Java Fundamental Course

Object-Oriented Thinking Task

Juan Felipe Florez Giraldo

A public coffee/vending machine at a

1. Describe the requirements for the chosen system: functional and non-functional.

Functinal:

* The sistem must accept payment in cash an credit/debit cards.
* The sistem must provide a variety of coffee options, such as espresso, latte, cappuccino, erc.
* The sistem must have a user interface for selecting beverage options ans completing transactions.
* The sistem must maintain an inventory of coffee ingredients.
* The sistem must dispense correct change if the payment exceeeds the cost.
* The sistem must maintan an inventroy of the coffee sales.
* The sistem must notify when a restock is needed.
* The sistem must offer customization options for beverages, allowing users to adjust factors like sugar levels, milk type, etc.

Non-Functional:

* The sistem should be reliable and availabe for use at all times.
* The sistem should provide quick service, minimizing wait times.
* The sistem should be easy to repair and maintain.
* The sistem should be designed with energy efficiency to optimize cost.
* The sistem should have robust error handling.
* The sistem should adhere to industry standars and regulations.
* The sistem should be accessible to users with disabilities, with accesibility standards to insure inclusivity for all customers.
* The sistem should be pleasing to the eye for a better service.

1. Design use cases for the system based on the requirements.

Purchase Beverage:

Actor: User

Description: User selects a beverage option from the user interface, customizes it if desired, inserts payment in cash or via credit/debit card, and completes the transaction.

Check Inventory:

Actor: Administrator/Staff

Description: Administrator or staff checks the inventory of coffee ingredients and sales records to monitor stock levels and identify when restocking is needed.

Refill Inventory:

Actor: Administrator/Staff

Description: Administrator or staff refills the inventory of coffee ingredients when restocking is needed.

Customize Beverage:

Actor: User

Description: User customizes their beverage by adjusting various options such as sugar levels, milk type, flavorings, etc.

View Transaction History:

Actor: Administrator/Staff

Description: Administrator or staff views the transaction history to analyze sales data and track revenue.

Report System Error:

Actor: User/Administrator/Staff

Description: User, administrator, or staff reports a system error or malfunction for resolution.

1. Identify objects, classes, and relationships in the system.

Objects:

Coffee/Vending Machine

BeveragePayment

Inventory

User Interface

Transaction

Classes:

CoffeeMachine

BeverageOption

PaymentProcessor

InventoryManager

UserInterfaceController

TransactionRecord

CoffeeMachine - BeverageOption:

Relationship Type: Association

Description: CoffeeMachine has a "has-a" relationship with BeverageOption, as it offers various beverage options to users.

UML Representation: Simple association between CoffeeMachine and BeverageOption.

CoffeeMachine - PaymentProcessor:

Relationship Type: Association

Description: CoffeeMachine interacts with PaymentProcessor to handle payment transactions.

UML Representation: Simple association between CoffeeMachine and PaymentProcessor.

CoffeeMachine - InventoryManager:

Relationship Type: Association

Description: CoffeeMachine communicates with InventoryManager to manage inventory levels and restocking needs.

UML Representation: Simple association between CoffeeMachine and InventoryManager.

CoffeeMachine - UserInterfaceController:

Relationship Type: Association

Description: CoffeeMachine is linked with UserInterfaceController to provide a user-friendly interface.

UML Representation: Simple association between CoffeeMachine and UserInterfaceController.

Transaction - Payment:

Relationship Type: Dependency

Description: Transaction depends on Payment for processing payment transactions.

UML Representation: Dependency arrow pointing from Transaction to Payment.

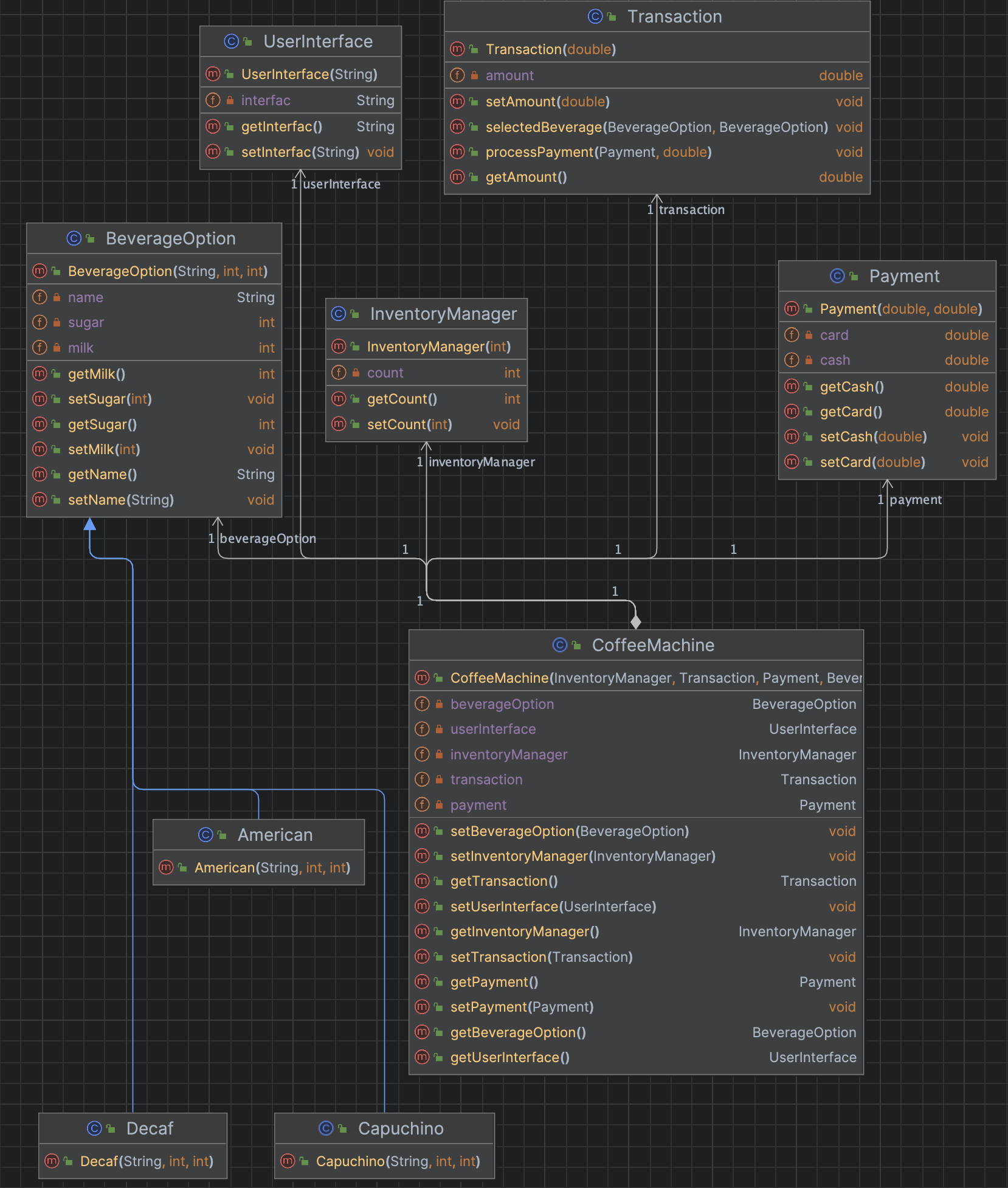
Transaction - BeverageOption:

Relationship Type: Dependency

Description: Transaction depends on BeverageOption for recording the specific beverage options purchased.

UML Representation: Dependency arrow pointing from Transaction to BeverageOption.

1. Design class diagrams picturing classes, their attributes, and relations in the system.



1. Optionally, convert diagrams into Java code.

