ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ thông tin và Truyền thông

Báo cáo Mẫu thiết kế phần mềm

Version 1.0

DP\_08

Vũ Hoàng Long – 20194106

Nguyễn Đức Thắng – 20194169

Nguyễn Văn Khoa - 20190166

*Hà Nội,* *7/2023*

Mục lục

Mục lục 1

1 Tổng quan 3

1.1 Mục tiêu 3

1.2 Phạm vi 3

1.3 Danh sách thuật ngữ 3

1.4 Danh sách tham khảo 3

2 Đánh giá thiết kế cũ 4

2.1 Nhận xét chung 4

2.2 Đánh giá các mức độ coupling và cohesion 4

2.2.1 Coupling 4

2.2.2 Cohesion 4

2.3 Đánh giá việc tuân theo SOLID 4

2.3.1 SRP 5

2.3.2 OCP 5

2.3.3 ... 5

2.4 Các vấn đề về Clean Code 5

2.4.1 Clear Name 5

2.4.2 Clean Function/Method 5

2.4.3 Clean Class 5

2.5 Các vấn đề khác 6

3 Đề xuất cải tiến 7

3.1 Vấn đề xxx và giải pháp 7

3.2 Vấn đề yyy và giải pháp 7

4 Tổng kết 8

4.1 Kết quả tổng quan 8

4.2 Các vấn đề tồn đọng 8

**Danh sách các minh họa**

**Error! No table of figures entries found.**

**Danh sách các bảng**

**Error! No table of figures entries found.**

# Tổng quan

*<Mục nội dung sau mô tả khái quát nội dung báo cáo>*

## Mục tiêu

<*Xác định mục đích sử dụng của báo cáo mẫu thiết kế phần mềm. Xác định đối tượng người đọc là ai trong quy trình xử lý phần mềm. Mô tả nội dung khái quát của báo cáo*>

Báo cáo mẫu thiết kế phần mềm sẽ đánh giá về các vấn đề của mã nguồn. Sau khi chỉ ra các vấn đề kể trên, báo cáo sẽ đề xuất các giải pháp chỉnh sửa mã nguồn.

Nội dung báo cáo gồm 2 phần chính:

* Đánh giá các vấn đề về mã nguồn: mức độ vi phạm coupling, cohesion; mức độ vi phạm các nguyên lí SOLID, các vấn đề về clean code.
* Đề xuất các giải pháp để chỉnh sửa mã nguồn.

## Phạm vi

<

*Trong mục này, cần làm rõ:*

1. *Mô tả khái quát phần mềm*
2. *Các chức năng chính của phần mềm*
3. *Cấu trúc mã nguồn*
4. *Các yêu cầu thêm cần cân nhắc cùng quá trình tái cấu trúc*
5. *Các hoạt động (Ví dụ: review, refactor) thực thi trên mã nguồn để đạt được mục tiêu kể trên*
6. *Kết quả dự kiến*

>

## Danh sách thuật ngữ

*<Danh sách và giải thích thuật ngữ sử dụng trong báo cáo. Bất kỳ giả định nào về kiến thức của người đọc cần cân nhắc thật kỹ>*

## 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

*<Danh sách các tài liệu sử dụng liên quan đến báo cáo. Bao gồm cả các tài liệu thêm cung cấp trong quá trình thực hiện tái cấu trúc>*

# Đánh giá thiết kế cũ

<*Mục này bao gồm kết quả code review trên mã nguồn case study>*

## Nhận xét chung

*<Đưa ra nhận xét chung về chất lượng mã nguồn case study hiện tại. Hiệu năng ứng biến của code đối với các kế hoạch update trong tương lai>*

## Đánh giá các mức độ coupling và cohesion

*<Sau quá trình học, đúc kết và giải thích ngắn gọn vai trò và ý nghĩa của coupling, cohesion.>*

### Coupling

*<Nhận xét tổng quát về mức độ phụ thuộc kết dính giữa các module>*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Các mức độ về Coupling* | *Module* | *Mô tả* | *Lý do* |
| 1 | Content coupling | Media | Package entity.media | Do truy cập trực tiếp vào phương thức getMediaById của MediaDAO |
| SessionInformation | Package controller | Do không có phương thức get, set ở trong lớp này nên khi cần truy cập thì truy cập trực tiếp vào |
| AuthenticationController | Package controller | Do truy cập trực tiếp vào mainUser, expiredTime ở trong SessionInformation |
| BaseController | Package controller | Do truy cập trực tiếp vào dữ liệu ở trong SessionInformation |
| PaymentController | Package controller | Do truy cập trực tiếp vào dữ liệu ở trong SessionInformation |
| ViewCartController | Package controller | Do truy cập trực tiếp vào phương thức calSubtotal trong SessionInformation |
| HomeScreenHandler | Package view.screen.home | Do truy cập trực tiếp vào phương thức getListMedia() trong lớp SessionInformation |
| 2 | Common coupling | SessionInformation,  Authentication, BaseController, Payment Controller, PlaceOrderController,  ViewCartController, Order, MediaHandler, HomeScreenHandler | Các lớp thuộc các package controller, entity.order, view.screen.home | Các lớp thuộc các package khác nhau tuy nhiên lại cùng đọc và thay đổi dữ liệu của lớp SessionInformation do đó giảm khả năng bảo trì hoặc tái sử dụng |
|  | Stamp coupling | PlaceOrderControlller | Lớp PlaceOrderController thuộc package controller dùng để xác nhận thông tin giao hang bao gồm phone, name, address | Lớp PlaceOrderController để xác nhận các thuộc tính phone, name, address của DeliveryInfo nên không cần truyền cả object vào |
| Cart | Lớp Cart thuộc package entity.cart có phương thức checkMediaInCart dùng để kiểm tra Media trong giỏ hàng | Lớp Cart có phương thức checkMediaInCart để trả lại CartItem thông qua id của media, do chỉ cần dùng id của media nên không cần phải truyền cả object Media vào |
|  | Data coupling |  |  |  |

### Cohesion

*<Nhận xét tổng quát về mức độ liên kết chặt chẽ giữa các submodule>*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Các mức độ về Cohesion* | *Module* | *Mô tả* | *Lý do* |
| 1 | Coincidental cohesion | DeliveryInfo | Lớp thuộc gói entity.shiping liên quan đến thông tin vận chuyển đơn hàng của order. | Phương thức calculateShippingFee nên được để trong lớp Order. Chức năng của lớp là lưu thông tin vấn chuyển, không phải là tính phí vận chuyển. |
| 2 | Logical cohesion | PopupScreen | Lớp thuộc package views.screen.popup dùng để hiển thị các popup cho các lớp view như success, error, loading | Lớp PopupScreen đang phải hiển thị nhiều kiểu popup thông qua các phương thức khác nhau: success, error, loading. Các phương thức này tuy thực hiện cùng kiểu chức năng nhưng lại có cách thực hiện khác nhau. |
| 3 | Temporal cohesion | HomeScreenHandler và các lớp thuộc gói views | Lớp thuộc package views.screen.home để hiển thị màn hình home | Hai phương thức setupData và setupFunctionality được thực hiện theo một tiến trình nhưng không có liên quan nào khác. |

## Đánh giá việc tuân theo SOLID

*<Nếu phải chỉnh sửa mã nguồn theo các yêu cầu phát sinh thì bản thiết kế và mã nguồn ban đầu có tuân theo các nguyên lý thiết kế SOLID hay không?>*

### SRP

Nhìn chung source code đã đáp ứng khá tốt nguyên lí SRP. Các module chỉ làm một nhiệm vụ liên quan đến một nghiệp vụ nào đó. Tuy nhiên, vẫn còn một số module vi phạm nguyên lí SRP:

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | PopupScreen | Lớp thuộc package views.screen.popup dùng để hiển thị các popup cho các lớp view như success, error, loading | Lớp PopupScreen đang phải hiển thị nhiều kiểu popup thông qua các phương thức khác nhau: success, error, loading. Các phương thức này tuy thực hiện cùng kiểu chức năng nhưng lại có cách thực hiện khác nhau. |
| 2 | DeliveryInfo | Lớp thuộc gói entity.shiping liên quan đến thông tin vận chuyển đơn hàng của order. | Phương thức calculateShippingFee nên được để trong lớp Order. Chức năng của lớp là lưu thông tin vấn chuyển, không phải là tính phí vận chuyển. |

### OCP

*<Nhận xét chung về tình trạng thiết kế và mã nguồn trên nguyên lý SRP>*

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | PopupScreen | Lớp thuộc gói views.screen.popup dùng để hiển thị các popup cho các lớp view như success, error, loading | Trong contructor, nếu có 1 popup mới có liểu style khác với điều kiện khác (thêm tham số mới khác với undecorated) thì sẽ phải thay đổi constructor bằng cách thêm câu lệnh if. |
| 2 | MediaDAO | Lớp thuộc gói dao.media dùng để kết nối với database của các lớp Media | Phương thức getAllMedia mới chỉ lấy thông tin chung của media. Nếu có yêu cầu lấy media với instance là của các lớp con, sẽ cần sử dụng câu lệnh điều kiện switch case xét theo type media để gọi đến phương thức của lớp con tương ứng. |
| 3 | InterbankPayloadConverter | Lớp thuộc gói subsystem.interbank dùng để chuyển đổi các trường cho request, response cho chức năng thanh toán. Thanh toán phụ thuộc vào hình thức thanh toán (card). | Khi có một phương thức thanh toán mới, thông tin thanh toán sẽ thay đổi, request, response với interbank cũng sec thay đổi. |

### DIP

*<Tương tự>*

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Mô tả* | *Lý do* |
| 1 | DeliveryInfo | Lớp thuộc gói entity.shiping liên quan đến thông tin vận chuyển đơn hàng của order. | Lớp DeliveryInfo sử dụng trực tiếp phương thức calculateDistance của lớp DistanceCalculator. Lớp Order phụ thuộc lớp DeliveryInfo, đồng thời DeliveryInfo cũng phụ thuộc Order. |
| 2 | MediaDAO | Lớp thuộc gói dao.media dùng để kết nối với database của các lớp Media | Phương thức getAllMedia mới chỉ lấy thông tin chung của media. Nếu có yêu cầu lấy media với instance là của các lớp con, sẽ cần sử dụng câu lệnh điều kiện switch case xét theo type media để gọi đến phương thức của lớp con tương ứng. |

## Các vấn đề về Clean Code

### Clear Name

*<Nhận xét tình trạng mã nguồn ban đầu có đáp ứng clear name hay không?>*

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Module* | *Phần vi phạm* | *Sửa đổi* |
| 1 | DeliveryInfo | Chỉ số 1.2 trong phương thức calculateShippingFee khá khó hiểu. | DISTANCE\_RATIO = 1.2 |

### Clean Function/Method

Mã nguồn đã đáp ứng tốt về clean method, không cần sửa đổi gì thêm.

### Clean Class

Mã nguồn đã đáp ứng tốt về clean class, không cần sửa đổi gì thêm.

# Đề xuất cải tiến

Một số yêu cầu thay đổi có thể dẫn đến vi phạm các mức độ cohesion, coupling, nguyên lí SOLID của mã nguồn được tóm tắt như sau:

(1) Khi thêm một loại Media mới dẫn đến thay đổi cách lấy thông tin của Media.

(2) Khi thêm một loại popup mới, phải thêm phương thức mới cho popup, lâu dần sẽ khiến popup thành god class.

(3) Khi thay đổi thư viện tính khoảng cách hoặc thay đổi cách tính giá ship, cần thay đổi nhiều ở lớp Order.

(4) Khi có thêm phương thức thanh toán mới, cần sửa đổi mã nguồn để xử lí được thông tin thanh toán mới.

## Vấn đề (1) và giải pháp

Lớp MediaDAO đã vi phạm nguyên lí OCP, do phương thức getAllMedia mới chỉ lấy thông tin chung của media. Nếu có yêu cầu lấy media với instance là của các lớp con, sẽ cần sử dụng câu lệnh điều kiện switch case xét theo type media để gọi đến phương thức của lớp con tương ứng.

Giải pháp đề xuất: sử dụng mẫu thiết kế Factory Method để tạo ra từng loại Media riêng, lấy thông tin theo từng Media.

Sơ đồ class minh họa giải pháp:

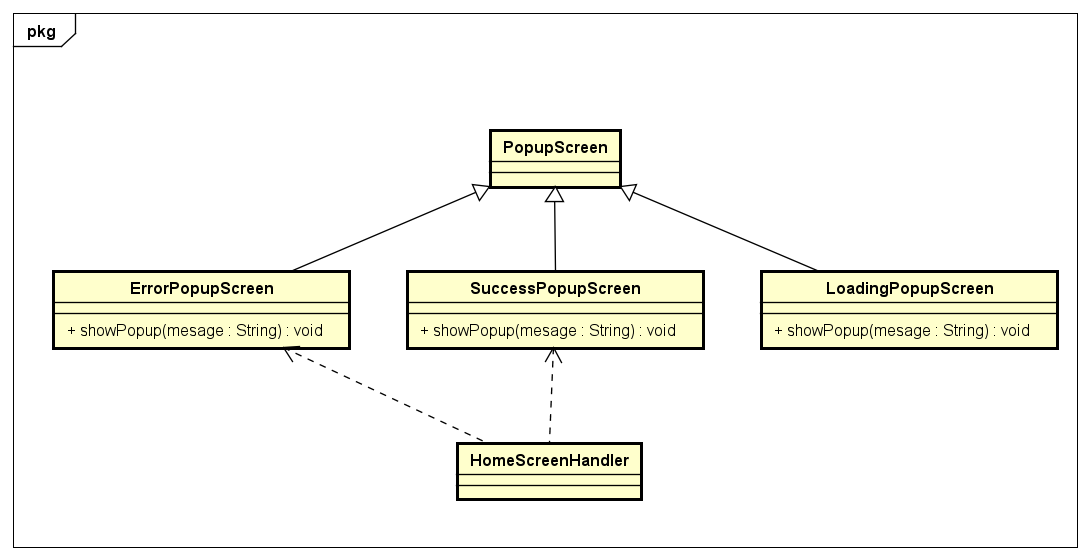
Mã nguồn mô tả giải pháp:

## Vấn đề (2) và giải pháp

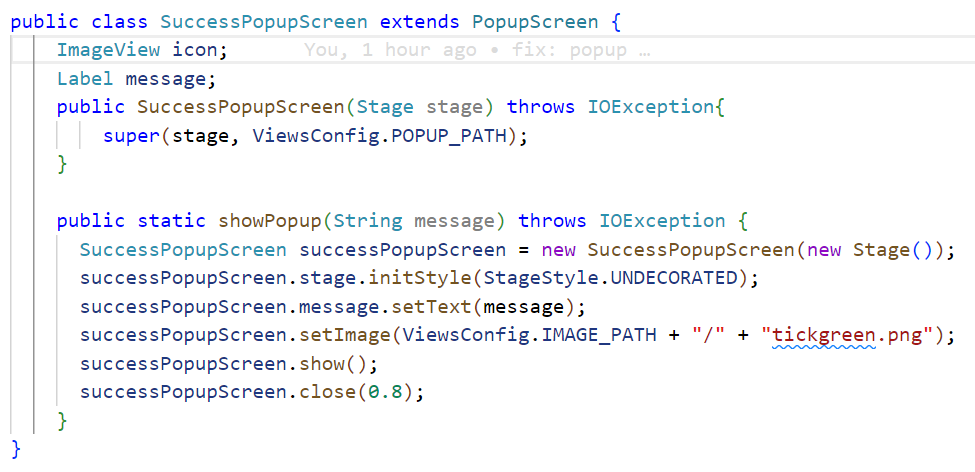
Lớp PopupScreen hiện đang có 2 phương thức success, loading, error. Nếu thêm một loại popup mới, ta cần thêm một phương thức mới. Điều này làm cho PopupScreen có thể trở thành god class. Điều này khiến PopupScreen vi phạm logical coupling và nguyên lí SRP.

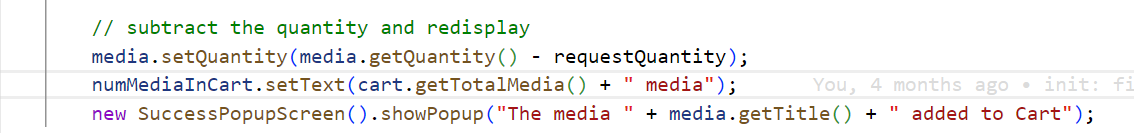
Giải pháp: tạo ra các lớp con tương ứng với mỗi loại popup. Nếu Screen nào cần dùng popup sẽ gọi đến từng lớp popup con. Mỗi lớp popup con sẽ được tạo bằng các cách khác nhau, có image khác nhau, thời gian hiện/ẩn khác nhau.

Sơ đồ class minh họa:



Mã nguồn minh họa:





# Tổng kết

***<Đúc kết lại hiệu năng của thiết kế sau khi tái cấu trúc. Mô tả các vấn đề còn sót lại hoặc chưa kịp thực hiện. >***

## Kết quả tổng quan

*<So sánh với dự kiến ban đầu, kết quả đã thực hiện được hoàn thiện được bao nhiêu kỳ vọng*>

## Các vấn đề tồn đọng

*<Nêu ra các vấn đề còn sót lại trong mã nguồn. Đánh giá khả năng xử lý*>