1) Use case description

Use Case: Process Sale

Actors: Cashier, Customer

Preconditions:

- The cashier is logged into the POS system.
- The customer has goods to purchase.

Postconditions:

- The sale is recorded in the system.
- Inventory is updated.
- A receipt is printed for the customer.

Main Flow:

1. The cashier initiates a new sale transaction in the POS system.
2. The cashier scans the barcode of each item the customer wishes to purchase.
3. The system retrieves the name and price of each item from the catalog.
4. The system checks the inventory for the availability of each item.
5. The system displays the total amount due to the cashier.
6. The cashier informs the customer of the total amount.
7. The customer selects a payment method (cash, credit card, or check).
8. The system processes the payment:
    - If the payment is successful, proceed to step 9.
    - If the payment fails, inform the customer and allow them to retry or select another method.
9. The system updates the inventory to deduct the sold items.
10. The system generates a receipt and prints it for the customer.
11. The cashier hands the receipt to the customer and completes the transaction.

Alternative Flows:

- **Payment Failure:** If payment fails, the cashier can re-attempt payment or cancel the transaction.
- **Item Not Found:** If an item's barcode is not recognized, the cashier can manually enter the item details.

Use Case: Handle Return

Actors: Cashier, Customer

Preconditions:

- The cashier is logged into the POS system.
- The customer has items to return, along with the original receipt.

Postconditions:

- The return is recorded in the system.
- Inventory is updated to reflect the returned items.
- A return receipt is printed for the customer.

Main Flow:

1. The cashier initiates a return transaction in the POS system.
2. The cashier asks the customer for the original receipt and the items being returned.
3. The system verifies the receipt and checks if the items are eligible for return.
4. The cashier scans the barcode of each item being returned.
5. The system retrieves the original sale price of each item and informs the cashier of the total refund amount.
6. The cashier confirms the return with the customer.
7. The customer selects a refund method (cash, credit card reversal, store credit).
8. The system processes the refund:
    - If the refund is successful, proceed to step 9.
    - If the refund fails, inform the customer and allow for alternate arrangements.
9. The system updates the inventory to add the returned items back to stock.
10. The system generates a return receipt and prints it for the customer.
11. The cashier hands the return receipt to the customer and completes the return transaction.

Alternative Flows:

- **Return Not Eligible:** If items are not eligible for return (e.g., past return window), inform the customer and cancel the transaction.
- **Refund Failure:** If the refund fails, the cashier can explain the situation and discuss alternative arrangements.

2) Identify Entity/Boundary Control Objects

Entity Objects

These represent the core data and business logic of the system.

1. **Product**
2. **Customer**
3. **SaleTransaction**
4. **ReturnTransaction**
5. **Receipt**

Boundary Objects

These represent the user interface and interaction points.

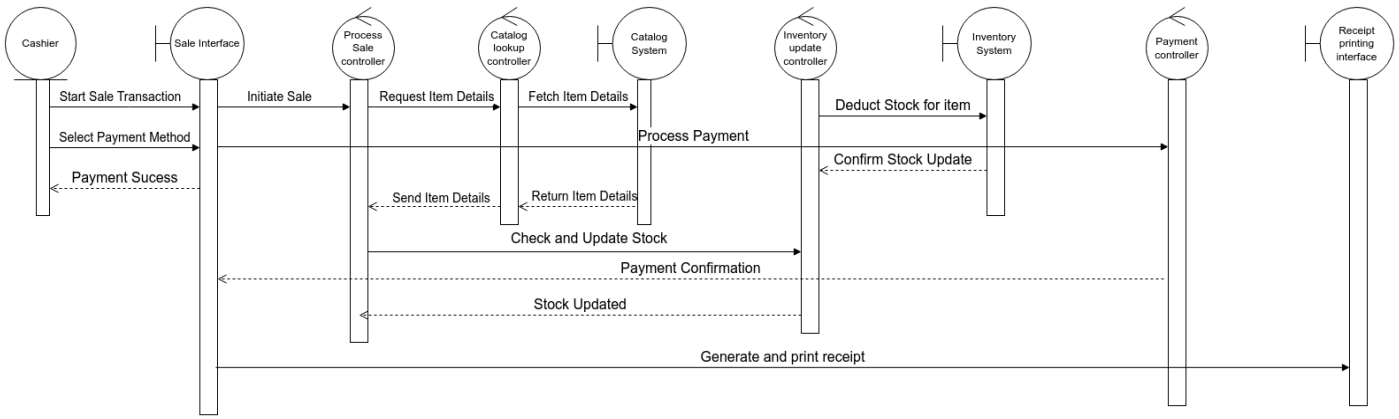
1. **CashierInterface**
2. **CustomerInterface**
3. **AdminInterface**

Control Objects

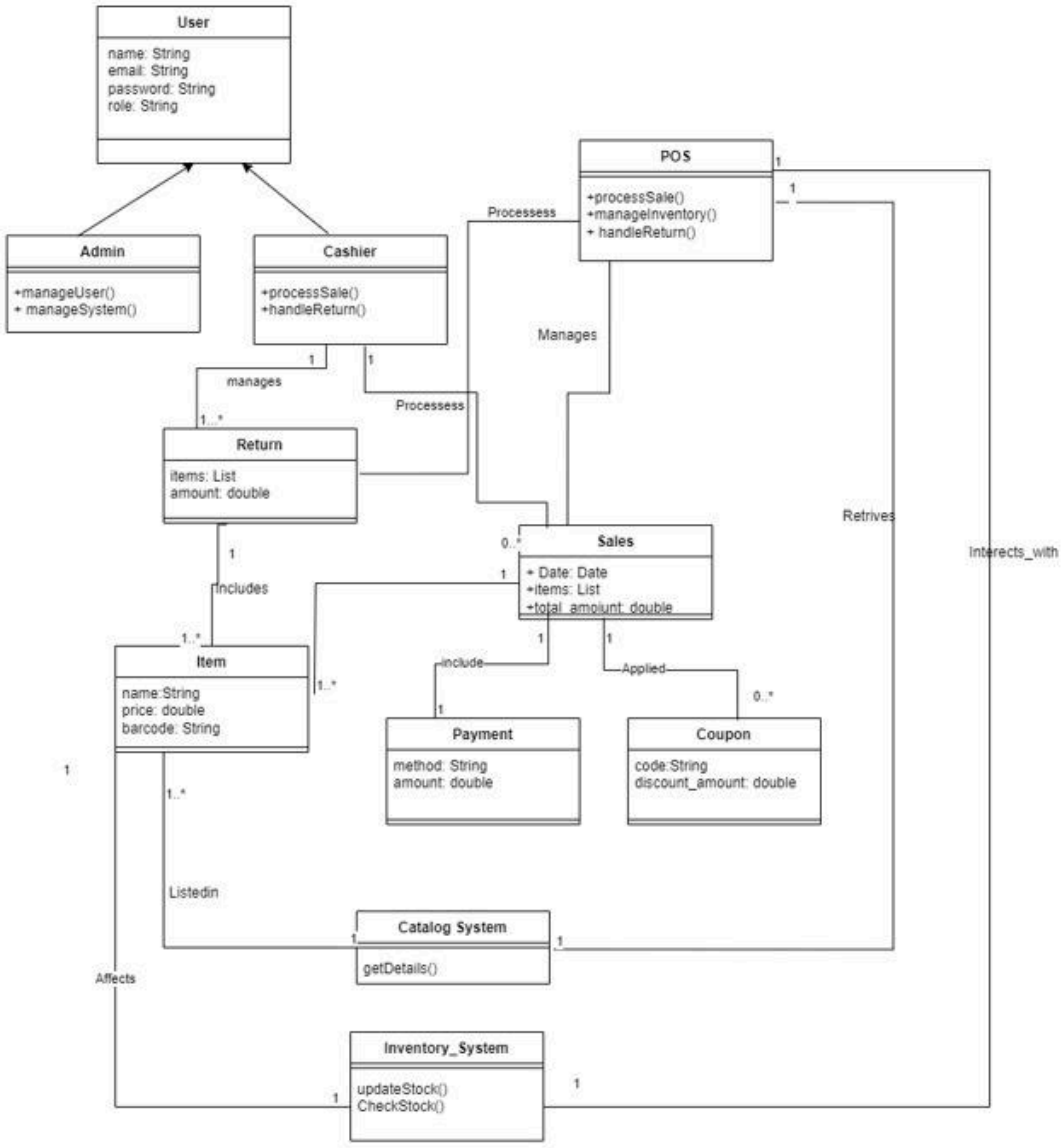
These manage the flow of the application and coordinate between entities and boundaries.

1. **SaleController**
2. **ReturnController**
3. **InventoryController**
4. **UserController**

3) Develop Sequence Diagrams

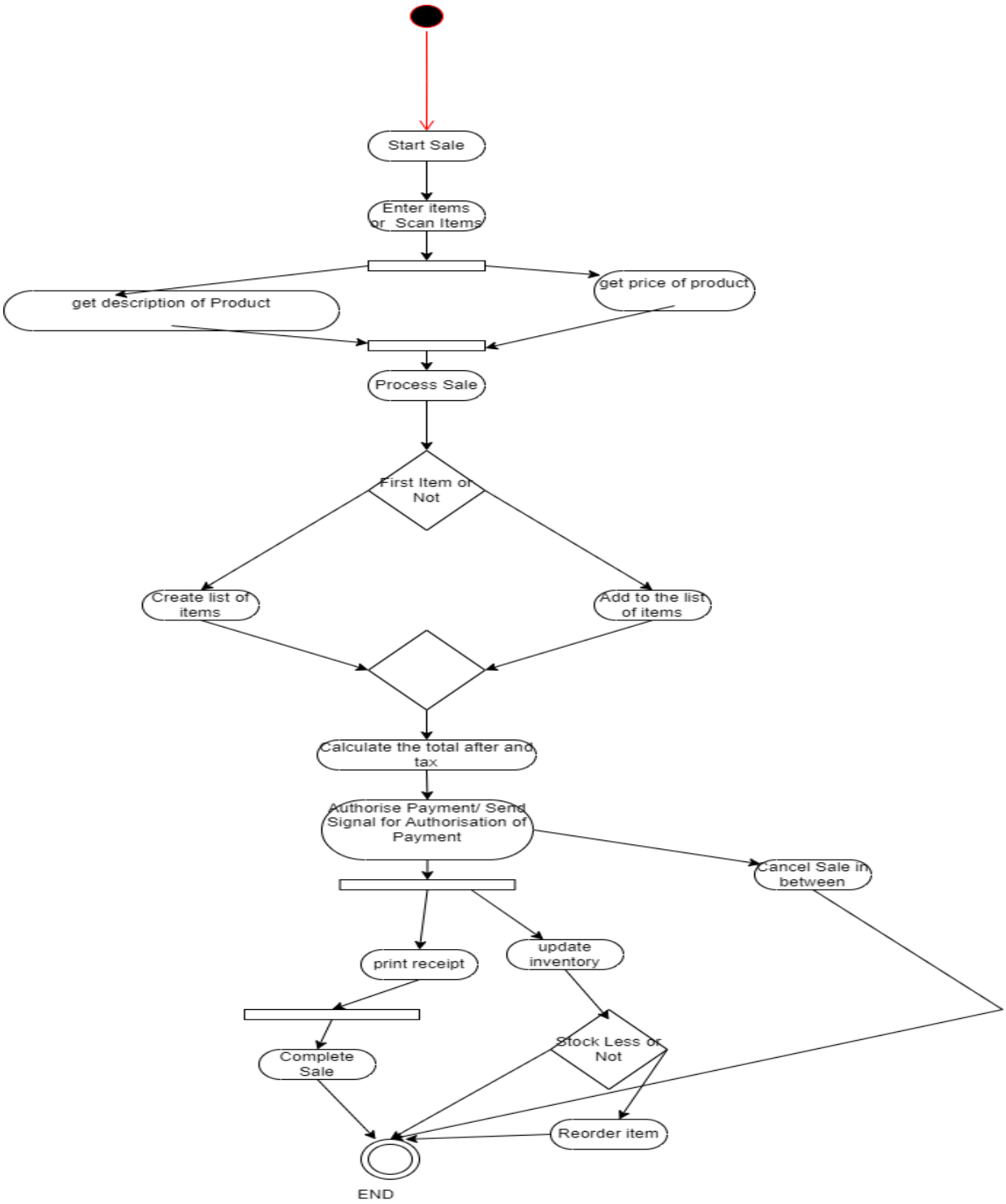


4) Develop Analysis Domain Models



5) Develop an activity diagram for "Process Sale" and "Handle Return" use cases.

a) Process Sale



b) Handle Return