Prompt Use Case diagram
Consider that you are a Solutions Architect
Create **use-case diagram** in Plant UML format for the following functionality

Authentication

Customer will have the following Attributes: Username (unique), Password, Email, Postal Address, Member/Non-member status.

Manager will have the following Attributes: Hardcoded Username, Password, Email Address.

Login/Logout: Customers and managers can log in and log out; login enables logout and checkout, with a 30-minute session timeout. No password recovery available.

Membership

Customers can sign up to become members or remain non-members. Membership status is captured as a boolean value.

Shopping Cart-Customers can add one or more books to the shopping cart. Duplicate copies of the same book are not allowed.

Inventory Management

Book will have following Attributes: Title, Author, Price, Stock/Quantity.

Stop-Order Management: Each book has a stop-order status (on/off). If stock falls below a threshold, managers are notified unless a stop-order is applied.

Manager will perform following Actions: Update stock quantities, change book prices, set reorder thresholds, and view transaction logs. If stock reaches the reorder threshold, an email is sent to the manager, and items are automatically reordered unless a stop-order is active.

Promotions

Promotion Attributes: Percentage discount, expiration date.

Promotion Use: Available to members only; one promotion code per purchase.

Manager Actions: Create/update promotions, define discounts and expiration dates. Promotions are auto-deleted after expiration, and emails are sent to all members.

Checkout

Checkout is restricted to logged-in customers. During checkout:

A 16-digit credit card number is collected.

Transactions are logged and recorded.

Order details are sent via email.

Inventory stock is decremented based on purchased items.

PlantUML generated by GitHub Copilot

```
@startuml
```

```
actor Customer
actor Manager

usecase "Sign Up" as UC1
usecase "Login" as UC2
usecase "Logout" as UC3
usecase "Auto Logout" as UC4
usecase "Checkout" as UC5
usecase "Add to Cart" as UC6
usecase "Maintain Inventory" as UC7
usecase "Update Stock" as UC8
usecase "Set Prices" as UC9
usecase "View Logs" as UC10
usecase "Specify Stop-Orders" as UC11
usecase "Create Promotions" as UC12
usecase "Email Promotions" as UC13
usecase "Auto Delete Promotions" as UC14

Customer --> UC1
Customer --> UC2
Customer --> UC3
Customer --> UC4
Customer --> UC5
Customer --> UC6

Manager --> UC2
Manager --> UC3
Manager --> UC7
Manager --> UC8
Manager --> UC9
Manager --> UC10
Manager --> UC11
Manager --> UC12
Manager --> UC13
Manager --> UC14

UC2 --> UC3 : <<include>>
UC2 --> UC5 : <<include>>
UC2 --> UC4 : <<include>>
UC12 --> UC13 : <<include>>
UC12 --> UC14 : <<include>>

@enduml
```

Similarity Score

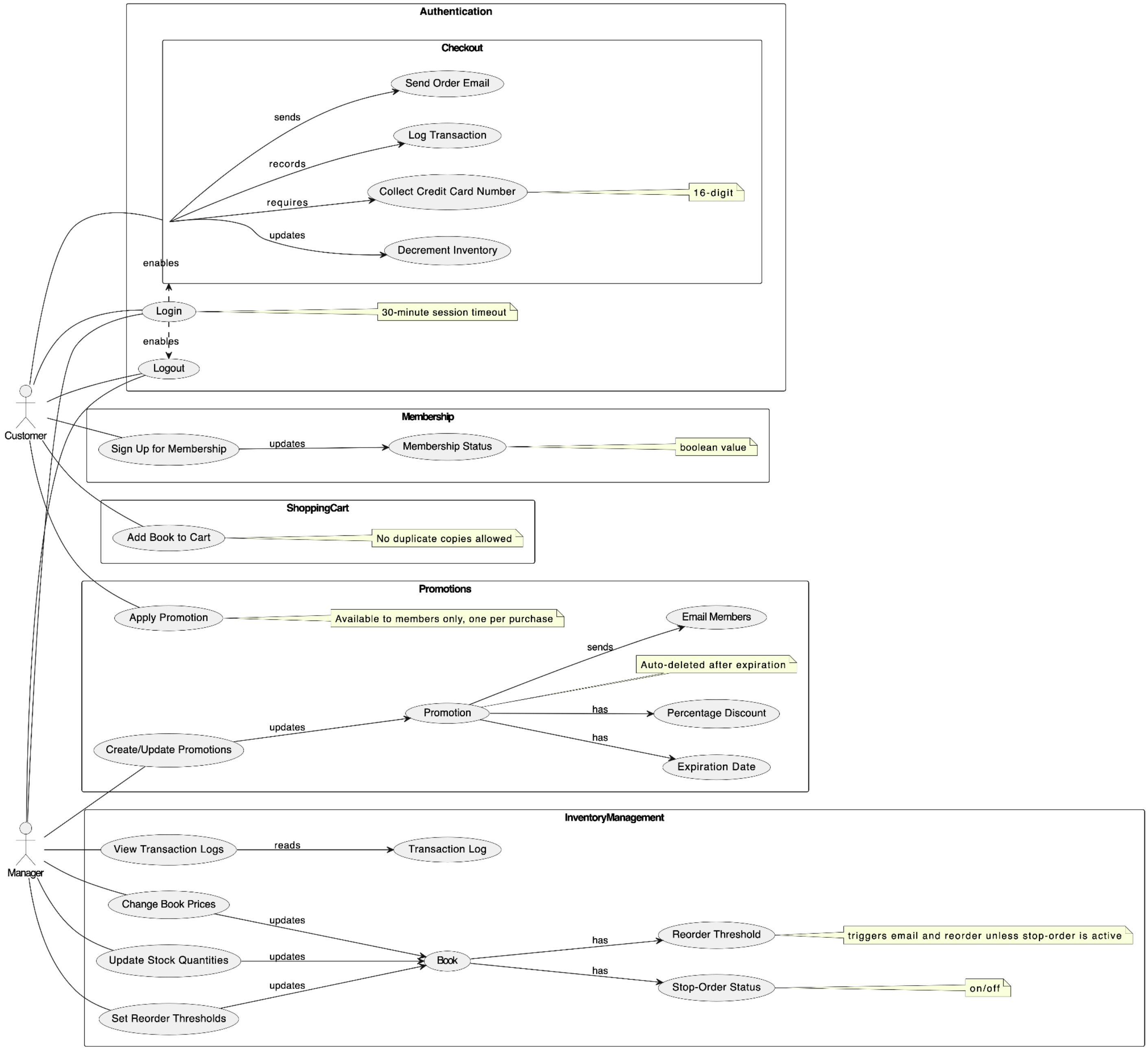
V1	13
V2	18

E1	17
E2	26
Mv	12
ME	12
Node Similarity Ratio	0.66666667
Edge Similarity Ratio	0.46153846
Similarity Score	0.56410256

PlantUML generated by BlackBox AI

```
@startuml
left to right direction

actor Customer
actor Manager

rectangle Authentication {
  Customer -- (Login)
  Customer -- (Logout)
  Manager -- (Login)
  Manager -- (Logout)
  (Login) .> (Logout) : enables
  (Login) .> (Checkout) : enables
  note right of (Login) : 30-minute session timeout
}

rectangle Membership {
  Customer -- (Sign Up for Membership)
  (Sign Up for Membership) --> (Membership Status) : updates
  note right of (Membership Status) : boolean value
}

rectangle ShoppingCart {
  Customer -- (Add Book to Cart)
  note right of (Add Book to Cart) : No duplicate copies allowed
}
```

```
rectangle InventoryManagement {
    Manager -- (Update Stock Quantities)
    Manager -- (Change Book Prices)
    Manager -- (Set Reorder Thresholds)
    Manager -- (View Transaction Logs)
    (Update Stock Quantities) --> (Book) : updates
    (Change Book Prices) --> (Book) : updates
    (Set Reorder Thresholds) --> (Book) : updates
    (View Transaction Logs) --> (Transaction Log) : reads
    (Book) --> (Stop-Order Status) : has
    note right of (Stop-Order Status) : on/off
    (Book) --> (Reorder Threshold) : has
    note right of (Reorder Threshold) : triggers email and reorder unless stop-order is active
}

rectangle Promotions {
    Manager -- (Create/Update Promotions)
    (Create/Update Promotions) --> (Promotion) : updates
    (Promotion) --> (Expiration Date) : has
    (Promotion) --> (Percentage Discount) : has
    note right of (Promotion) : Auto-deleted after expiration
    (Promotion) --> (Email Members) : sends
    Customer -- (Apply Promotion)
    note right of (Apply Promotion) : Available to members only, one per purchase
}

rectangle Checkout {
    Customer -- (Checkout)
    (Checkout) --> (Collect Credit Card Number) : requires
    note right of (Collect Credit Card Number) : 16-digit
    (Checkout) --> (Log Transaction) : records
    (Checkout) --> (Send Order Email) : sends
    (Checkout) --> (Decrement Inventory) : updates
}
```



@enduml

V1	13
V2	23
E1	18

E2	29
Mv	8
ME	16
Node Similarity Ratio	0.34782609
Edge Similarity Ratio	0.55172414
Similarity Score	0.44977511

PlantUML generated by ChatGPT

```
@startuml

actor Customer
actor Manager

rectangle "Authentication" {
    Customer -- (Login)
    Customer -- (Logout)
    Manager -- (Login)
    Manager -- (Logout)
}

rectangle "Membership" {
    Customer -- (Sign Up for Membership)
}

rectangle "Shopping Cart" {
    Customer -- (Add Book to Cart)
    Customer -- (Remove Book from Cart)
    Customer -- (View Cart)
}

rectangle "Inventory Management" {
    Manager -- (Update Stock)
    Manager -- (Change Book Prices)
    Manager -- (Set Reorder Threshold)
    Manager -- (View Transaction Logs)
    Manager -- (Receive Low Stock Alerts)
    Manager -- (Auto-Reorder Books)
    (Auto-Reorder Books) ..> (Receive Low Stock Alerts) : "Triggered When Threshold Reached"
}

rectangle "Promotions" {
    Manager -- (Create Promotion)
    Manager -- (Update Promotion)
    Manager -- (Set Expiry Date)
    Manager -- (Send Promotion Emails)
    Customer -- (Apply Promotion Code)
}

rectangle "Checkout" {
    Customer -- (Proceed to Checkout)
    (Proceed to Checkout) -- (Enter Credit Card Details)
    (Proceed to Checkout) -- (Receive Order Confirmation Email)
    (Proceed to Checkout) -- (Inventory Stock Decrement)
    (Proceed to Checkout) -- (Log Transaction)
}

@enduml
```

V1	13
V2	22
E1	18
E2	22
Mv	12
ME	16
Node Similarity Ratio	0.54545455
Edge Similarity Ratio	0.72727273
Similarity Score	0.63636364

Adjusted Prompt
Create **use-case diagram** in Plant UML format for the following functionality

Authentication

Customer will have the following Attributes: Username (unique), Password, Email, Postal Address, Member/Non-member status.

Manager will have the following Attributes: Hardcoded Username, Password, Email Address.

Login/Logout: Customers and managers can log in and log out; login enables logout and checkout, with a 30-minute session timeout. No password recovery available.

Membership

Customers can sign up to become members or remain non-members. Membership status is captured as a boolean value.

Shopping Cart-Customers can add one or more books to the shopping cart. Duplicate copies of the same book are not allowed.

Inventory Management

Book will have following Attributes: Title, Author, Price, Stock/Quantity.

Stop-Order Management: Each book has a stop-order status (on/off). If stock falls below a threshold, managers are notified unless a stop-order is applied.

Manager will perform following Actions: Update stock quantities, change book prices, set reorder thresholds, and view transaction logs.
System will send an email to the manager, If stock reaches the reorder threshold, . items are automatically reordered unless a stop-order is active.

Promotions

Promotion Attributes: Percentage discount, expiration date.

Promotion Use: Available to members only; one promotion code per purchase.

Manager Actions: Create/update promotions, define discounts and expiration dates. Promotions are auto-deleted after expiration, and emails are sent to all members.

Checkout

Checkout is restricted to logged-in customers. During checkout:

A 16-digit credit card number is collected.

Transactions are logged and recorded.

Order details are sent via email.

Inventory stock is decremented based on purchased items.

**```plantuml**

~~~
@startuml
actor Customer
actor Manager

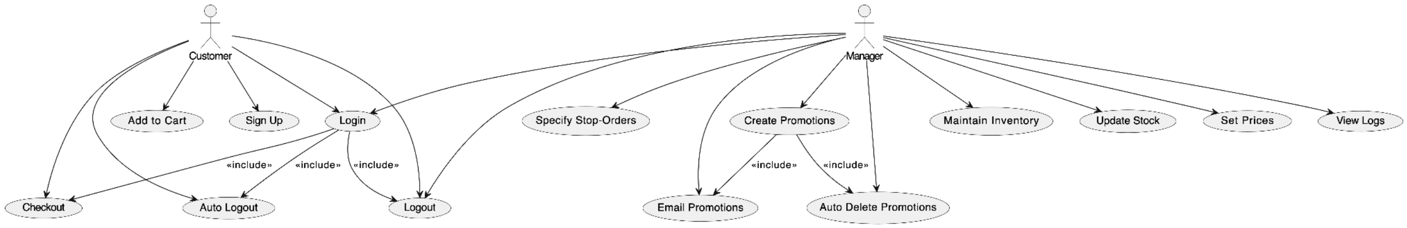
usecase "Sign Up" as UC1
usecase "Login" as UC2
usecase "Logout" as UC3
usecase "Auto Logout" as UC4
usecase "Checkout" as UC5
usecase "Add to Cart" as UC6
usecase "Maintain Inventory" as UC7
usecase "Update Stock" as UC8
usecase "Set Prices" as UC9
usecase "View Logs" as UC10
usecase "Specify Stop-Orders" as UC11
usecase "Create Promotions" as UC12
usecase "Email Promotions" as UC13
usecase "Auto Delete Promotions" as UC14

Customer --> UC1
Customer --> UC2
Customer --> UC3
Customer --> UC4
Customer --> UC5
Customer --> UC6

Manager --> UC2
Manager --> UC3
Manager --> UC7
Manager --> UC8
Manager --> UC9
Manager --> UC10
Manager --> UC11
Manager --> UC12
Manager --> UC13
Manager --> UC14

UC2 --> UC3 : <<include>>
UC2 --> UC5 : <<include>>
UC2 --> UC4 : <<include>>
UC12 --> UC13 : <<include>>
UC12 --> UC14 : <<include>>

@enduml
```
~~~



Similarity Score

V1	13
V2	18
E1	17
E2	26
Mv	13
ME	13
Node Similarity Ratio	0.72222222
Edge Similarity Ratio	0.5
Similarity Score	0.61111111

Prompt

Consider that you are a Solutions Architect
Create class diagram for the following functionality
Create class diagram in Plant UML Diagram for the following system

Authentication- Customer will have the following Attributes: Username (unique), Password, Email, Postal Address, Member/Non-member status.

Manager will have the following Attributes: Hardcoded Username, Password, Email Address.

Login/Logout: Customers and managers can log in and log out; login enables logout and checkout, with a 30-minute session timeout. No password recovery available.

Membership-Customers can sign up to become members or remain non-members. Membership status is captured as a boolean value.

Shopping Cart-Customers can add one or more books to the shopping cart. Duplicate copies of the same book are not allowed.

Inventory Management

Book will have following Attributes: Title, Author, Price, Stock/Quantity.

Stop-Order Management: Each book has a stop-order status (on/off). If stock falls below a threshold, managers are notified unless a stop-order is applied.

Manager will perform following Actions: Update stock quantities, change book prices, set reorder thresholds, and view transaction logs. If stock reaches the reorder threshold, an email is sent to the manager, and items are automatically reordered unless a stop-order is active.

Promotions

Promotion Attributes: Percentage discount, expiration date.

Promotion Use: Available to members only; one promotion code per purchase.

Manager Actions: Create/update promotions, define discounts and expiration dates. Promotions are auto-deleted after expiration, and emails are sent to all members.

Checkout

Checkout is restricted to logged-in customers. During checkout:

A 16-digit credit card number is collected.

Transactions are logged and recorded.

Order details are sent via email.

Inventory stock is decremented based on purchased items.

PlantUML generated by GitHubCopilot

```
@startuml

class Customer {
  - String username
  - String password
  - String email
  - String postalAddress
  - boolean memberStatus
  + login()
  + logout()
  + signUp()
  + addToCart(Book book)
  + checkout(Cart cart, String creditCardNumber)
}

class Manager {
  - String username
  - String password
  - String email
  + login()
  + logout()
  + updateStock(Book book, int quantity)
  + changePrice(Book book, double newPrice)
  + setReorderThreshold(Book book, int threshold)
  + viewTransactionLogs()
  + createPromotion(Promotion promotion)
}

class Book {
  - String title
  - String author
  - double price
  - int stock
  - boolean stopOrderStatus
}

class Cart {
  - List<Book> books
  + addBook(Book book)
  + removeBook(Book book)
}

class Promotion {
  - double discountPercentage
  - Date expirationDate
}

class InventoryService {
  + checkStock(Book book)
  + updateInventory(Cart cart)
  + notifyManager(Book book)
}

class EmailService {
  + sendOrderEmail(String orderDetails)
  + sendPromotionEmail(Promotion promotion)
  + sendReorderNotification(Book book, String managerEmail)
}

Customer --> Cart : has
Customer --> Book : adds
Manager --> Book : manages
Manager --> Promotion : creates
InventoryService --> Book : manages
InventoryService --> Manager : notifies
EmailService --> Manager : notifies
EmailService --> Customer : notifies

@enduml
```

Similarity Score

V1	10
V2	7
E1	13
E2	7
Mv	7
ME	7
Node Similarity Ratio	0.7
Edge Similarity Ratio	0.53846154
Similarity Score	0.61923077

PlantUML Generated by BlackBoxAI

```
@startuml

' Classes
class Customer {
    +String username (unique)
    +String password
    +String email
    +String postalAddress
    +boolean isMember
    +login()
    +logout()
    +signUpForMembership()
    +addToCart(Book book)
    +checkout()
}
```

```
class Manager {
    +String username (hardcoded)
    +String password
    +String email
    +login()
    +logout()
    +updateStock(Book book, int quantity)
    +changePrice(Book book, double newPrice)
    +setReorderThreshold(Book book, int threshold)
    +viewTransactionLogs()
    +createPromotion(Promotion promotion)
    +updatePromotion(Promotion promotion)
}

class Book {
    +String title
    +String author
    +double price
    +int stockQuantity
    +boolean stopOrderStatus
    +int reorderThreshold
    +decrementStock(int quantity)
}

class ShoppingCart {
    +List<Book> books
    +addBook(Book book)
    +removeBook(Book book)
    +checkout()
}

class Promotion {
    +double percentageDiscount
    +Date expirationDate
    +applyPromotion()
    +sendEmailToMembers()
}

class Transaction {
    +String transactionId
    +Date transactionDate
    +double totalAmount
    +String customerUsername
    +logTransaction()
}

class Checkout {
    +String creditCardNumber
    +processPayment()
    +sendOrderEmail()
    +decrementInventory(List<Book> books)
}

' Relationships
Customer "1" --> "1" ShoppingCart : owns
Customer "1" --> "0..*" Transaction : performs
Customer "1" --> "0..*" Promotion : applies
Manager "1" --> "0..*" Book : manages
Manager "1" --> "0..*" Promotion : creates/updates
ShoppingCart "1" --> "0..*" Book : contains
Checkout "1" --> "1" Transaction : creates
Checkout "1" --> "0..*" Book : decrements
Book "1" --> "0..1" Promotion : has

' Notes
note top of Customer : Login enables logout and checkout.\n30-minute session timeout.\nNo password recovery.
note top of Manager : Hardcoded username.\nManages inventory and promotions.
note top of Book : Stop-order status (on/off).\nReorder threshold triggers email and reorder unless stop-order is active.
note top of Promotion : Auto-deleted after expiration.\nEmails sent to all members.
note top of Checkout : Restricted to logged-in customers.\n16-digit credit card number collected.\nOrder details sent via email.

@enduml
```

V1	9
V2	7
E1	15
E2	10
Mv	5
ME	10
Node Similarity Ratio	0.55555556
Edge Similarity Ratio	0.66666667

Similarity Score	0.61111111

PlantUML Generated by ChatGPT

```
@startuml

class Customer {
    - username: String
    - password: String
    - email: String
    - postalAddress: String
    - isMember: Boolean
    + login()
    + logout()
    + signUpForMembership()
    + addToCart(Book)
    + checkout()
}

class Manager {
    - username: String
    - password: String
    - email: String
    + login()
    + logout()
    + updateStock(Book, int)
    + changeBookPrice(Book, float)
    + setReorderThreshold(Book, int)
    + viewTransactionLogs()
    + createPromotion(Promotion)
    + updatePromotion(Promotion)
    + setPromotionExpiry(Promotion, Date)
}

class Book {
    - title: String
    - author: String
    - price: float
    - stock: int
    - stopOrder: Boolean
}

class ShoppingCart {
    - books: List<Book>
    + addBook(Book)
    + removeBook(Book)
    + viewCart()
}

class Promotion {
    - discount: float
    - expirationDate: Date
    + applyPromotionCode(String)
}

class Checkout {
    - creditCardNumber: String
    + processPayment()
    + logTransaction()
    + sendOrderEmail()
    + decrementInventory(Book)
}

Customer -- ShoppingCart
Customer -- Checkout
Customer -- Promotion
Manager -- Book
Manager -- Promotion
Manager -- Checkout
ShoppingCart -- Book
Checkout -- Book

@enduml
```

V1	9
V2	6
E1	15
E2	8
Mv	5
ME	8
Node Similarity Ratio	0.55555556
Edge Similarity Ratio	0.53333333

Similarity Score	0.54444444

Prompt -Sequence diagram
Consider that you are a Solutions Architect
Create sequence diagram for the following functionality
Create Plant UML Sequence Diagram for the following system

Authentication- Customer will have the following Attributes: Username (unique), Password, Email, Postal Address, Member/Non-member status.

Manager will have the following Attributes: Hardcoded Username, Password, Email Address.

Login/Logout: Customers and managers can log in and log out; login enables logout and checkout, with a 30-minute session timeout. No password recovery available.

Membership-Customers can sign up to become members or remain non-members. Membership status is captured as a boolean value.

Shopping Cart-Customers can add one or more books to the shopping cart. Duplicate copies of the same book are not allowed.

Inventory Management

Book will have following Attributes: Title, Author, Price, Stock/Quantity.

Stop-Order Management: Each book has a stop-order status (on/off). If stock falls below a threshold, managers are notified unless a stop-order is applied.

Manager will perform following Actions: Update stock quantities, change book prices, set reorder thresholds, and view transaction logs. If stock reaches the reorder threshold, an email is sent to the manager, and items are automatically reordered unless a stop-order is active.

Promotions

Promotion Attributes: Percentage discount, expiration date.

Promotion Use: Available to members only; one promotion code per purchase.

Manager Actions: Create/update promotions, define discounts and expiration dates. Promotions are auto-deleted after expiration, and emails are sent to all members.

Checkout

Checkout is restricted to logged-in customers. During checkout:

A 16-digit credit card number is collected.

Transactions are logged and recorded.

Order details are sent via email.

Inventory stock is decremented based on purchased items.

Plant UML Generated by GitHub Copilot

```
@startuml

actor Customer
actor Manager

participant "AuthenticationService" as AuthService
participant "MembershipService" as MemberService
participant "CartService" as CartService
participant "InventoryService" as InventoryService
participant "PromotionService" as PromoService
participant "CheckoutService" as CheckoutService
participant "EmailService" as EmailService

Customer -> AuthService: login(username, password)
AuthService -> Customer: loginSuccess()
Customer -> AuthService: logout()
AuthService -> Customer: logoutSuccess()

Customer -> MemberService: signup(username, password, email, postalAddress,
memberStatus)
MemberService -> Customer: signupSuccess()

Customer -> CartService: addToCart(bookId)
CartService -> InventoryService: checkStock(bookId)
InventoryService -> CartService: stockAvailable()
CartService -> Customer: bookAddedToCart()

Manager -> InventoryService: updateStock(bookId, quantity)
InventoryService -> Manager: stockUpdateSuccess()

Manager -> InventoryService: changePrice(bookId, newPrice)
InventoryService -> Manager: priceUpdateSuccess()

Manager -> InventoryService: setReorderThreshold(bookId, threshold)
InventoryService -> Manager: thresholdSetSuccess()

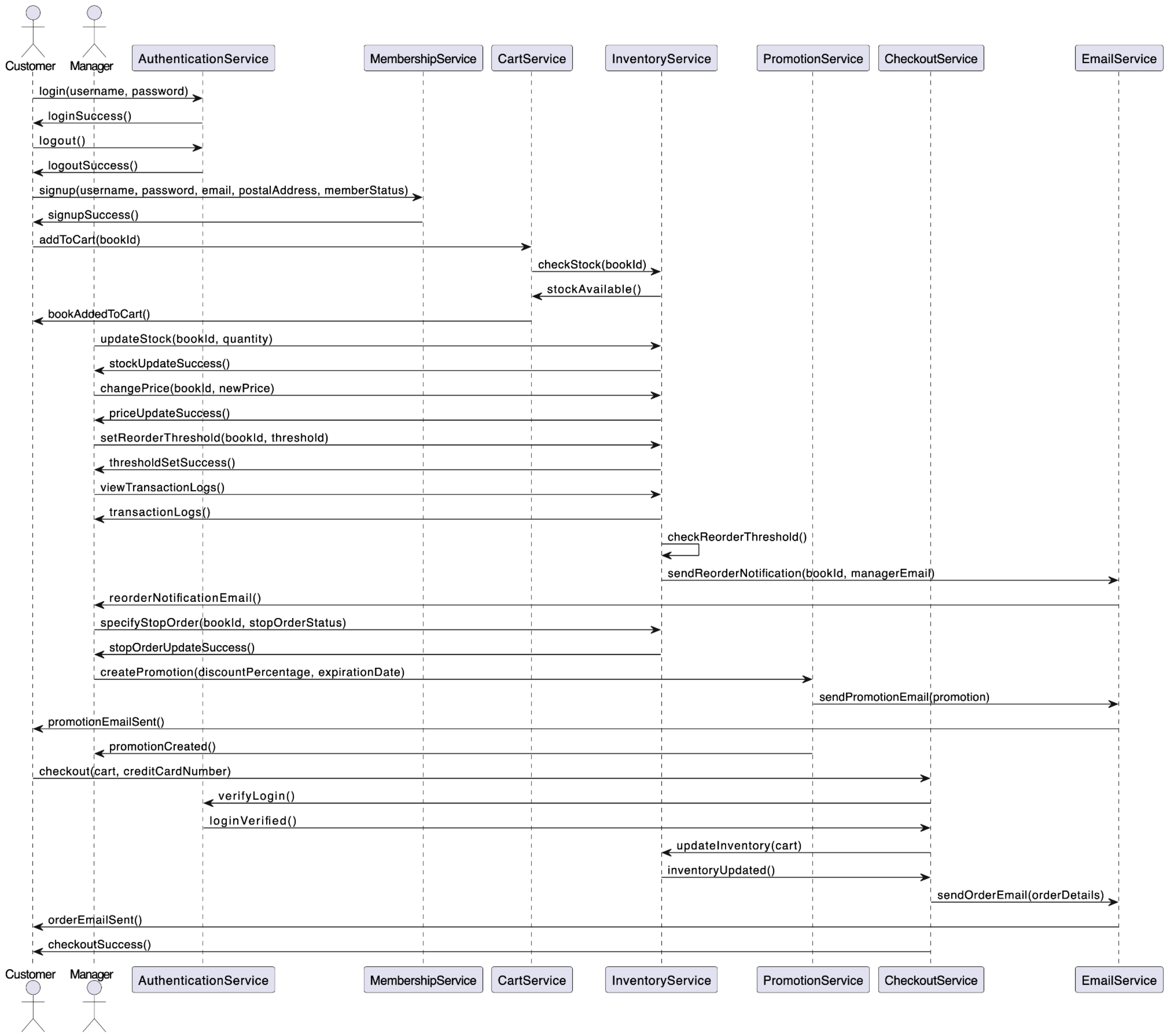
Manager -> InventoryService: viewTransactionLogs()
InventoryService -> Manager: transactionLogs()
InventoryService -> InventoryService: checkReorderThreshold()
InventoryService -> EmailService: sendReorderNotification(bookId, managerEmail)
EmailService -> Manager: reorderNotificationEmail()

Manager -> InventoryService: specifyStopOrder(bookId, stopOrderStatus)
InventoryService -> Manager: stopOrderUpdateSuccess()

Manager -> PromoService: createPromotion(discountPercentage, expirationDate)
PromoService -> EmailService: sendPromotionEmail(promotion)
EmailService -> Customer: promotionEmailSent()
PromoService -> Manager: promotionCreated()

Customer -> CheckoutService: checkout(cart, creditCardNumber)
CheckoutService -> AuthService: verifyLogin()
AuthService -> CheckoutService: loginVerified()
CheckoutService -> InventoryService: updateInventory(cart)
InventoryService -> CheckoutService: inventoryUpdated()
CheckoutService -> EmailService: sendOrderEmail(orderDetails)
EmailService -> Customer: orderEmailSent()
CheckoutService -> Customer: checkoutSuccess()

@enduml
```



Similarity Score

V1	8
V2	8
E1	18
E2	28
Mv	6
ME	16

Node Similarity Ratio	0.75
Edge Similarity Ratio	0.57142857
Similarity Score	0.66071429

PlantUML Generated by ChatGPT

```
@startuml

actor Customer
actor Manager
participant System
participant ShoppingCart
participant Inventory
participant TransactionLog

Customer -> System: Login(username, password)
System -> System: Validate Credentials
System -> Customer: Session Started (30 min timeout)

Customer -> System: Logout()
System -> Customer: Session Terminated

Customer -> System: Sign Up for Membership()
System -> System: Update Membership Status
System -> Customer: Membership Status Updated

Customer -> ShoppingCart: Add Book(Book)
ShoppingCart -> System: Validate Book (No Duplicates)
System -> ShoppingCart: Book Added

Manager -> System: Update Stock(Book, Quantity)
System -> Inventory: Update Stock Levels
Inventory -> Manager: Stock Updated

Manager -> System: Change Book Price(Book, Price)
System -> Inventory: Update Book Price
Inventory -> Manager: Price Updated

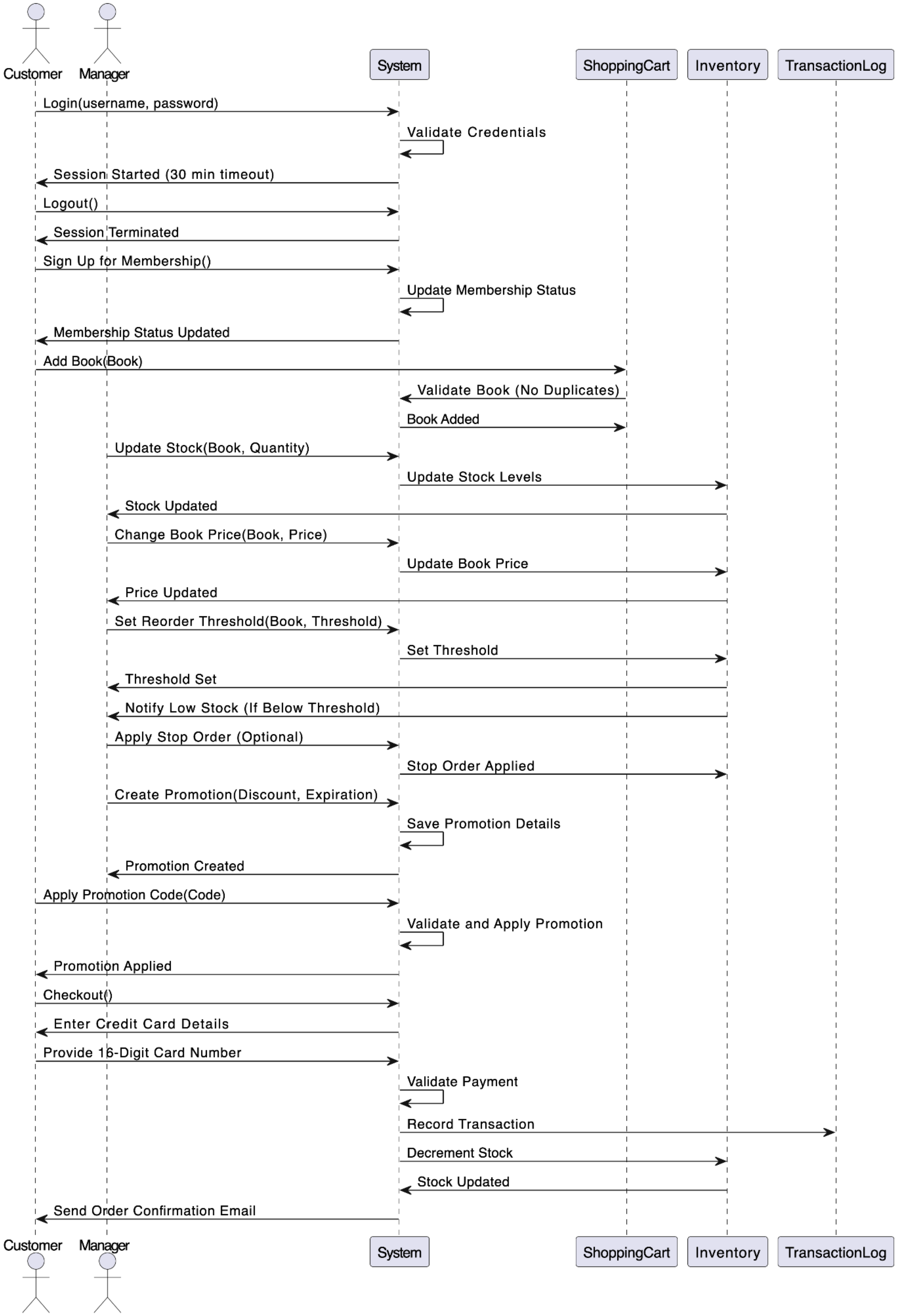
Manager -> System: Set Reorder Threshold(Book, Threshold)
System -> Inventory: Set Threshold
Inventory -> Manager: Threshold Set
Inventory -> Manager: Notify Low Stock (If Below Threshold)
Manager -> System: Apply Stop Order (Optional)
System -> Inventory: Stop Order Applied

Manager -> System: Create Promotion(Discount, Expiration)
System -> System: Save Promotion Details
System -> Manager: Promotion Created

Customer -> System: Apply Promotion Code(Code)
System -> System: Validate and Apply Promotion
System -> Customer: Promotion Applied

Customer -> System: Checkout()
System -> Customer: Enter Credit Card Details
Customer -> System: Provide 16-Digit Card Number
System -> System: Validate Payment
System -> TransactionLog: Record Transaction
System -> Inventory: Decrement Stock
Inventory -> System: Stock Updated
System -> Customer: Send Order Confirmation Email

@enduml
```


Customer
Manager
System
ShoppingCart
Inventory
TransactionLog
Login(username, password)
Validate Credentials
Session Started (30 min timeout)
Logout()
Session Terminated
Sign Up for Membership()
Update Membership Status
Membership Status Updated
Add Book(Book)
Validate Book (No Duplicates)
Book Added
Update Stock(Book, Quantity)
Update Stock Levels
Stock Updated
Change Book Price(Book, Price)
Update Book Price
Price Updated
Set Reorder Threshold(Book, Threshold)
Set Threshold
Threshold Set
Notify Low Stock (If Below Threshold)
Apply Stop Order (Optional)
Stop Order Applied
Create Promotion(Discount, Expiration)
Save Promotion Details
Promotion Created
Apply Promotion Code(Code)
Validate and Apply Promotion
Promotion Applied
Checkout()
Enter Credit Card Details
Provide 16-Digit Card Number
Validate Payment
Record Transaction
Decrement Stock
Stock Updated
Send Order Confirmation Email
Customer
Manager
System
ShoppingCart
Inventory
TransactionLog

V1	9
V2	6
E1	30
E2	40
Mv	5
ME	27
Node Similarity Ratio	0.55555556
Edge Similarity Ratio	0.675
Similarity Score	0.61527778

PlantUML Generated by BlackBox.AI

```
@startuml
actor Customer
actor Manager
participant "Authentication System" as Auth
participant "Shopping Cart" as Cart
participant "Checkout System" as Checkout
participant "Inventory System" as Inventory
participant "Transaction Log" as Transaction
participant "Email Service" as Email
participant "Promotion System" as Promotion

' Login Process
Customer -> Auth : login(username, password)
Auth -> Customer : login success

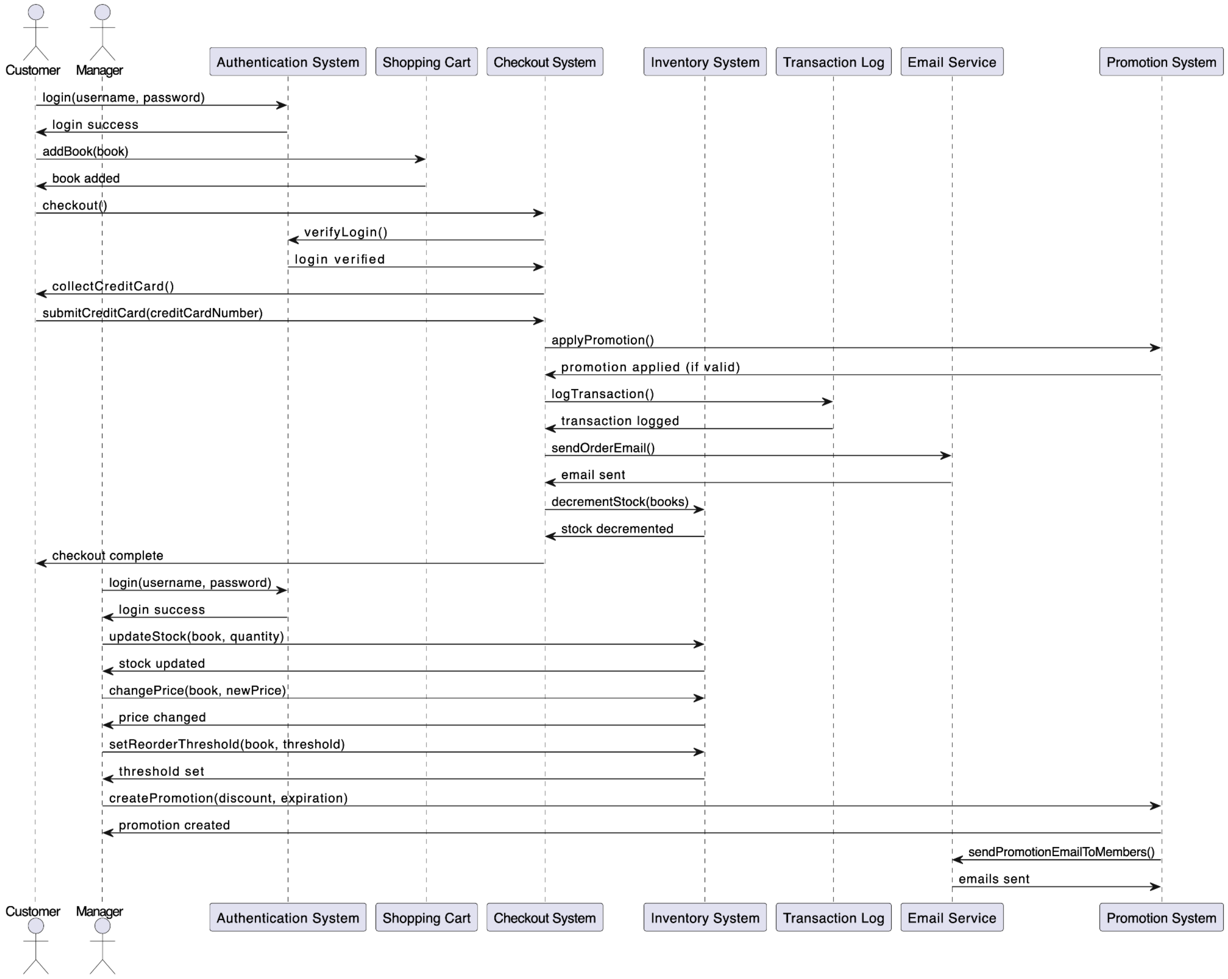
' Add Book to Cart
Customer -> Cart : addBook(book)
Cart -> Customer : book added

' Checkout Process
```

```
Customer -> Checkout : checkout()
Checkout -> Auth : verifyLogin()
Auth -> Checkout : login verified
Checkout -> Customer : collectCreditCard()
Customer -> Checkout : submitCreditCard(creditCardNumber)
Checkout -> Promotion : applyPromotion()
Promotion -> Checkout : promotion applied (if valid)
Checkout -> Transaction : logTransaction()
Transaction -> Checkout : transaction logged
Checkout -> Email : sendOrderEmail()
Email -> Checkout : email sent
Checkout -> Inventory : decrementStock(books)
Inventory -> Checkout : stock decremented
Checkout -> Customer : checkout complete

' Manager Actions
Manager -> Auth : login(username, password)
Auth -> Manager : login success
Manager -> Inventory : updateStock(book, quantity)
Inventory -> Manager : stock updated
Manager -> Inventory : changePrice(book, newPrice)
Inventory -> Manager : price changed
Manager -> Inventory : setReorderThreshold(book, threshold)
Inventory -> Manager : threshold set
Manager -> Promotion : createPromotion(discount, expiration)
Promotion -> Manager : promotion created
Promotion -> Email : sendPromotionEmailToMembers()
Email -> Promotion : emails sent

@enduml
```



V1	9
V2	8
E1	30
E2	40
Mv	7
ME	27
Node Similarity Ratio	0.77777778
Edge Similarity Ratio	0.675

Similarity Score	0.72638889

Prompt-State diagram

Consider that you are a Solutions Architect
Create state diagram in the Plant UML Diagram for the following system
Authentication- Customer will have the following Attributes: Username (unique), Password, Email, Postal Address, Member/Non-member status.
Manager will have the following Attributes: Hardcoded Username, Password, Email Address.
Login/Logout: Customers and managers can log in and log out; login enables logout and checkout, with a 30-minute session timeout. No password recovery available.
Membership-Customers can sign up to become members or remain non-members. Membership status is captured as a boolean value.
Shopping Cart-Customers can add one or more books to the shopping cart. Duplicate copies of the same book are not allowed.
Inventory Management
Book will have following Attributes: Title, Author, Price, Stock/Quantity.
Stop-Order Management: Each book has a stop-order status (on/off). If stock falls below a threshold, managers are notified unless a stop-order is applied.
Manager will perform following Actions: Update stock quantities, change book prices, set reorder thresholds, and view transaction logs. If stock reaches the reorder threshold, an email is sent to the manager, and items are automatically reordered unless a stop-order is active.
Promotions
Promotion Attributes: Percentage discount, expiration date.
Promotion Use: Available to members only; one promotion code per purchase.
Manager Actions: Create/update promotions, define discounts and expiration dates. Promotions are auto-deleted after expiration, and emails are sent to all members.
Checkout
Checkout is restricted to logged-in customers. During checkout:
A 16-digit credit card number is collected.
Transactions are logged and recorded.
Order details are sent via email.
Inventory stock is decremented based on purchased items.

PlantUML for GitHub Copilot

```
Login and Logout State Diagram
@startuml
[*] --> LoggedOut

LoggedOut : User is logged out
LoggedOut --> LoggedIn : login()

LoggedIn : User is logged in
LoggedIn --> LoggedOut : logout()
LoggedIn --> LoggedOut : sessionTimeout()
```

```
@enduml
```

Signup State Diagram

```
@startuml
[*] --> NotSignedUp

NotSignedUp : User is not signed up
NotSignedUp --> SignedUp : signup()

SignedUp : User is signed up
@enduml
```

Membership State Diagram

```
@startuml
[*] --> NonMember

NonMember : User is not a member
NonMember --> Member : becomeMember()

Member : User is a member
@enduml
```

Add to Cart State Diagram

```
@startuml
[*] --> EmptyCart

EmptyCart : Shopping cart is empty
EmptyCart --> NonEmptyCart : addBook()

NonEmptyCart : Shopping cart has items
NonEmptyCart --> EmptyCart : removeBook()
@enduml
```

Maintain Inventory State Diagram

```
@startuml
[*] --> Inventory

Inventory : Inventory management
Inventory --> Inventory : updateStock()
Inventory --> Inventory : specifyStopOrder()
Inventory --> Inventory : notifyReorder()
@enduml
```

Specify Stop-Order State Diagram

```
@startuml
[*] --> StopOrderOff

StopOrderOff : Stop-order is off
StopOrderOff --> StopOrderOn : specifyStopOrder(on)

StopOrderOn : Stop-order is on
StopOrderOn --> StopOrderOff : specifyStopOrder(off)
@enduml
```

Notify Reorder State Diagram

```
@startuml
[*] --> AboveThreshold

AboveThreshold : Stock is above reorder threshold
AboveThreshold --> BelowThreshold : stockFallsBelowThreshold()

BelowThreshold : Stock is below reorder threshold
BelowThreshold --> AboveThreshold : stockReplenished()
@enduml
```

Create Promotions State Diagram

```
@startuml
[*] --> NoPromotion

NoPromotion : No active promotion
NoPromotion --> ActivePromotion : createPromotion()

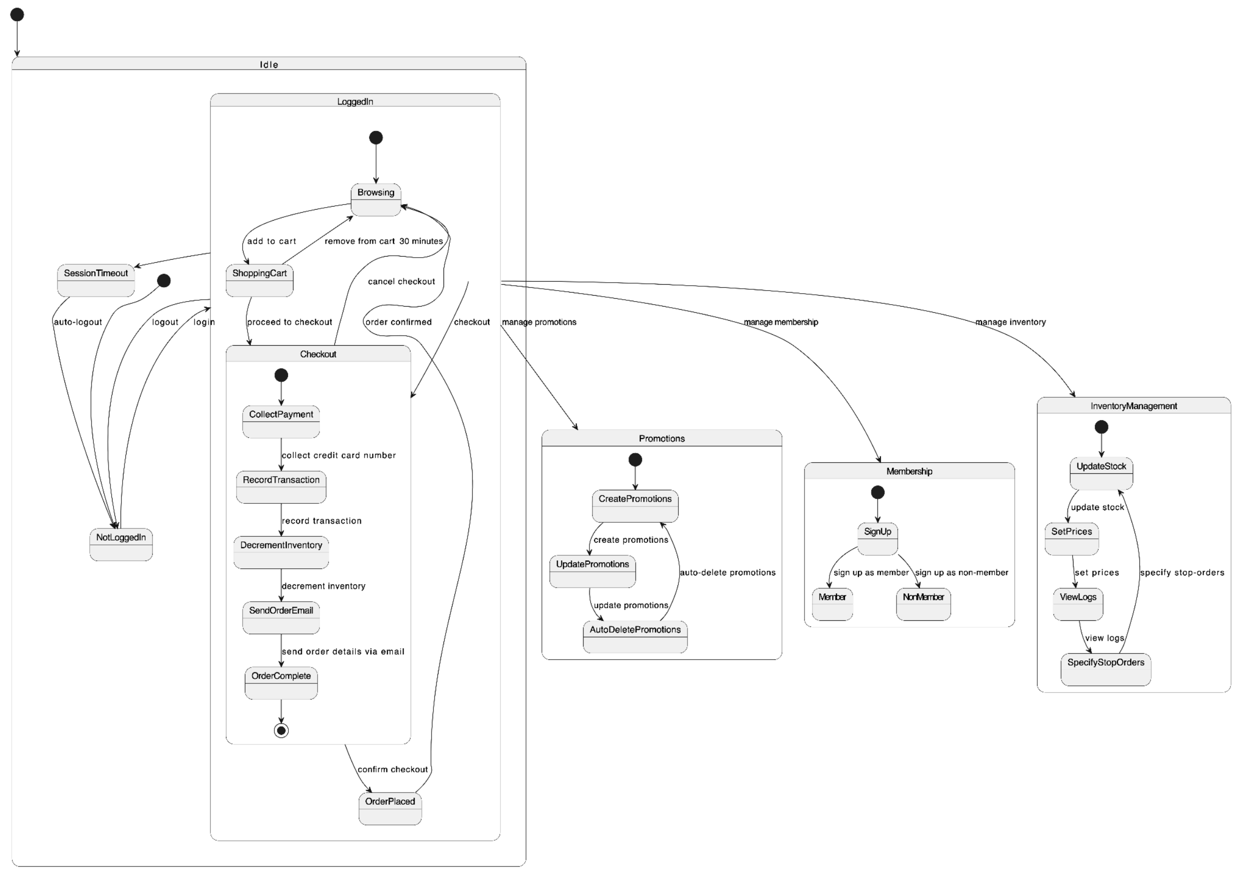
ActivePromotion : Promotion is active
ActivePromotion --> NoPromotion : promotionExpires()
@enduml
```

Checkout State Diagram

```
@startuml
[*] --> NotCheckedOut

NotCheckedOut : User has not checked out
NotCheckedOut --> CheckedOut : checkout()

CheckedOut : User has checked out
@enduml
```



Similarity Score

V1	15
V2	13
E1	25
E2	20
Mv	12
ME	16
Node Similarity Ratio	0.8
Edge Similarity Ratio	0.64

Similarity Score	0.72

PlantUML for ChatGPT

```
@startuml
left to right direction
state "Not Logged In" as NotLoggedIn
state "Logged In" as LoggedIn
state "Membership Selection" as Membership
state "Shopping Cart" as ShoppingCart
state "Inventory Management" as InventoryMgmt
state "Promotions" as Promotions
state "Checkout Process" as Checkout
state "Transaction Logged" as TransactionLogged

[*] --> NotLoggedIn
NotLoggedIn --> LoggedIn: Login
LoggedIn --> NotLoggedIn: Logout

LoggedIn --> Membership: Sign Up for Membership
Membership --> LoggedIn: Membership Updated

LoggedIn --> ShoppingCart: Add Book
ShoppingCart --> ShoppingCart: Validate Book (No Duplicates)
ShoppingCart --> LoggedIn: Book Added

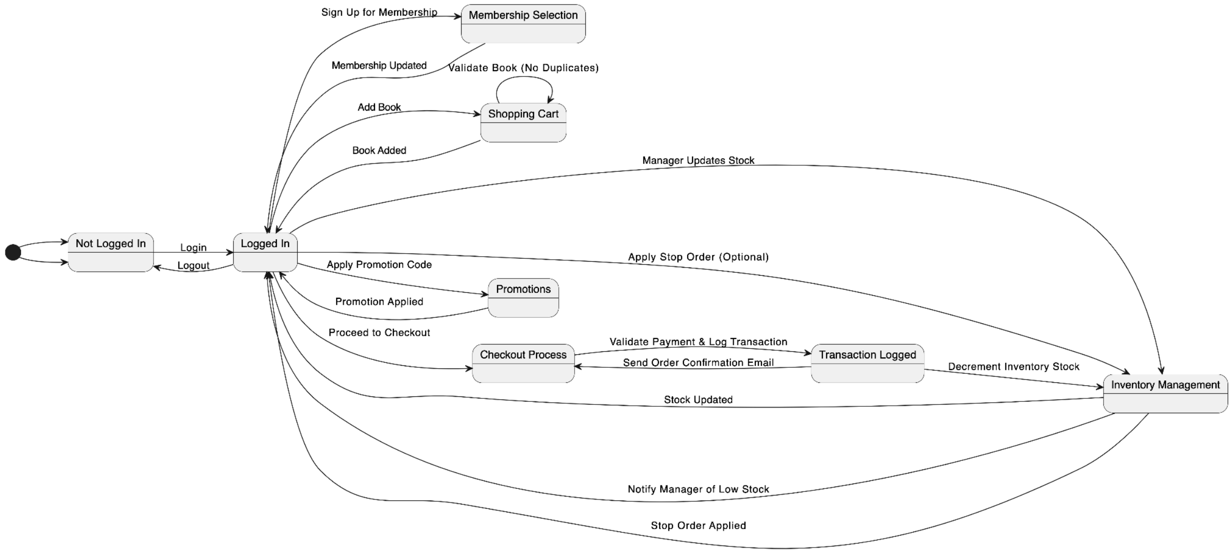
LoggedIn --> Checkout: Proceed to Checkout
Checkout --> TransactionLogged: Validate Payment & Log Transaction
TransactionLogged --> Checkout: Send Order Confirmation Email
TransactionLogged --> InventoryMgmt: Decrement Inventory Stock

LoggedIn --> InventoryMgmt: Manager Updates Stock
InventoryMgmt --> LoggedIn: Stock Updated

LoggedIn --> Promotions: Apply Promotion Code
Promotions --> LoggedIn: Promotion Applied

InventoryMgmt --> LoggedIn: Notify Manager of Low Stock
LoggedIn --> InventoryMgmt: Apply Stop Order (Optional)
InventoryMgmt --> LoggedIn: Stop Order Applied

[*] --> NotLoggedIn
@enduml
```



V1	15
V2	8
E1	25
E2	18
Mv	7
ME	17
Node Similarity Ratio	0.46666667
Edge Similarity Ratio	0.68
Similarity Score	0.57333333

PlantUML for BlackBoxAI

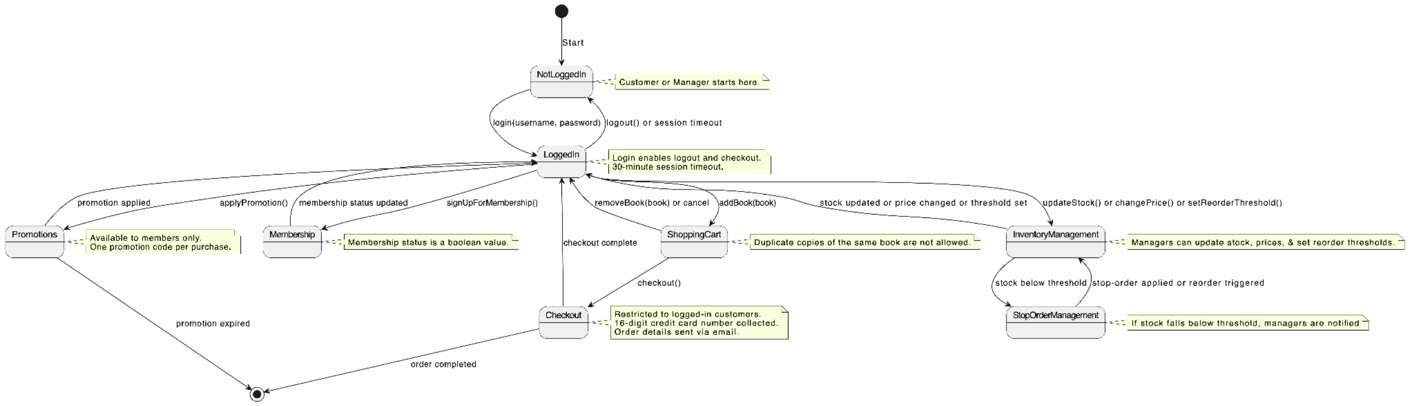
```
@startuml
top to bottom direction

' States
[*] --> NotLoggedIn : Start
NotLoggedIn --> LoggedIn : login(username, password)
```

```
LoggedIn --> NotLoggedIn : logout() or session timeout
LoggedIn --> ShoppingCart : addBook(book)
ShoppingCart --> LoggedIn : removeBook(book) or cancel
ShoppingCart --> Checkout : checkout()
Checkout --> LoggedIn : checkout complete
LoggedIn --> Membership : signUpForMembership()
Membership --> LoggedIn : membership status updated
LoggedIn --> Promotions : applyPromotion()
Promotions --> LoggedIn : promotion applied
LoggedIn --> InventoryManagement : updateStock() or changePrice() or setReorderThreshold()
InventoryManagement --> LoggedIn : stock updated or price changed or threshold set
InventoryManagement --> StopOrderManagement : stock below threshold
StopOrderManagement --> InventoryManagement : stop-order applied or reorder triggered
Promotions --> [*] : promotion expired
Checkout --> [*] : order completed

' Notes
note right of NotLoggedIn : Customer or Manager starts here.
note right of LoggedIn : Login enables logout and checkout.\n30-minute session timeout.
note right of ShoppingCart : Duplicate copies of the same book are not allowed.
note right of Checkout : Restricted to logged-in customers.\n16-digit credit card number collected.\nOrder details sent via email.
note right of Membership : Membership status is a boolean value.
note right of Promotions : Available to members only.\nOne promotion code per purchase.
note right of InventoryManagement : Managers can update stock, prices, & set reorder thresholds.
note right of StopOrderManagement : If stock falls below threshold, managers are notified

@enduml
```


Start
NotLoggedIn
Customer or Manager starts here.
login(username, password)
logout() or session timeout
LoggedIn
Login enables logout and checkout.
30-minute session timeout.
promotion applied
applyPromotion()
membership status updated
signUpForMembership()
removeBook(book) or cancel
addBook(book)
stock updated or price changed or threshold set
updateStock() or changePrice() or setReorderThreshold()
Promotions
Available to members only.
One promotion code per purchase.
Membership
Membership status is a boolean value.
checkout complete
ShoppingCart
Duplicate copies of the same book are not allowed.
InventoryManagement
Managers can update stock, prices, & set reorder thresholds.
checkout()
stock below threshold
stop-order applied or reorder triggered
Checkout
Restricted to logged-in customers.
16-digit credit card number collected.
Order details sent via email.
StopOrderManagement
If stock falls below threshold, managers are notified
promotion expired
order completed

V1	15
V2	8
E1	25
E2	16
Mv	6
ME	15
Node Similarity Ratio	0.4
Edge Similarity Ratio	0.6
Similarity Score	0.5

Use case diagram-ChatGPT→Copilot

V1	18
V2	20
E1	28
E2	28
Mv	14
ME	17
Node Similarity Ratio	0.7
Edge Similarity Ratio	0.60714286
Similarity Score	0.65357143

Use case diagram-ChatGPT→BlackBoxAI

V1	17
V2	22
E1	29
E2	22
Mv	16
ME	22
Node Similarity Ratio	0.72727273
Edge Similarity Ratio	0.75862069
Similarity Score	0.74294671

Use case diagram-Copilot→BlackBoxAI

V1	13
V2	22
E1	18
E2	22
Mv	12
ME	16
Node Similarity Ratio	0.54545455
Edge Similarity Ratio	0.72727273

Similarity Score	0.63636364

Class diagram-ChatGPT→Copilot

V1	7
V2	6
E1	8
E2	8
Mv	5
ME	6
Node Similarity Ratio	0.71428571
Edge Similarity Ratio	0.75
Similarity Score	0.73214286

Class diagram-ChatGPT→BlackBoxAi

V1	7
V2	6
E1	8
E2	10
Mv	6
ME	6
Node Similarity Ratio	0.85714286

Edge Similarity Ratio	0.6
Similarity Score	0.72857143

Class Diagram Copilot→ BlackBoxAI

V1	7
V2	7
E1	8
E2	10
Mv	5
ME	5
Node Similarity Ratio	0.71428571
Edge Similarity Ratio	0.5
Similarity Score	0.60714286

Sequence Diagram Copilot→ BlackBoxAI

V1	7
V2	7
E1	8
E2	10
Mv	5
ME	5

Node Similarity Ratio	0.71428571
Edge Similarity Ratio	0.5
Similarity Score	0.72714286

Sequence Diagram Copilot→ChatGPT

V1	9
V2	6
E1	37
E2	35
Mv	4
ME	30
Node Similarity Ratio	0.44444444
Edge Similarity Ratio	0.81081081
Similarity Score	0.62762763

State Diagram Copilot→ChatGPT

V1	15
V2	8
E1	16
E2	15
Mv	8

ME	14
Node Similarity Ratio	0.53333333
Edge Similarity Ratio	0.875
Similarity Score	0.70416667

State diagram-Copilot-BlackBoxAI

V1	15
V2	8
E1	18
E2	15
Mv	7
ME	14
Node Similarity Ratio	0.46666667
Edge Similarity Ratio	0.77777778
Similarity Score	0.62222222

State diagram-ChatGPT-BlackBoxAI

V1	8
V2	8
E1	18
E2	15

Mv	5
ME	14
Node Similarity Ratio	0.625
Edge Similarity Ratio	0.77777778
Similarity Score	0.70138889