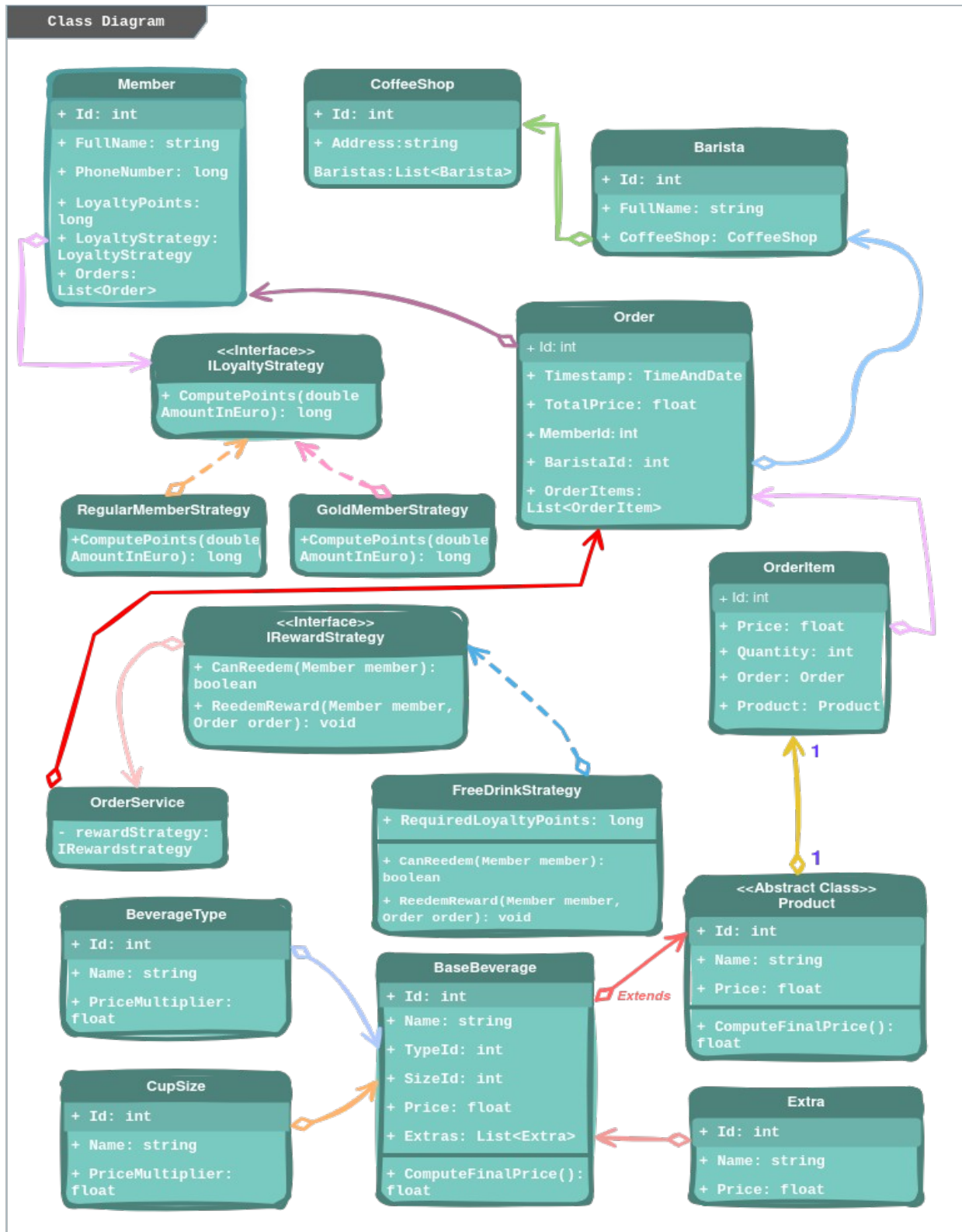*Chitu Bianca-Ioana*

Internship Dynamics & Azure / Full Stack 2026 - Rezolvare

Problem 1

Solution breakdown:

- The membership system based on loyalty points is comprised of a Member object that holds information about the customer, the list of all order history, keeps track of their loyalty points count and of their member status.

- The membership status is decided through Strategy Pattern, where the point conversion from order total to amount of points differs from level to level of their membership. This allows for further extension to multiple levels of membership.

- The beverage customization system was divided into a base beverage that can be a combination of a beverage type (e.g. espresso, cappuccino, latte) and a cup size (Small, Medium, Large). For additional topping and such, there is a list of items for each customized beverage.

- The customized beverage item is a concrete implementation for a Product. This allows for further menu extension, where Product can be any other item (e.g. pastry, juice, tea).

- An OrderItem is an object added to the order by the barista, which has information about the product name, quantity and price (dependant on quantity).

- An Order object will contain information about the order put in my the barista, identified in the object by BaristaID, the time stamp, the order total price, the MemeberID in case the customer is a member and scans their ID and list of order items.

- Converting points into redeemable free drinks was solved also through Strategy Pattern, so that conditions of any other added rewards can be different (e.g. different requirement for different drink sizes, the reward can also be a percentage off an order).

- I have proposed two separate strategy interfaces for the accumulation of loyalty points system and redeeming rewards system to account for the fact that each functionality needs to be easily expandable.

Class Diagram

Member
+ Id: int
+ FullName: string
+ PhoneNumber: long
+ LoyaltyPoints: long
+ LoyaltyStrategy: LoyaltyStrategy
+ Orders: List<Order>

CoffeeShop
+ Id: int
+ Address:string
Baristas:List<Barista>

Barista
+ Id: int
+ FullName: string
+ CoffeeShop: CoffeeShop

<<Interface>> ILoyaltyStrategy
+ ComputePoints(double AmountInEuro): long

Order
+ Id: int
+ Timestamp: TimeAndDate
+ TotalPrice: float
+ MemberId: int
+ BaristaId: int
+ OrderItems: List<OrderItem>

RegularMemberStrategy
+ComputePoints(double AmountInEuro): long

GoldMemberStrategy
+ComputePoints(double AmountInEuro): long

OrderItem
+ Id: int
+ Price: float
+ Quantity: int
+ Order: Order
+ Product: Product

<<Interface>> IRewardStrategy
+ CanReedem(Member member): boolean
+ ReedemReward(Member member, Order order): void

OrderService
- rewardStrategy: IRewardstrategy

FreeDrinkStrategy
+ RequiredLoyaltyPoints: long
+ CanReedem(Member member): boolean
+ ReedemReward(Member member, Order order): void

<<Abstract Class>> Product
+ Id: int
+ Name: string
+ Price: float
+ ComputeFinalPrice(): float

1

1

BeverageType
+ Id: int
+ Name: string
+ PriceMultiplier: float

BaseBeverage
+ Id: int
+ Name: string
+ TypeId: int
+ SizeId: int
+ Price: float
+ Extras: List<Extra>
+ ComputeFinalPrice(): float

Extends

CupSize
+ Id: int
+ Name: string
+ PriceMultiplier: float

Extra
+ Id: int
+ Name: string
+ Price: float

Chitu Bianca-Ioana

Entity Relation Diagram

CoffeeShops

PK	Id
	Address

Baristas

PK	Id
	FullName
FK	CoffeeShopID

Extras

PK	Id
	Name
	Price

Orders

PK	Id
	Timestamp
	TotalPrice
FK	MemberID
FK	BaristaID

BeverageExtras

FK	ExtraID
FK	BeverageID

OrderItems

PK	Id
	Price
	Quantity
FK	ProductID
FK	OrderId

BaseBeverages

PK	Id
FK	BevTypeID
FK	CupSizeID
	Price

Product

PK	Id
FK	Name
FK	Price

Members

PK	Id
	FullName
	PhoneNumber
	LoyaltyPoints

BeverageType

PK	Id
	Name
	PriceMultiplier

CupSize

PK	Id
	Name
	PriceMultiplier