+ getStationPanels(): ArrayList<StationPanel>
StationPanel
- serialVersionUID: long
- buttonPanel: Jpanel
- itemListPanel: JPanel
- reactionPanel: JPanel
- balancePanel: JPanel
- mainPane: JSplitPane
- left: JSplitPane
- right: JSplitPane
+ id: int
- setup(): void
- buildButtonPanel(): void
- buildReactionPanel(state: String
- lookUp(): JLayeredPane
- PLU(): JLayeredPane
- member(): JLayeredPane
- bags() JLayeredPane
- setBalancePanel(): void
- setItemListPanel(): void
GUIHelper
stationList: ArrayList<SelfCheckoutStation>
+ getStationList(): ArrayList<SelfCheckoutStation>
+ setStationList(stationList: ArrayList<SelfCheckoutStation>): void
HardwareSimulator
- instance : static HardwareSimulator
- hardwareSim : Jframe()
- buttonHolder : Jpanel()
+ HardwareSimulator()
+ getInstance() : static HardwareSimulator
- buildSimulator()

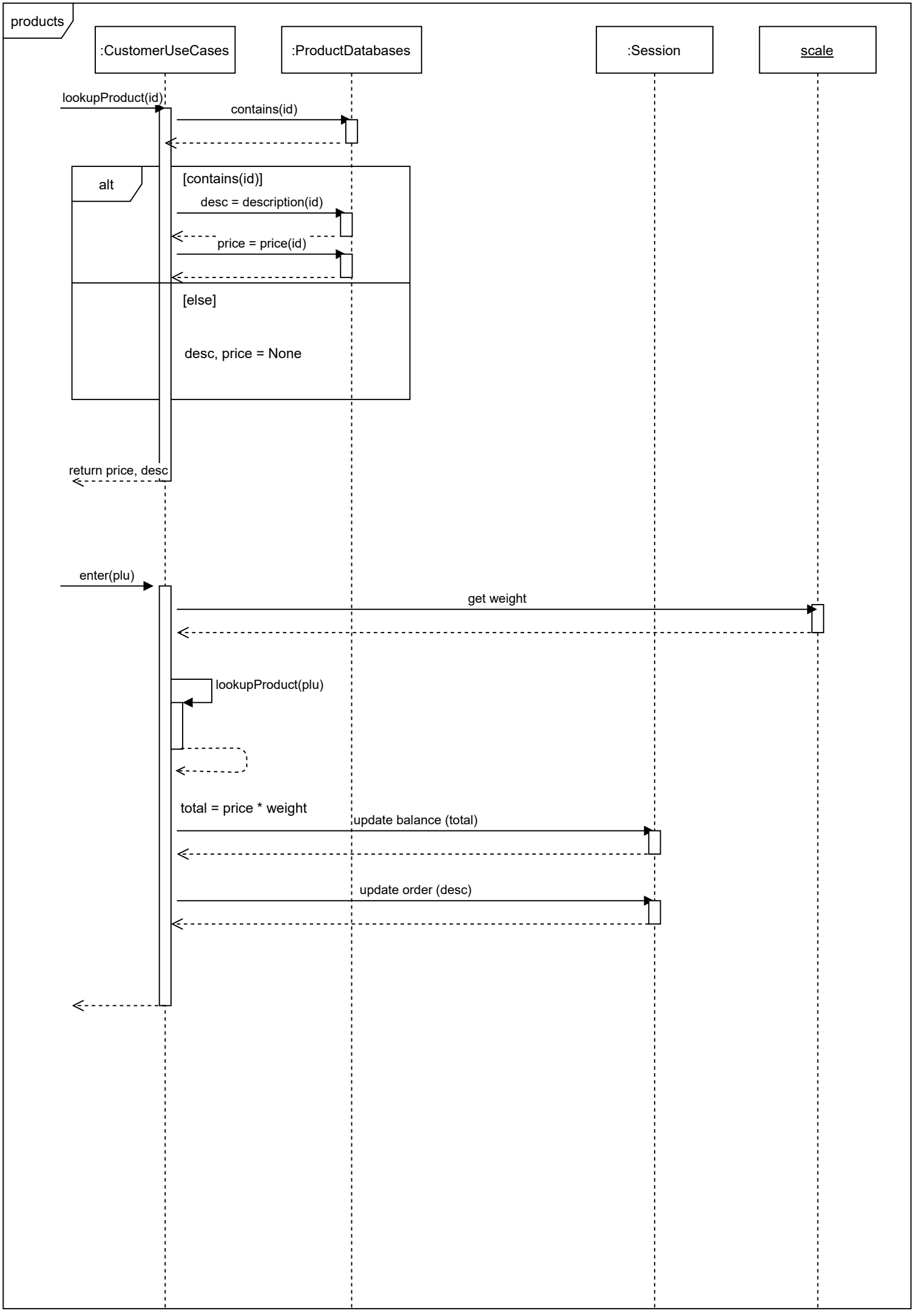

CoinDispenserListenerSoftware
- stationUsage : StationUsage
+ CoinDispenserListenerSoftware(stationUsage : StationUsage)
+ coinsFull(dispenser : CoinDispenser)
+ coinsEmpty(dispenser : CoinDispenser)
+ coinAdded(dispenser : CoinDispenser)
+ coinRemoved(dispenser : CoinDispenser, coin : Coin)
+ coinsLoaded(dispenser : CoinDispenser, coin : Coin...)
+ coinsUnoaded(dispenser : CoinDispenser, coin : Coin...)
CoinStorageUnitListenerSoftware
- session : Session
+ coinsFull(unit : CoinStorageUnit)
+ coinAdded(unit : CoinStorageUnit)
+ coinsLoaded(unit : CoinStorageUnit)
+ coinsUnloaded(unit : CoinStorageUnit)
CoinValidatorListenerSoftware
-session : Session
- paymentService : PaymentService
+ CoinValidatorListenerSoftware(session : Session)
+ validCoinDetected(validator : CoinValidator, value : BigDecimal)
+ invalidCoinDetected(validator : CoinValidator)
PLUScannerListenerSoftware
+weightChanged(scale : ElectronicScale, weightInGrams : double)
+ overload(scale : ElectronicScale)
+outOfOverload(scale : ElectronicScale)
BanknoteSlotListenerSoftware
- changeCounter: ChangeCounter
+ banknoteEjected(slot: banknoteSlot): void
+ banknoteRemoved(slot: BanknoteSlot): voic
CardReaderListenerSoftware
- session : Session
+ mCard : MembershipCard
+ cardInserted(reader : CardReader)
+ cardRemoved(reader : CardReader)
+ cardTapped(reader : CardReader)
+ cardSwiped(reader : CardReader)
+ cardDataRead(reader : CardReader, data : CardData)
CoinTrayListenerSoftware
- changeCounter : ChangeCounter
+coinAdded(tray : CoinTray)
BanknoteStorageUnitListenerSoftware
+ session : Session
+ banknotesFull(unit : BanknoteStorageUnit)
+ banknoteAdded(unit : BanknoteStorageUnit)
+banknotesLoaded(unit : BanknoteStorageUnit)
+ banknotesUnloaded(unit : BanknoteStorageUnit)
BarcodeScannerListenerSoftware
- session: Session
+ barcodeScanned(barcodeScanner: BarcodeScanner, barcode, Barcode
banknoteValidatorListenerSoftware
-session: Session
- paymentService : PaymentService
+ BanknoteValidatorListenerSoftware(session : Session)
+ validBanknoteDetected(validator: BanknoteValidator, currency: Currency, value, int)
+ invalidBanknoteDetected(validator: BanknoteValidator)
BanknoteDispenserListenerSoftware
+ stationUsage : StationUsage
+ BanknoteDispenserListenerSoftware(station: StationUsage)
+ bankntoesFull(dispenser : BanknoteDispenser)
+ banknotesEmpty(dispenser : BanknoteDispenser)
+ banknoteAdded(dispenser : BanknoteDispenser, banknote: Banknote)
+ banknoteRemoved(dispenser : BanknoteDispenser, banknote: Banknote)
+ banknotesLoaded(dispenser : BanknoteDispenser, banknote: Banknote)
+ banknotesUnloaded(dispenser : BanknoteDispenser, banknote : Banknote)
BaggingAreaListenerSoftware
- session : Session
- currentWerght : double
+ BaggingAreaListenerSoftware(session: Session, CUS: CustomerUseCases)
+ checkConform(scale: ElectronicScale, item: Item)
+ weightChanged(scale: ElectronicScale, item: Item)
+ overload(scale: ElectronicScale)
+ outOfOverload(scale: ElectronicScale)

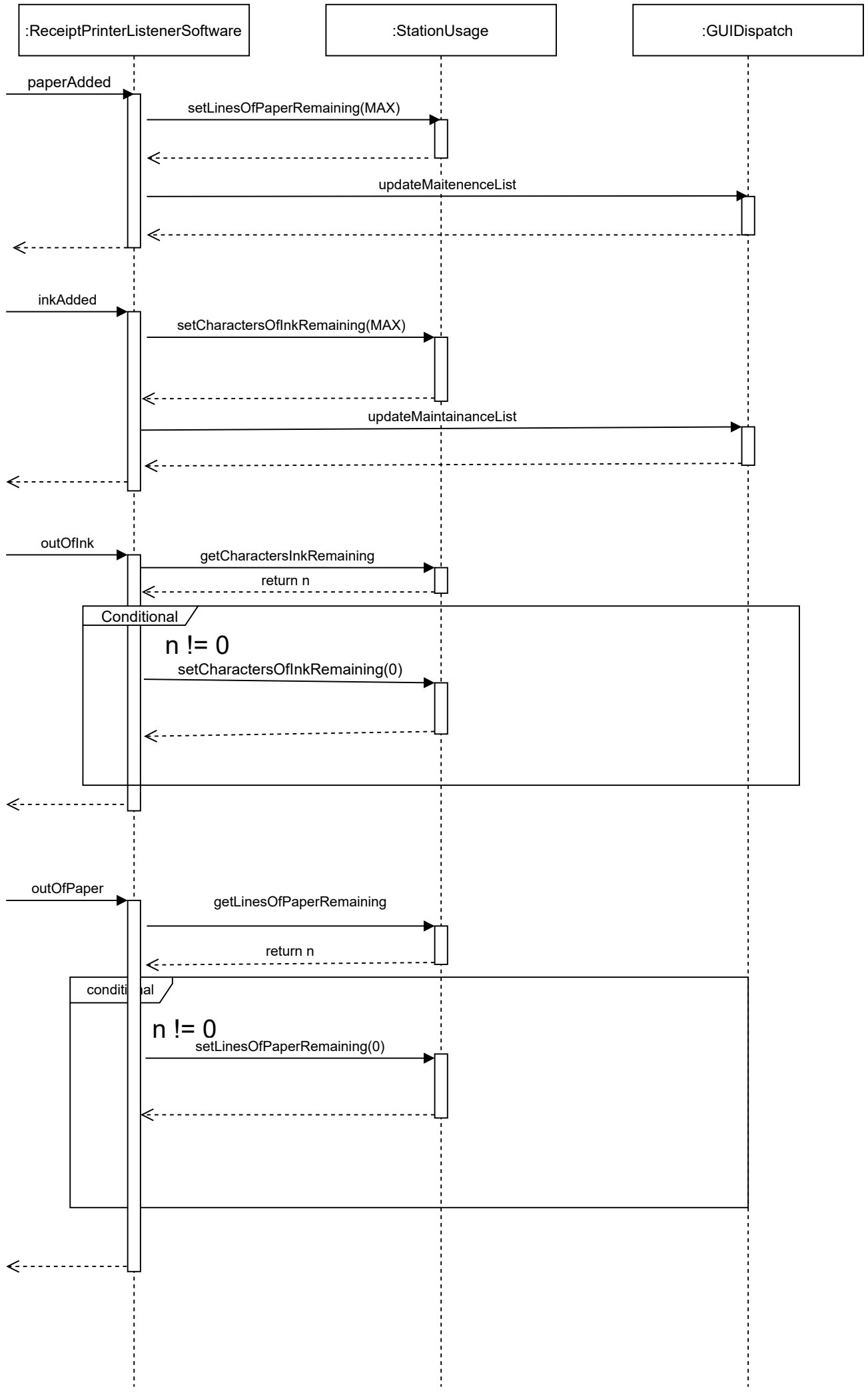

SelfCheckoutStation
ProductDatabases
GiftcardDatabase
Session
+ restartScanning()
BigDecimal
PlasticBags
- NumberOfBagsUsed: double
- price_per_bag: double
+ setprice_per_bag(double)
+ get_price_per_bag(): double
+ get_total_bags_price(): double
+ setNumberOfBagsUsed(double)
+ getNumberOfBagsUsed(): double
CustomerUseCases
- session: Session
+ returnToAddingItems()
+ lookupProduct(Barcode): List<Object>
+ lookupProduct(PriceLookupCode): List<Object>
+ entersPLUCode(String): BigDecimal
+ RemoveaScannedItem(Barcode)
+ enter_number_of_plastic_bags(int): BigDecimal
+ membershipCardReader(CardData)
Barcode
PriceLookupCode
Item
BaggingAreaListenerSoftware
- session: Session
- currentWeight: double
- CUS: CustomerUseCases
+ checkConform(ElectronicScale, Item): bool
+ weightChanged(ElectronicScale, double)

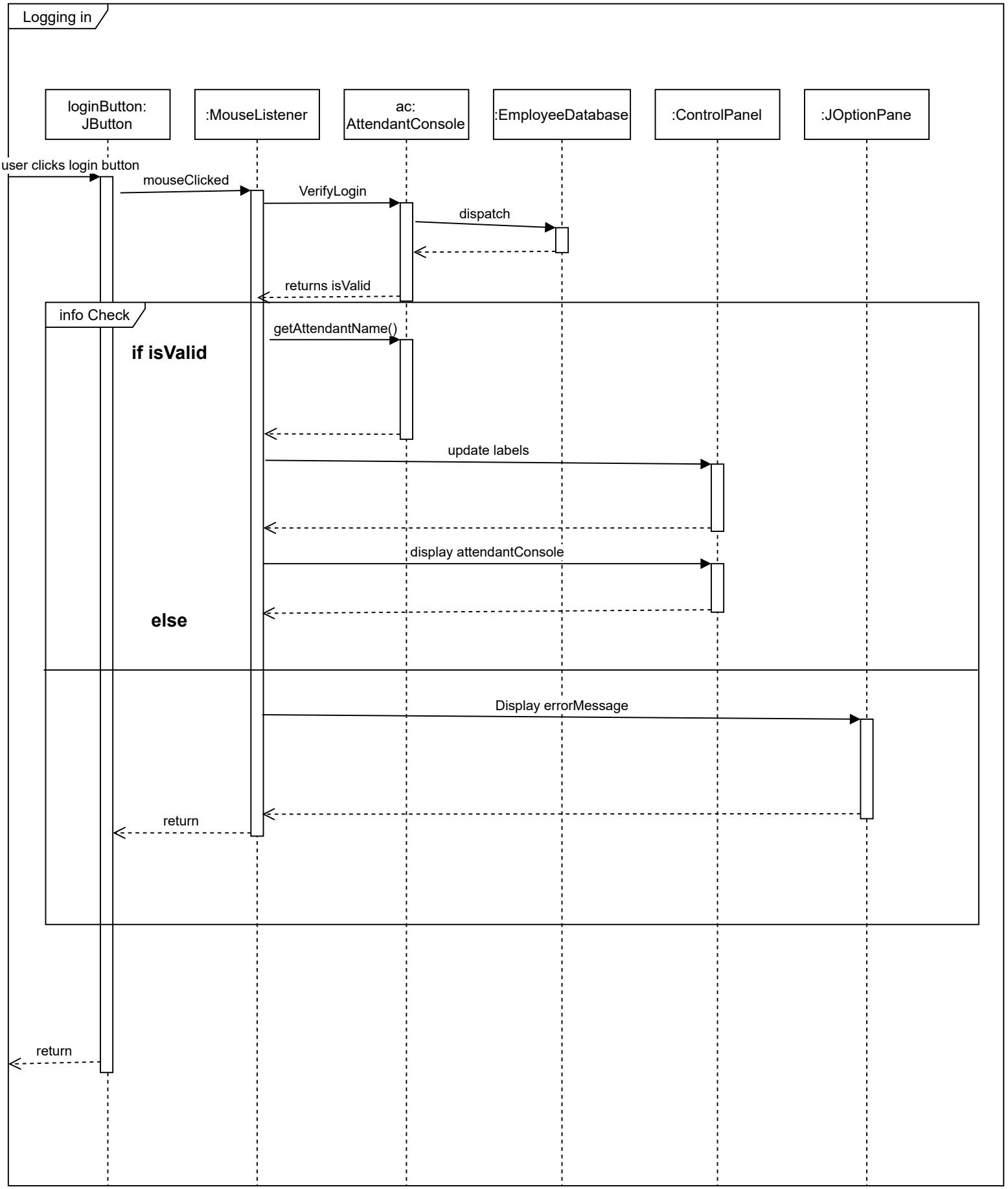

:Session
pluCode:PLUCodedProduct
BP:BarcodedProduct
input:Scanner
scale:ElectronicScale
pluCode.getDescription()
return PLUdescription
BP.getDescription()
return BPdescription
alt
[in = "n"]
input.nextLine()
return in
[else in == "y"]
scale.remove(item)
[else]
shouldRemoveProduct()


:AttendantConsole
:Item
:ElectronicScale
input:Scanner
ElectronicScale(int, int)
return scale
return item
listener.checkConform(scale, item)
alt
input.nextLine()
return in
[in = "n"]
[else in == "y"]
[else]
session.shouldRemoveProduct()


:BaggingAreaListenerSoftware
:Item
:ElectronicScale
:SimulationException
ElectronicScale(int, int)
return scale
return item
item.getWeight()
return itemWeight
scale.getCurrentWeight()
return lastWeight
scale.add(item)
return
scale.getCurrentWeight()
return thisWeight
alt
[thisWeight-lastWeight != itemWeight]
SimulationException("That is not the expected item.")
return message


products
:CustomerUseCases
:ProductDatabases
:Session
scale
lookupProduct(id)
contains(id)
alt
[contains(id)]
desc = description(id)
price = price(id)
[else]
desc, price = None
return price, desc
enter(plu)
get weight
lookupProduct(plu)
total = price * weight
update balance (total)
update order (desc)


:ReceiptPrinterListenerSoftware
:StationUsage
:GUIDispatch
paperAdded
setLinesOfPaperRemaining(MAX)
updateMaitenenceList
inkAdded
setCharactersOfInkRemaining(MAX)
updateMaintainanceList
outOfInk
getCharactersInkRemaining
return n
Conditional
n != 0
setCharactersOfInkRemaining(0)
outOfPaper
getLinesOfPaperRemaining
return n
n != 0
setLinesOfPaperRemaining(0)


Logging in
loginButton: JButton
:MouseListener
ac: AttendantConsole
:EmployeeDatabase
:ControlPanel
:JOptionPane
user clicks login button
mouseClicked
VerifyLogin
dispatch
returns isValid
info Check
getAttendantName()
if isValid
update labels
display attendantConsole
else
Display errorMessage
return
return

Press "start" button

start to use this machine

entersPLUCode(pluCode)
isOwnBagPlaced(bag)[bag == "own bag"]/ return true

RemoveItem()/
print "Enter the barcode of the item to be removed: ",
"Enter the PLU code of the item to be removed: ",
"Submit"

barcodeScanned(barcodeScanner, barcode)

Weight

weightChanged(scale, weightInGrams)[thisWeight - lastWeight == itemWeight]

checkConform(scale, item)

Find the item

weight item

toPayment()

Bag

shouldnotBeBagged()[ShouldbeBagged = false]

press "Return to adding items"
returnToAddingItems()

Scan barcode of item

enter_number_of_plastic_bags(number_of_bags_used)

Remove items

Not add to bag

press "Return to adding items"
returnToAddingItems()

isOwnBagPlaced(bag)[bag == "own bag"]/ return true

lookupProduct(barcode)
lookupProduct(code)

enters number of plastic bags used

press "Return to adding items"
returnToAddingItems()

look up item

Add to customers' own bag

checkOwnBagPlaced(customersBag)
[not use own bag]/ add 0.5 per bag

press "Return to adding items"
returnToAddingItems()

Add to plastic bag

toPayment()

toPayment()

toPayment()

toPayment()

toPayment()

Payment

make a payment

processCardPayment(cardData, amount)[cardIssuer != null]
[holdNumber != -1]/ return false
handleCardPayment(CardData cardData)
cardDataRead(reader, data)

use membership card

press "Pay by Card"
calculatePoints(balance, points)
membershipCardReader(data)

Press "Pay by Cash"

pay by card

giftcardPayment
(giftcardNumber)
press "Pay by giftCard"

use gift card

pay by cash

validBanknoteDetected(validator,
currency, value) / update balance

press "Print Receipt"
printRecipt()/ return Receipt
printer(reciptContent)/return Receipt
generateReceiptContent()/return content

validCoinDetected(validator, value)/ update balance

pay by banknote

coin

dispenseChange()[banknoteCount > 0]

coinAdded(tray)/ coin ++
dispenseChange()[coinCount > 0]

recieve banknote changes

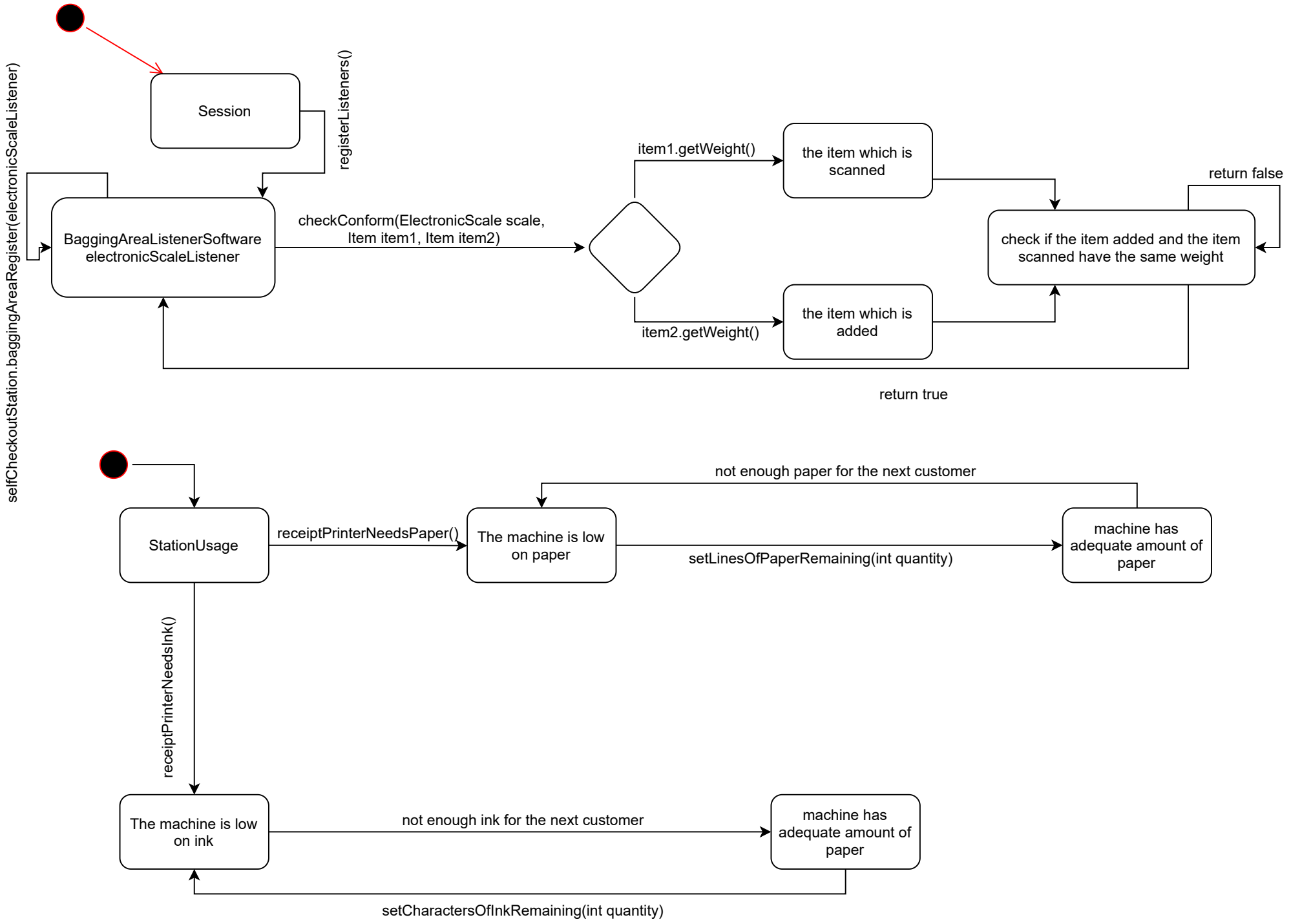
press "Print Receipt"
printRecipt()/ return Receipt
printer(reciptContent)/
returnReceipt
generateReceiptContent()/return content

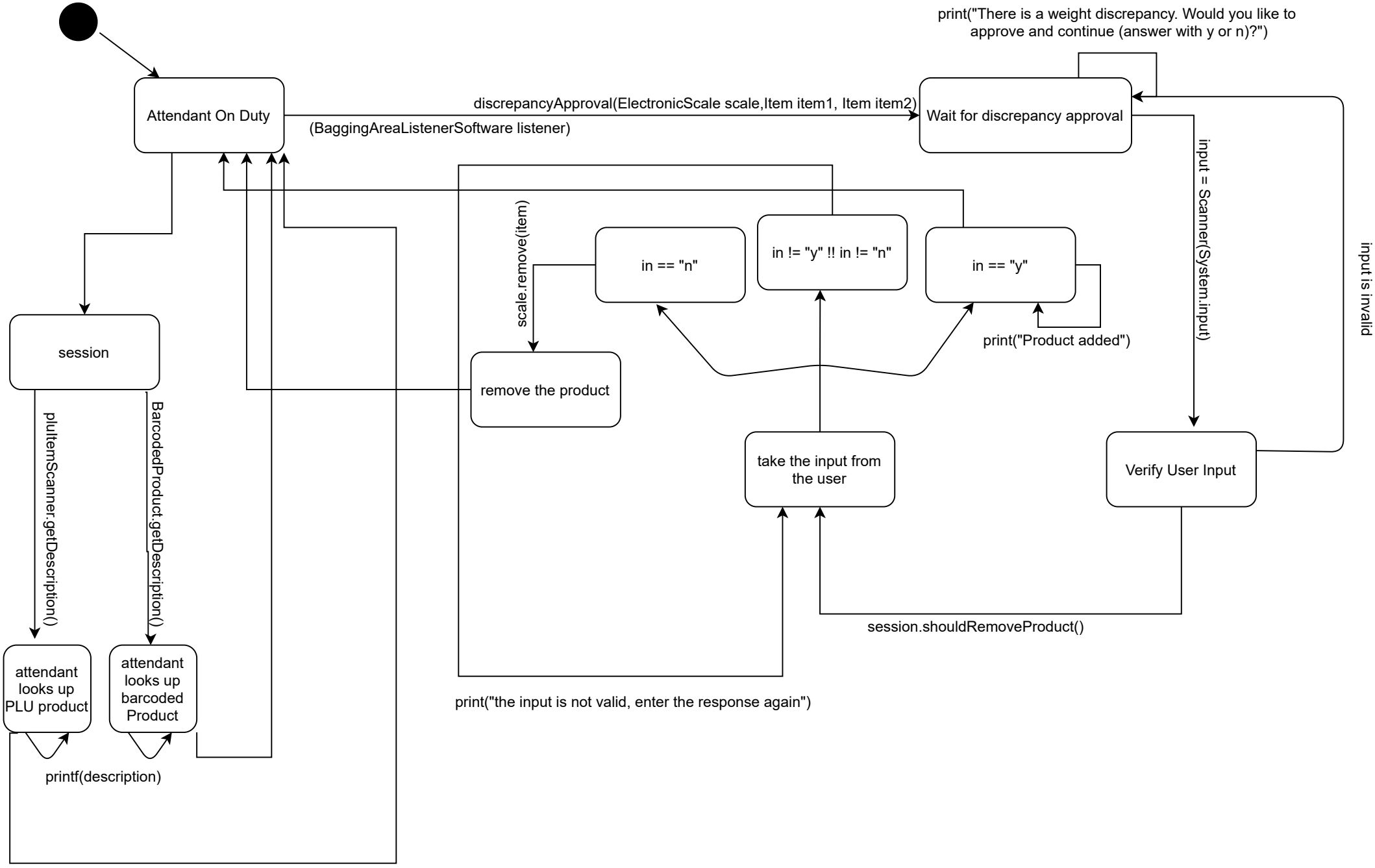
recieve coin changes

press "Print Receipt"
printRecipt()/ return Receipt
printer(reciptContent)/
returnReceipt
generateReceiptContent()/return content

get teceipt

newPaperAmount()[newAmount <= 2000]/ print "Paper low. Please replace the paper roll."
newInkAmount()[newAmount <= 20000]/ print "Ink low. Please replace the ink cartridge."


selfCheckoutStation.baggingAreaRegister(electronicScaleListener)
Session
registerListeners()
BaggingAreaListenerSoftware
electronicScaleListener
checkConform(ElectronicScale scale,
Item item1, Item item2)
item1.getWeight()
the item which is
scanned
item2.getWeight()
the item which is
added
check if the item added and the item
scanned have the same weight
return false
return true
StationUsage
receiptPrinterNeedsPaper()
The machine is low
on paper
not enough paper for the next customer
setLinesOfPaperRemaining(int quantity)
machine has
adequate amount of
paper
receiptPrinterNeedsInk()
The machine is low
on ink
not enough ink for the next customer
machine has
adequate amount of
paper
setCharactersOfInkRemaining(int quantity)


print("There is a weight discrepancy. Would you like to approve and continue (answer with y or n)?")
Attendant On Duty
discrepancyApproval(ElectronicScale scale,Item item1, Item item2)
(BaggingAreaListenerSoftware listener)
Wait for discrepancy approval
input = Scanner(System.input)
input is invalid
scale.remove(item)
in == "n"
in != "y" !! in != "n"
in == "y"
print("Product added")
session
remove the product
take the input from the user
Verify User Input
pluItemScanner.getDescription()
BarcodedProduct.getDescription()
attendant looks up PLU product
attendant looks up barcoded Product
printf(description)
session.shouldRemoveProduct()
print("the input is not valid, enter the response again")