Scenario: A public vending machine

Requirements

Functional Requirements
The vending machine must deliver the selected product after payment
The vending machine must allow users to select an available product
The vending machine must list the available products price
The vending machine must accept cash for the selected product
The vending machine must keep track of the available products
The vending machine must be able to add and refill products
The vending machine must allow the user to cancel the purchase before product selection

Non-functional Requirements
The vending machine should be available 24/7
The vending machine should deliver the paid products in less than 12 seconds
The vending machine should display an error in case of detected malfunctions
The vending machine should have spare change at all times
The vending machine products should be in good condition
The vending machine should prevent product theft
The vending machine must accept the correct amount of cash for the products

Use Cases

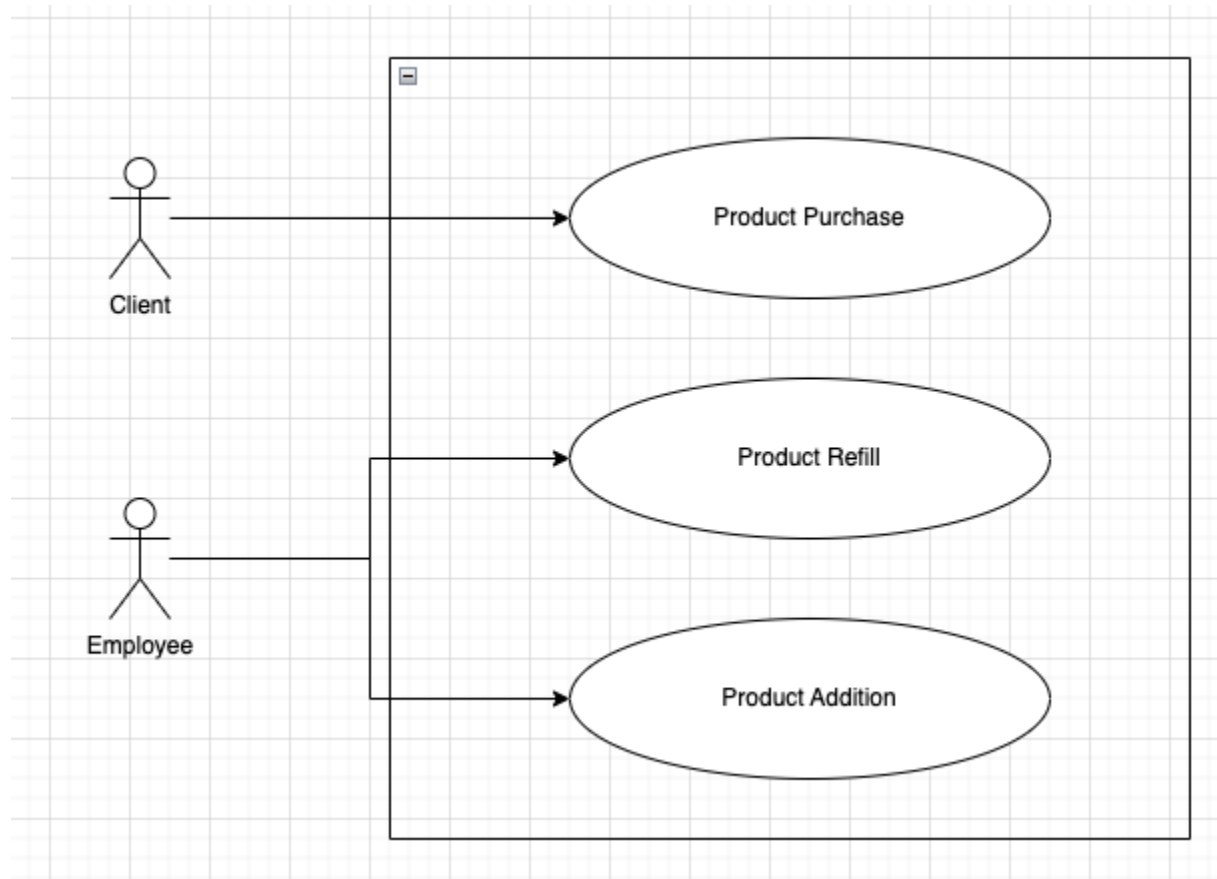| Use Case	Product purchase
Primary Actor	Client
Success Scenario	1. Client inserts cash into the vending machine. 2. Client selects the desired product. 3. Vending machine delivers the selected product. 4. Vending machine returns change, if needed. 5. Client retrieves product.
Extensions:	2a. Vending machine checks for product availability

Use Case	Product purchase cancelation
Primary Actor	Client
Success Scenario	1. Client inserts cash into the vending machine 2. Client cancels the process 3. Vending machine returns cash

Use Case	Invalid product selection
Primary Actor	Client
Success Scenario	1. Client inserts cash into vending machine 2. Client selects an incorrect product 3. Vending machine notifies the user 4. Vending machine awaits for a new product selection or process cancellation

Use Case	Invalid amount of cash
Primary Actor	Client
Success Scenario	1. Client inserts cash into vending machine 2. Client selects a product with insufficient amount of cash 3. Vending machine notifies the user 4. Vending machine awaits for more cash or process cancellation
Extension	4a. Vending machine returns cash in case of order cancellation 4b. Vending machine awaits for cash until the required amount is inserted

Use Case	Product refill
Primary Actor	Employee
Success Scenario	1. Employee opens vending machine 2. Employee refills products 3. Employee closes vending machine 4. Vending machine checks for changes in stock

Use Case	Product addition
Primary Actor	Employee
Success Scenario	1. Employee opens vending machine 2. Employee adds new product 3. Employee adds price tag 4. Employee adds the new product and location to the vending machine system 5. Employee closes vending machine 6. Vending machine checks for changes in stock

CRC Cards

Product	
Responsibility	**Collaborator**
Knows product name	ProductSlot
Knows product price	

ProductSlot	
Responsibility	**Collaborator**
Knows stored products	Vending Machine
Manages stored products	ProductSlot
Knows slot location	

VendingMachine	
Responsibility	**Collaborator**
Knows product slots	ProductSlot
Manages product slots	VendingMachine
Sells products	
Handles cash	

Class Diagram

<<Interface>>
Product

- price: Float
- name: String

+ getPrice(): Float
+ getName(): String

0..n

⊟ Snack 1

- price: Float

- name: String

+ getPrice(): Float

+ getName(): String

⊟ Snack 2

- price: Float

- name: String

+ getPrice(): Float

+ getName(): String

⊟ Snack 1

- price: Float

- name: String

+ getPrice(): Float

+ getName(): String

⊟ ProductSlot

- location: String

- products: Product[]

+ getLocation(): String
+ getProducts(): Product[]
+ addProduct(): Product
+ removeProduct(): void
+ replaceProducts(Product[]) : void

1

0..n

1

⊟ VendingMachine

- productSlots[] : ProductSlot[]

+ field: type

+ getProductSlots[] : ProductSlot[]
+ refillSlot(Product product, String location): void
+ receiveCash(): int
+ returnCash(int amount): int
+ sellProduct(String location): Boolean
+ scanProductSlots(): void