**Topic: Public vending machine at a university**

**Polina Batova**

**Functional Requirements**

The system must …

* Give users goods after the payment
* Accept users’ payment
* Check if there is a product on the slot
* Not allow users to buy items from empty slots
* Ask the user to verify and confirm the purchase
* Turn maintenance mode on and off for different maintenance tasks
* Notice the attempted shake
* Send an alarm to the technician in case of shaking
* Lock itself in case of shaking

**Non-functional Requirements**

The system should be …

* User-friendly
* Available 24/7 (why not?)
* Updatable
* Internet-connected (to accept payment)
* Resistant to shaking

**Use-cases**

1. Title: Give a product

Primary actor: Customer

Success scenario:

* 1. User enters the slot number of the item they want to buy into the control panel.
  2. The system asks the user to verify and confirm the purchase.
  3. The user confirms their choice on the control panel.
  4. The system asks the user for the corresponding payment for the product
  5. The user pays for the product
  6. The system gives the chosen product to the customer
  7. The system becomes ready to serve next customers

Extensions:

1. If a customer enters a non-existent slot number or an empty slot number, the system displays an error message and then becomes ready to accept the customer's request again.
2. If a customer refuses to buy, the system returns to an initial stage and becomes ready to accept the customer's request again.
3. If the payment fails, the system asks the customer to pay again.
4. If the payment fails again, the system displays an error message, returns to an initial stage and then becomes ready to accept the customer's request again.
5. Title: Maintenance

Primary actor: Technician

Success scenario:

* + - 1. Technician opens the machine with a special code or tool.
      2. The system turns maintenance mode on.
      3. Technician performs necessary maintenance tasks like software updates, restoking of fixing problems.
      4. Technician closes the vending machine.
      5. Vending machine exits maintenance mode and returns to an initial stage and becomes ready to accept customers ‘orders.

Extensions:

1. If the maintenance mode was not activated the technician should try to close and open the machine again. If the problem stays, the technician tries trouble shooting themselves or with the help of technical support.
2. If the maintenance mode was not disabled the technician should try trouble shooting themselves or with the help of technical support.
3. Title: Shaking

Primary actor: Customer

Success scenario:

* 1. Customer attempts to shake the machine.
  2. The system detects abnormal movement/shaking.
  3. The system triggers an alarm or alert sound.
  4. The system locks itself for further operations.
  5. The system sends an alert to the technician.
  6. The system displays a warning message on the control panel and does not work until the technician unlocks it.

**Classes**

1. Vending Machine

It gives (dispensers) customers products, checks if a product is on a slot, supports different modes like maintenance and working, locks itself if it is necessary.

1. Product

It is a representation of a real item that can be bought in the vending machine. It has quantity that can be increased or decreased by different methods.

1. Control Panel

It accepts users’ commands, asks users verification and confirmation, and displays different messages like errors, warnings, or instructions.

1. Shake Detector

It detects shaking and alerts technician about the incident.

1. Payment

It is a representation of a user’s payment that contains its type and sum and can be validate.

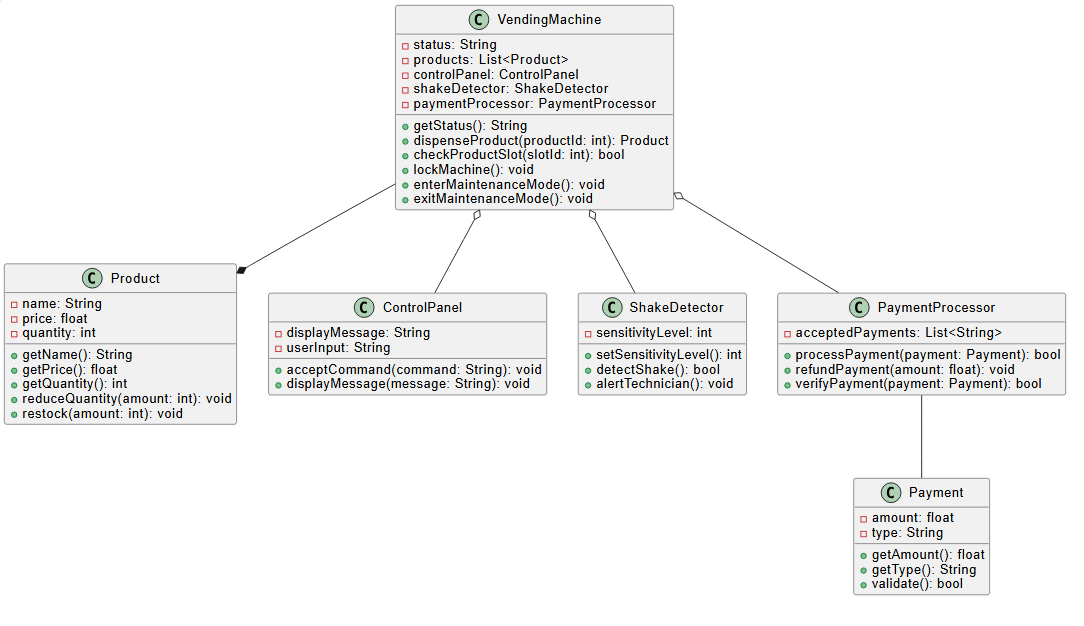
1. Payment Processor

It processes and accepts users’ payment, refunds and verifies payments.

**Relationships**

1. Vending Machine has a product (Aggregation). Moreover, the machine has a lot of products.
2. Vending Machine owns only one Control Panel (Composition).
3. Vending Machine owns only one Shake Detector (Composition).
4. Vending Machine owns only one Payment Processor (Composition).
5. Payment is processed by Payment Processor.

**Class Diagram**



It was created using PlantUML based on a simple text representation.

*@startuml*

*class VendingMachine {*

*-status: String*

*-products: List<Product>*

*-controlPanel: ControlPanel*

*-shakeDetector: ShakeDetector*

*-paymentProcessor: PaymentProcessor*

*+getStatus(): String*

*+dispenseProduct(productId: int): Product*

*+checkProductSlot(slotId: int): bool*

*+lockMachine(): void*

*+enterMaintenanceMode(): void*

*+exitMaintenanceMode(): void*

*}*

*class Product {*

*-name: String*

*-price: float*

*-quantity: int*

*+getName(): String*

*+getPrice(): float*

*+getQuantity(): int*

*+reduceQuantity(amount: int): void*

*+restock(amount: int): void*

*}*

*class ControlPanel {*

*-displayMessage: String*

*-userInput: String*

*+acceptCommand(command: String): void*

*+displayMessage(message: String): void*

*}*

*class ShakeDetector {*

*-sensitivityLevel: int*

*+setSensitivityLevel(): int*

*+detectShake(): bool*

*+alertTechnician(): void*

*}*

*class Payment {*

*-amount: float*

*-type: String*

*+getAmount(): float*

*+getType(): String*

*+validate(): bool*

*}*

*class PaymentProcessor {*

*-acceptedPayments: List<String>*

*+processPayment(payment: Payment): bool*

*+refundPayment(amount: float): void*

*+verifyPayment(payment: Payment): bool*

*}*

*VendingMachine --\* Product*

*VendingMachine o-- ControlPanel*

*VendingMachine o-- ShakeDetector*

*VendingMachine o-- PaymentProcessor*

*PaymentProcessor -- Payment*

*@enduml*