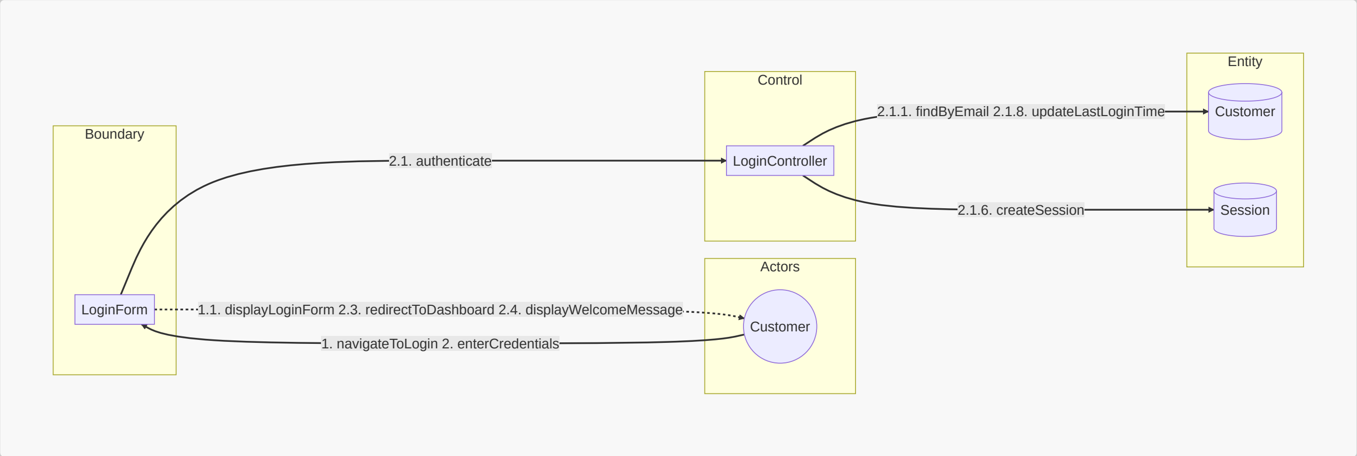
CardHub: Collaboration and VOPC Diagrams

Levente Szabo - 618088

1. Customer Login

Collaboration Diagram

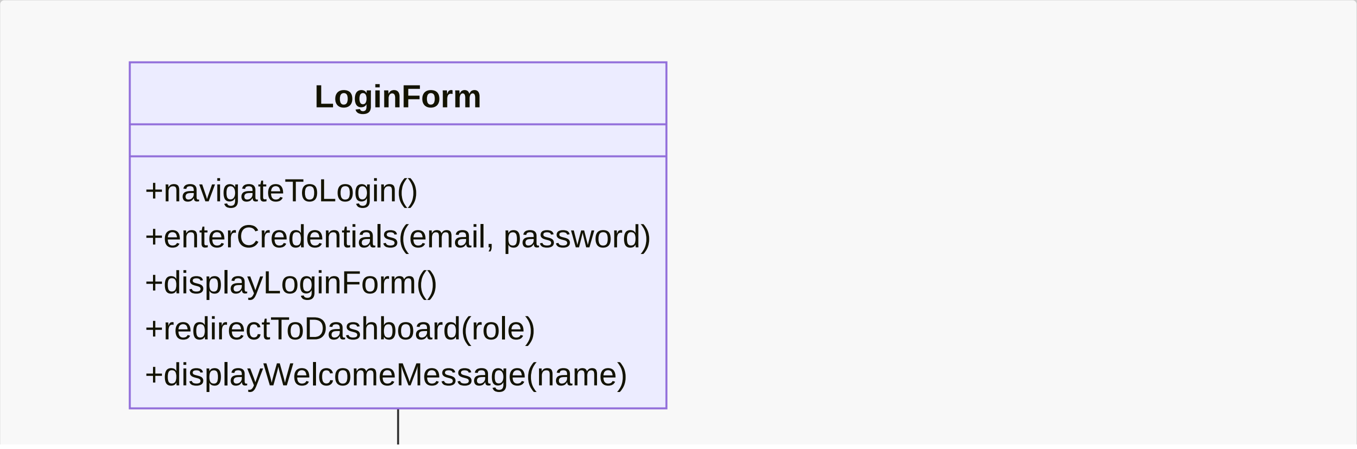
The collaboration diagram shows how objects interact during the login process. Numbers indicate message sequence.

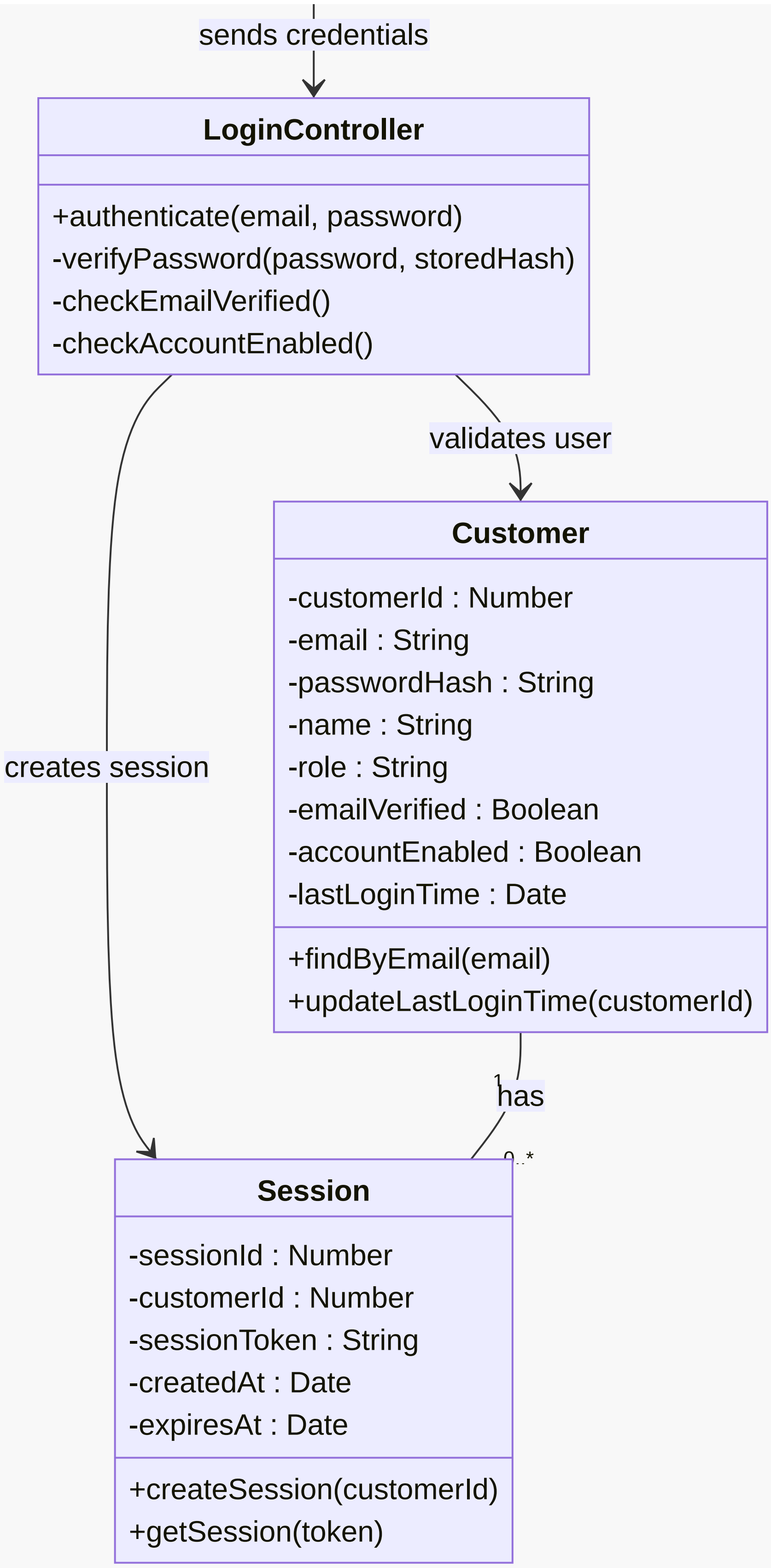

Boundary
LoginForm
Control
LoginController
Entity
Customer
Session
Actors
Customer
2.1. authenticate
2.1.1. findByEmail 2.1.8. updateLastLoginTime
2.1.6. createSession
1.1. displayLoginForm 2.3. redirectToDashboard 2.4. displayWelcomeMessage
1. navigateToLogin 2. enterCredentials

Message Flow:

- 1. Customer navigates to login page
    - 1.1. LoginForm displays the login form
- 2. Customer enters email and password
    - 2.1. LoginForm sends credentials to LoginController for authentication
        - 2.1.1. LoginController looks up customer by email
        - 2.1.3-2.1.5. LoginController verifies password and account status
        - 2.1.6. LoginController creates a new session
        - 2.1.8. LoginController updates last login time
        - 2.2-2.4. Success response flows back to customer

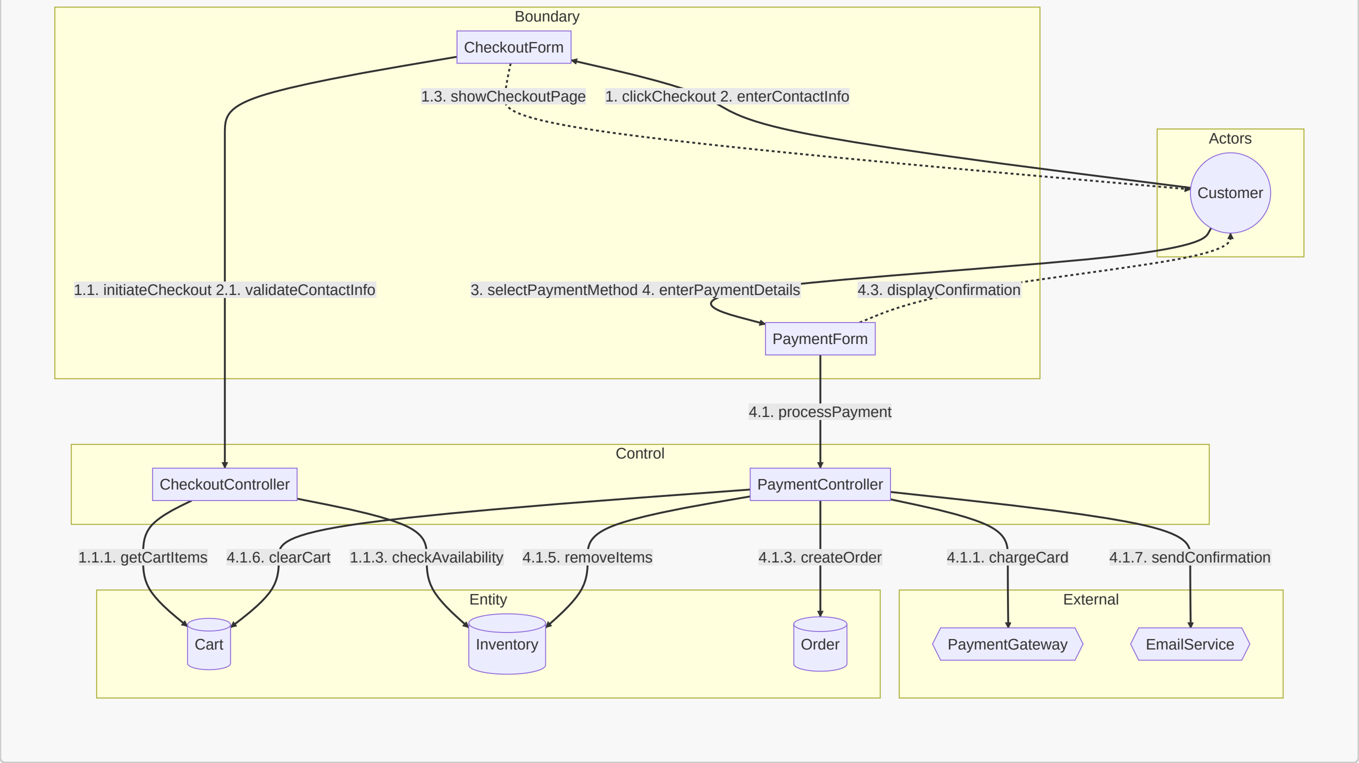
VOPC Diagram


LoginForm
+navigateToLogin()
+enterCredentials(email, password)
+displayLoginForm()
+redirectToDashboard(role)
+displayWelcomeMessage(name)



2. Customer Purchase Checkout

Collaboration Diagram

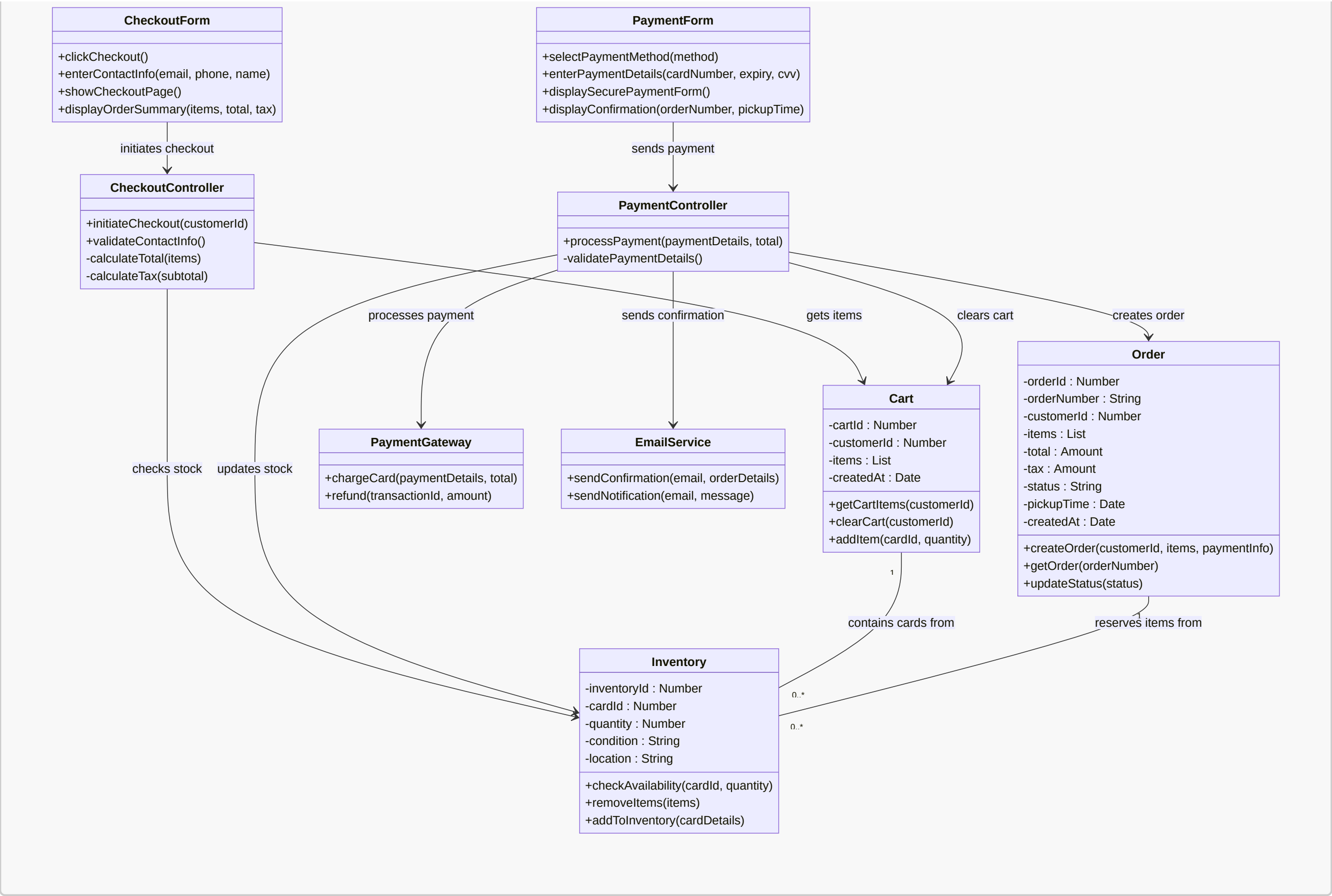


Message Flow:

1. Customer clicks checkout 1.1. CheckoutController gets cart items and checks inventory 1.2-1.3. Order summary displayed to customer
2. Customer enters contact info for pickup 3-4. Customer enters payment details 4.1. PaymentController processes payment through gateway 4.1.3-4.1.6. Order created, inventory updated, cart cleared 4.1.7. Confirmation email sent 4.3. Confirmation displayed to customer

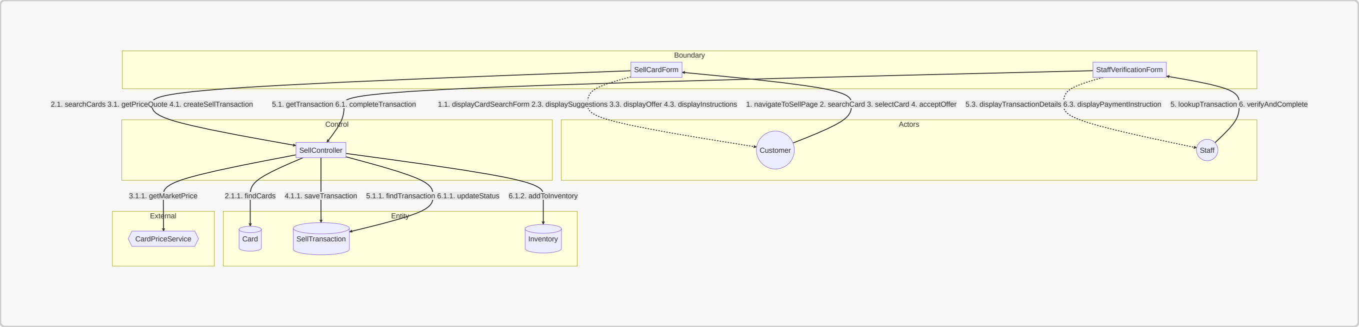
---

VOPC Diagram



3. Sell Card to Store

Collaboration Diagram

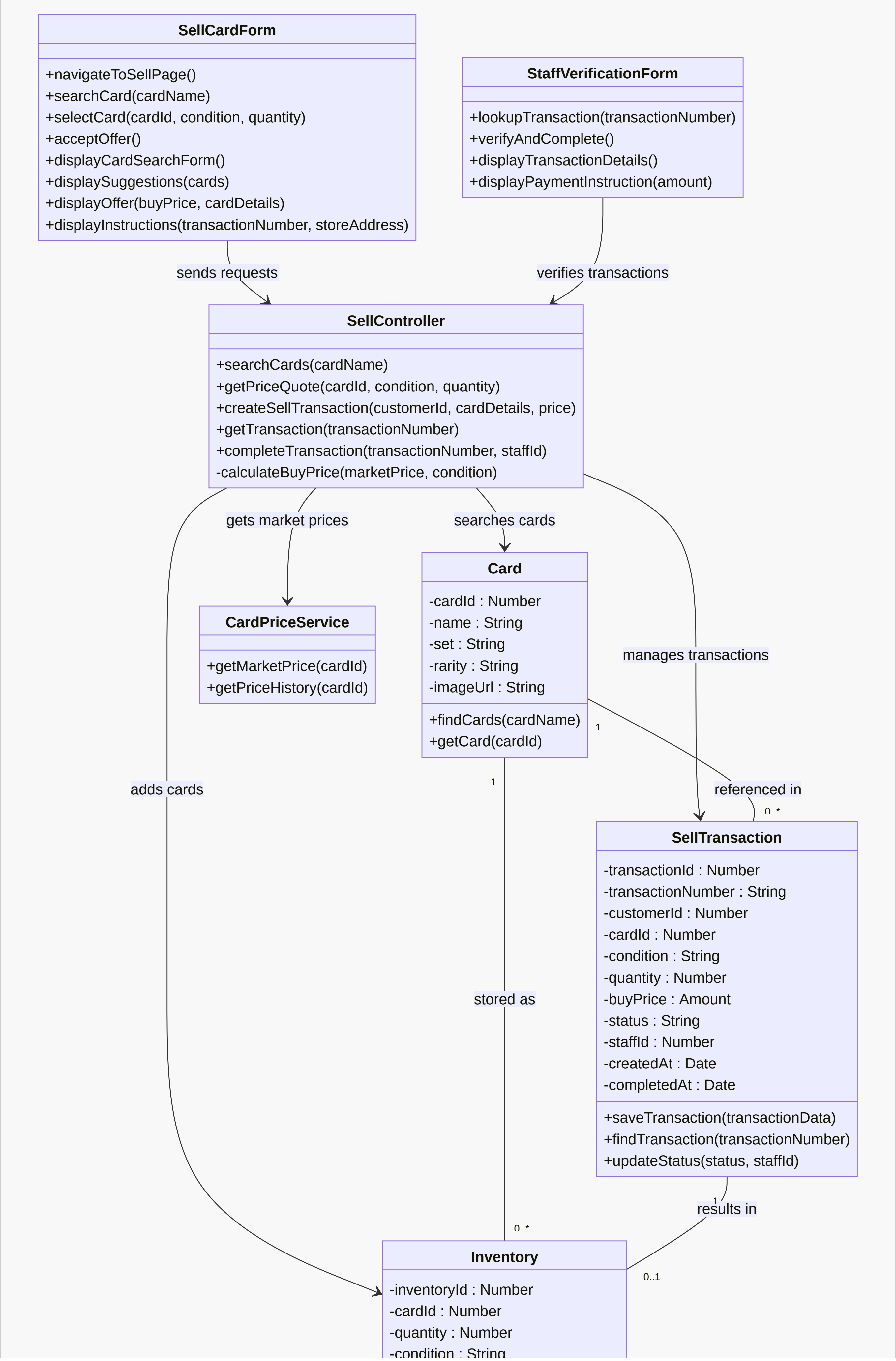

Boundary
SellCardForm
StaffVerificationForm
2.1. searchCards 3.1. getPriceQuote 4.1. createSellTransaction
5.1. getTransaction 6.1. completeTransaction
1.1. displayCardSearchForm 2.3. displaySuggestions 3.3. displayOffer 4.3. displayInstructions
1. navigateToSellPage 2. searchCard 3. selectCard 4. acceptOffer
5.3. displayTransactionDetails 6.3. displayPaymentInstruction
5. lookupTransaction 6. verifyAndComplete
Control
SellController
Actors
Customer
Staff
3.1.1. getMarketPrice
2.1.1. findCards
4.1.1. saveTransaction
5.1.1. findTransaction 6.1.1. updateStatus
6.1.2. addToInventory
External
CardPriceService
Entity
Card
SellTransaction
Inventory

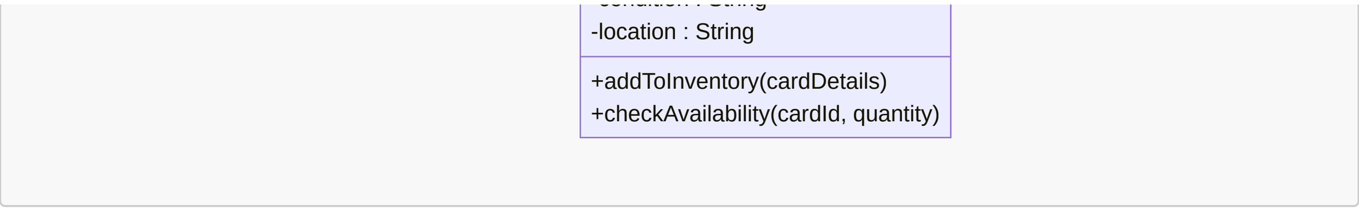
Message Flow:

1. Customer navigates to sell page
2. Customer searches for card, system shows matches
3. Customer selects card and condition, system calculates buy price 3.1.1. SellController fetches market price from external service 3.1.3. Buy price calculated (60% of market, adjusted for condition)
4. Customer accepts offer, transaction created
5. Staff looks up transaction when customer arrives
6. Staff verifies card and completes transaction 6.1.2. Card added to inventory

---

VOPC Diagram


SellCardForm
+navigateToSellPage()
+searchCard(cardName)
+selectCard(cardId, condition, quantity)
+acceptOffer()
+displayCardSearchForm()
+displaySuggestions(cards)
+displayOffer(buyPrice, cardDetails)
+displayInstructions(transactionNumber, storeAddress)
StaffVerificationForm
+lookupTransaction(transactionNumber)
+verifyAndComplete()
+displayTransactionDetails()
+displayPaymentInstruction(amount)
sends requests
verifies transactions
SellController
+searchCards(cardName)
+getPriceQuote(cardId, condition, quantity)
+createSellTransaction(customerId, cardDetails, price)
+getTransaction(transactionNumber)
+completeTransaction(transactionNumber, staffId)
-calculateBuyPrice(marketPrice, condition)
gets market prices
searches cards
Card
-cardId : Number
-name : String
-set : String
-rarity : String
-imageUrl : String
+findCards(cardName)
+getCard(cardId)
CardPriceService
+getMarketPrice(cardId)
+getPriceHistory(cardId)
manages transactions
1
referenced in
0..*
1
adds cards
SellTransaction
-transactionId : Number
-transactionNumber : String
-customerId : Number
-cardId : Number
-condition : String
-quantity : Number
-buyPrice : Amount
-status : String
-staffId : Number
-createdAt : Date
-completedAt : Date
+saveTransaction(transactionData)
+findTransaction(transactionNumber)
+updateStatus(status, staffId)
stored as
1
results in
0..*
0..1
Inventory
-inventoryId : Number
-cardId : Number
-quantity : Number
-condition : String


-condition : String
-location : String
+addToInventory(cardDetails)
+checkAvailability(cardId, quantity)