ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ thông tin và Truyền thông

Báo cáo Mẫu thiết kế phần mềm

Version 1.0

Nhóm: DP2022.07

Danh sách thành viên

1. Nguyễn Trung Kiên – MSSV: 20194087
2. Nguyễn Đức Dũng - MSSV: 20194028
3. Lê Đức Anh – MSSV: 20193978
4. Lê Văn Phước – MSSV: 20194140

*Hà Nội, 07/2023*

**Mục lục**

[1](#_heading=h.3whwml4) Tổng quan 3

[1.1](#_heading=h.1pxezwc) Mục tiêu 3

[1.2](#_heading=h.49x2ik5) Phạm vi 3

[1.2.1. Mô tả khái quát phần mềm 3](#_heading=h.vt9qjyaomv3g)

[1.2.2. Các chức năng chính của phần mềm 3](#_heading=h.ijkt6v5n9tcm)

[1.2.3 .Cấu trúc mã nguồn 4](#_heading=h.2p2csry)

[1.2.4 Các yêu cầu thêm cần cân nhắc cùng quá trình tái cấu trúc 4](#_heading=h.147n2zr)

[1.2.5 *Các hoạt động (Ví dụ: review, refactor) thực thi trên mã nguồn để đạt được mục tiêu kể trên* 5](#_heading=h.3o7alnk)

[1.2.6. *Kết quả dự kiến* 6](#_heading=h.qx3c43gfp7xv)

[1.3](#_heading=h.32hioqz) Danh sách thuật ngữ 6

[1.4](#_heading=h.1hmsyys) Danh sách tham khảo 6

[2](#_heading=h.41mghml) Đánh giá thiết kế cũ 7

[2.1](#_heading=h.2grqrue) Nhận xét chung 7

[2.2](#_heading=h.vx1227) Đánh giá các mức độ coupling và cohesion 7

[2.2.1](#_heading=h.3fwokq0) Coupling 7

[2.2.2](#_heading=h.1v1yuxt) Cohesion 12

[2.3](#_heading=h.4f1mdlm) Đánh giá việc tuân theo SOLID 13

[2.3.1](#_heading=h.2u6wntf) SRP 13

[2.3.2](#_heading=h.19c6y18) OCP 15

[2.3.3](#_heading=h.3tbugp1) LSV 15

[2.3.4](#_heading=h.28h4qwu) ISP 16

[2.3.5](#_heading=h.nmf14n) DIP 16

[2.4](#_heading=h.37m2jsg) Các vấn đề về Clean Code 16

[2.4.1](#_heading=h.1mrcu09) Clear Name 16

[2.4.2](#_heading=h.46r0co2) Clean Function/Method 17

[2.4.3](#_heading=h.2lwamvv) Clean Class 20

[2.5](#_heading=h.111kx3o) Các vấn đề khác 20

[3](#_heading=h.3l18frh) Đề xuất cải tiến 21

[1. Thêm mặt hàng Media mới 21](#_heading=h.206ipza)

[2. Thêm màn hình 24](#_heading=h.4k668n3)

[3. Thay đổi yêu cầu khi load giao diện 30](#_heading=h.2zbgiuw)

[4. Thay đổi cách tính khoảng cách, sử dụng thư viện mới 31](#_heading=h.1egqt2p)

[5 . Thêm phương thức thanh toán mới 32](#_heading=h.3ygebqi)

[6. Thay đổi công thức tính phí vận chuyển 34](#_heading=h.2dlolyb)

[4](#_heading=h.sqyw64) Tổng kết 36

[4.1](#_heading=h.3cqmetx) Kết quả tổng quan 36

[4.2](#_heading=h.1rvwp1q) Các vấn đề tồn đọng 36

# Tổng quan

## Mục tiêu

Mục đích sử dụng của báo cáo mẫu thiết kế phần mềm:

* Mô tả khái quát về phần mềm, chức năng của phần mềm
* Chỉ ra, đánh giá những vi phạm của source code
* Đề xuất những cải tiến cho source code

Đối tượng người đọc:

* Những thành viên trong đội phát triển dự án
* Khách hàng của dự án

Khái quát nội dung báo cáo

* Tổng quan về phần mềm
* Đánh giá thiết kế cũ của dự án
* Những vi phạm source code mà cách cải tiến
* Tổng quan những kết quả đạt được, những vẫn để còn sót lại

## Phạm vi

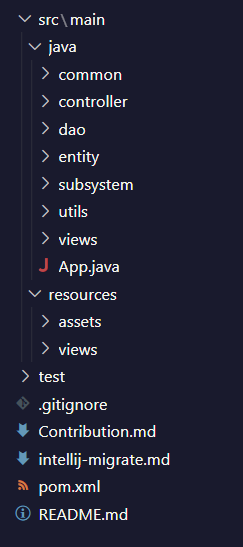
### 1.2.1. Mô tả khái quát phần mềm

Phần mềm là 1 web bán hàng với các sản phẩm: sách, đĩa CD và DVD. Người dùng muốn mua sản phẩm nào sẽ thêm sản phẩm đó vào giỏ hàng. Sau đó người dùng có thể thanh toán bằng hình thức credit card. Sau đó người dùng điền các thông tin vận chuyển để sản phẩm được giao đến đúng người.

### 1.2.2. Các chức năng chính của phần mềm

* Đăng nhập
* Các chức năng liên quan đến giỏ hàng: thêm sản phẩm, xem giỏ hàng
* Xem hóa đơn
* Điền thông tin vận chuyển
* Thanh toán

### 1.2.3 .Cấu trúc mã nguồn



### 1.2.4 Các yêu cầu thêm cần cân nhắc cùng quá trình tái cấu trúc

*-Thêm mặt hàng media mới*

*-Thay đổi khoảng cách tính sử dụng thư viện mới*

*-Thêm phương thức thanh toán mới*

*-Thay đổi lại cộng thức tính phí vận chuyển*

*-Thêm màn hình xem chi tiết sản phẩm*

*-Thay đổi yêu cầu khi load giao diện hiện tại*

*-Cập nhật lại chức năng hủy đơn hàng*

### 1.2.5 *Các hoạt động (Ví dụ: review, refactor) thực thi trên mã nguồn để đạt được mục tiêu kể trên*

* Review mã nguồn: Đánh giá, xem xét và đánh giá lại mã nguồn để tìm các vấn đề, lỗi, thiếu sót và cải thiện chất lượng code. Các hoạt động review có thể bao gồm:
  + Code review: Đánh giá từng phần code để tìm lỗi logic, kiến trúc không tốt, đặt tên không rõ ràng, v.v.
  + Design review: Xem xét kiến trúc tổng thể của hệ thống, đảm bảo tính mô-đun, tách biệt và tái sử dụng.
  + Performance review: Kiểm tra hiệu suất của mã nguồn và tìm cách tối ưu hóa nếu cần.
* Refactor code: Tái cấu trúc và cải thiện mã nguồn mà không thay đổi hành vi của nó. Các hoạt động refactor có thể bao gồm:
  + Đặt tên biến, hàm và lớp rõ ràng và mô tả chính xác mục đích và nhiệm vụ của chúng.
  + - Tách các hàm lớn thành các hàm nhỏ hơn và đơn giản hóa logic.
* Loại bỏ code trùng lặp và tạo ra các phương thức hoặc lớp tái sử dụng.
  + Cải thiện cấu trúc code, tuân thủ các nguyên tắc như nguyên tắc SOLID và design patterns.
* Testing: Tạo và thực thi các bộ kiểm thử đáng tin cậy để đảm bảo rằng mã nguồn hoạt động chính xác và không gây ra lỗi. Các loại kiểm thử có thể bao gồm:
  + Unit tests: Kiểm tra từng phần nhỏ của mã nguồn để đảm bảo tính đúng đắn và đáng tin cậy.
  + Integration tests: Kiểm tra tương tác giữa các thành phần của hệ thống để đảm bảo tính liên kết và tính toàn vẹn dữ liệu.
  + Acceptance tests: Kiểm tra xem hệ thống có đáp ứng được yêu cầu kinh doanh và mong đợi của người dùng không.
* Tổ chức mã nguồn: Sắp xếp và cấu trúc lại mã nguồn để nó dễ đọc và dễ hiểu hơn. Các hoạt động tổ chức có thể bao gồm:
  + Xóa code không sử dụng hoặc không cần thiết.
  + Sắp xếp lại mã nguồn theo thứ tự logic và logic.
  + Đặt comment chú thích rõ ràng và mô tả về các phần quan trọng của code.

### 1.2.6. *Kết quả dự kiến*

Kết quả dự kiến sẽ là một mã nguồn dễ đọc, dễ hiểu, dễ kiểm thử, dễ tái sử dụng và dễ tham gia cộng tác. Các tên biến, tên hàm và tên lớp được đặt một cách rõ ràng và mô tả chính xác mục đích và nhiệm vụ của chúng. Cấu trúc code được sắp xếp một cách hợp lý và dễ theo dõi, giúp giảm độ phức tạp và tăng tính tái sử dụng của mã nguồn.

## Danh sách thuật ngữ

## Danh sách tham khảo

1. Centers for Medicare & Medicaid Services. (n.d.). *System Design Document Template.* Retrieved from Centers for Medicare & Medicaid Services: https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SystemDesignDocument.docx

2. Cornell University How We Refactor and How We Document it? On the Use of Supervised Machine Learning Algorithms to Classify Refactoring Documentation

Retrived from www.elsevier.com/locate/eswa

# Đánh giá thiết kế cũ

## Nhận xét chung

Mã nguồn đáp ứng được những chức năng cơ bản của phần mềm. Tuy nhiên

* Mã nguồn hiện tại khó để mở rộng, nếu cần phát triển tính năng mới thì rất khó khăn.
* Vi phạm các quy tắc SOLID, clean code.
* Chưa áp dụng nhiều design pattern.
* Vi phạm các quy tắc về coupling, cohesion

## Đánh giá các mức độ coupling và cohesion

### Coupling

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Các mức độ về Coupling* | *Module* | *Mô tả* | *Lý do* |
| 1 | Content coupling | ApplicationProgrammingInterface.java | Một lớp trừu tượng có chức năng cung cấp một giao diện để thực thi các yêu cầu API (Application Programming Interface) | Lớp đã sử dụng chỉ định truy cập setAccessible (true) để get thuộc tính lớp khác. Việc đó đã vi phạm content coupling |
| 2 | Content coupling | MyMap.java | Lớp này cung cấp các phương thức để chuyển đổi giữa MyMap và chuỗi JSON, cũng như chuyển đổi giữa đối tượng Java và MyMap | Việc sử dụng chỉ định truy cập để get, set các thuộc tính của lớp khác đã vi phạm content coupling |
| 3 | Content coupling | media/Book.java, CD.java, DVD.java | Đại diện cho 1 entity cụ thể, và cung cấp các phương thức để quản lý và truy xuất các thuộc tính của entity đó | Các lớp trên public các thuộc tính của mình. Không chỉ định private cho các thuộc tính của lớp. Các lớp khác có thể lấy, sửa đổi các thuộc tính của lớp mà không cần qua method get, set |
| 4 | Data Coupling | AuthenticationController | Trong đoạn code, chúng ta có các phương thức như isAnonymousSession(), getMainUser(), login(), logout(), và md5(). Các phương thức này sử dụng dữ liệu (data) từ lớp SessionInformation để thực hiện các chức năng như kiểm tra phiên đăng nhập, lấy thông tin người dùng chính, đăng nhập, đăng xuất và mã hóa thông tin. | Data Coupling giúp tách biệt logic xử lý đăng nhập (authentication) và quản lý thông tin người dùng (user information). Điều này giúp tạo ra một sự phân tách rõ ràng giữa các thành phần trong hệ thống, từ đó tăng tính linh hoạt và khả năng tái sử dụng của mã nguồn.  Ngoài ra phương thức md5() được sử dụng để mã hóa mật khẩu trước khi kiểm tra đăng nhập. Việc sử dụng mã hóa mật khẩu có thể ảnh hưởng đến việc xử lý dữ liệu trong hệ thống và có thể liên quan đến các thành phần khác |
| 5 | Data Coupling | PlaceOrderController | Đoạn code trên là một lớp PlaceOrderController trong hệ thống quản lý đặt hàng (AIMS project). Lớp này quản lý quá trình đặt hàng và xử lý thông tin giao hàng. | Phụ thuộc vào lớp cha BaseController, vì nó kế thừa từ BaseController và sử dụng các phương thức của lớp cha.  Sử dụng lớp SessionInformation để truy cập thông tin về giỏ hàng và kiểm tra tính khả dụng của sản phẩm (checkAvailabilityOfProduct()).  Sử dụng lớp Order để tạo đơn hàng mới (createOrder()) và lớp Invoice để tạo hóa đơn mới (createInvoice()). Điều này chỉ ra một phụ thuộc dữ liệu giữa PlaceOrderController và các lớp Order và Invoice.  Sử dụng lớp DeliveryInfo để xử lý thông tin giao hàng (processDeliveryInfo()). Ngoài ra, lớp PlaceOrderController sử dụng các phương thức validate như validatePhoneNumber(), validateName(), validateAddress() để kiểm tra tính hợp lệ của thông tin giao hàng |
| 6 | Data Coupling | HomeController, PaymentController, PlaceOrderController, ViewCartController | Các lớp quản lý lần lượt các luồng sự kiện của trang home, payment, placeorder, viewcart | Các lớp kế thừa từ lớp BaseController, do đó có một phụ thuộc dữ liệu vào lớp cha. |
| 7 | Common Couling | Utils.java | Lớp quản lý việc hiển thị ngày tháng và trả về đối tượng Logger | Các lớp có thể truy cập vào dữ liệu global là DateFormat, vi phạm common coupling |
| 8 | Common Coupling | SessionInformation.java | Lớp quản lý các thông tin của 1 session, bao gồm: user, cartInstance và thời gian truy cập | Các dữ liệu User, cartInstance và localDateTime đều là dữ liệu global và có thể được truy cập bởi các module khác, dẫn đến common coupling |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Các mức độ về Coupling | Module | Mô tả | Ảnh hưởng |
| 9 | Stamp coupling | Controller.BaseController | Truyền vào 1 tham số object media nhưng không cần thiết có thể truyền vào cái id của media đó | Truyền thừa thông tin, dẫn đến sự phụ thuộc mạnh mẽ giữa các thành phần và làm giảm tính độc lập của chúng. |
| 10 | Stamp coupling | function checkMediaInCart trong Cart.java | Vì chỉ sử dụng 1 thuộc tính id trong biến media truyền vào | Truyền thừa thông tin, dẫn đến sự phụ thuộc mạnh mẽ giữa các thành phần và làm giảm tính độc lập của chúng. |
| 11 | Stamp coupling | Function caculateShippingDee trong DeliveryInfo.java | Tham số order không được sử dụng | Truyền thừa thông tin, dẫn đến sự phụ thuộc mạnh mẽ giữa các thành phần và làm giảm tính độc lập của chúng. |
| 12 | Stamp coupling | Function refund trong InterbankSubsystemController.java | Vì biến card không được sử dụng | Truyền thừa thông tin, dẫn đến sự phụ thuộc mạnh mẽ giữa các thành phần và làm giảm tính độc lập của chúng. |
| 13 | Stamp coupling | Function setupData trong  view.screen.home.LoginScreenHandler | Tham số dto không được sử dụng | Truyền thừa thông tin |
| 14 | Stamp coupling | Function setupData trong  view.screen.intro.IntroScreenHandler | Tham số dto không được sử dụng | Truyền thừa thông tin |
| 15 | Control coupling | PlaceOrderController.java | Tùy thuộc vào giá trị của info mà điểu chỉnh luồng hoạt động | hệ thống trở nên phức tạp hơn và khó khăn trong việc hiểu và bảo trì. |
| 16 | Control coupling | AuthencationController.java | Tùy thuộc vào giá trị của mainUser mà điểu chỉnh luồng hoạt động | Hệ thống trở nên phức tạp hơn và khó khăn trong việc hiểu và bảo trì. |
| 17 | Control coupling | Interbank.InterBankPayloadConverter.java | Tùy thuộc vào giá trị của responseText mà điểu chỉnh luồng hoạt động |  |

### Cohesion

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Các mức độ về Cohesion* | *Module* | *Mô tả* | *Lý do* |
| 1 | Communicational | Entity/cart/cart.java | Đại diện cho 1 entity cụ thể, và cung cấp các phương thức để quản lý và truy xuất các thuộc tính của entity đó | Các method trong class đều sử dụng dữ liệu từ cartItem |
| 2 | Procedural Cohesion | View/screen//\*\*.java | Class có chức năng cung cấp các method phục vụ cho việc chuẩn bị dữ liệu và hiện thị thông tin lên màn hình | Các method setupData, setupFunctionality, show, getBController đều có liên quan đến chức năng hiển thị |
| 3 | Coincidental cohesion | Utils.java | Lớp quản lý việc hiển thị ngày tháng và trả về đối tượng Logger | Các phương thức định dạng ngày tháng và trả về đối tượng Logger hoàn toàn không liên quan đến nhau |
| 4 | Logical cohesion | PlaceOrderController.java | Lớp quản lý các phương thức xác nhận đơn hàng, cùng với đó là tạo order và hóa đơn | Các phương thức validate không thực sự liên quan đến nhau |
| 5 | Logical cohesion | PopupScreen.java | Lớp quản lý các giao diện popup | Các phương thức hiển thị message popup không liên quan đến nhau |
| 6 | Temporal cohesion | Các class screenHandler | Các lớp quản lý việc hiển thị các giao diện | Các phương thức liên quan đến trình tự thực hiện |

## Đánh giá việc tuân theo SOLID

### SRP

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | AuthenticationController | Chịu trách nhiệm quản lý quá trình xác thực và phiên đăng nhập của người dùng trong hệ thống. Nó cung cấp các phương thức để kiểm tra trạng thái phiên đăng nhập, đăng nhập, đăng xuất và mã hóa mật khẩu. | Ngoài các method liên quan đến Authen (Login, logout, getMainUser, isAnonymousSession) thì method md5 lại không liên quan. Nên tách ra thành một class thuộc package util |
| 2 | PaymentController | Chịu trách nhiệm điều khiển quá trình thanh toán trong hệ thống. Nó cung cấp các phương thức để kiểm tra tính hợp lệ của thẻ tín dụng, thực hiện thanh toán đơn hàng và xóa giỏ hàng. | Có method getExpirationDate không liên quan đến các phương thức còn lại |
| 3 | PlaceOrderController | chịu trách nhiệm điều khiển quy trình đặt hàng trong hệ thống. Nó cung cấp các phương thức để kiểm tra tính khả dụng của sản phẩm, tạo đơn hàng và hóa đơn, xử lý thông tin vận chuyển và xác thực thông tin. | Class nên chỉ xử các phương thức liên quan đến placeOrder, các method validate nên tách thành một class riêng |
| 4 | Order.java | Vi phạm | Các method về DeliveryInfo nên viết trong class DeliveryInfo |
| 5 | InterbankPayloadConverter | Vi phạm | Các method getToday, extrachPaymentTransaction nên tách ra class khác |
| 6 | MediaDAO | Module này đảm nhiệm các hoạt động chung liên quan đến việc truy xuất và cung cấp thông tin về các đối tượng media từ cơ sở dữ liệu(Vi phạm SRP) | Thực hiện quá nhiều chức năng khác nhau như: Lấy danh sách tất cả các media từ cơ sở dữ liệu (getAllMedia); Lấy thông tin của một media cụ thể bằng ID từ cơ sở dữ liệu(getMediaById); Cập nhật một trường cụ thể của media bằng ID trong cơ sở dữ liệu(updateMediaFieldById) |
| 7 | ApplicationProgrammingInterface | Lớp này xử lý các phương thức GET và POST cùng với việc thiết lập kết nối và phương thức allowMethods. (Vi phạm SRP) | Thực hiện quá nhiều chức năng khác nhau như: Gửi yêu cầu GET đến một URL và trả về phản hồi(get); Gửi yêu cầu POST đến một URL và trả về phản hồi(post); Thiết lập kết nối HTTP(setupConnection); Thiết lập phương thức HTTP cho kết nối(allowMethods).  Ngoài ra còn chịu trách nhiệm về việc khởi tạo các đối tượng Logger và DateFormat |
| 8 | MyMap | Lớp này lưu trữ cặp key-value trong một đối tượng Map, biến đổi các đối tượng thành chuỗi JSON, biến đổi các đối tượng thành đối tượng Map, trích xuất thuật ngữ từ một chuỗi và giải thích một chuỗi JSON và chuyển đổi thành đối tượng MyMap. Vi phạm SRP | Thực hiện quá nhiều chức năng khác nhau như: Chuyển đổi đối tượng MyMap thành một chuỗi JSON(toJSON); chuyển đổi đối tượng bất kỳ thành một MyMap đại diện cho tên thuộc tính và giá trị của đối tượng(toMyMap); Trích xuất thuật ngữ từ một chuỗi JSON(getNextTerm).  Ngoài ra lớp này còn thừa kế từ LinkedHashMap và lưu trữ các cặp key-value trong một đối tượng Map. |
| 9 | HomeScreenHandler | Xử lý màn hình trang chủ của ứng dụng | Thực hiện quá nhiều chức năng khác nhau như: Lớp này quản lý giao diện màn hình chính; Xử lý logic liên quan đến hiển thị danh sách media, xử lý sự kiện click, xử lý đăng nhập, quản lý giỏ hàng, cập nhật dữ liệu và nhiều hơn nữa.  Ngoài ra phương thức “update” thực hiện nhiều chức năng, bao gồm kiểm tra số lượng media, xử lý giỏ hàng và hiển thị thông báo |
| 10 | InvoiceScreenHandler | Phụ trách việc thiết lập dữ liệu cho giao diện hóa đơn. | Thực hiện quá nhiều chức năng khác nhau như: Lớp này quản lý giao diện màn hình hóa đơn; Phương thức “setupData” chịu trách nhiệm thiết lập dữ liệu cho các thành phần giao diện người dùng (label, vbox) và xử lý danh sách invoice.getOrder().getListOrderMedia() để tạo ra các MediaInvoiceScreenHandler và thêm chúng vào vboxItems; Phương thức “setupFunctionality” không thực hiện bất kỳ chức năng nào; Phương thức “confirmInvoice” chịu trách nhiệm chuyển hướng sang màn hình thanh toán (PaymentScreen) khi người dùng xác nhận hóa đơn. |

### OCP

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | BookDAO | Lớp vi phạm OCP | Nếu muốn thêm 1 sản phẩm thì cần thêm 1 hàm getMediaById() |
| 2 | CDDAO | Lớp vi phạm OCP | Nếu muốn thêm 1 sản phẩm thì cần thêm 1 hàm getMediaById() |
| 3 | DVDDAO | Lớp vi phạm OCP | Nếu muốn thêm 1 sản phẩm thì cần thêm 1 hàm getMediaById() |

### LSV

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | Media | Vi phạm LSV | Có nhiều method get set mà các class con extend nó k dùng đến (setMediaURL,..) |
| 2 | HomeScreenHandler | Chịu trách nhiệm quản lý giao diện và xử lý logic liên quan đến màn hình chính của ứng dụng | Lớp không dùng 2 phương thức setPreviousScreen và getPreviousScreen của lớp cha là baseScreenHandler |

### ISP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Module* | | *Mô tả* | *Lý do* |
| 1 | HomeScreenHandler | Lớp vi phạm ISP | | Lớp không dùng 2 phương thức setPreviousScreen và của lớp cha là baseScreenHandler |

### DIP

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | InterbankInterface,java |  | Phụ thuộc vào Credit card mà không phải 1 abstract class |

## Các vấn đề về Clean Code

### Clear Name

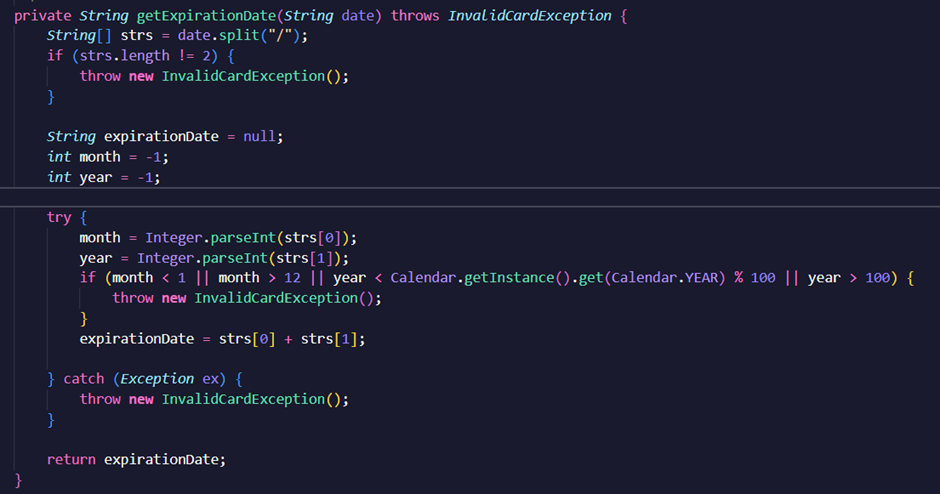
***-*** *Clean name trong codebase còn 1 vài chỗ chưa được rõ ràng*

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Class* | *Mô tả* | *Chỉnh sửa* |
| *1* | *AuthenticationController* | *Tên biến chưa thể hiện rõ chức năng* | *md5 -> md5Encryption* |
| *2* | *Utils.MyMap* | *tên MyMap không thể hiện được nhiệm vụ của lớp này.* |  |
| *3* | *InterbankSubsystem* | *Biến viết tắt không có nghĩa* | *ctrl ->* |
| *4* | *PopupScreen* | *Chưa rõ nghĩa khi gọi* | *Method success ->showPopupSuccess* |
|  | *PaymentController* | *Biến viết tắt không có nghĩa* | *Strs ->* |

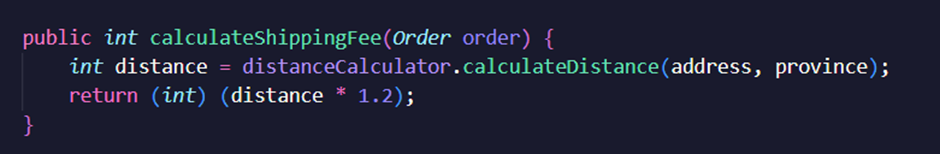
### Clean Function/Method

*Mã nguồn ban đầu chưa đáp ứng được clean function/method tại một số class.*

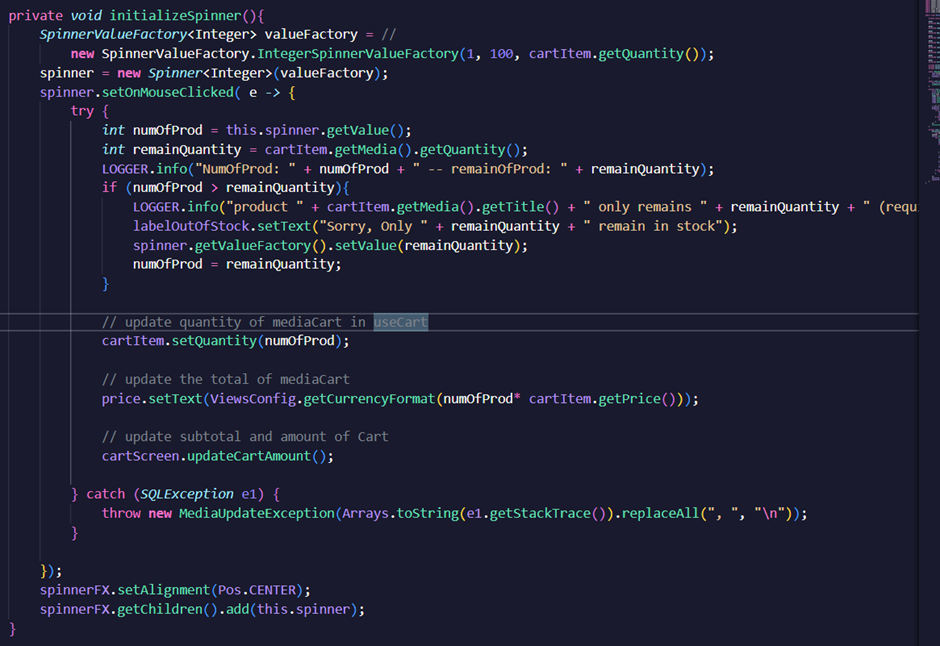
*Trong mã nguồn ban đầu, phương thức getExpirationDate() của lớp PaymentController chứa một biểu thức điều kiện phức tạp Một cách để cải thiện mã nguồn là tạo một phương thức boolean riêng, ví dụ như isValidExpirationDate(String date), để kiểm tra xem ngày hết hạn có hợp lệ hay không. Phương thức này sẽ trả về true nếu ngày hết hạn hợp lệ và false nếu không hợp lệ.*

**

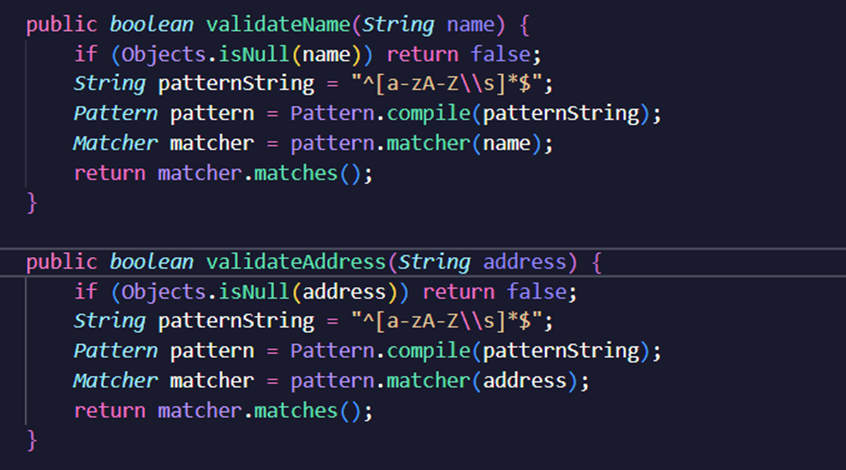
*Hàm tính phí ở class DeliveryInfo có phép nhân distance\*1.2 mà không rõ con số 1.2 có ý nghĩa gì. Cần phải đặt một biến cụ thể và cho nó giá trị bằng 1.2.*

**

*Tại class views.screen.cart.MediaHandler, method initializeSpinner(), có 2 nhiệm vụ, tạo ra Spinner và xử lý sự kiện khi click vào spinner. Bằng cách tách các nhiệm vụ thành các phương thức riêng biệt, bạn cải thiện tính dễ đọc và bảo trì của code bằng cách tách biệt logic của từng phần.*

**

*Trong PlaceOrderController : Cả hai method validateName() và validateAddress() đều có cùng logic xử lý nên bị duplicate code vì vậy cần viết lại thành 1 hàm*



### Clean Class

*Đoạn mã nguồn ban đầu có một số lớp không đáp ứng về clean class vì bị vi phạm vào một hoặc nhiều trong các yếu tố như tính đơn nhiệm, Cohesion, Coupling; nhưng về mặt tên lớp hay quy tắc viết mã nguồn vẫn được áp dụng đầy đủ*

## Các vấn đề khác

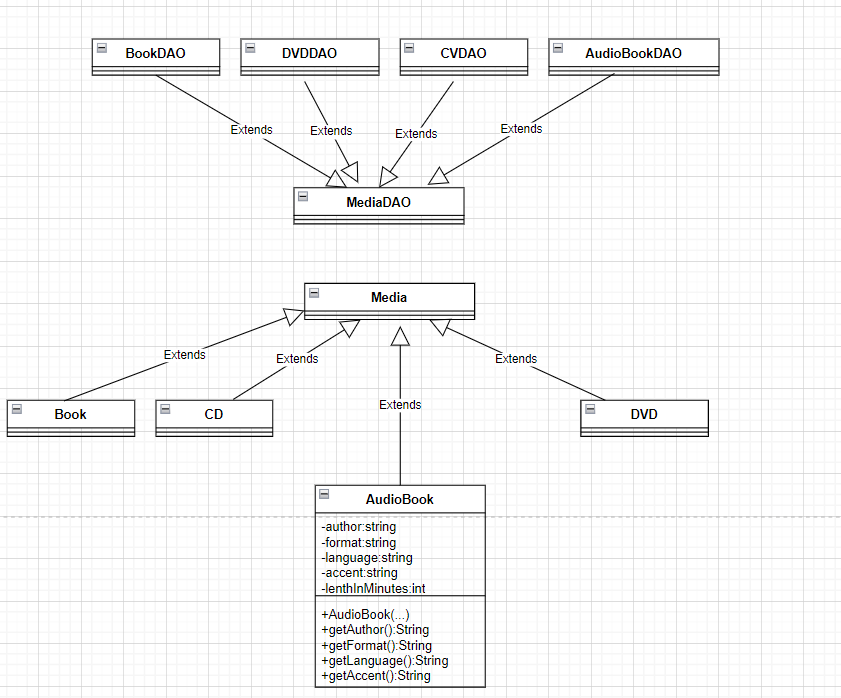
|  |  |
| --- | --- |
| *Vấn đề* | *Nội dung* |
| *Documentation* | *Trong readme không có phần hướng dẫn add “distance-api”* |
| *Unit Testing* | *Chưa tổng quát được chương trình chỉ một số test cho một số chức năng như Login, tính khoảng cách hoặc một số giá trị như địa chỉ, tên, số điện thoại* |

# Đề xuất cải tiến

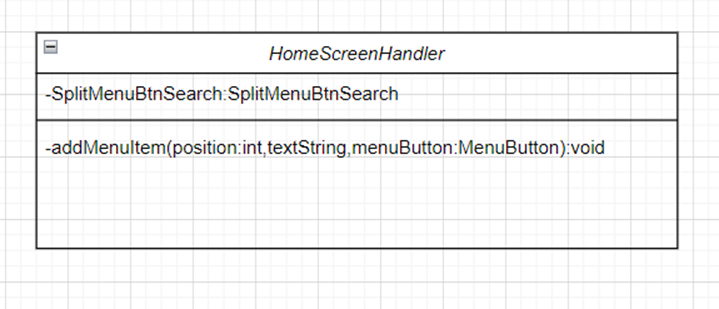
## 1. Thêm mặt hàng Media mới

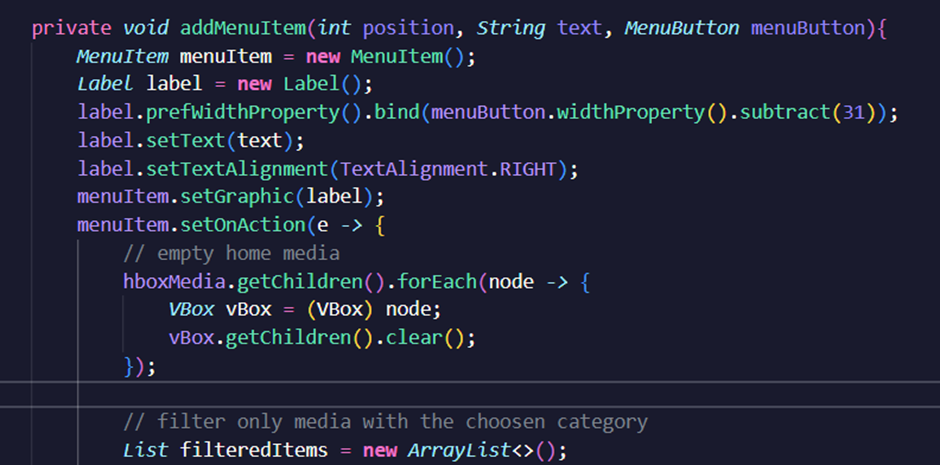
AudioBook - Thông tin về loại mặt hàng mới như sau: AudioBook + String author, ví dụ: Paulo Coelho + String format, ví dụ: mp3 AIMS: An Internet Media Store 1 + String language, ví dụ: English + String accent, ví dụ: Male - North America + int lengthInMinutes, ví dụ: 226

Do mặt hàng AudioBook là dạng mặt hàng Media và thông tin có thể lưu trong bảng Media trong cơ sở dữ liệu do cùng có một số thuộc tính như: id, title, price, quanity,… nên ta có thể xây dựng lớp AudioBook kế thừa lớp Media, lớp AudioBookDAO kế thừa lớp MediaDAO

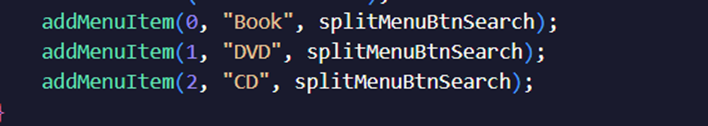
****

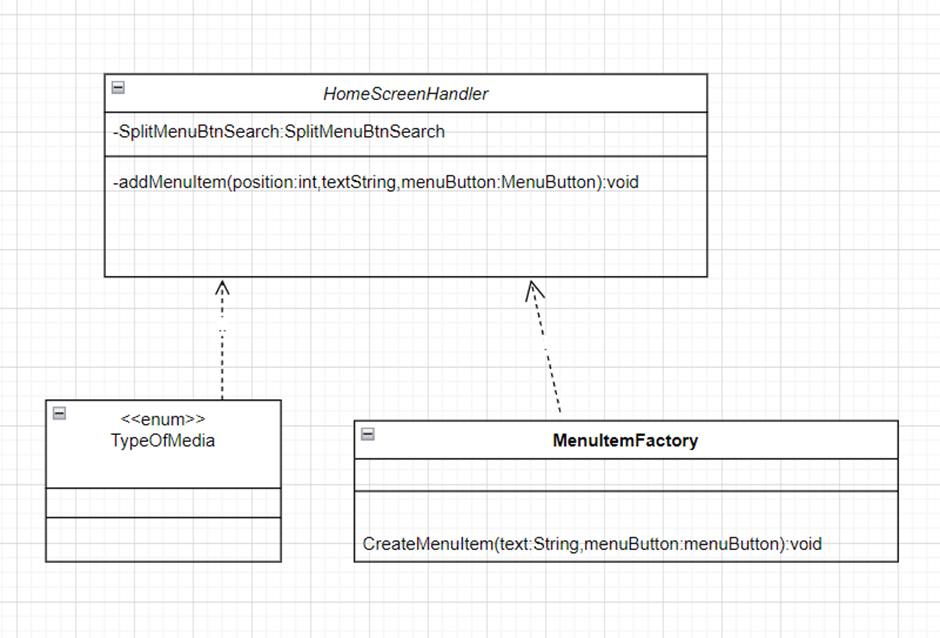
Khi thực hiện thêm mới một mặt hàng Media, class views.home.HomeScreenHandler sẽ vi phạm nguyên lý thiết kế OCP. Ngoài ra ở class này, tại hàm addMenuItem(), khởi tạo một MenuItem và xử lý logic sự kiện nên được tách riêng.

****

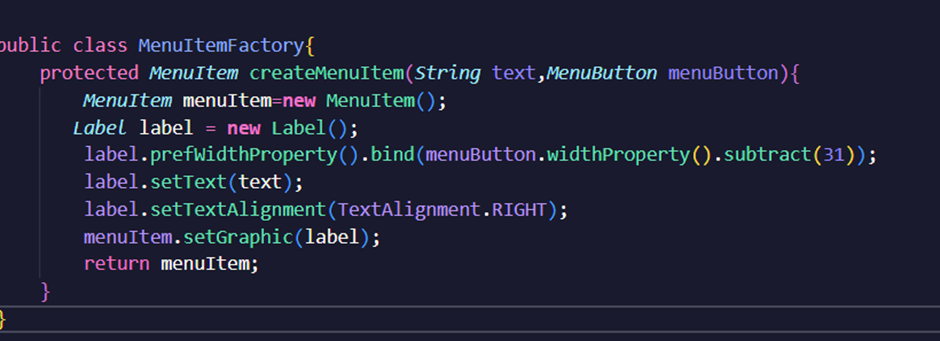
****

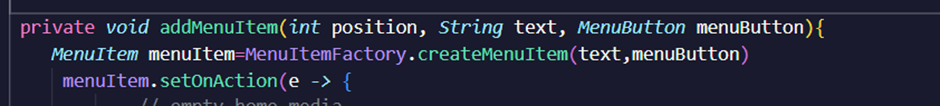
Khi thêm sản phẩm mới, cần viết thêm một lần hàm addMenuItem()

****

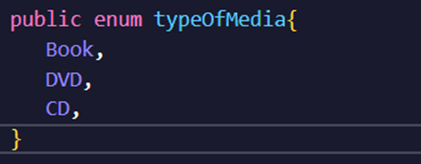
Thiết kế mới 🎉****

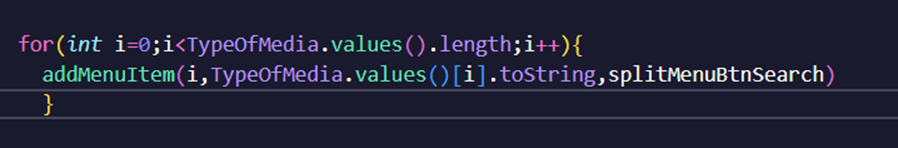
Tạo ra một class MenuItemFactory, đảm nhận chức năng tạo giao diện cho menuItem ở class HomeScreenHandler

****

****

Tạo ra một class MenuItemFactory, đảm nhận chức năng tạo giao diện cho menuItem ở class HomeScreenHandler

****

****

Tạo ra một enum chứa các thể loại Media, sau đó trong HomeScreenHandler, dùng một vòng for để addMenuItem() từng thể loại Media.

## 2. Thêm màn hình

Xem chi tiết sản phẩm Tại màn hình chính (home screen), người dùng có thể xem danh sách các sản phẩm. Bên cạnh nút “Add to cart” tại mỗi sản phẩm, chương trình cần bổ sung thêm nút “View details”. Nếu khách hàng quan tâm sản phẩm nào, có thể chọn để xem chi tiết. Tại màn hình xem chi tiết, tất cả thông tin của sản phẩm được hiển thị đầy đủ để khách hàng xem xét, kèm theo lựa chọn số lượng và nút “Add to cart”.

**Sử dụng Factory method**

A screenshot of a computer screen

Description automatically generated

A screen shot of a computer program

Description automatically generatedA screen shot of a computer program

Description automatically generated

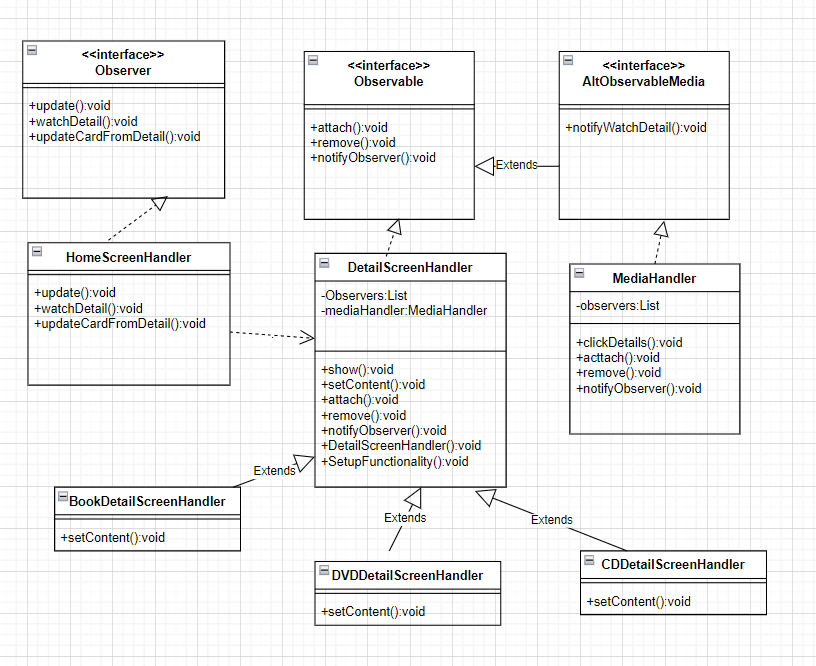
A screenshot of a computer program

Description automatically generated

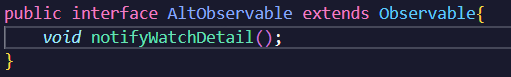
A screen shot of a computer program

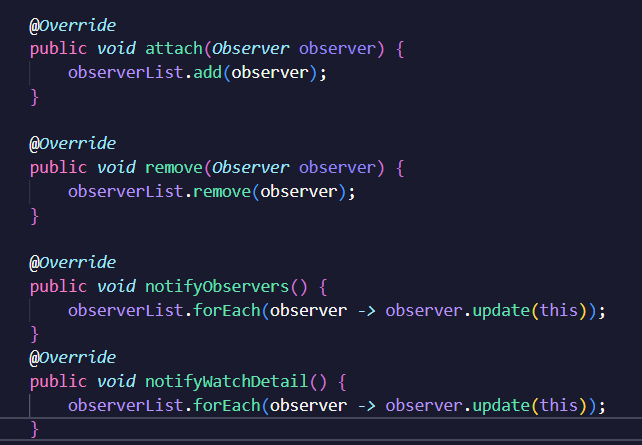
Description automatically generated

Màn Home sẽ subscribe MediaHandler và DetailScreen . Khi có sự kiện AddToCard sẽ thêm sản phẩm vào giỏ hàng. Nếu sự kiện “Xem chi tiết” thì màn Home điều hướng trang chi tiết sản phẩm. Với từng loại sản phẩm sẽ có một trang chi tiết riêng. Ở đây ta sẽ áp dụng Template Pattern cho từng trang chi tiết của sản phẩm. Lý do : Các trang chi tiết sản phẩm chỉ khách nhau setContent(), còn lại với phương thức khởi tạo, setBasicContent sẽ là như nhau.

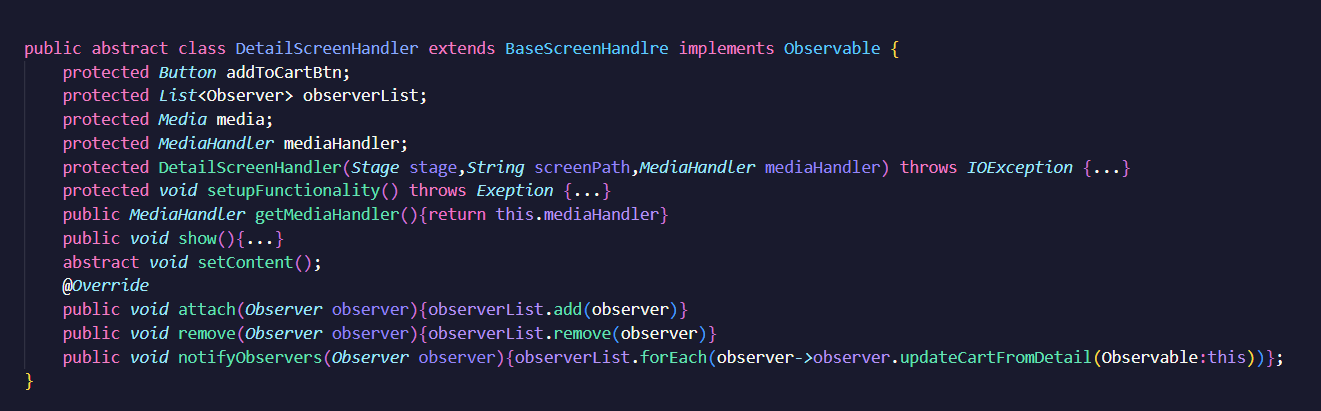
****

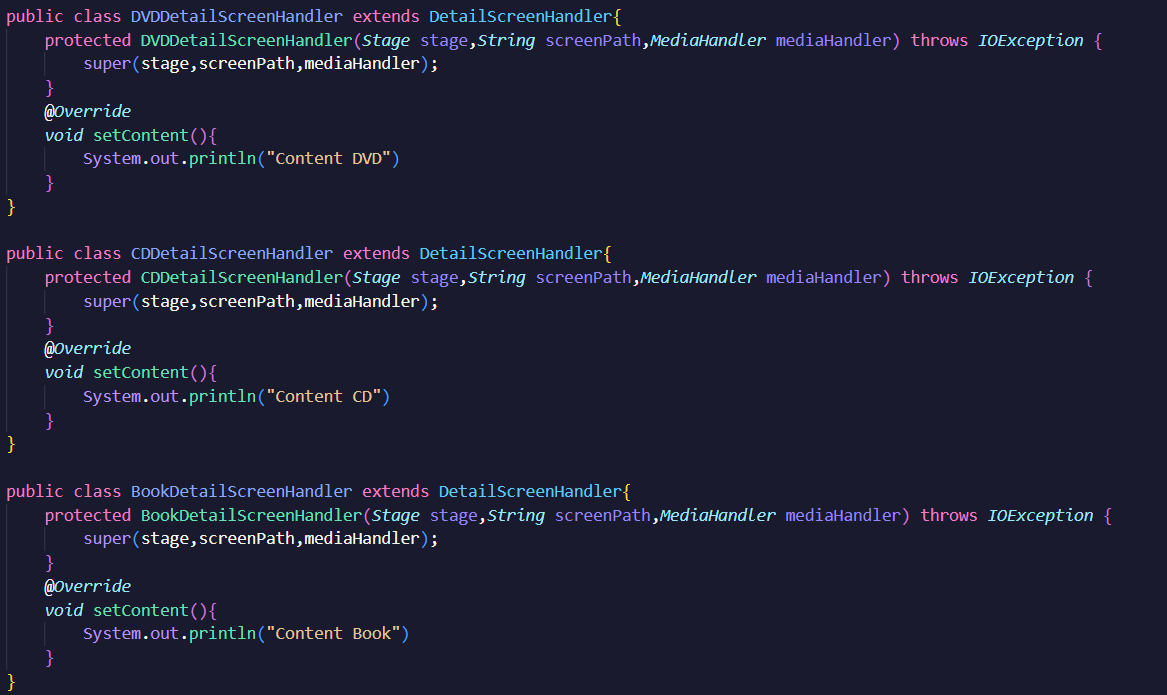
MediaHandler thêm một event notifyWatchDetail() do đó tạo một interface AltObservableMedia kế thừa Observable đã có đảm bảo tính ISP trong SOLID, và MediaHandler implement lớp AltObservableMedia

****

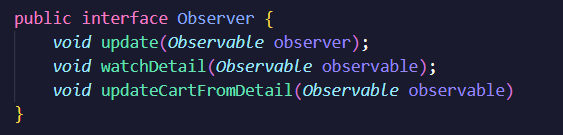
****

Tạo ra Abstract Class DetailScreenHandler phụ trách việc hiển thị trang chi tiết, theo đó có BookDetailScreenHandler, DVDDetailScreenHandler, CDDetailScreenHanlder kế thừa DetailScreenHandler - Ở đây áp dụng Template Method.

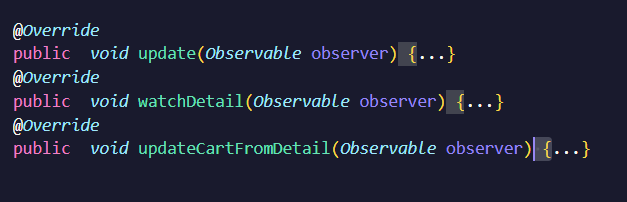
****

****

- Observer ngoài method update thêm các method watchDetail(), updateCardFromDetail()

****

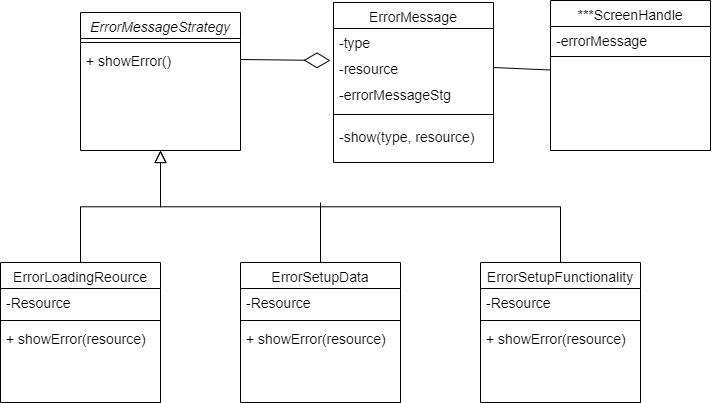
Tại màn HomeScreenHandler override lại các phương thức

****

## 3. Thay đổi yêu cầu khi load giao diện

Hiện tại, trên các màn hình, nếu xảy ra lỗi IOException sẽ hiện lên thông báo “Error when loading resources”. Yêu cầu mới: Thay vì hiển thị thông báo lỗi trên màn hình popup, chương trình hiển thị trực tiếp lỗi màu đỏ trên phía trên cùng của các trang. Thay vì thông báo “Error when loading resources”, cần bổ sung thêm lỗi khi thao tác với resource nào, và cụ thể lỗi (VD: không tìm thấy,…).

Giải pháp: Sử dụng Strategy pattern

Biểu đồ lớp: ****

**Mã nguồn minh họa  
**

## 4. Thay đổi cách tính khoảng cách, sử dụng thư viện mới

A graph with text and numbers

Description automatically generated with medium confidenceThay vì sử dụng thư viện cũ DistanceCalculator, ta sử dụng một thư viện mới là alt-distance-api.jar có interface gần giống với distance-api.jar.

A black background with white text

Description automatically generated

A screen shot of a computer screen

Description automatically generatedA screen shot of a computer program

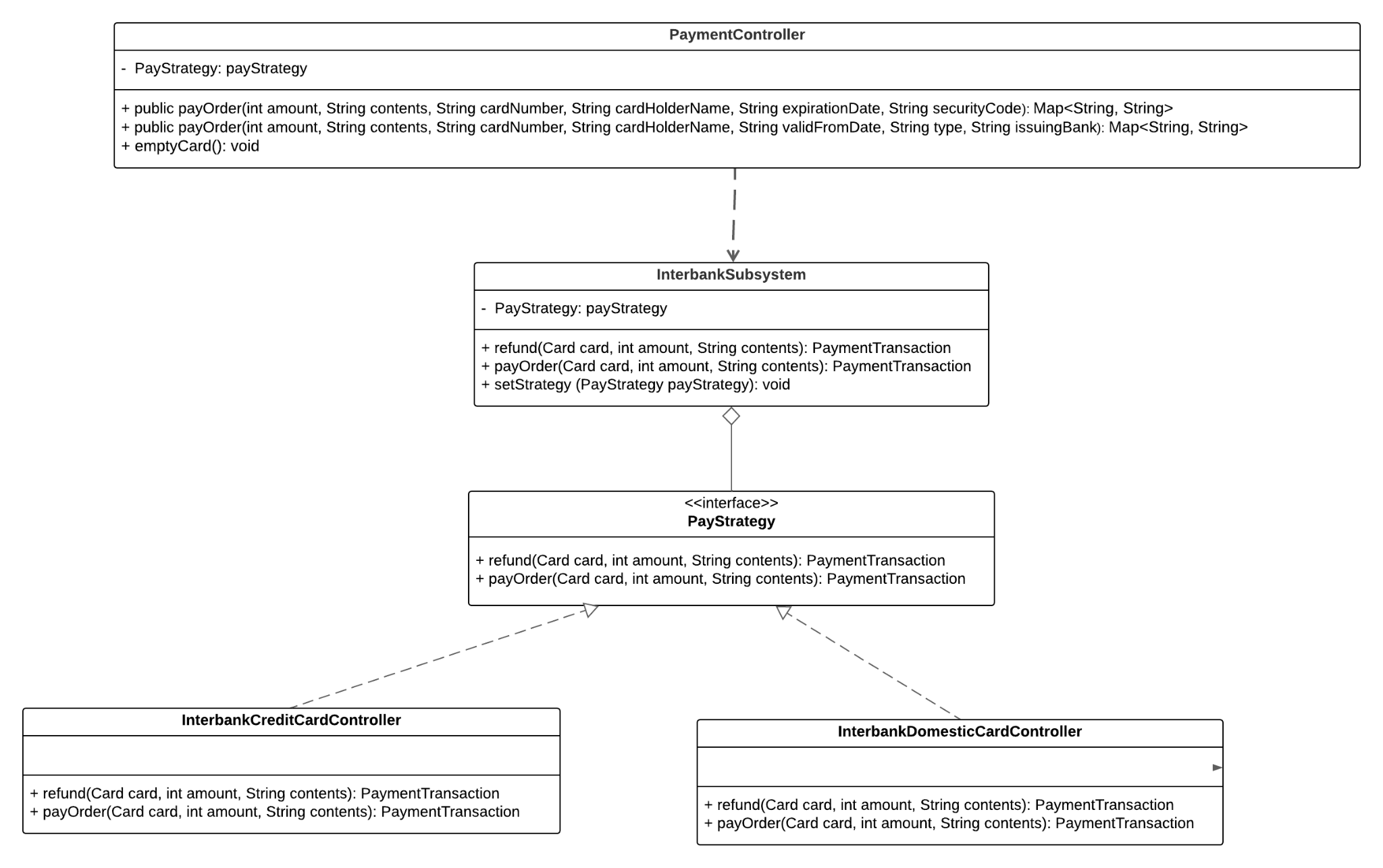
Description automatically generatedA screen shot of a computer program

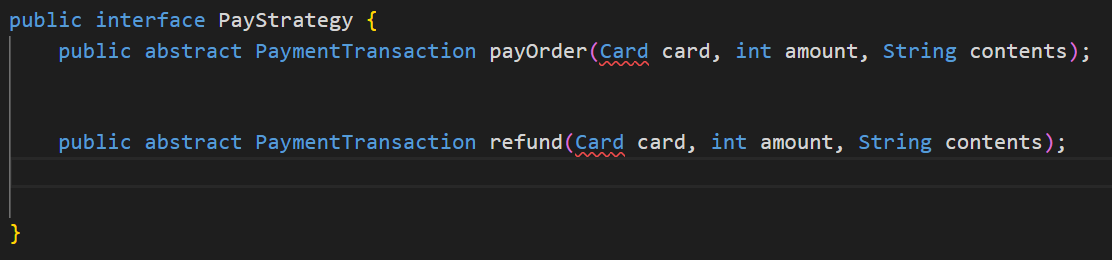
Description automatically generated

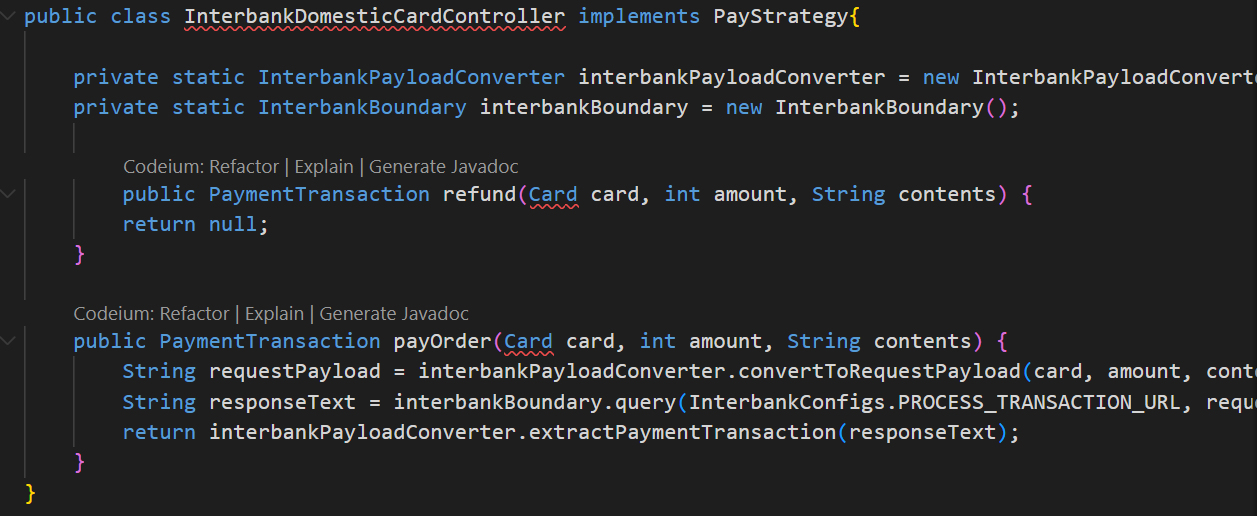
## 5 . Thêm phương thức thanh toán mới

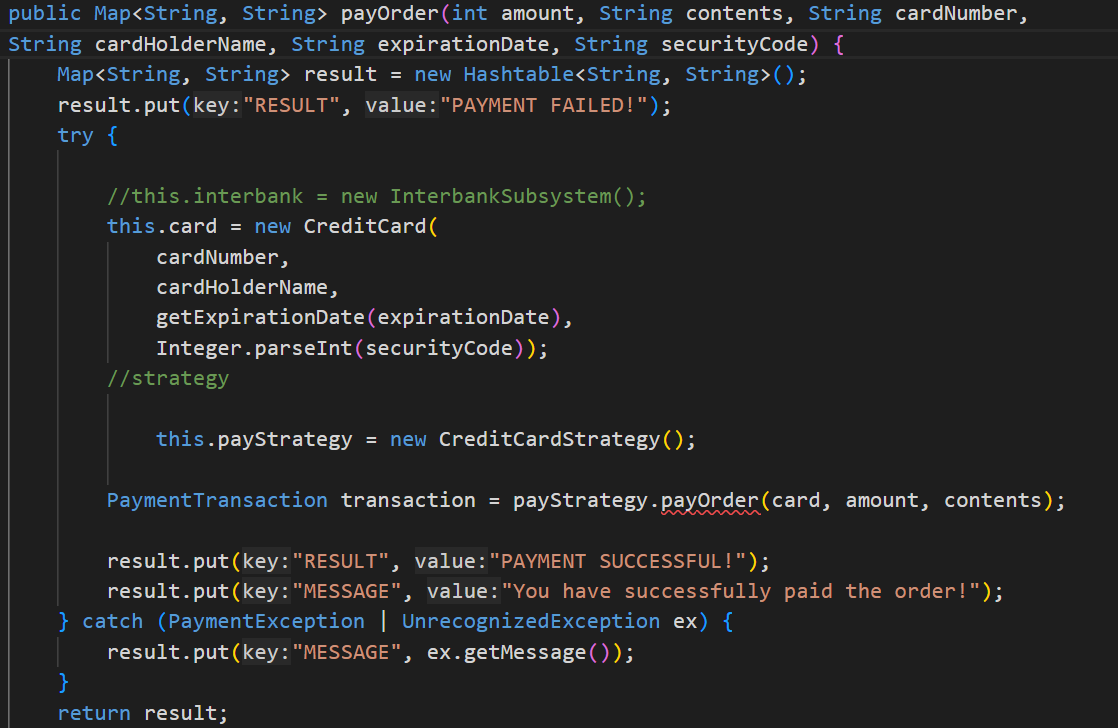
Thẻ nội địa (Domestic Card) 1. Khách hàng có thể lựa chọn phương thức thanh toán thẻ tín dụng (credit card) hoặc thẻ nội địa (domestic card). Thông tin về loại thẻ thanh toán mới như sau: + Type: Domestic Debit Card + Issuing Bank, ví dụ VietinBank + Card number 16 kí tự là chữ số + Valid-from date, ví dụ: 12/33 + Cardholder’s name, ví dụ: DO MINH HIEU Với thẻ nội địa, chương trình vẫn kết nối với API của Interbank, chỉ thay đổi thông tin của phương thức thanh toán.

**Giải pháp: sử dụng Strategy pattern**

****

****

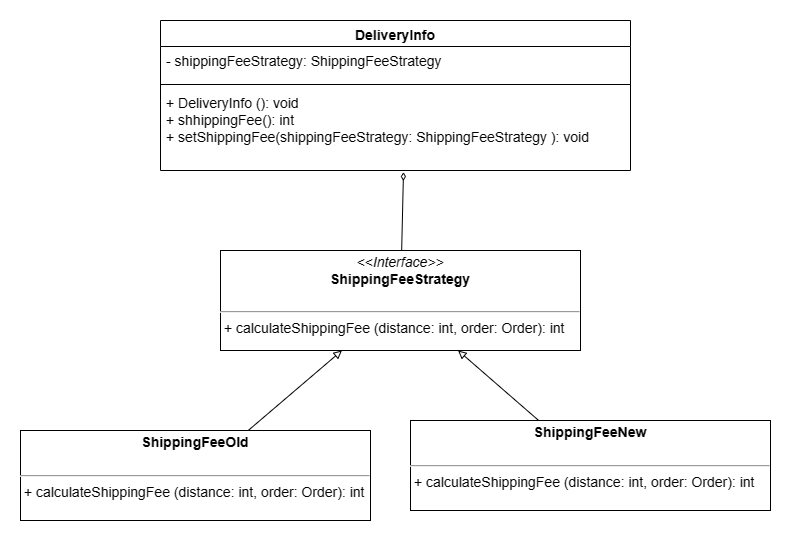
****

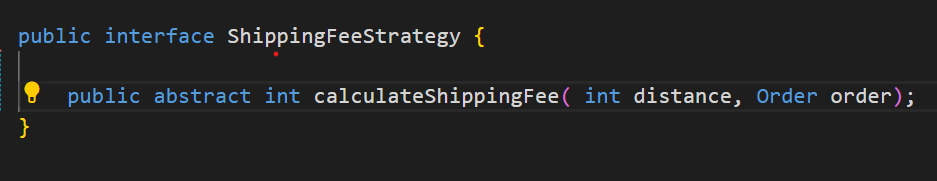
****

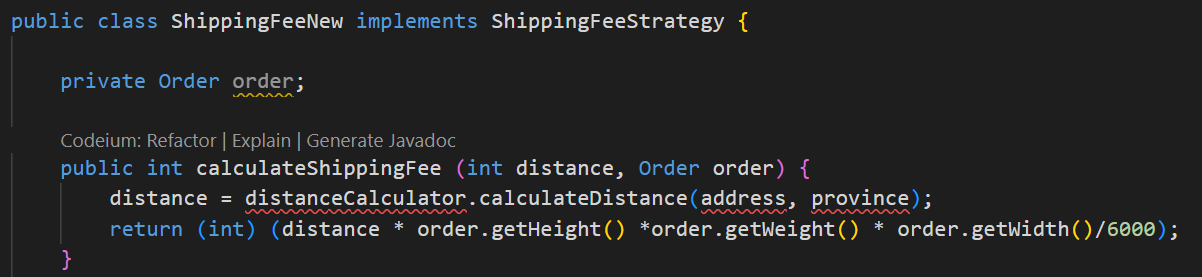
## 6. Thay đổi công thức tính phí vận chuyển

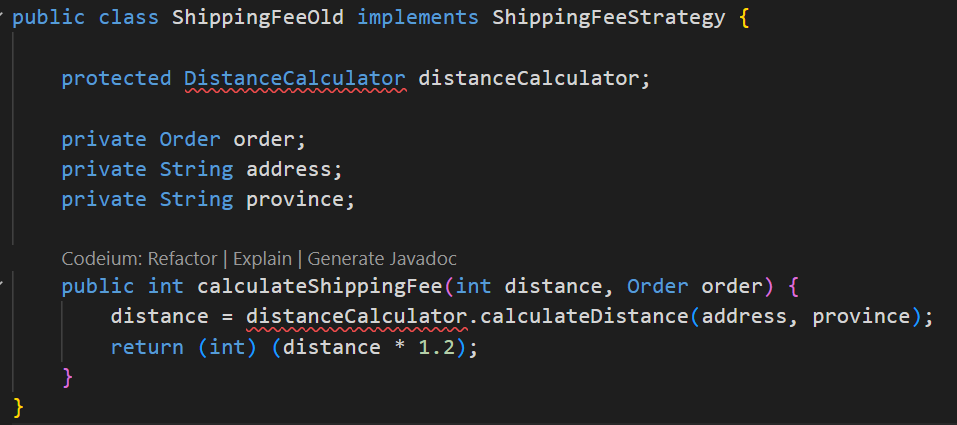
Phí vận chuyển sẽ được tính dựa theo khối lượng thực tế, khoảng cách, và độ cồng kềnh của kiện hàng. Ví dụ: Kích thước của kiện hàng: chiều dài, chiều rộng, chiều cao. Độ cồng kềnh của kiện hàng có thể lấy tượng trưng: Độ cồng kềnh (kg) = Chiều dài (cm) x Chiều rộng (cm) x Chiều cao (cm) / 6000. Khối lượng và độ cồng kềnh (đơn vị: kg) lấy hệ số là 10 (đ/kg); khoảng cách (đơn vị: km, mặc định trả về từ thư viện Distance-API) lấy hệ số là 1. Yêu cầu này có thể kết hợp cùng yêu cầu số 4.

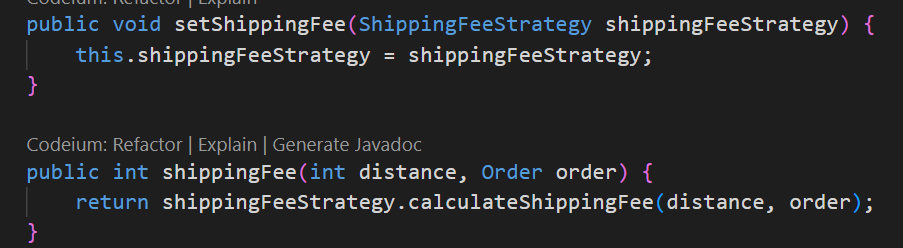
**Giải pháp: sử dụng Strategy pattern**

****

****

****

****

****

# Tổng kết

## Kết quả tổng quan

So với dự kiến ban đầu, nhóm cũng đã chỉ ra được những vi phạm của source code cũ. Và với 7 vấn đề được nêu ra thì nhóm đã giải quyết được 6 vấn đề. Bằng cách sử dụng các design pattern. Mã nguồn sẽ dễ phát triển và bảo trì hơn trong tương lai

## Các vấn đề tồn đọng

Các vấn đề còn sót lại:

- Với class ApplicationProgrammingInterface ( class có chức năng call api) mới chỉ có 2 method là get, put. Sau này có thể mở rộng lên nhiều method: delete, patch, options. Có thể thêm method mới được.

- Với class PopupScreen sau này có thể có nhiều lại popup (hiện tại chỉ có error, success, loading ) sau này mở rộng thì cần phải cải tiến lại code.

- Với vấn đề: Cập nhật lại chức năng hủy đơn hàng thì nhóm cũng chưa đưa ra được giải quyết