supermarket

Products:products

Productcount:int

Supermarket()

Add product(name: const char\*, price : double, stock: int)void

Displayproducts(): void

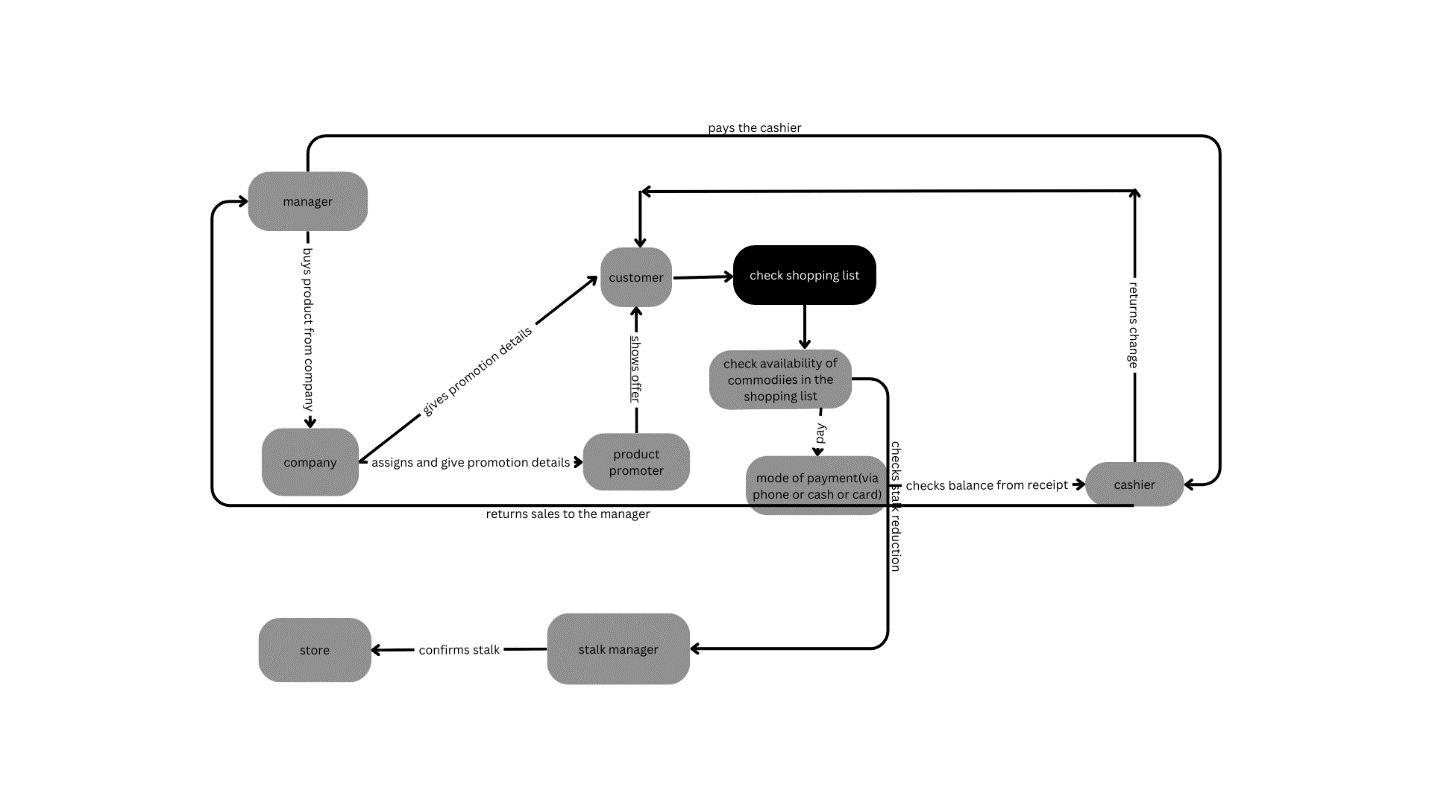
ProcessSale(ProductName: const char, quantity : int): void

**PRODUCT**

Name : char{50}

Price: double

Stock : int



#include <iostream>

#include <iomanip>

#include <string.h>

// Product class representing items in the supermarket

class Product {

public:

char name[50];

double price;

int stock;

Product() : price(0.0), stock(0) {

memset(name, 0, sizeof(name));

}

Product(const char\* name, double price, int stock) : price(price), stock(stock) {

strncpy(this->name, name, sizeof(this->name));

this->name[sizeof(this->name) - 1] = '\0'; // Ensure null-terminated string

}

};

// Supermarket class to manage products and transactions

class Supermarket {

private:

Product\* products;

int productCount;

public:

Supermarket() : products(NULL), productCount(0) {}

~Supermarket() {

delete[] products;

}

// Add a product to the supermarket inventory

void addProduct(const char\* name, double price, int stock) {

Product\* newProducts = new Product[productCount + 1];

for (int i = 0; i < productCount; ++i) {

newProducts[i] = products[i];

}

newProducts[productCount++] = Product(name, price, stock);

delete[] products;

products = newProducts;

}

// Display available products

void displayProducts() {

std::cout << "Available Products:\n";

std::cout << std::setw(20) << std::left << "Name" << std::setw(10) << "Price" << "Stock\n";

for (int i = 0; i < productCount; ++i) {

const Product& product = products[i];

std::cout << std::setw(20) << std::left << product.name << std::setw(10) << product.price << product.stock << "\n";

}

}

// Process a sale

void processSale(const char\* productName, int quantity) {

for (int i = 0; i < productCount; ++i) {

Product& product = products[i];

if (strcmp(product.name, productName) == 0) {

if (product.stock >= quantity) {

double totalCost = product.price \* quantity;

std::cout << "Sale successful. Total: $" << totalCost << "\n";

product.stock -= quantity;

} else {

std::cout << "Insufficient stock for " << productName << ". Available: " << product.stock << "\n";

}

return;

}

}

std::cout << "Product not found.\n";

}

};

int main() {

Supermarket supermarket;

// Add products to the supermarket inventory

supermarket.addProduct("Milk", 2.5, 50);

supermarket.addProduct("Bread", 1.0, 100);

supermarket.addProduct("Eggs", 1.75, 30);

// Display available products

supermarket.displayProducts();

// Process a sale

char productName[50];

int quantity;

std::cout << "Enter the product name and quantity for the sale (e.g., Milk 3): ";

std::cin >> productName >> quantity;

supermarket.processSale(productName, quantity);

return 0;