E-Web-Store

AN ONLINE SHOPPING SYSTEM

Group Members: 1005012
1005035

Subsystems

- User Panel Management

- Warehouse Management

- Order and Delivery System

1. User Panel Management

1. Use Case Glossary

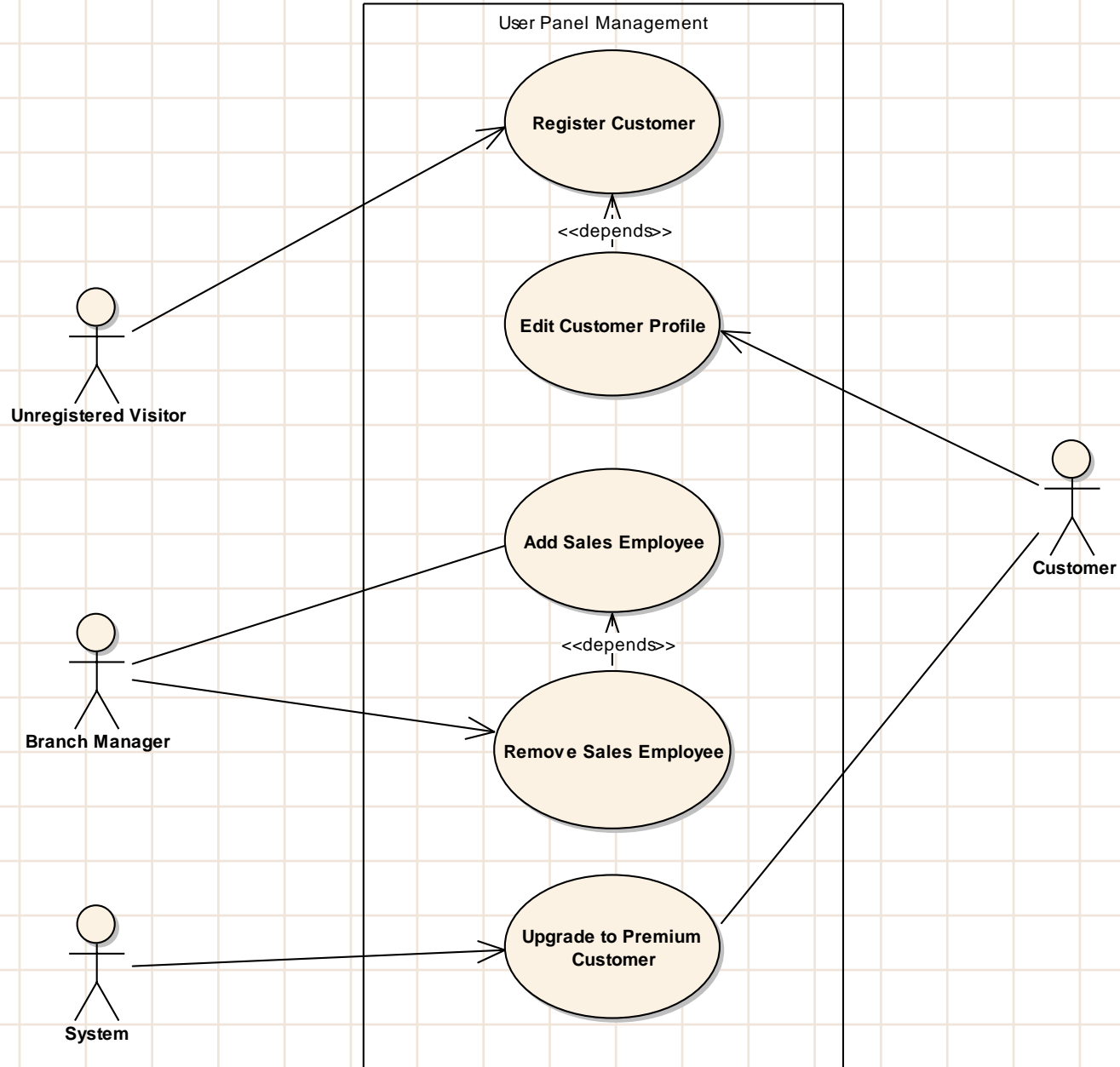| Name	Description	Actors
1.1 Register customer	A general person registers as a customer for the shop	Unregistered visitor
1.2 Edit customer profile	A customer edits his/her profile	Customer
1.3 Add sales employee	A branch manager adds a sales employee to the corresponding branch	Branch manager
1.4 Remove sales employee	A branch manager removes the profile of a sales employee of the corresponding branch	Branch manager
1.5 Upgrade to premium customer	System upgrades a customer to a premium customer status if certain conditions are met	System, Customer

uc 1. User Panel Management

User Panel Management

Register Customer

<<depends>>

Edit Customer Profile

Add Sales Employee

<<depends>>

Remove Sales Employee

Upgrade to Premium Customer

Unregistered Visitor

Branch Manager

System

Customer

Fig : Use Case Diagram – User Panel Management

act 1. User Panel Management

Customer creates account → [valid data] → Register new customer account → Customer Account

[invalid data]

Update customer account → [valid data]

[invalid data]

Show customer account ← «datastore» Customer Accounts ← Update database

Add sales person → [valid data] → Register new sales person account → Sales Person Account

[invalid data]

«datastore» Sales Person Accounts ← Update database

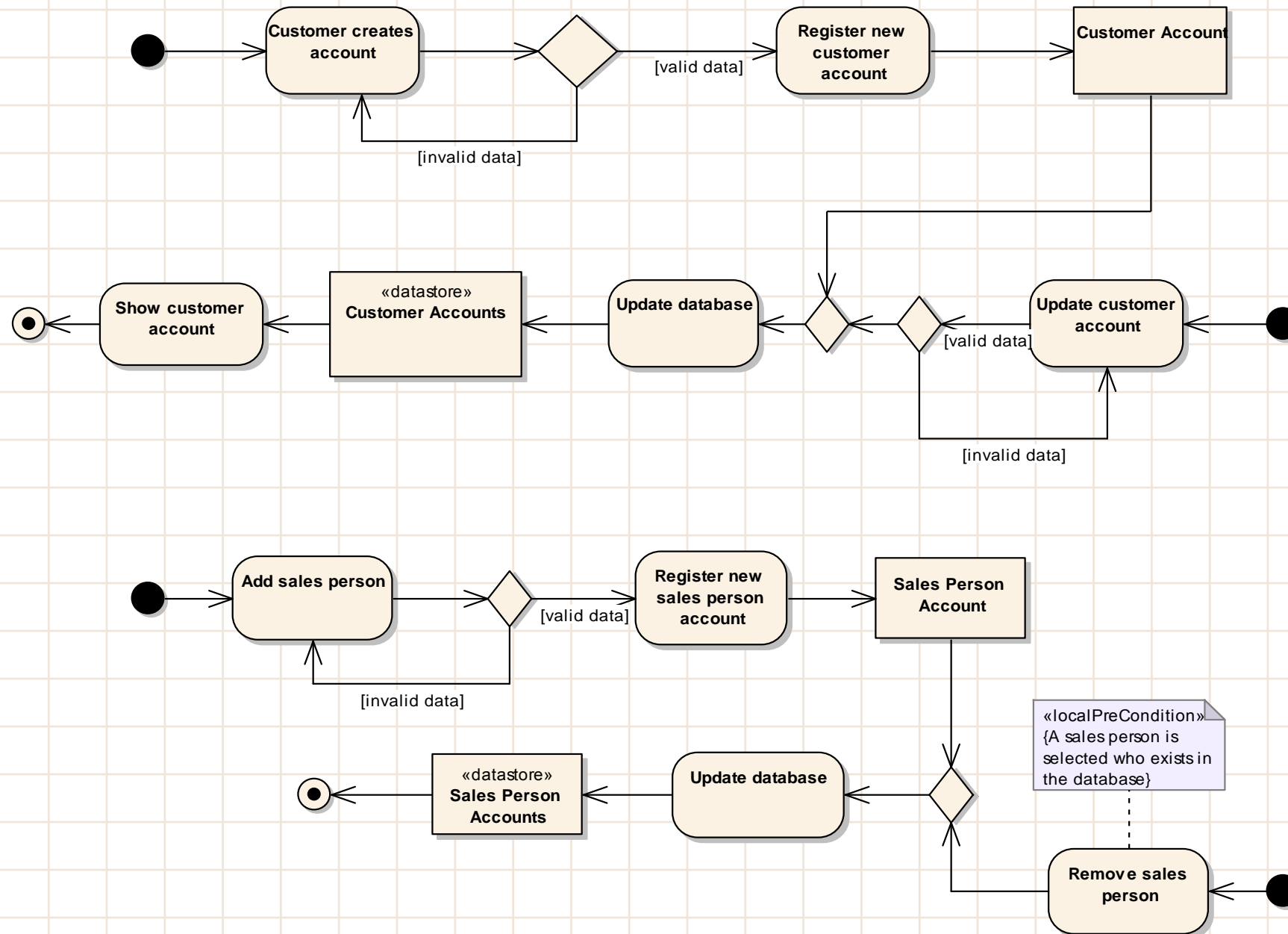«localPreCondition» {A sales person is selected who exists in the database}

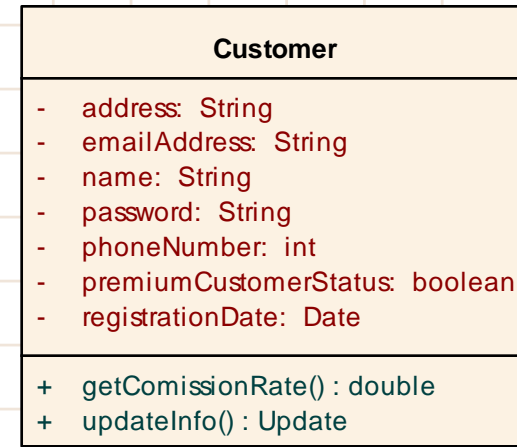Remove sales person

Fig: Control Flow – User Panel Management
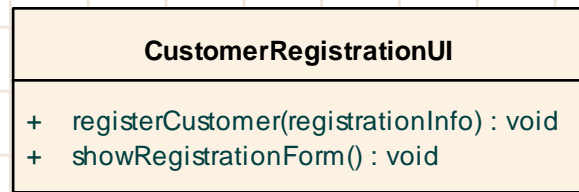
Fig: Class Diagram –
1.1 Register Customer

class 1.2 EditCustomerProfile

EditCustomerProfileUI

+ displayCustomerProfile() : void
+ updateCustomerInfo(newCustomerInfo, customer) : void

Customer

- address: String
- emailAddress: String
- name: String
- password: String
- phoneNumber: int
- premiumCustomerStatus: boolean
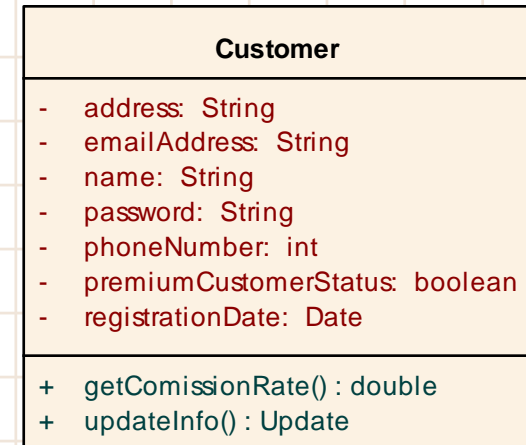- registrationDate: Date

+ getComissionRate() : double
+ updateInfo() : Update
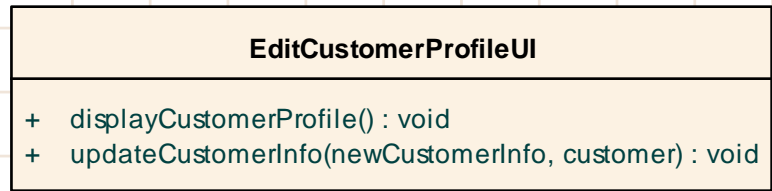
Fig: Class Diagram –
1.2 Edit Customer Profile

class 1.3 Add Sales Employee

AddSalesEmployeeUI

+　addSalesEmployee(salesEmployee) : void
+　showRegistrationForm() : void

SalesEmployee

-　address: String
-　available: boolean
-　branchName: String
-　email: String
-　joinDate: Date
-　name: String

+　isAvailable() : boolean
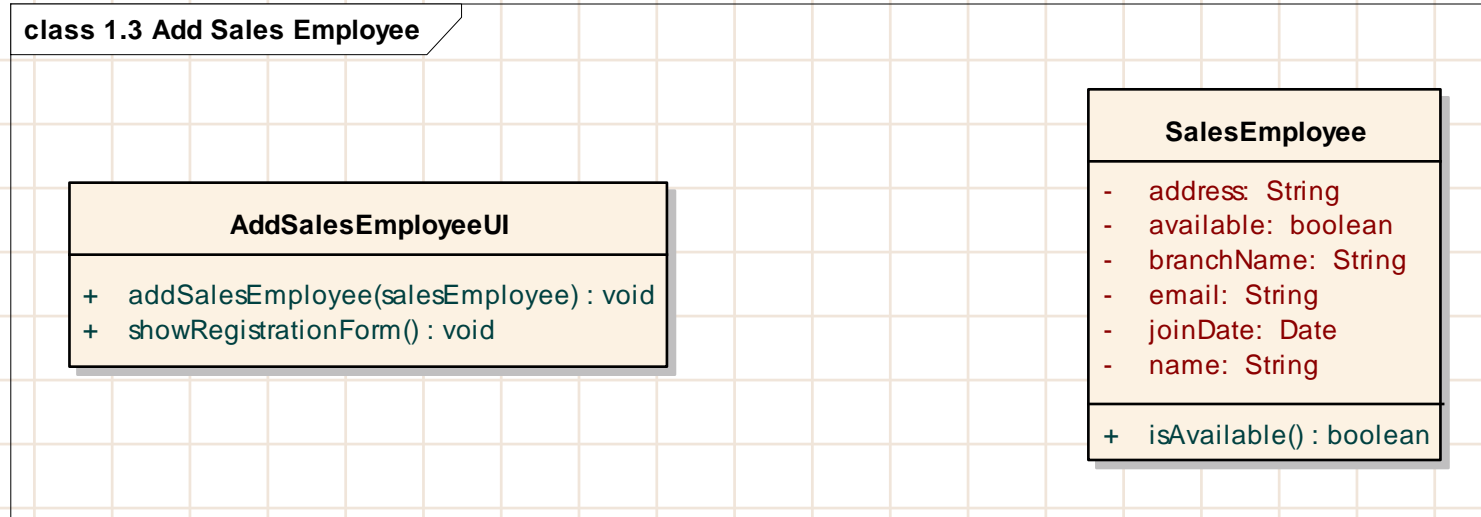
Fig: Class Diagram – 1.3
Add Sales Employee

class 1.4 Remove Sales Employee

RemoveSalesEmployeeUI

+ displayUI() : void
+ removeSalesEmployee(salesEmployee) : void

SalesEmployee

- address: String
- available: boolean
- branchName: String
- email: String
- joinDate: Date
- name: String

+ isAvailable() : boolean
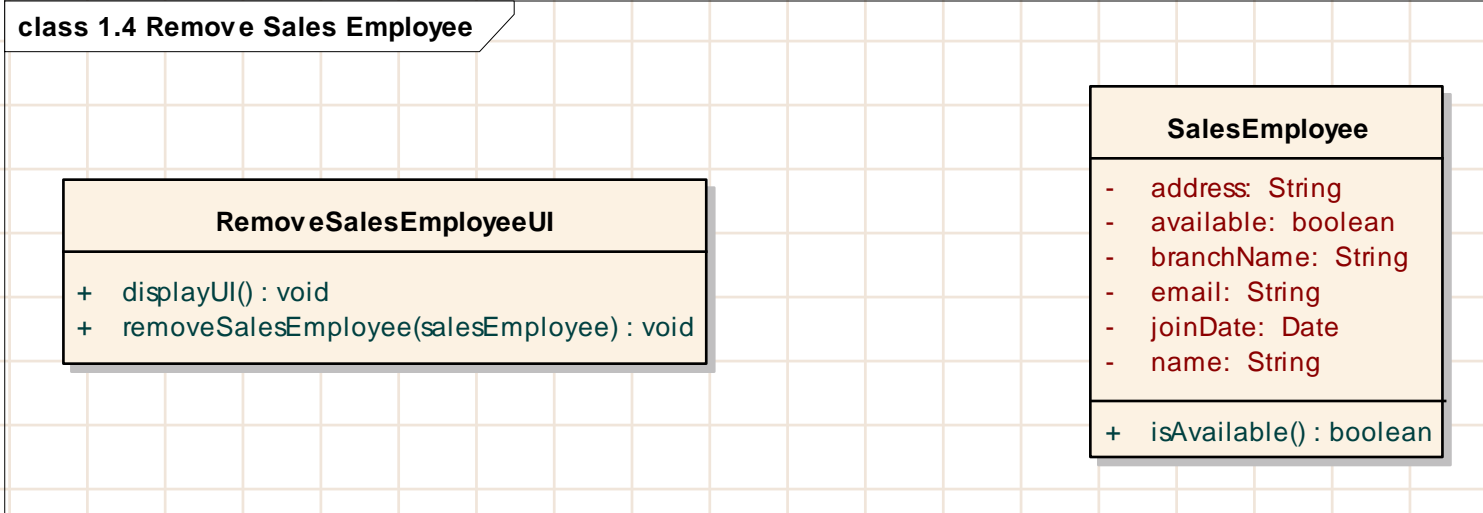
Fig: Class Diagram –
1.4 Remove Sales Employee

2. Warehouse Management

2. Use Case Glossary

| Name	Description	Actors
2.1 Add products	A branch manager adds some products to the corresponding branch	Branch manager
2.2 Update Product Information	A branch manager updates any product information necessary	Branch Manager
2.3 Products withdrawal	A sales employee specified by the branch manager withdraws some products from the warehouse according to some order	Sales employee

uc 2. Warehouse Management

Warehouse Management

Add Products

Update Product Information

Product Withdrawal

Branch Manager

Sales Person
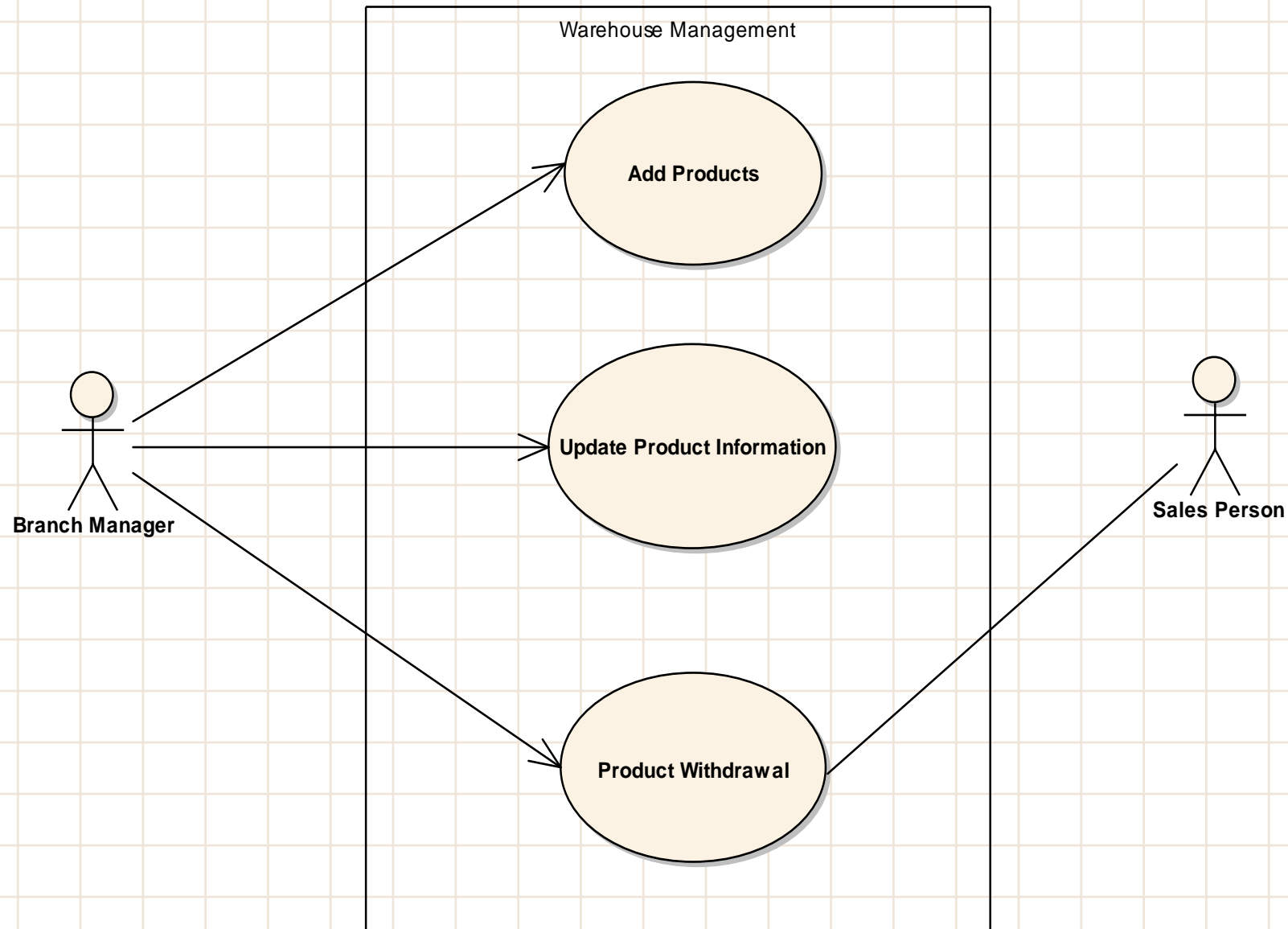
Fig : Use Case Diagram – Warehouse Management
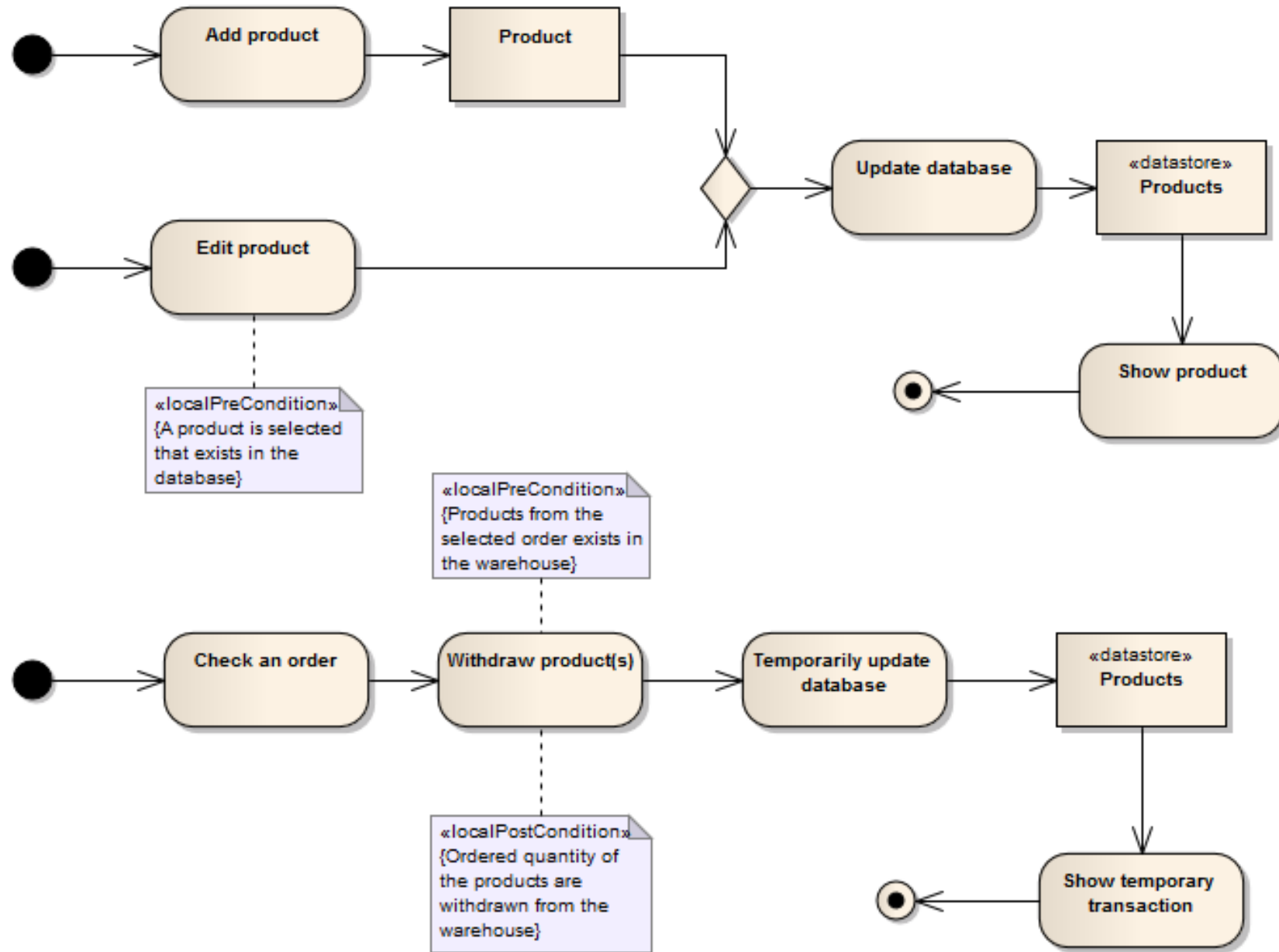
Fig: Control Flow – Warehouse Management

class 2.1 Add Product

AddProductUI

+ addProduct(form) : void
+ displayAddProductForm() : void

Product

- branchName: String
- brand: String
- category: String
- orderQuantity: int
- price: double
- productDetails: String
- productID: int
- productImage: Image
- productInStrock: int
- productName: String
- soldCount: int
- subCategory: String
- withdrawCount: int

+ getFinalPrice(premiumUserStatus) : double
+ returnProduct(returnAmount) : void
+ updatePrice(newPrice) : void
+ withdraw(withdrawAmount) : void

Fig: Class Diagram –
2.1 Add Product

class 2.2 Update Product Information

UpdateProductUI

+   loadProductInfo(productID) : void
+   updateProduct(newProductInfo, product) : void

Product

-   branchName: String
-   brand: String
-   category: String
-   orderQuantity: int
-   price: double
-   productDetails: String
-   productID: int
-   productImage: Image
-   productInStrock: int
-   productName: String
-   soldCount: int
-   subCategory: String
-   withdrawCount: int

+   getFinalPrice(premiumUserStatus) : double
+   returnProduct(returnAmount) : void
+   updatePrice(newPrice) : void
+   withdraw(withdrawAmount) : void

Fig: Class Diagram –
2.2 Update Product Information

class 2.3 Product Withdrawal

ProductWithdrawalUI

+ loadProducts(orders) : void
+ withdrawProduct(product) : void

Product

- branchName: String
- brand: String
- category: String
- orderQuantity: int
- price: double
- productDetails: String
- productID: int
- productImage: Image
- productInStrock: int
- productName: String
- soldCount: int
- subCategory: String
- withdrawCount: int

+ getFinalPrice(premiumUserStatus) : double
+ returnProduct(returnAmount) : void
+ updatePrice(newPrice) : void
+ withdraw(withdrawAmount) : void

Order

- deliveryLocation: String
- orderProducts: Product[]

+ addProductToCart(Product) : void
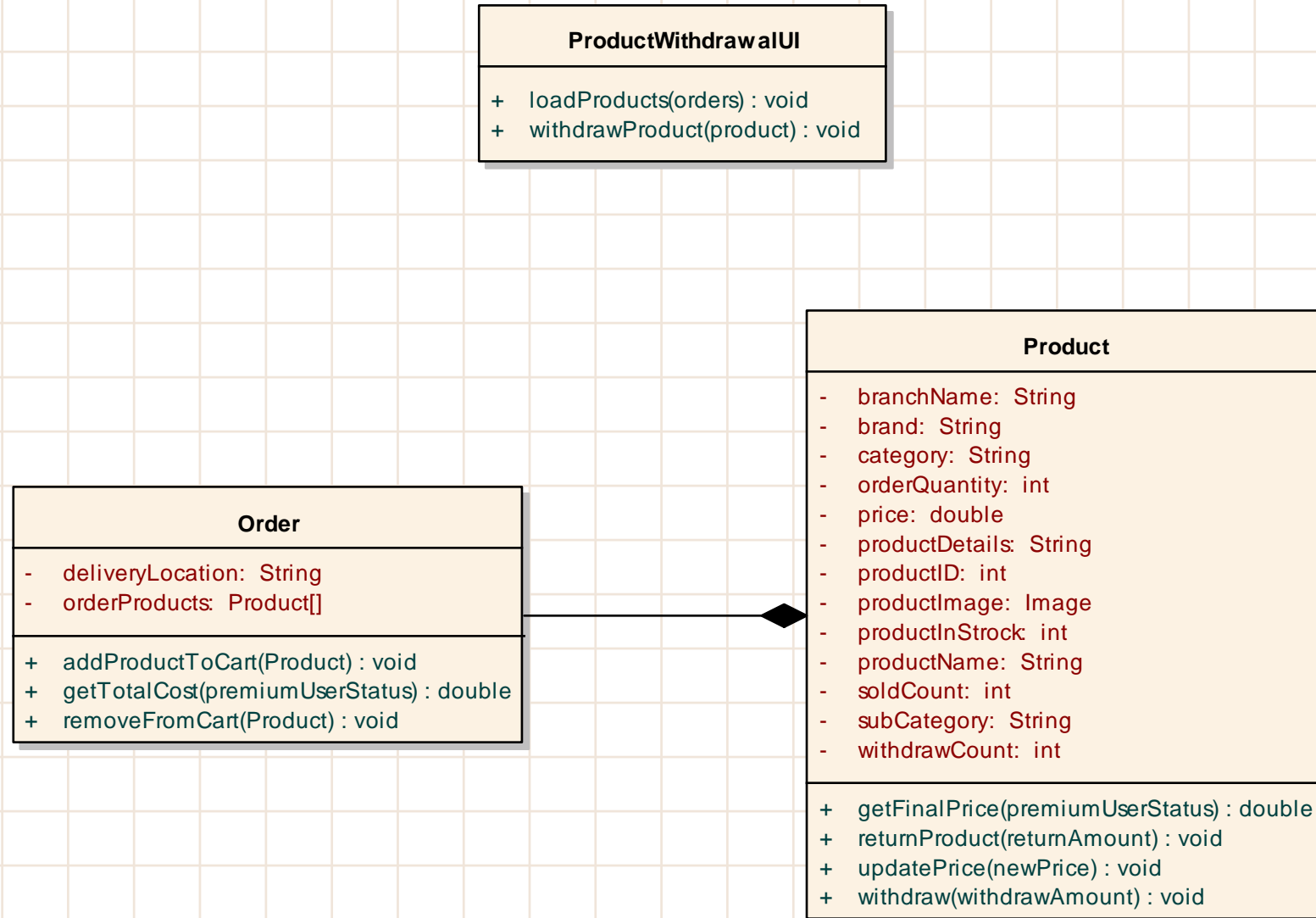+ getTotalCost(premiumUserStatus) : double
+ removeFromCart(Product) : void

Fig: Class Diagram –
2.3 Product Withdrawal

3. Order and Delivery System

3. Use Case Glossary

| Use Case	Description	Participant Actors
3.1 Order Products	Customer orders one or more products providing delivery place	Customer, Branch Manager
3.2 Assign Sales Person for Delivery	Branch manager assigns a Sales Person for a delivery	Branch Manager, Sales Person
3.3 Collect Deliverable Products	Sales Person collects products for delivery	Sales Person
3.4 Deliver Products	Product is delivered to the Customer and payment received	Sales Person, Customer
3.5 Confirm Delivery	Sales Person reports delivery confirmation or failure to the Branch Manager	Sales Person, Branch Manager
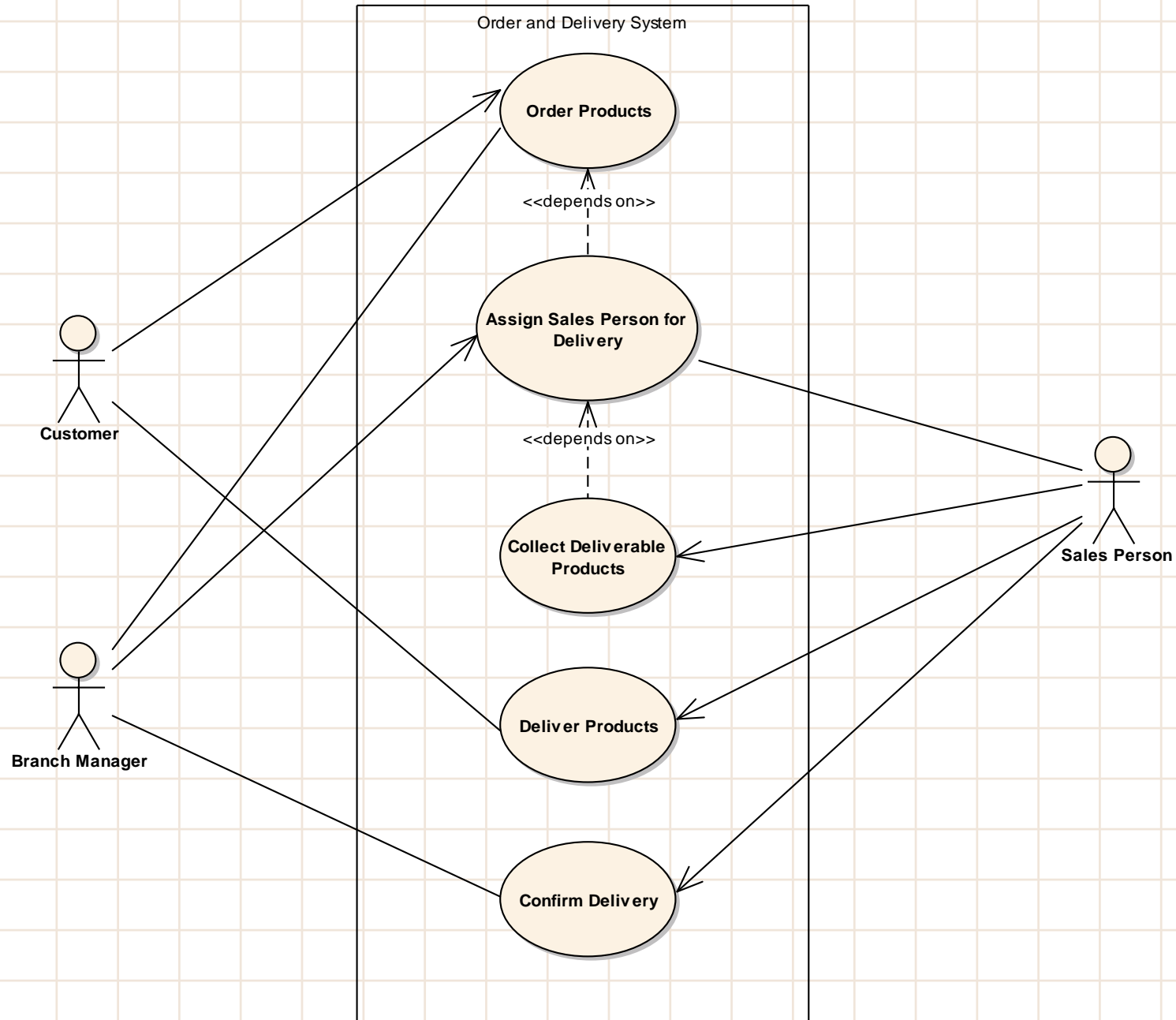
Fig : Use Case Diagram – Order and Delivery System
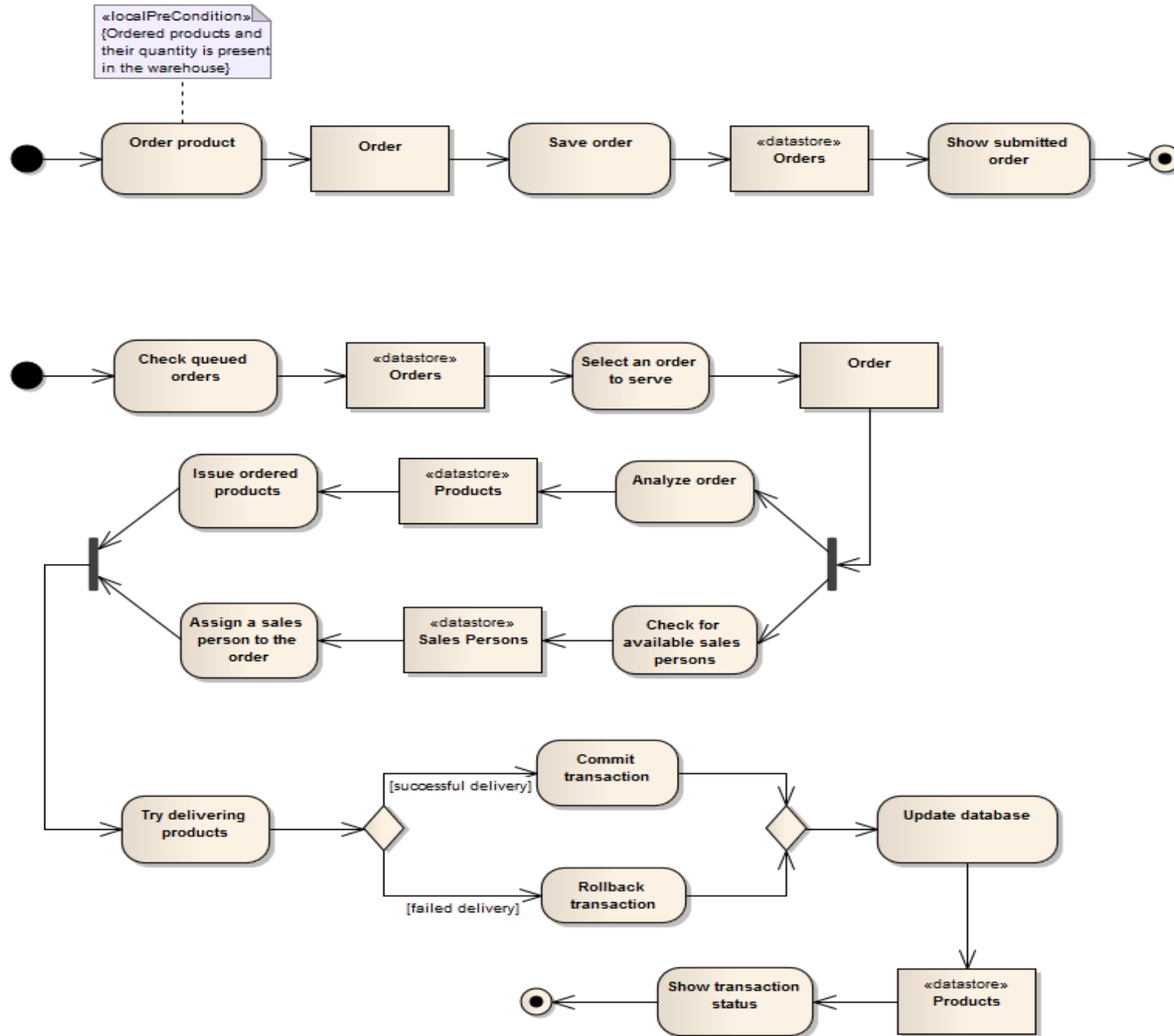
Fig: Control Flow – Order and Delivery System

BrowseAndOrderProductsUI

+ addProductToCart(Product) : void
+ checkout() : void
+ removeFromCart(Product) : void

Order

- deliveryLocation: String
- orderProducts: Product[]

+ addProductToCart(Product) : void
+ getTotalCost(premiumUserStatus) : double
+ removeFromCart(Product) : void

Product

- branchName: String
- brand: String
- category: String
- orderQuantity: int
- price: double
- productDetails: String
- productID: int
- productImage: Image
- productInStrock: int
- productName: String
- soldCount: int
- subCategory: String
- withdrawCount: int

+ getFinalPrice(premiumUserStatus) : double
+ returnProduct(returnAmount) : void
+ updatePrice(newPrice) : void
+ withdraw(withdrawAmount) : void

Fig: Class Diagram –
3.1 Order Products

class 3.2 Assign Sales Person For Delivery

OrderAssignmentUI

+ assignOrder(order, salesEmployee) : void
+ getAvailableSalesEmployee() : SalesEmployee[]

Order

- deliveryLocation: String
- orderProducts: Product[]

+ addProductToCart(Product) : void
+ getTotalCost(premiumUserStatus) : double
+ removeFromCart(Product) : void

DeliveryAssignmentManager

- branchName: String
- orders: Order[]
- salesEmployee: SalesEmployee[]

+ assignOrder(salesPerson, order) : void
+ getAvailableSalesPersons() : SalesPerson[]
+ getNextOrder() : Order

SalesEmployee

- address: String
- available: boolean
- branchName: String
- email: String
- joinDate: Date
- name: String

+ isAvailable() : boolean
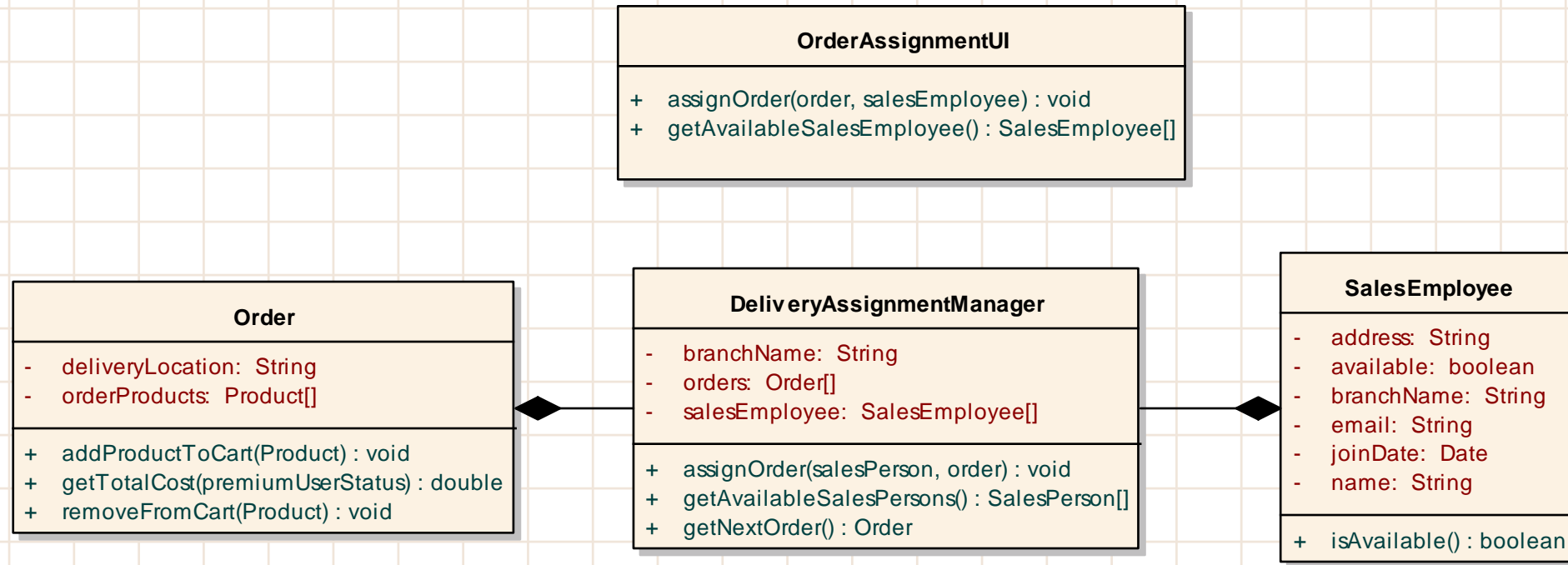
Fig: Class Diagram –
3.2 Assign Sales Person for
Delivery

class 3.5 Confirm Delivery

DeliveryConfirmUI

+ confirmDelivery() : void
+ failedDelivery() : void

SalesEmployee

- address: String
- available: boolean
- branchName: String
- email: String
- joinDate: Date
- name: String

+ isAvailable() : boolean

DeliveryConfirmationManager

- order: Order
- salesEmployee: SalesEmployee

+ confirmDelivery() : void
+ failedDelivery() : void

Order

- deliveryLocation: String
- orderProducts: Product[]

+ addProductToCart(Product) : void
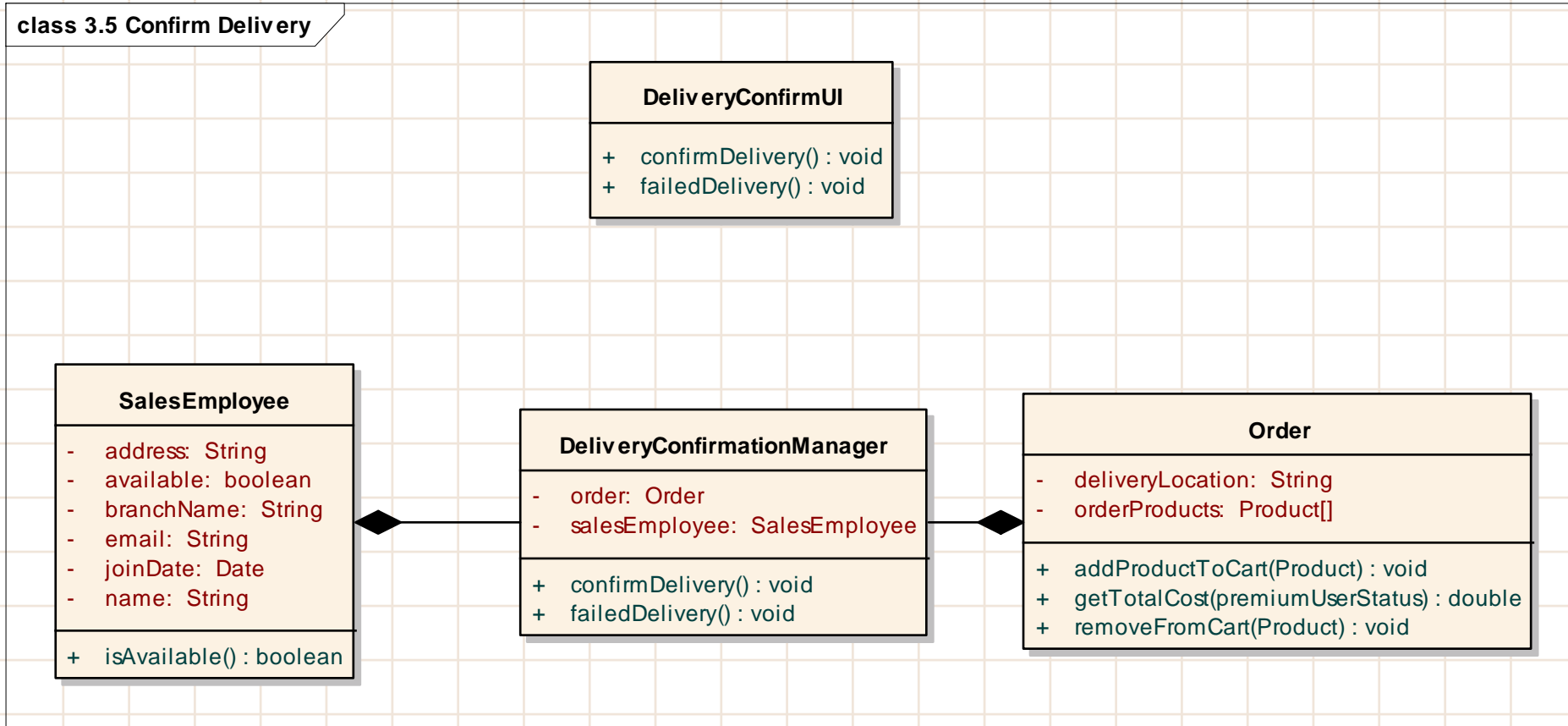+ getTotalCost(premiumUserStatus) : double
+ removeFromCart(Product) : void

Fig: Class Diagram –
3.5 Confirm Delivery

Thank You