SENG 300 - Iteration 3

Diagrams

Michelle Loi - 30019557

Alisha Nasir - 30140704

Zainab Bari - 30154224

Rheanna Fielder - 30092060

Jason Ngu - 30145770

Brian Tran - 30064686

Duong Tran - 30113765

Imran Haji - 30141571

Sahil Brar - 30021440

Alina Mansuri - 30008370

Eugene Lee - 30137489

Ali Savab Pour - 30154744

Justin Clibbett - 30128271

Andy Tran - 30125341

Eungyo Song - 30079379

Anthony Krim - 30142199

Maira Khan - 30047942

Brett Lyle - 30103268

Nemanja Grujic - 30116614

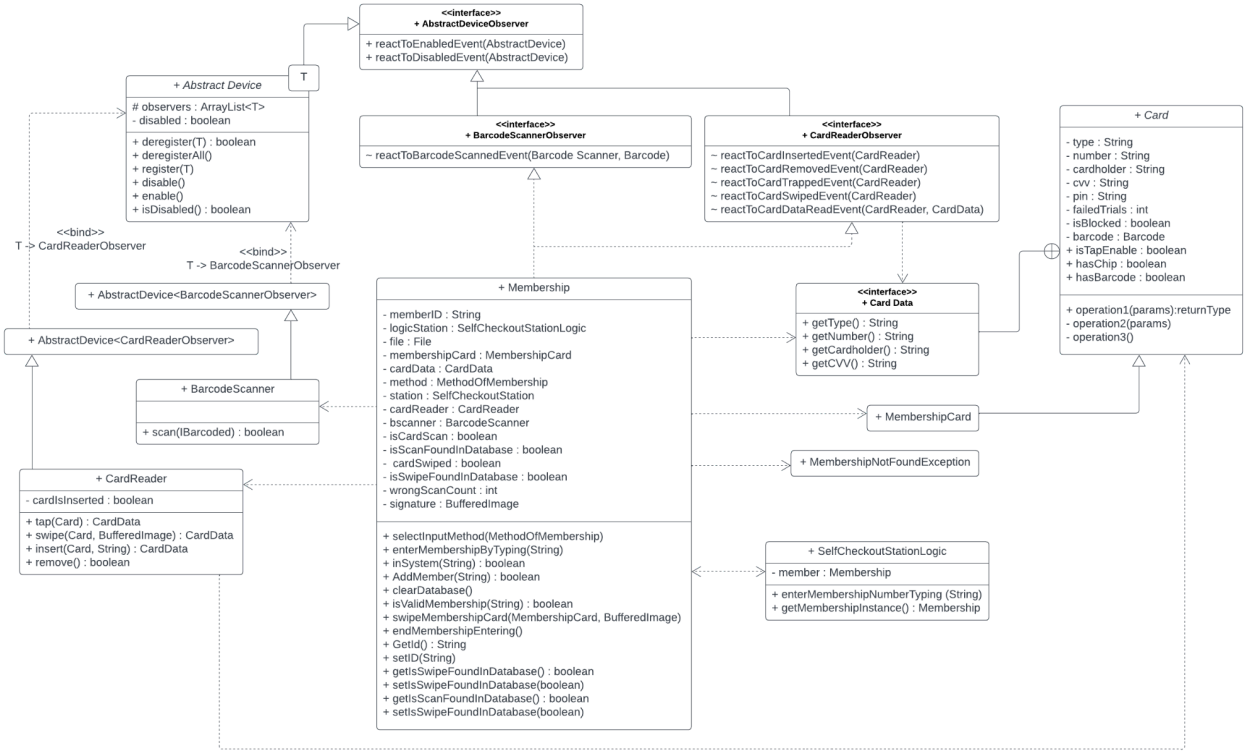
Aminreza Tavakoli - 30127594

Brian Vo - 10188952

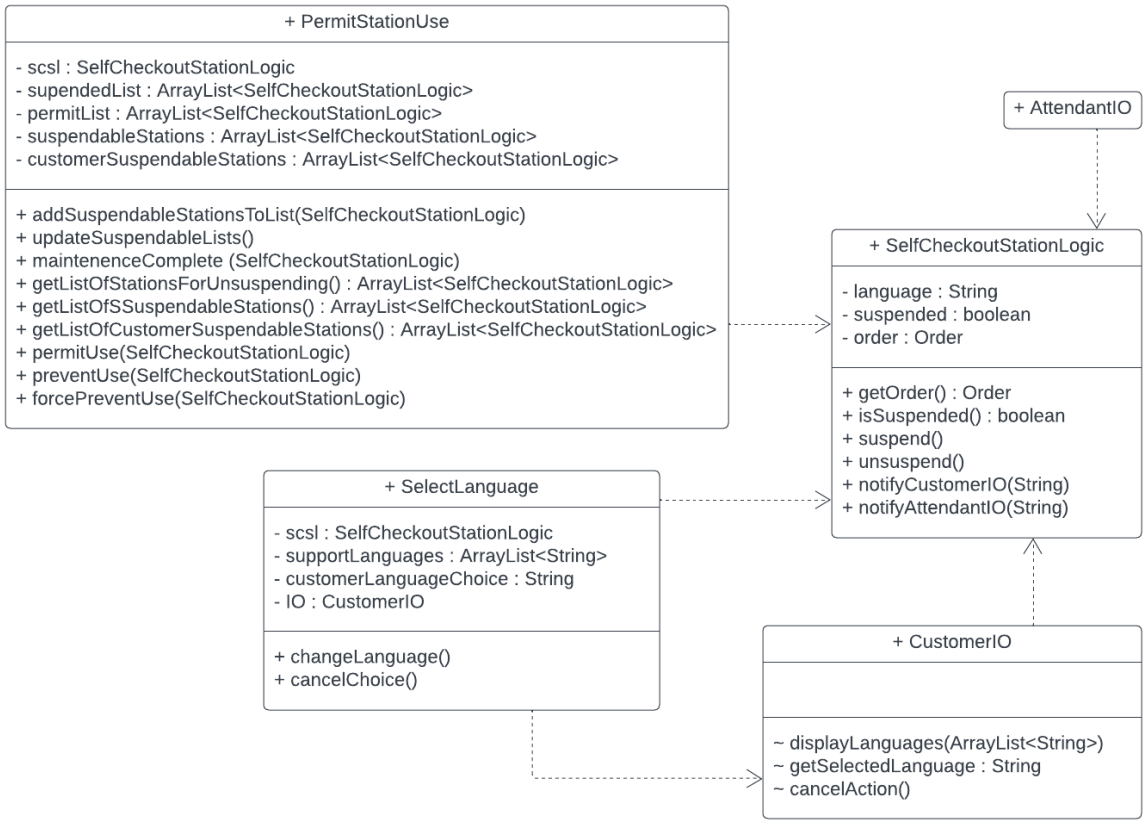
Nishan Soni 30147280

Caio Araujo 30148726

Adding Membership:



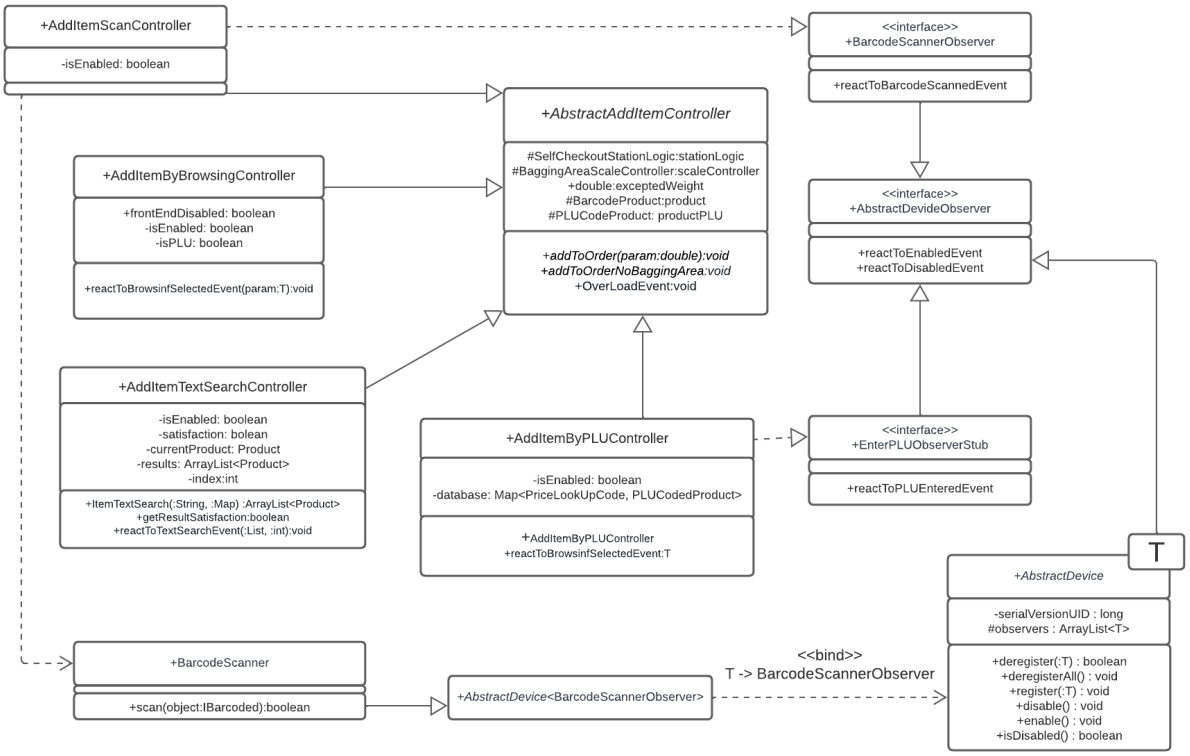
Station Usage Prevention and Language Options:



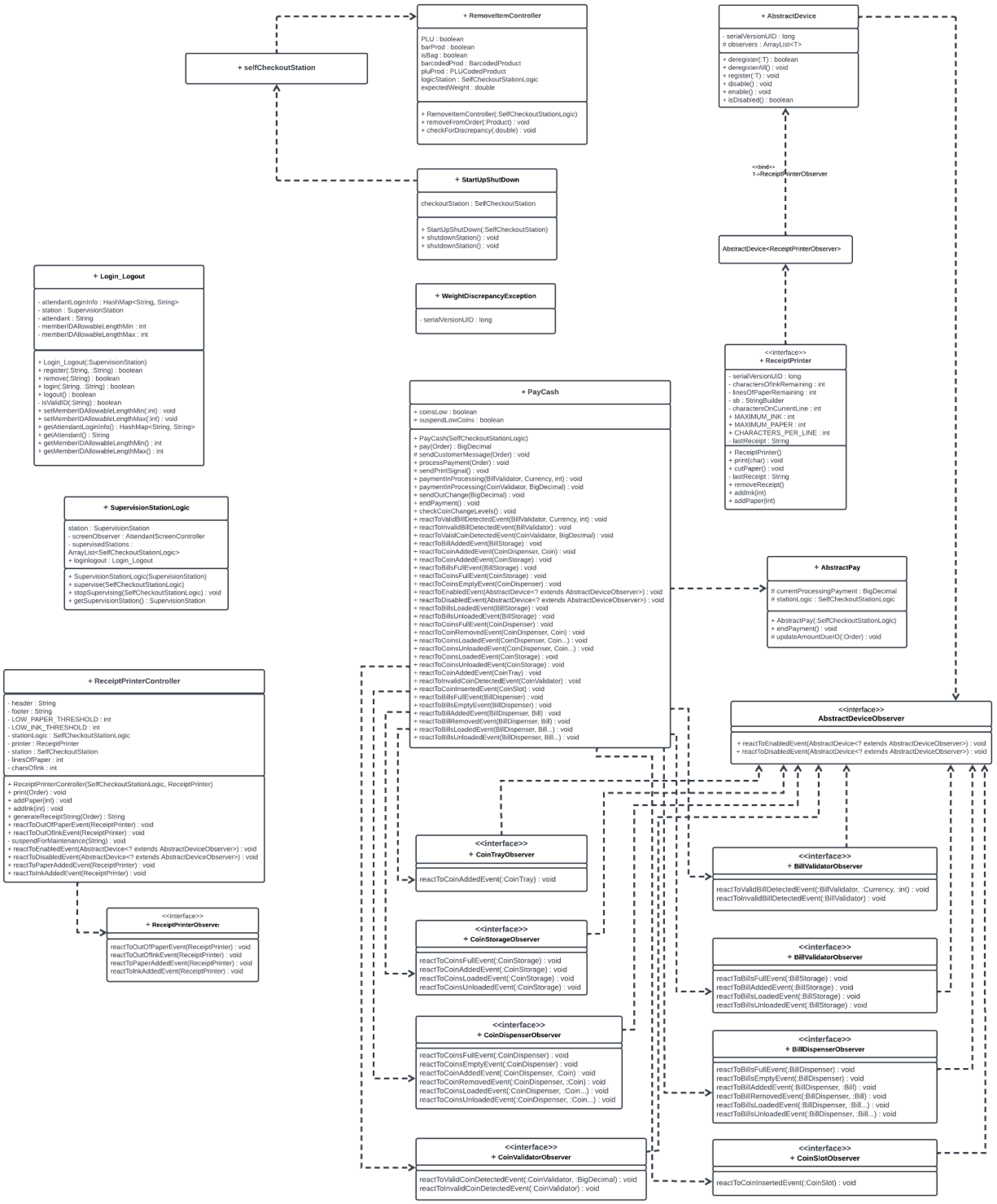
Pay:

+PayCard
-currentReader:CarReader
-cardData:CardData
-cardIssuer:CardIssuer
-method:MethodOfPayment
-station:SelfCheckoutStation
+pay(:Order,:MethodOfPayment,:CardIssuer):void
<<interface>>
+CardReaderObserver
+reactToCardInsertedEvent(:CardReader):void
+reactToCardRemovedEvent(:CardReader):void
+reactToCardTappedEvent(:CardReader):void
+reactToCardSwipedEvent(:CardReader):void
+reactToCardDataEvent(:CardReader):void
+AbstractPay
#currentProcessingPayment:BigDecimal
#SelfCheckoutStationLogic:stationLogic
+endPayment:void
#updateAmountDueIO(:Order):void
+CardReader
+tap(card:Card):CardData
+swipe(:Card,:BufferedImage):CardData
+inserted(:Card,:String):CardData
+remove():boolean
+PayWithGiftCard
-currentReader:CarReader
-cardData:CardData
-cardIssuer:CardIssuer
-method:MethodOfPayment
-station:SelfCheckoutStation
+pay(:Order,:MethodOfPayment,:CardIssuer):void
#sendCustomerMessage(:Order):void
+AbstractDevice<CardReaderObserver>
+PayCash
+pay(:Order):BigDecimal
#sendCustomerMessage(:Order):void
+processPayment(:Order):void
+sendPrintSignal:void
+paymentInProcessing(:BillValidator,:Currency,:int):void
+paymentInProcessing(:CoinValidator,:BigDecimal):void
+sendOutChange(:BigDecimal):void
{exceptions={SimulationException,OverloadException
DisabledException,EmptyException}}
+endPayment:void
<<interface>>
+CoinDispenserObserver
+reactToCoinsFullEvent(:CoinDispenser): void
+reactToCoinsEmptyEvent(:CoinDispenser):void
+reactToCoinAddedEvent(:CoinDispenser, :Coin):void
+reactToCoinRemovedEvent(:CoinDispenser, :Coin):void
+reactToCoinsLoadedEvent(:CoinDispenser, :Coin):void
+reactToBCoinsUnloadedEvents(:CoinDispenser, :Coin):void
<<interface>>
+BillValidatorObserver
+reactToValidDetectedEvent(:BillValidator, :Currency, :int): void
+reactToInvalidBillDetectedEvent(:validator):void
<<interface>>
+CoinValidatorObserver
+reactToValidCoinDetectedEvent(:CoinValidator): void
+reactToInvalidCoinDetectedEvent(:CoinValidator):void
<<interface>>
+BillDispenserObserver
+reactToBillsFullEvent(:BillDispenser): void
+reactToBillsEmptyEvent(:BillDispenser):void
+reactToBillAddedEvent(:BillDispenser, :Bill):void
+reactToBillRemovedEvent(:BillDispenser, :Bill):void
+reactToBillsLoadedEvent(:BillDispenser, :Bill):void
+reactToBillsUnloadedEvents(:BillDispenser,
:Bill):void
+ReceiptPrinterController
-header:String{readOnly}
-footer:String{readOnly}
-LOW_PAPER_THRESHOLD:int{readOnly}
-LOW_INK_THRESHOLD:int{readOnly}
-stationLogic:SelfCheckoutStationLogic
-printer:ReceiptPrinter
-station:SelfCheckoutStation
-int:linesOfPaper
-int:charsOfInk
+print(:Order):void
+addPaper(:int):void {exception={OverloadException}}
+addInk(:int):void {exception={OverloadException}}
+generateReceiptString(:Order):String
<<interface>>
+BillStorageObserver
+reactToBillsFullEvent(:BillStorage): void
+reactToBillAddedEvent(:BillStorage):void
+reactToBillsLoadedEvent(:BillStorage):void
+reactToBillsUnloadedEvents(:BillStorage):void
<<interface>>
+CoinStorageObserver
+reactToCoinsFullEvent(:CoinStorage): void
+reactToCoinAddedEvent(:CoinStorage):void
+reactToCoinsLoadedEvent(:CoinStorage):void
+reactToCoinsUnloadedEvent(:CoinStorage):void
<<interface>>
+CoinTrayObserver
+reactToCoinAddedEvent(:CoinTray): void
<<interface>>
+CoinSlotObserver
+reactToCoinInsertedEvent(:CoinSlot): void
<<interface>>
+ReceiptPrinterObserver
+reactToOutOfPaperEvent(:ReceiptPrinter):void
+reactToOutOfInkEvent(:ReceiptPrinter):void
+reactToPaperAddedEvent(ReceiptPrinter):void
+reactToInkAddedEvent(ReceiptPrinter):void
+CoinDispenser
+size():int
+load(Coin:coin):void{exceptions=
{SimulationException,OverloadException}}
+unload():List<Coin>
+connect(UnidirectionalChannel<Coin>:sink):void
+getCapacity():int
+BillValidator
+accept(bill: Bill):boolean
+ hasSpace():boolean
+CoinValidator
+accept(coin:Coin) boolean
+isValid(coin:Coin):boolean
+connnect(:overflowSink):void
+BillDispenser
+size():int
+load(Bill:bills):void{exceptions=
{SimulationException,OverloadException}}
+unload():List<Bill>
+connect(:sink):void
+getCapacity():int
+emit():boolean
+CoinStorage
+getCapacity():int
+coinGetCount():int
+load(Coin:coins):void{exceptions=
{SimulationException,OverloadException}}
+unload():List<Coin>
+accept(coin:Coin):boolean
+CoinTray
+accept(coin:Coin):boolean
+collectionsCoins():List<Coin>
+CoinSlot
+connect(:sink):void
+accept(coin:Coin):boolean
<<bind>>
T -> CardReaderObserver
T
+AbstractDevice
-serialVersionUID : long
#observers : ArrayList<T>
+deregister(:T) : boolean
+deregisterAll() : void
+register(:T) : void
+disable() : void
+enable() : void
+isDisabled() : boolean
<<interface>>
+AbstractDeviceObserver
+reactToEnabledEvent
+reactToDisabledEvent
+AbstractDevice<CoinDispenserObserver>
<<bind>>
T -> CoinDispenserObserver
+AbstractDevice<BillValidatorObserver>
<<bind>>
T -> BillValidatorObserver
+AbstractDevice<CoinValidatorObserver>
<<bind>>
T -> CoinValidatorObserver
+AbstractDevice<BillDispenserObserver>
<<bind>>
T -> BillDispenserObserver
+AbstractDevice<CoinStorageObserver>
<<bind>>
T -> CoinStorageObserver
+AbstractDevice<CoinTrayObserver>
<<bind>>
T -> CoinTrayObserver
+AbstractDevice<CoinSlotObserver>
<<bind>>
T -> CoinSlotObserver

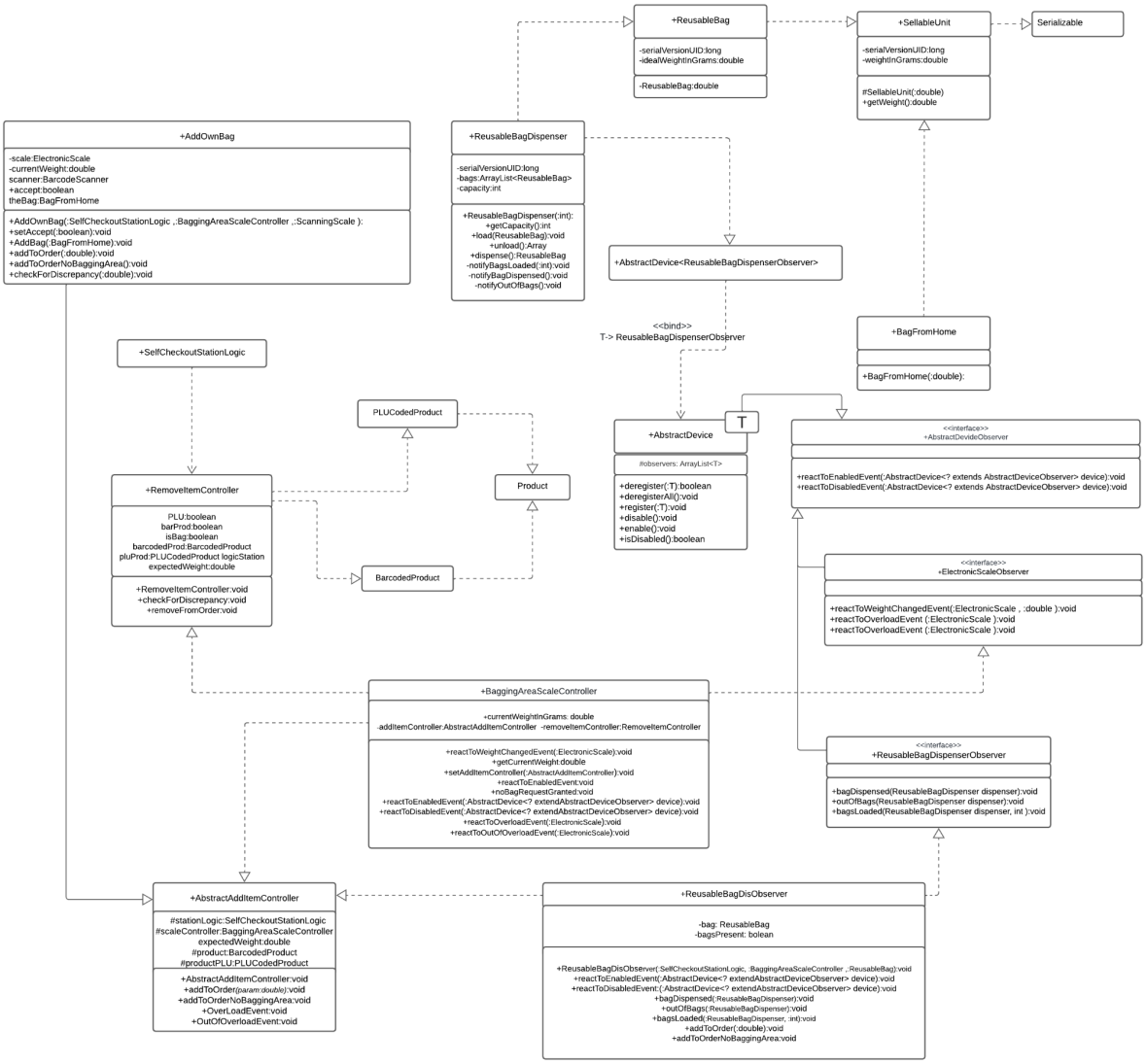
Add Item:


+AddItemScanController
-isEnabled: boolean
<<interface>>
+BarcodeScannerObserver
+reactToBarcodeScannedEvent
+AbstractAddItemController
#SelfCheckoutStationLogic:stationLogic
#BaggingAreaScaleController:scaleController
+double:exceptedWeight
#BarcodeProduct:product
#PLUCodeProduct: productPLU
+addToOrder(param:double):void
+addToOrderNoBaggingArea:void
+OverLoadEvent:void
+AddItemByBrowsingController
+frontEndDisabled: boolean
-isEnabled: boolean
-isPLU: boolean
+reactToBrowsinfSelectedEvent(param:T):void
<<interface>>
+AbstractDevideObserver
+reactToEnabledEvent
+reactToDisabledEvent
+AddItemTextSearchController
-isEnabled: boolean
-satisfaction: bolean
-currentProduct: Product
-results: ArrayList<Product>
-index:int
+ItemTextSearch(:String, :Map) :ArrayList<Product>
+getResultSatisfaction:boolean
+reactToTextSearchEvent(:List, :int):void
+AddItemByPLUController
-isEnabled: boolean
-database: Map<PriceLookUpCode, PLUCodedProduct>
+AddItemByPLUController
+reactToBrowsinfSelectedEvent:T
<<interface>>
+EnterPLUObserverStub
+reactToPLUEnteredEvent
T
+AbstractDevice
-serialVersionUID : long
#observers : ArrayList<T>
+deregister(:T) : boolean
+deregisterAll() : void
+register(:T) : void
+disable() : void
+enable() : void
+isDisabled() : boolean
+BarcodeScanner
+scan(object:IBarcoded):boolean
+AbstractDevice<BarcodeScannerObserver>
<<bind>>
T -> BarcodeScannerObserver

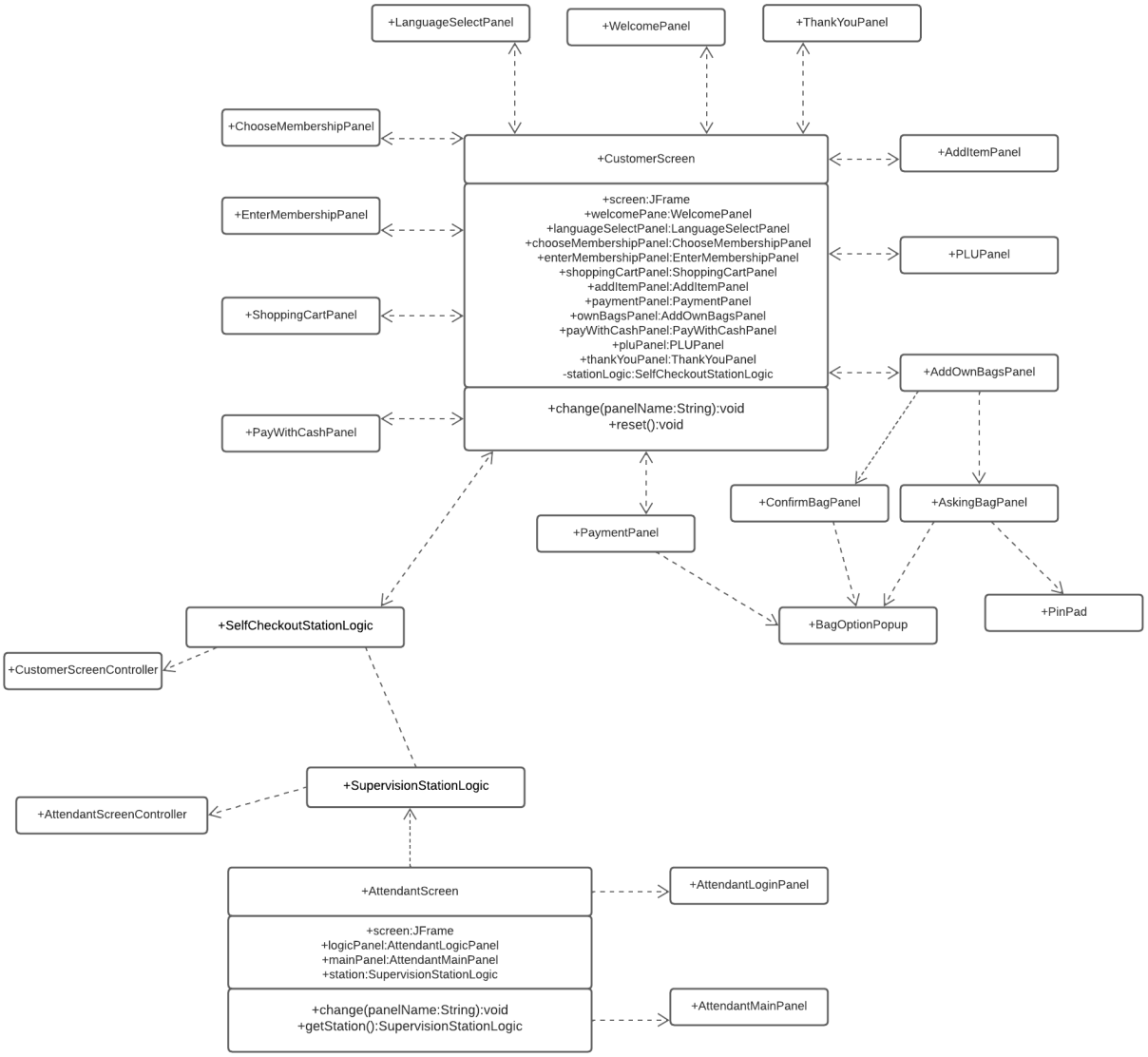
Attendant Maintenance & Attendant Actions:



Bagging:


+ReusableBag
-serialVersionUID:long
-idealWeightInGrams:double
-ReusableBag:double
+SellableUnit
-serialVersionUID:long
-weightInGrams:double
#SellableUnit(:double)
+getWeight():double
Serializable
+AddOwnBag
-scale:ElectronicScale
-currentWeight:double
scanner:BarcodeScanner
+accept:boolean
theBag:BagFromHome
+AddOwnBag(:SelfCheckoutStationLogic ,:BaggingAreaScaleController ,:ScanningScale):
+setAccept(:boolean):void
+AddBag(:BagFromHome):void
+addToOrder(:double):void
+addToOrderNoBaggingArea():void
+checkForDiscrepancy(:double):void
+ReusableBagDispenser
-serialVersionUID:long
-bags:ArrayList<ReusableBag>
-capacity:int
+ReusableBagDispenser(:int):
+getCapacity():int
+load(ReusableBag):void
+unload():Array
+dispense():ReusableBag
-notifyBagsLoaded(:int):void
-notifyBagDispensed():void
-notifyOutOfBags():void
+AbstractDevice<ReusableBagDispenserObserver>
<<bind>>
T-> ReusableBagDispenserObserver
+BagFromHome
+BagFromHome(:double):
+SelfCheckoutStationLogic
PLUCodedProduct
Product
BarcodedProduct
+RemoveItemController
PLU:boolean
barProd:boolean
isBag:boolean
barcodedProd:BarcodedProduct
pluProd:PLUCodedProduct logicStation
expectedWeight:double
+RemoveItemController:void
+checkForDiscrepancy:void
+removeFromOrder:void
+AbstractDevice
T
#observers: ArrayList<T>
+deregister(:T):boolean
+deregisterAll():void
+register(:T):void
+disable():void
+enable():void
+isDisabled():boolean
<<interface>>
+AbstractDevideObserver
+reactToEnabledEvent(:AbstractDevice<? extends AbstractDeviceObserver> device):void
+reactToDisabledEvent(:AbstractDevice<? extends AbstractDeviceObserver> device):void
<<interface>>
+ElectronicScaleObserver
+reactToWeightChangedEvent(:ElectronicScale , :double):void
+reactToOverloadEvent (:ElectronicScale):void
+reactToOverloadEvent (:ElectronicScale):void
+BaggingAreaScaleController
+currentWeightInGrams: double
-addItemController:AbstractAddItemController -removeItemController:RemoveItemController
+reactToWeightChangedEvent(:ElectronicScale):void
+getCurrentWeight:double
+setAddItemController(:AbstractAddItemController):void
+reactToEnabledEvent:void
+noBagRequestGranted:void
+reactToEnabledEvent(:AbstractDevice<? extendAbstractDeviceObserver> device):void
+reactToDisabledEvent(:AbstractDevice<? extendAbstractDeviceObserver> device):void
+reactToOverloadEvent(:ElectronicScale):void
+reactToOutOfOverloadEvent(:ElectronicScale):void
<<interface>>
+ReusableBagDispenserObserver
+bagDispensed(ReusableBagDispenser dispenser):void
+outOfBags(ReusableBagDispenser dispenser):void
+bagsLoaded(ReusableBagDispenser dispenser, int):void
+AbstractAddItemController
#stationLogic:SelfCheckoutStationLogic
#scaleController:BaggingAreaScaleController
expectedWeight:double
#product:BarcodedProduct
#productPLU:PLUCodedProduct
+AbstractAddItemController:void
+addToOrder(param:double):void
+addToOrderNoBaggingArea:void
+OverLoadEvent:void
+OutOfOverloadEvent:void
+ReusableBagDisObserver
-bag: ReusableBag
-bagsPresent: bolean
+ReusableBagDisObserver(:SelfCheckoutStationLogic, :BaggingAreaScaleController , :ReusableBag):void
+reactToEnabledEvent(:AbstractDevice<? extendAbstractDeviceObserver> device):void
+reactToDisabledEvent:(:AbstractDevice<? extendAbstractDeviceObserver> device):void
+bagDispensed(:ReusableBagDispenser):void
+outOfBags(:ReusableBagDispenser):void
+bagsLoaded(:ReusableBagDispenser, :int):void
+addToOrder(:double):void
+addToOrderNoBaggingArea:void

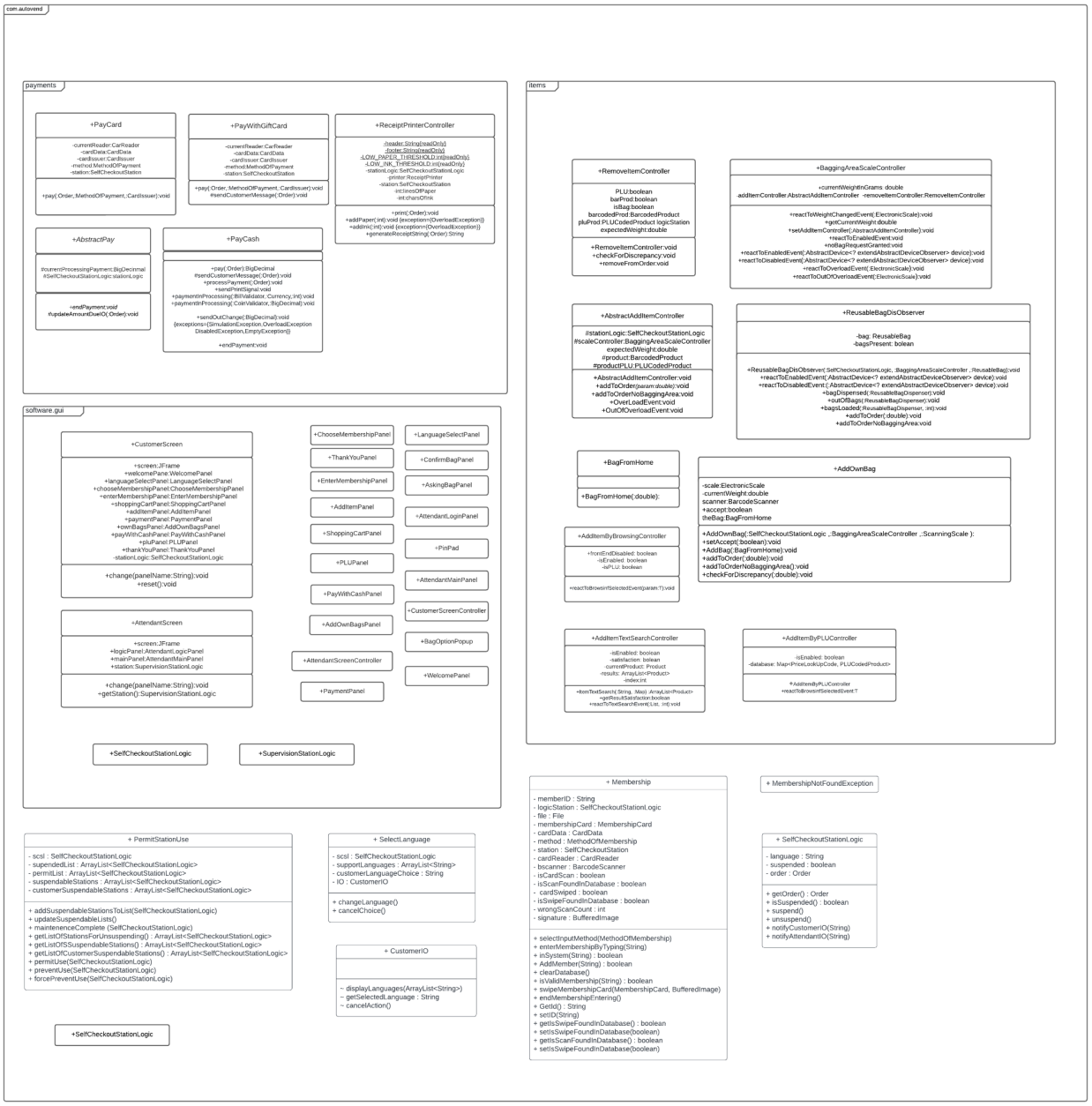
GUI:



<u>Observations:</u>

We have shown classes that are interacting with the two main classes in the GUI package (AttendantScreen, CustomerScreen).

Relationships:

- AddOwnBagPanel and AddOwnBag
- PLUPanel and AddItemByPLUController
- AddItemPanel and AddItemByBrowsingController / AddItemTextSearchController
- PaymentPanel and PayWithGiftCard / PayCard
- PayWithCashPanel and PayCash
- ChooseMembershipPanel and Membership
- AttendantLoginPanel and Login_Logout

General Overview: