I see. You want me to create a class diagram and a sequence diagram for the grocery store system based on the description you provided. I'll try to do that. Here is my attempt:

1) Class Diagram

```plantuml

@startuml

class GroceryStore {

- stores: Store[]

+ addStore(store: Store): void

+ removeStore(store: Store): void

+ getStores(): Store[]

+ findStoreById(id: string): Store

}

class Store {

- storeId: string

- stockItems: StockItem[]

- customers: Customer[]

+ addStockItem(item: StockItem): void

+ removeStockItem(item: StockItem): void

+ getStockItems(): StockItem[]

+ addCustomer(customer: Customer): void

+ removeCustomer(customer: Customer): void

+ getCustomers(): Customer[]

+ findCustomerById(id: string): Customer

}

class StockItem {

- itemId: string

- itemName: string

- itemQuantity: int

- itemPrice: float

}

abstract class Customer {

- customerId: string

- customerType: CustomerType

- registered: boolean

- membershipCard: MembershipCard

}

class OnlineCustomer {

- order: Order

}

class WalkInCustomer {

}

class Order {

- orderId: string

- orderItems: OrderItem[]

- orderTotal: float

- paymentMethod: PaymentMethod

}

class OrderItem {

- stockItem: StockItem

- quantity: int

}

enum CustomerType {

ONLINE

WALK\_IN

}

enum PaymentMethod {

CASH\_ON\_DELIVERY

CREDIT\_CARD

}

class MembershipCard {

- cardId: string

- discount: float

}

GroceryStore "1" -- "0..\*" Store

Store "0..\*" -- "0..\*" StockItem

Store "0..\*" -- "0..\*" Customer

Customer <|-- OnlineCustomer

Customer <|-- WalkInCustomer

OnlineCustomer "1" -- "0..1" Order

Order "1" -- "0..\*" OrderItem

OrderItem "\*" -- "1" StockItem

Customer "0..1" -- "0..1" MembershipCard

@enduml

```

2) Sequence Diagram (Any two activities)

a) Online customer places an order and pays by credit card.

```plantuml

@startuml

actor OnlineCustomer as OC

participant GroceryStore as GS

participant Store as S

participant Order as O

participant CreditCard as CC

OC -> GS : findStoreById(storeId)

activate GS

GS -> S : return store

activate S

S --> GS : store

deactivate S

GS --> OC : store

deactivate GS

OC -> S : findStockItemById(itemId)

activate S

S -> O : createOrder(orderId)

activate O

O --> S : order

deactivate O

S --> OC : stockItem

deactivate S

OC -> O : addOrderItem(stockItem, quantity)

activate O

O -> O : updateOrderTotal()

O --> OC : orderItem added

deactivate O

OC -> O : selectPaymentMethod(CREDIT\_CARD)

activate O

O -> CC : validateCreditCard(cardNumber, expiryDate, cvv)

activate CC

CC --> O : credit card valid/invalid

deactivate CC

alt credit card valid

O -> CC : chargeCreditCard(orderTotal)

activate CC

CC --> O : payment successful/failed

deactivate CC

alt payment successful

O --> OC : order confirmed

else payment failed

O --> OC : order cancelled

end

else credit card invalid

O --> OC : order cancelled

end

deactivate O

@enduml

```

b) Walk-in customer buys items and gets a membership card.

```plantuml

@startuml

actor WalkInCustomer as WC

participant GroceryStore as GS

participant Store as S

participant MembershipCard as MC

WC -> GS : findStoreById(storeId)

activate GS

GS -> S : return store

activate S

S --> GS : store

deactivate S

GS --> WC : store

deactivate GS

WC -> S : selectStockItems(itemIds)

activate S

S -> WC : return stockItems

WC -> S : payByCash(stockItems)

S -> MC : createMembershipCard(cardId, discount)

activate MC

MC --> S : membershipCard

deactivate MC

S -> WC : return membershipCard and receipt

deactivate S

@enduml

```