```cpp
#include <iostream>

#include <vector>

#include <string>

#include <memory>    // For smart pointers

#include <thread>    // For multithreading

#include <map>       // For STL map (e.g., product catalog)

#include <algorithm> // For STL algorithms (e.g., sort)


using namespace std;


// --- Product Class ---

class Product {

private:

    int id;

    string name;

    double price;


public:

    Product(int id, string name, double price) : id(id), name(name), price(price) {}


    int getId() const { return id; }

    string getName() const { return name; }

    double getPrice() const { return price; }

};


// --- Order Class ---

class Order {

private:

    vector<shared_ptr<Product>> products;
    double totalAmount;

public:
    Order() : totalAmount(0) {}

    void addProduct(shared_ptr<Product> product) {
        products.push_back(product);
        totalAmount += product->getPrice();
    }

    double getTotalAmount() const {
        return totalAmount;
    }

    void displayOrderDetails() const {
        cout << "Order Details:\n";
        for (const auto& product : products) {
            cout << "Product: " << product->getName() << " | Price: $" << product->getPrice() << endl;
        }
        cout << "Total Amount: $" << totalAmount << endl;
    }
};

// --- Customer Class ---
class Customer {
private:
    string name;
    shared_ptr<Order> order;
```

```cpp
public:
    Customer(string name) : name(name), order(make_shared<Order>()) {}

    string getName() const { return name; }

    void addToOrder(shared_ptr<Product> product) {
        order->addProduct(product);
    }

    void checkout() const {
        cout << "Customer: " << name << endl;
        order->displayOrderDetails();
    }
};

// --- Multithreaded Order Processing ---
void processOrder(shared_ptr<Customer> customer) {
    cout << "Processing order for customer: " << customer->getName() << endl;
    this_thread::sleep_for(chrono::seconds(2)); // Simulate time for processing
    customer->checkout();
}

// --- Main Shopping System ---
int main() {
    // --- Product Catalog (Using STL Map) ---
    map<int, shared_ptr<Product>> productCatalog;
    productCatalog[1] = make_shared<Product>(1, "Laptop", 1200.99);
    productCatalog[2] = make_shared<Product>(2, "Smartphone", 699.99);
    productCatalog[3] = make_shared<Product>(3, "Headphones", 199.99);
    // --- Display Available Products ---
    cout << "Available Products:\n";
    for (const auto& item : productCatalog) {
        cout << "ID: " << item.second->getId() << " | Name: " << item.second->getName() << " | Price: $" <<
item.second->getPrice() << endl;
    }

    // --- Create Customer and Add Products to Order ---
    auto customer1 = make_shared<Customer>("Alice");
    customer1->addToOrder(productCatalog[1]); // Adding Laptop
    customer1->addToOrder(productCatalog[3]); // Adding Headphones

    // --- Create Another Customer ---
    auto customer2 = make_shared<Customer>("Bob");
    customer2->addToOrder(productCatalog[2]); // Adding Smartphone

    // --- Process Orders Concurrently ---
    thread thread1(processOrder, customer1);
    thread thread2(processOrder, customer2);

    thread1.join();
    thread2.join();

    return 0;
}
```