TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ MÔN**

**MẪU THIẾT KẾ**

**PHẦN MỀM QUẢN LÝ BÁN HÀNG**

**CHO CỬA HÀNG TIỆN LỢI**

*Người hướng dẫn*: **THẦY VŨ ĐÌNH HỒNG**

*Người thực hiện*:**VŨ LƯƠNG NGỌC BAN – 52000010**

**LƯU KIẾN VĂN – 51800832**

**ĐẶNG PHÙNG THIÊN ÂN – 52000622**

**NGUYỄN KHƯƠNG VIỆT TIẾN -** **52000474**

Lớp **: 20050201**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ MÔN**

**MẪU THIẾT KẾ**

**PHẦN MỀM QUẢN LÝ BÁN HÀNG**

**CHO CỬA HÀNG TIỆN LỢI**

*Người hướng dẫn*: **THẦY VŨ ĐÌNH HỒNG**

*Người thực hiện*: **VŨ LƯƠNG NGỌC BAN – 52000010**

**LƯU KIẾN VĂN – 51800832**

**ĐẶNG PHÙNG THIÊN ÂN – 52000622**

**NGUYỄN KHƯƠNG VIỆT TIẾN - 52000474**

Lớp **: 20050201**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Chúng em gửi lời cảm ơn chân thành đến thầy Vũ Đình Hồng đã nhiệt tình giảng dạy, hỗ trợ chúng em trong môn học Mẫu thiết kế. Qua môn học này, chúng em đã học được những kiến thức mới về các mẫu thiết kế, cách áp dụng chúng vào các dự án thực tế.

Chúng em cũng muốn gửi lời cảm ơn đến Ban giám hiệu và các giảng viên khác đã hỗ trợ chúng em có môi trường học tập và nghiên cứu. Chúng em tin rằng những kiến thức học được từ môn học này sẽ giúp chúng em phát triển trong kỹ năng của mình trong tương lai.

Em xin chân thành cảm ơn !

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của thầy Vũ Đình Hồng. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào chúng xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Ban*

*Văn*

*Ân*

*Tiến*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Dưới đây là đồ án cuối kỳ môn Mẫu thiết kế, đồ án trình bày về cách áp dụng các mẫu thiết kế vào phần mềm quản lý bán hàng cho cửa hàng tiện lợi

Để thực hiện đồ án này, chúng em đã sử dụng các kiến thức về ngôn ngữ lập trình C#, hệ cơ sở dữ liệu Microsoft SQL Server và các design pattern để nâng cao chất lượng phần mềm.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc133099314)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc133099315)

[TÓM TẮT iv](#_Toc133099316)

[MỤC LỤC 1](#_Toc133099317)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 3](#_Toc133099318)

[CHƯƠNG 1 – MỞ ĐẦU 5](#_Toc133099319)

[CHƯƠNG 2 – CÁC MẪU THIẾT KẾ ĐƯỢC ÁP DỤNG 6](#_Toc133099320)

[2.1 Singleton pattern 6](#_Toc133099321)

[2.1.1 Lý do áp dụng 6](#_Toc133099322)

[2.1.2 Sơ đồ lớp 6](#_Toc133099323)

[2.1.3 Code áp dụng mẫu thiết kế: 7](#_Toc133099324)

[2.2 Strategy pattern 8](#_Toc133099325)

[2.2.1 Lý do áp dụng 8](#_Toc133099326)

[2.2.2 Sơ đồ lớp 8](#_Toc133099327)

[2.2.3 Code áp dụng mẫu thiết kế: 9](#_Toc133099328)

[2.3 Simple Factory 14](#_Toc133099329)

[2.3.1 Lý do áp dụng 14](#_Toc133099330)

[2.3.2 Sơ đồ lớp 14](#_Toc133099331)

[2.3.3 Code áp dụng mẫu thiết kế: 15](#_Toc133099332)

[2.4 Abstract Factory 16](#_Toc133099333)

[2.4.1 Lý do áp dụng 16](#_Toc133099334)

[2.4.2 Sơ đồ lớp 16](#_Toc133099335)

[2.4.3 Code áp dụng mẫu thiết kế: 17](#_Toc133099336)

[2.5 Command Pattern 19](#_Toc133099337)

[2.5.1 Lý do áp dụng 19](#_Toc133099338)

[2.5.2 Sơ đồ lớp 19](#_Toc133099339)

[2.5.3 Code áp dụng mẫu thiết kế: 19](#_Toc133099340)

[2.6 Template Method 20](#_Toc133099341)

[2.6.1 Lý do áp dụng 20](#_Toc133099342)

[2.6.2 Sơ đồ lớp 20](#_Toc133099343)

[2.6.3 Code áp dụng mẫu thiết kế: 21](#_Toc133099344)

[2.7 Observer Pattern 22](#_Toc133099345)

[2.7.1 Lý do áp dụng 22](#_Toc133099346)

[2.7.2 Sơ đồ lớp 22](#_Toc133099347)

[2.7.3 Code áp dụng mẫu thiết kế: 23](#_Toc133099348)

[2.8 Decorator Pattern 25](#_Toc133099349)

[2.8.1 Lý do áp dụng 25](#_Toc133099350)

[2.8.2 Sơ đồ lớp 25](#_Toc133099351)

[2.8.3 Code áp dụng mẫu thiết kế: 25](#_Toc133099352)

[2.9 MVC Pattern 28](#_Toc133099353)

[2.9.1 Lý do áp dụng 28](#_Toc133099354)

[2.9.2 Sơ đồ lớp 28](#_Toc133099355)

[2.9.3 Code áp dụng mẫu thiết kế: 29](#_Toc133099356)

[2.10 Proxy pattern 30](#_Toc133099357)

[2.10.1 Lý do áp dụng 30](#_Toc133099358)

[2.10.2 Sơ đồ lớp 30](#_Toc133099359)

[2.10.3 Code áp dụng mẫu thiết kế: 31](#_Toc133099360)

[TÀI LIỆU THAM KHẢO 32](#_Toc133099361)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 2.1 Một trong những sơ đồ lớp singleton pattern(1) 6](#_Toc133099362)

[Hình 2.2 Một trong những sơ đồ lớp singleton pattern (2) 7](#_Toc133099363)

[Hình 2.3 Code áp dụng singeton pattern 7](#_Toc133099364)

[Hình 2.4. Sơ đồ lớp strategy pattern doanh thu 8](#_Toc133099365)

[Hình 2.5 Sơ đồ lớp strategy pattern kiểm tra số liệu đầu vào 9](#_Toc133099366)

[Hình 2.6 Code áp dụng Strategy pattern (1) 9](#_Toc133099367)

[Hình 2.7 Code áp dụng Strategy pattern (2) 10](#_Toc133099368)

[Hình 2.8 Code áp dụng Strategy pattern (3) 10](#_Toc133099369)

[Hình 2.9 Code áp dụng Strategy pattern (4) 11](#_Toc133099370)

[Hình 2.10 Code áp dụng Strategy pattern (5) 11](#_Toc133099371)

[Hình 2.11 Code áp dụng Strategy pattern (6) 12](#_Toc133099372)

[Hình 2.12 Code áp dụng Strategy pattern (7) 12](#_Toc133099373)

[Hình 2.13 Code áp dụng Strategy pattern (8) 13](#_Toc133099374)

[Hình 2.14 Code áp dụng Strategy pattern (9) 13](#_Toc133099375)

[Hình 2.15 Sơ đồ lớp Simple factory 14](#_Toc133099376)

[Hình 2.16 Code áp dụng Simple factory (1) 15](#_Toc133099377)

[Hình 2.17 Code áp dụng Simple factory (2) 15](#_Toc133099378)

[Hình 2.18 Sơ đồ lớp Abstract factory 16](#_Toc133099379)

[Hình 2.19 Code áp dụng Abstract factory (1) 17](#_Toc133099380)

[Hình 2.20 Code áp dụng Abstract factory (2) 18](#_Toc133099381)

[Hình 2.21 Code áp dụng Abstract factory (3) 18](#_Toc133099382)

[Hình 2.22 Sơ đồ Command pattern 19](#_Toc133099383)

[Hình 2.23 Code áp dụng Command pattern 19](#_Toc133099384)

[Hình 2.24 Sơ đồ lớp Template Method 20](#_Toc133099385)

[Hình 2.25 Code áp dụng Template Method (1) 21](#_Toc133099386)

[Hình 2.26 Code áp dụng Template method (2) 21](#_Toc133099387)

[Hình 2.27 Sơ đồ Observer pattern 22](#_Toc133099388)

[Hình 2.28 Code áp dụng Observer pattern (1) 23](#_Toc133099389)

[Hình 2.29 Code áp dụng Observer pattern (2) 23](#_Toc133099390)

[Hình 2.30 Code áp dụng Observer pattern (3) 24](#_Toc133099391)

[Hình 2.31 Code áp dụng Observer pattern (4) 24](#_Toc133099392)

[Hình 2.32 Sơ đồ lớp Decorator pattern 25](#_Toc133099393)

[Hình 2.33 Code áp dụng Decorator pattern (1) 25](#_Toc133099394)

[Hình 2.34 Code áp dụng Decorator pattern (2) 26](#_Toc133099395)

[Hình 2.35 Code áp dụng Decorator pattern (3) 26](#_Toc133099396)

[Hình 2.36 Code áp dụng Decorator pattern (4) 27](#_Toc133099397)

[Hình 2.37 Code áp dụng Decorator pattern (5) 27](#_Toc133099398)

[Hình 2.38 Một trong những sơ đồ lớp MVC (1) 28](#_Toc133099399)

[Hình 2.39 Một trong những sơ đồ lớp MVC (2) 29](#_Toc133099400)

[Hình 2.40 Code áp dụng MVC (1) 29](#_Toc133099401)

[Hình 2.41 Code áp dụng MVC (2) 30](#_Toc133099402)

[Hình 2.42 Sơ đồ lớp Proxy pattern 30](#_Toc133099403)

[Hình 2.43 Code áp dụng Proxy pattern (1) 31](#_Toc133099404)

[Hình 2.43 Code áp dụng Proxy pattern (2) 31](#_Toc133099405)

CHƯƠNG 1 – MỞ ĐẦU

Phần mềm quản lý bán hàng cho cửa hàng tiện lợi là một dự án nhằm tạo ra một phần mềm hỗ trợ những nhân viên ở cửa hàng tiện lợi có thể dễ dàng hơn để hoàn thành công việc bán hàng của mình.

Phần mềm được thiết với giao diện thân thiện và dễ dàng sử dụng với đa số người dùng. Nhân viên cửa hàng có thể đăng nhập vào phần mềm thực hiện các chức năng chức năng thanh toán, quản lý sản phẩm cũng như doanh thu cửa hàng.

Tuy nhiên phần mềm ban đầu còn đơn giản, thiếu các chức năng và chưa tối ưu, cho nên chúng em sẽ áp dụng các mẫu thiết kế phù hợp để nâng cao chất lượng cho phần mềm.

Yêu cầu phần mềm sau khi áp dụng các mẫu thiết kế là phải đảm bảo phần mềm hoạt động tốt, giao diện thân thiện dễ tương tác, có đủ các chức năng cần thiết cũng như tối ưu chất lượng cho phần mềm.

CHƯƠNG 2 – CÁC MẪU THIẾT KẾ ĐƯỢC ÁP DỤNG

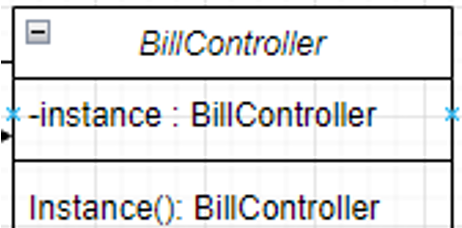
2.1 Singleton pattern

2.1.1 Lý do áp dụng

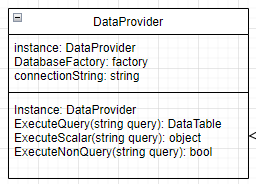
Dự án ban đầu chưa được xây dựng theo mô hình MVC và chưa có singleton, dẫn tới lãng phí tài nguyên hệ thống, ảnh hưởng hiệu suất hoạt động, cũng như có thể gây khó khăn trong việc kiểm soát truy cập đến các đối tượng, nguy cơ xảy ra lỗi và vấn đề bảo mật.

Vì vậy, sau khi tổ chức lại mô hình MVC thì chúng em đã áp dụng mẫu thiết kế singleton vào các hàm khởi tạo controller để đảm bảo tính duy nhất của đối tượng, quản lý tài nguyên hệ thống hiệu quả hơn cũng như dễ dàng kiểm soát và quản lý đối tượng trong hệ thống.

* + 1. Sơ đồ lớp

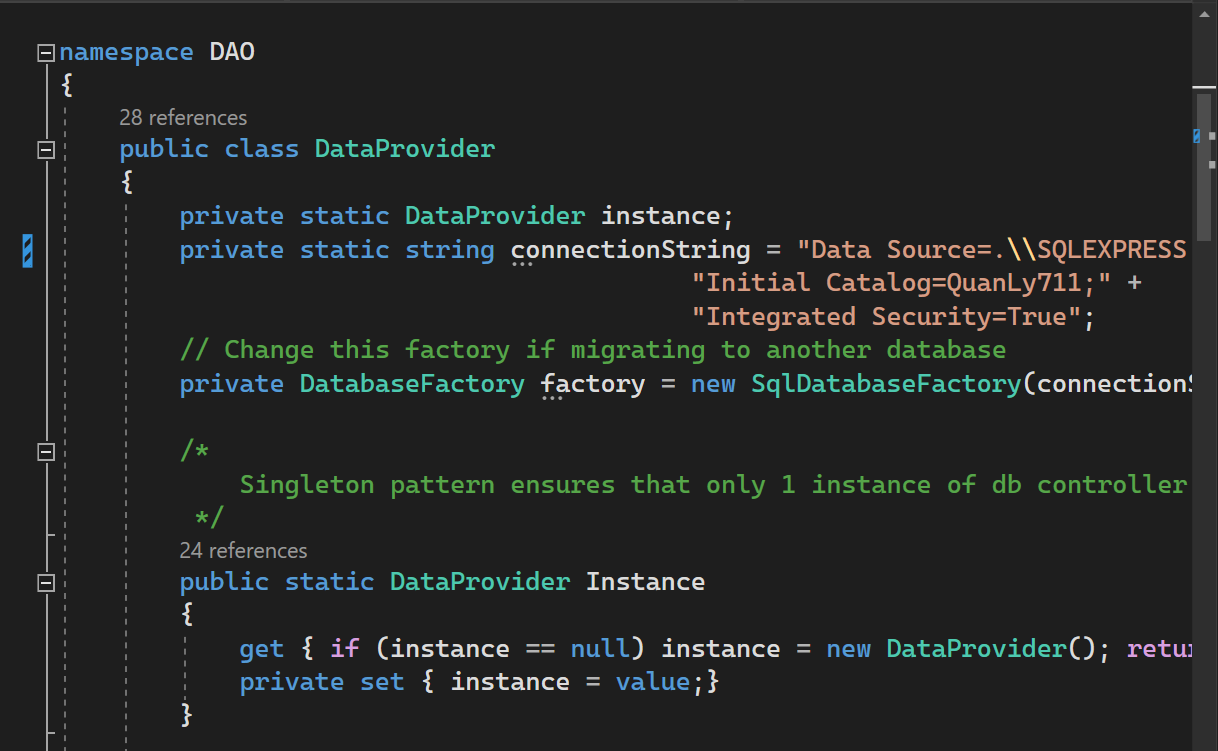


Hình .1 Một trong những sơ đồ lớp singleton pattern(1)



Hình .2 Một trong những sơ đồ lớp singleton pattern (2)

2.1.3 Code áp dụng mẫu thiết kế:



Hình .3 Code áp dụng singeton pattern

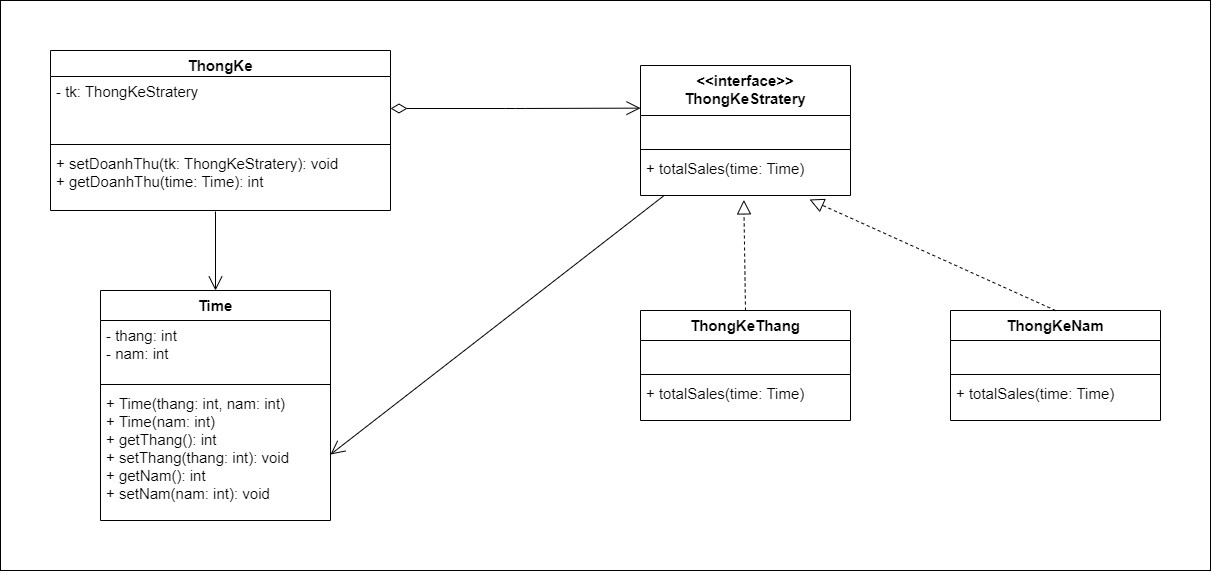
2.2 Strategy pattern

2.2.1 Lý do áp dụng

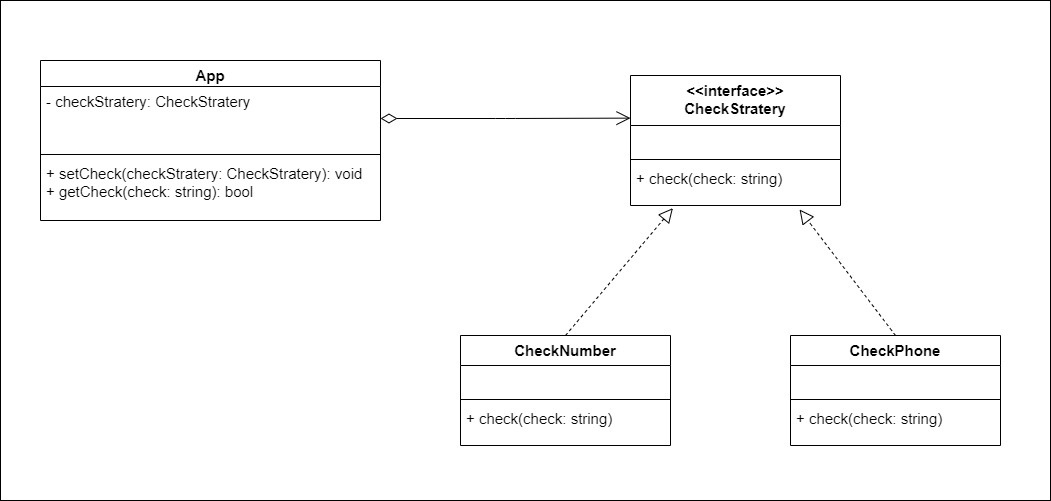
Trong dự án cũ có sử dụng các thuật toán để tính toán doanh thu và kiểm tra các số liệu đầu vào. Khi không có mẫu thiết kế strategy, việc bổ sung hoặc thay đổi các thuật toán sẽ khó khăn hơn vì phải sửa code ở mã nguồn thay vì chỉ cần tạo các lớp con, hoặc không thể tái sử dụng các thuật toán đó trong các tình huống khác nhau, làm cho việc phát triển hệ thống trở nên khó khăn và chậm chạp hơn.

Vì vậy, chúng em áp dụng mẫu thiết kế strategy vào chức năng tính doanh thu và kiểm tra số liệu đầu vào để giúp cho việc quản lý và thay đổi thuật toán trở nên dễ dàng, linh hoạt hơn, giảm sự phụ thuộc giữa các lớp trong hệ thống, cũng như khả năng tái sử dụng code và giải quyết các vấn đề liên quan đến thiết kế của hệ thống.

2.2.2 Sơ đồ lớp

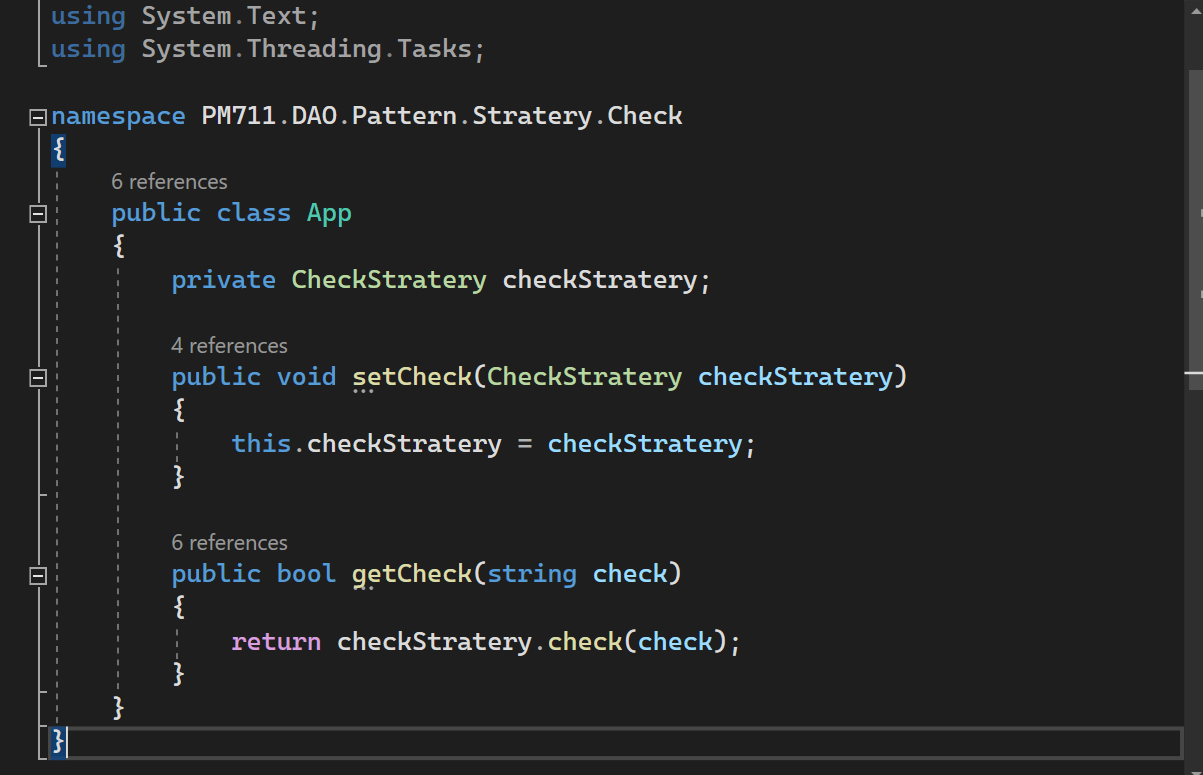


Hình 2.. Sơ đồ lớp strategy pattern doanh thu

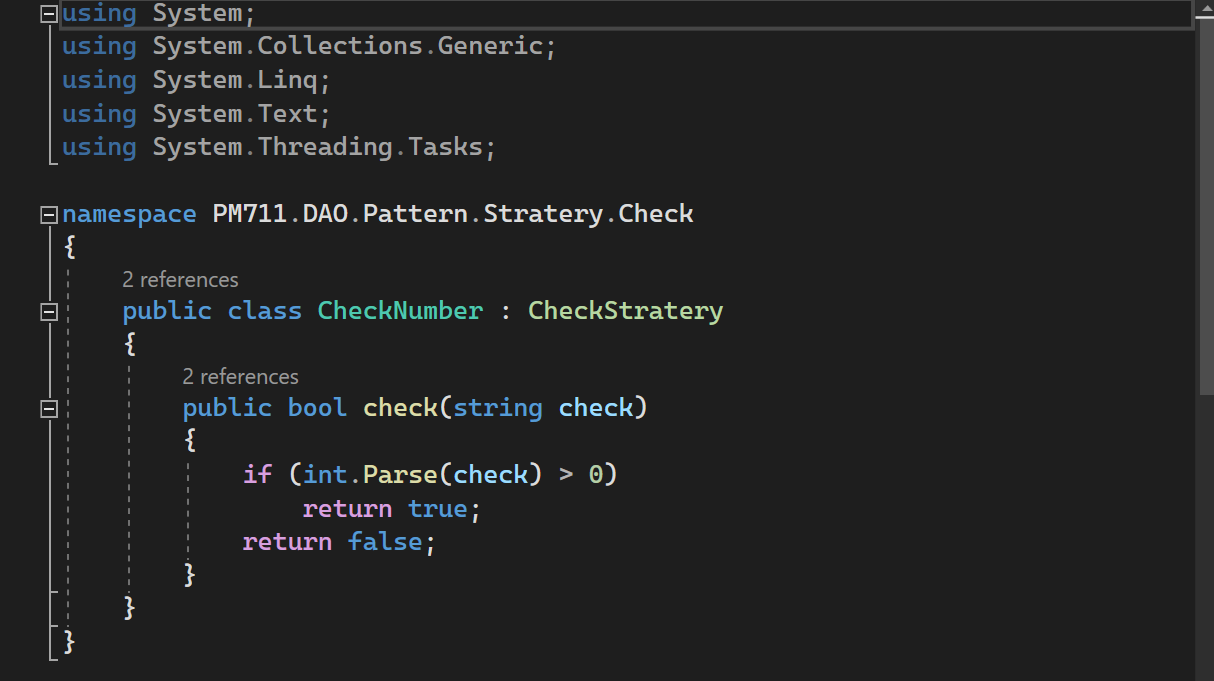


Hình 2. Sơ đồ lớp strategy pattern kiểm tra số liệu đầu vào

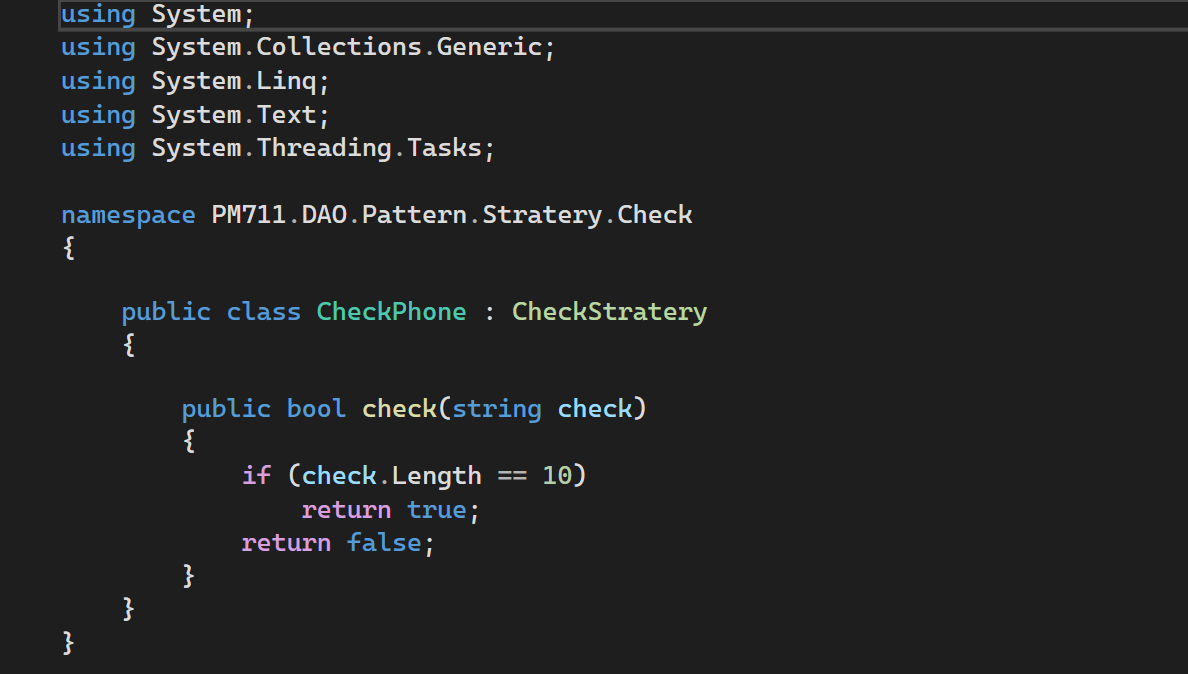
2.2.3 Code áp dụng mẫu thiết kế:



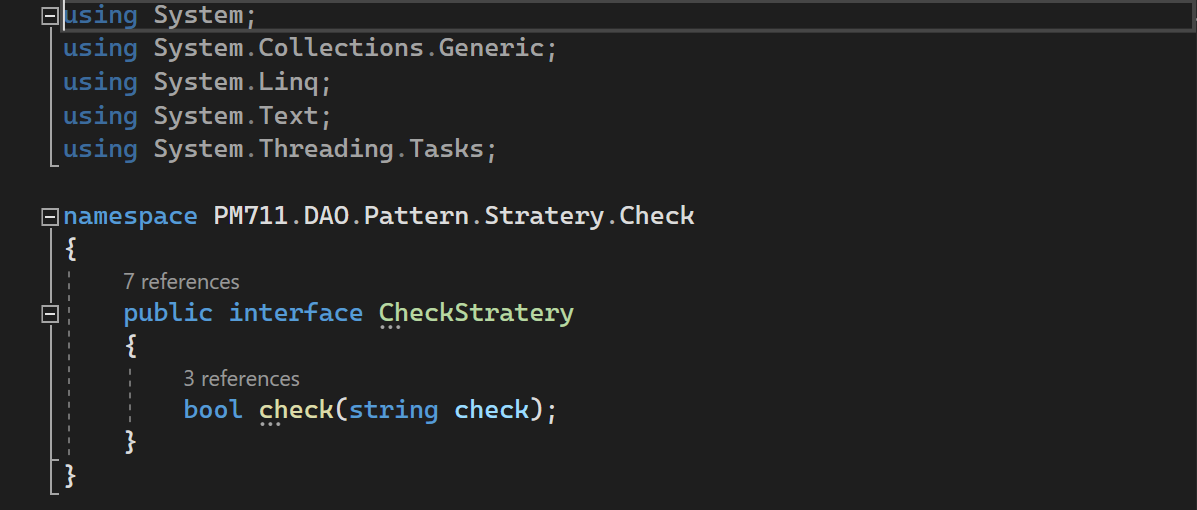
Hình 2. Code áp dụng Strategy pattern (1)



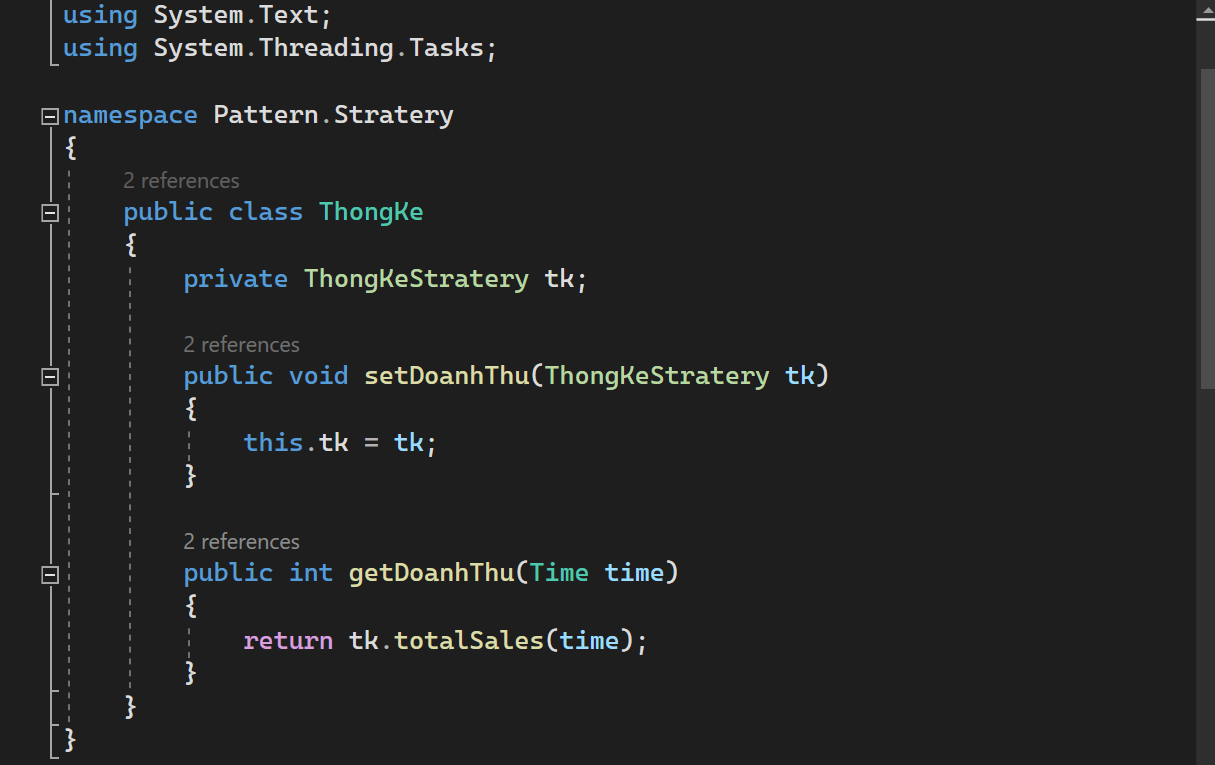
Hình 2. Code áp dụng Strategy pattern (2)



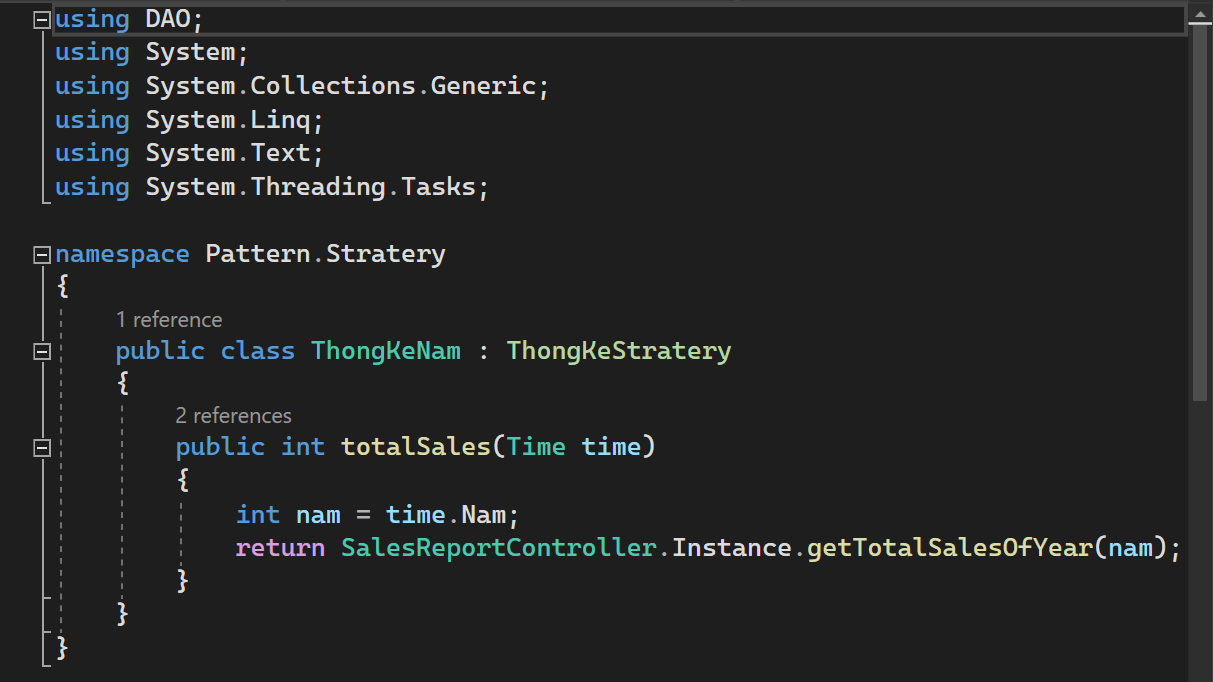
Hình 2. Code áp dụng Strategy pattern (3)



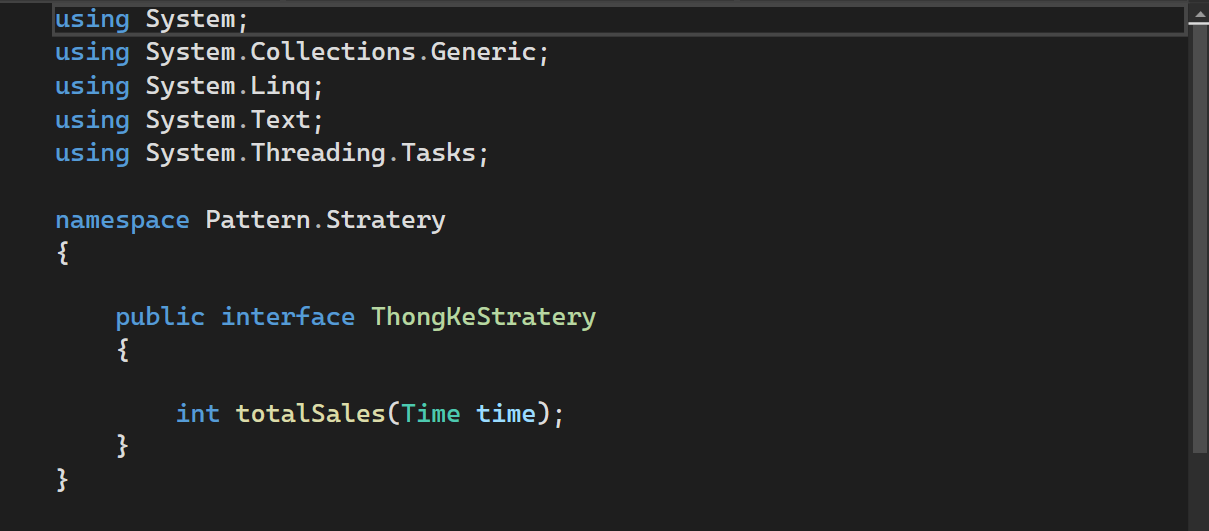
Hình 2. Code áp dụng Strategy pattern (4)



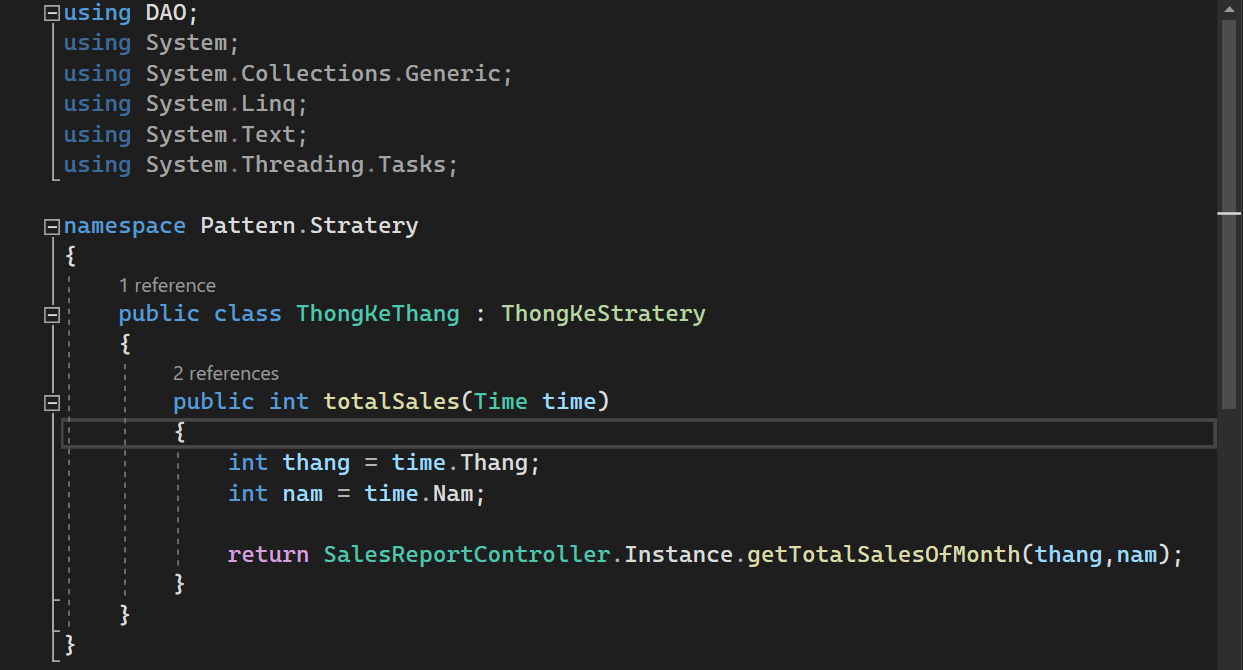
Hình 2. Code áp dụng Strategy pattern (5)



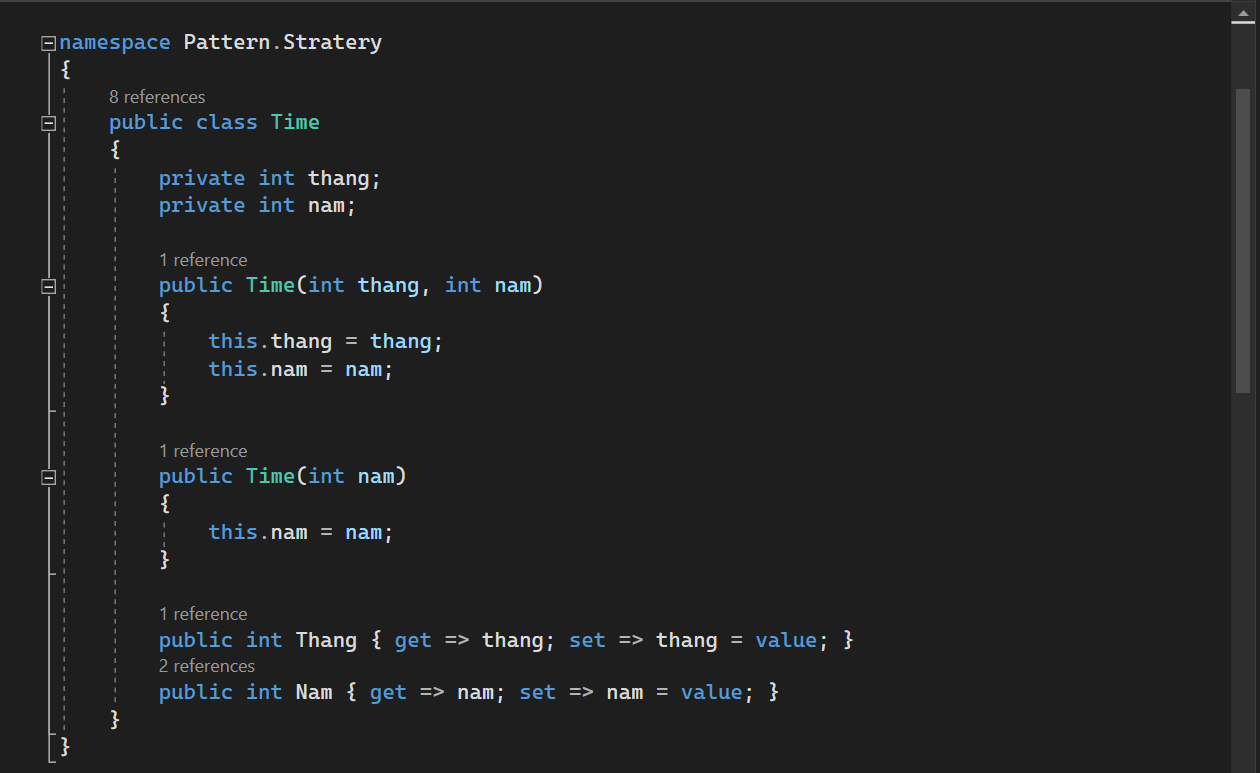
Hình 2. Code áp dụng Strategy pattern (6)



Hình 2. Code áp dụng Strategy pattern (7)



Hình 2. Code áp dụng Strategy pattern (8)



Hình 2. Code áp dụng Strategy pattern (9)

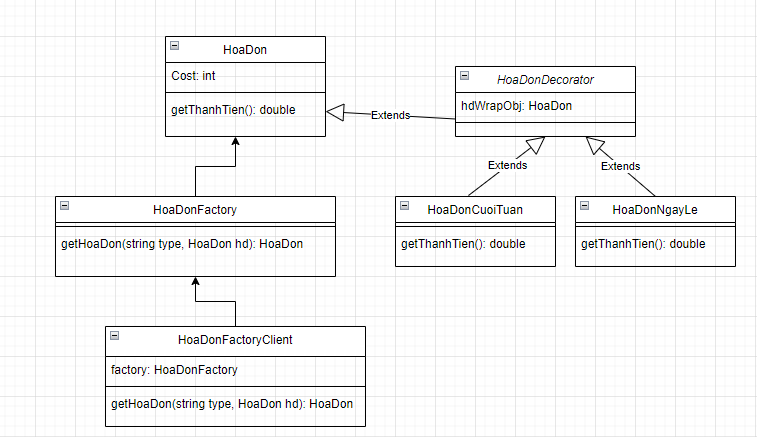
2.3 Simple Factory

2.3.1 Lý do áp dụng

Trong quá trình hoạt động, hệ thống cần tạo ra nhiều loại đối tượng khuyến mãi, trong dự án ban đầu không áp dụng Simple Factory, các đối tượng đó được tạo ra trực tiếp trong mã nguồn khiến cho mã nguồn trở nên phức tạp, khiến cho hệ thống trở nên khó bảo trì hoặc mở rộng.

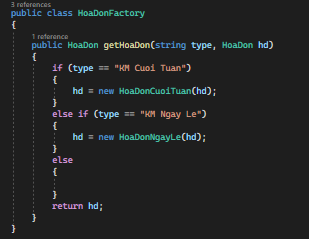
Vì vậy chúng em đã áp dụng Simple Factory vào dự án để có thể tạo ra các đối tượng khuyến mãi một cách linh hoạt hơn, cũng như dễ dàng kiểm thử và mở rộng hơn. Ngoài ra, việc áp dụng Simple Factory cũng hỗ trợ cho việc phát triển Decorator Pattern trong dự án.

2.3.2 Sơ đồ lớp

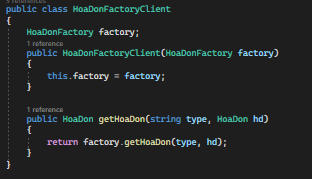


Hình 2. Sơ đồ lớp Simple factory

2.3.3 Code áp dụng mẫu thiết kế:



Hình 2. Code áp dụng Simple factory (1)



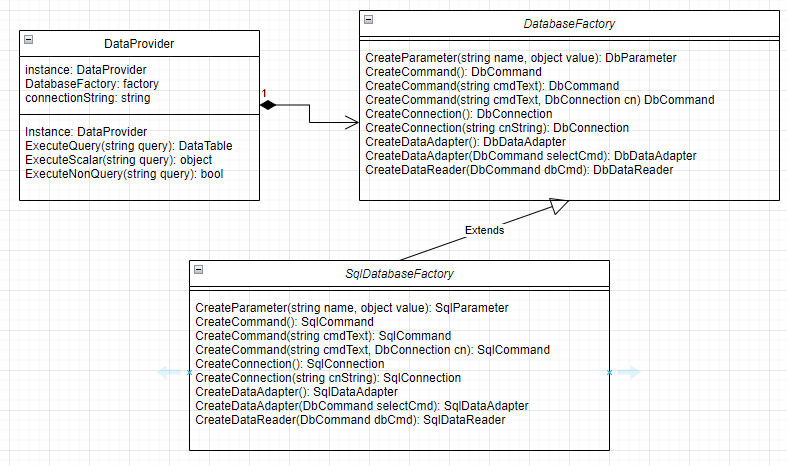
Hình 2. Code áp dụng Simple factory (2)

2.4 Abstract Factory

2.4.1 Lý do áp dụng

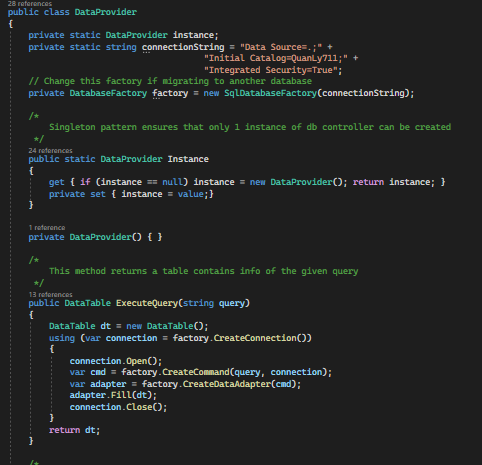
Trong dự án phần mềm, chúng em đã sử dụng Abstract Factory để giảm thiểu việc lặp lại các dòng code và tập trung tất cả các thao tác liên quan đến cơ sở dữ liệu vào một lớp. Đồng thời giúp việc chuyển đổi sang một hệ cơ sở dữ liệu khác dễ dàng hơn.

2.4.2 Sơ đồ lớp

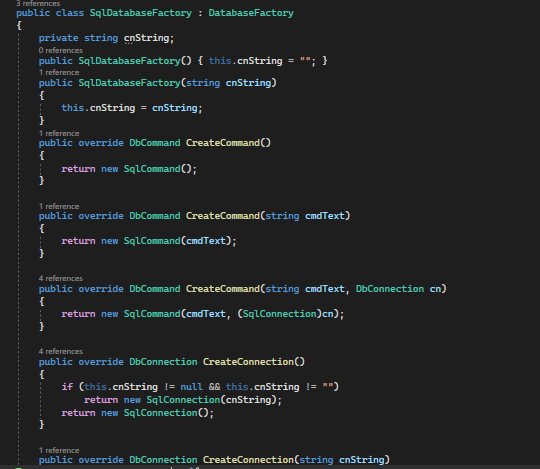


Hình 2. Sơ đồ lớp Abstract factory

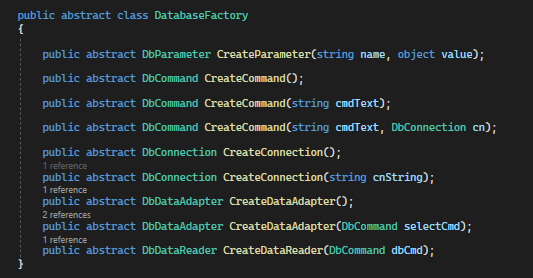
2.4.3 Code áp dụng mẫu thiết kế:



Hình 2. Code áp dụng Abstract factory (1)



Hình 2. Code áp dụng Abstract factory (2)



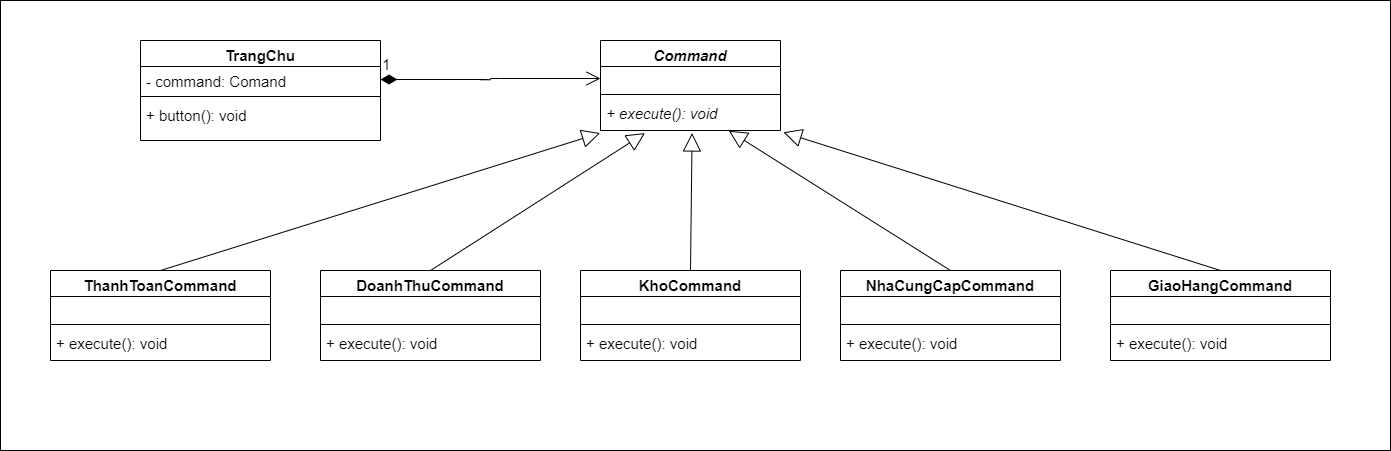
Hình 2. Code áp dụng Abstract factory (3)

2.5 Command Pattern

2.5.1 Lý do áp dụng

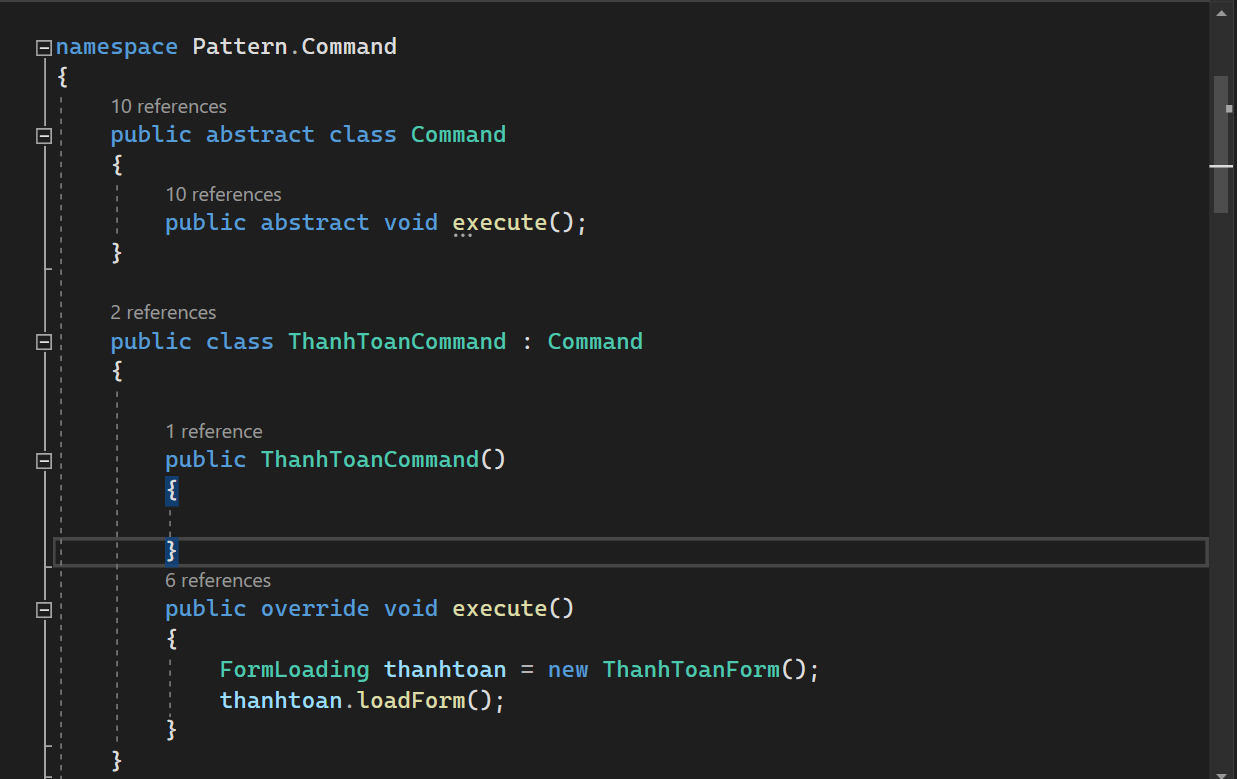
Trong quá trình hoạt động, phần mềm thường gặp khó khăn trong việc quản lý các yêu cầu và các hoạt động được thực hiện khi yêu cầu được gửi đến.Vì vậy chúng em sử dụng Command pattern để tăng tính linh hoạt của ứng dụng giúp tách biệt yêu cầu và hoạt động được thực hiện, giúp cho mã nguồn ứng dụng trở nên dễ dàng bảo trì hơn.

2.5.2 Sơ đồ lớp



Hình 2. Sơ đồ Command pattern

2.5.3 Code áp dụng mẫu thiết kế:



Hình 2. Code áp dụng Command pattern

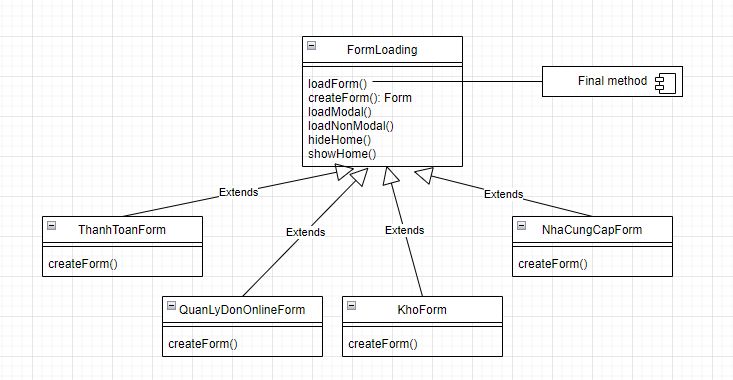
2.6 Template Method

2.6.1 Lý do áp dụng

Trong quá trình hoạt động của phần mềm, mỗi khi load một giao diện, hệ thống cần phải chạy một khố lượng code lặp lại, điều này ảnh hưởng đến hiệu suất của phần mềm và dễ gây ra lỗi.

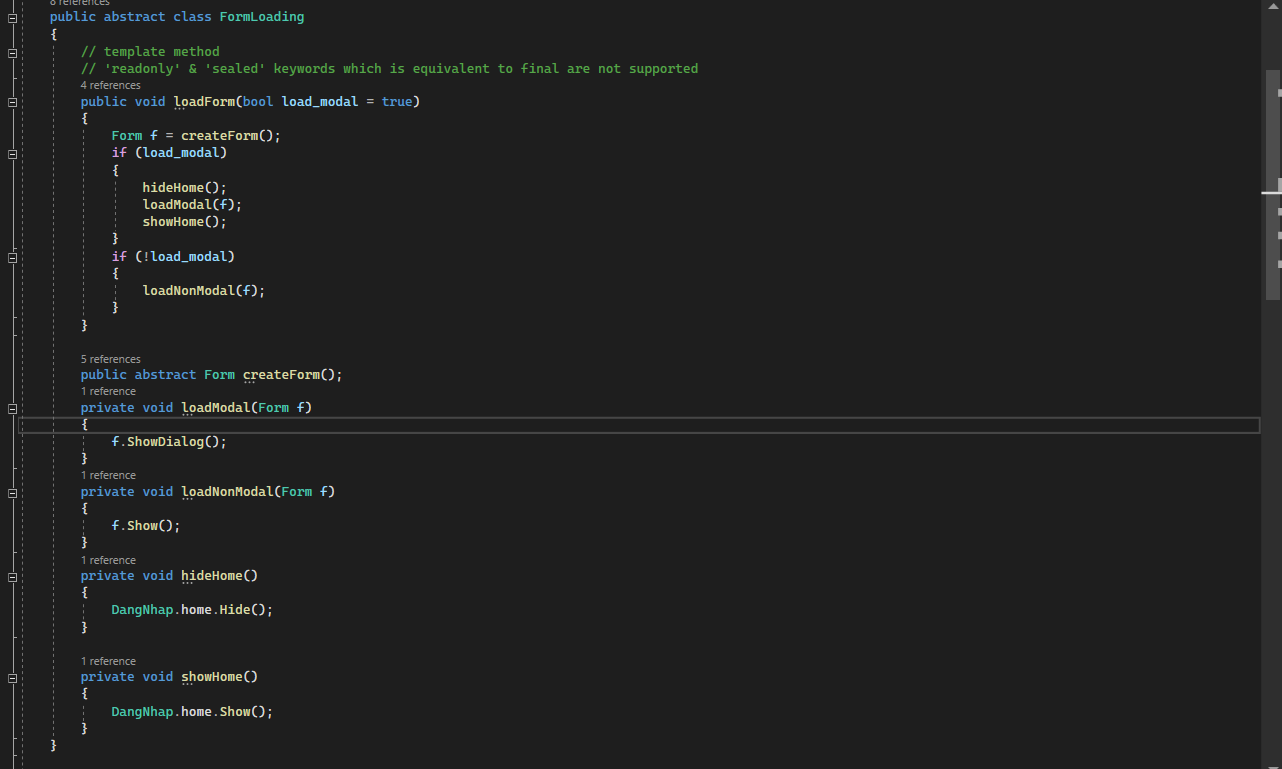
Vì vậy chúng em sử dụng Template Method để giúp giảm thiểu code lặp lại, tăng tính tái sử dụng, tránh các lỗi không đáng có và giúp dễ bảo trì mã nguồn hơn.

2.6.2 Sơ đồ lớp

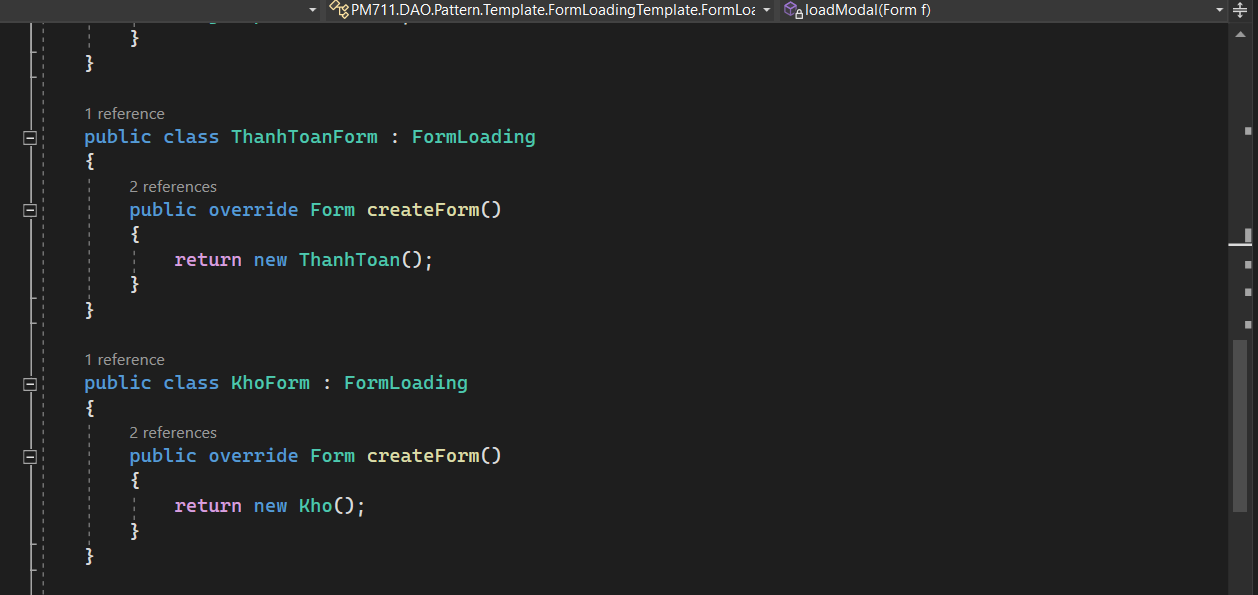


Hình 2. Sơ đồ lớp Template Method

2.6.3 Code áp dụng mẫu thiết kế:



Hình 2. Code áp dụng Template Method (1)



Hình 2. Code áp dụng Template method (2)

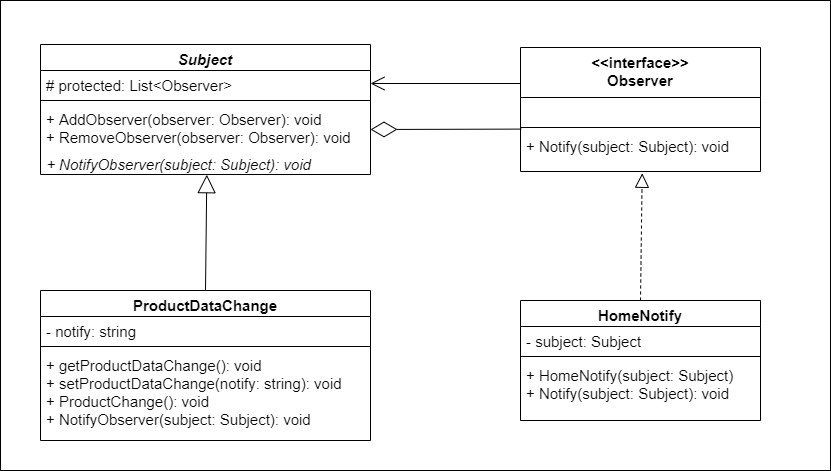
2.7 Observer Pattern

2.7.1 Lý do áp dụng

Trong dự án ban đầu, khi thêm sản phẩm mởi vào kho, thì nhân viên sử dụng hệ thống sẽ không được thông báo tự động mà phải tự truy cập vào kho để kiểm tra, dẫn tới các thiếu sót không đáng có.

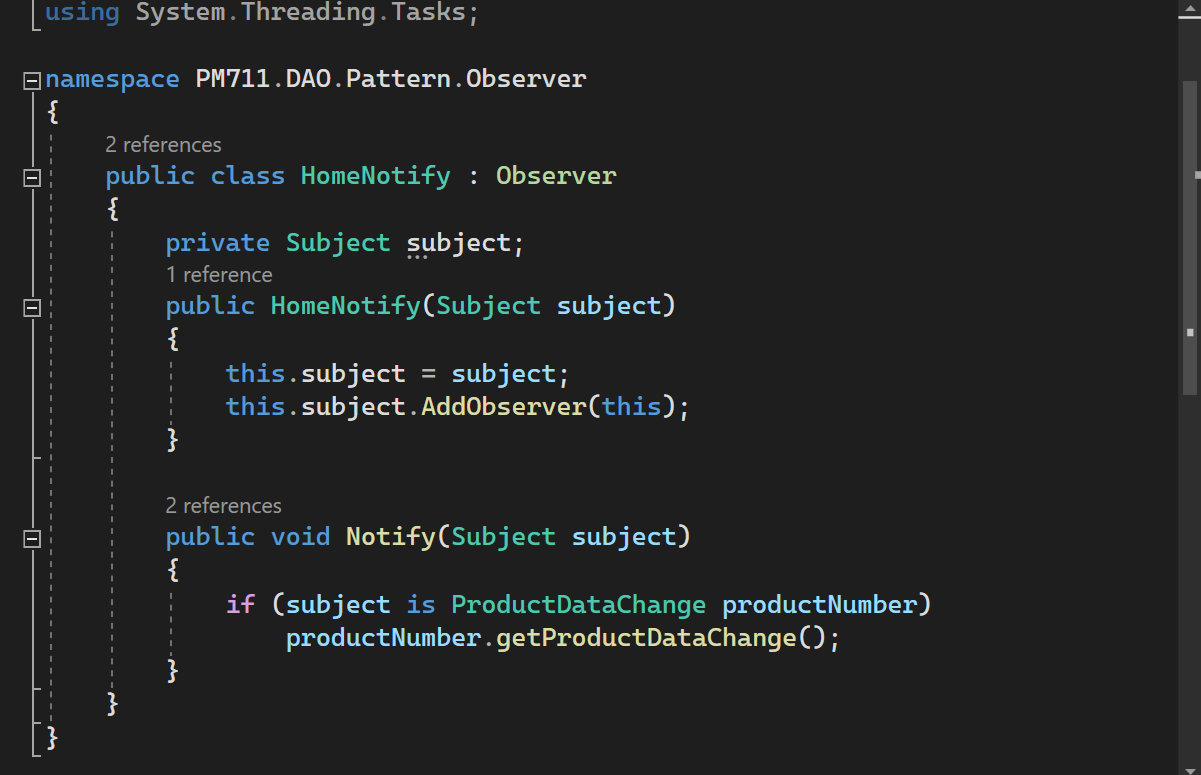
Vì vậy chúng em đã áp dụng observer pattern, để khi có sản phẩm mới được thêm vào kho, thì hệ thống sẽ tự động gửi thông báo đến trang chủ cho nhân viên, giúp đảm bảo quá trình làm việc của nhân viên được thuận tiện và hiệu quả, cũng như mở rộng sự tương tác của các đối tượng trong phần mềm.

2.7.2 Sơ đồ lớp

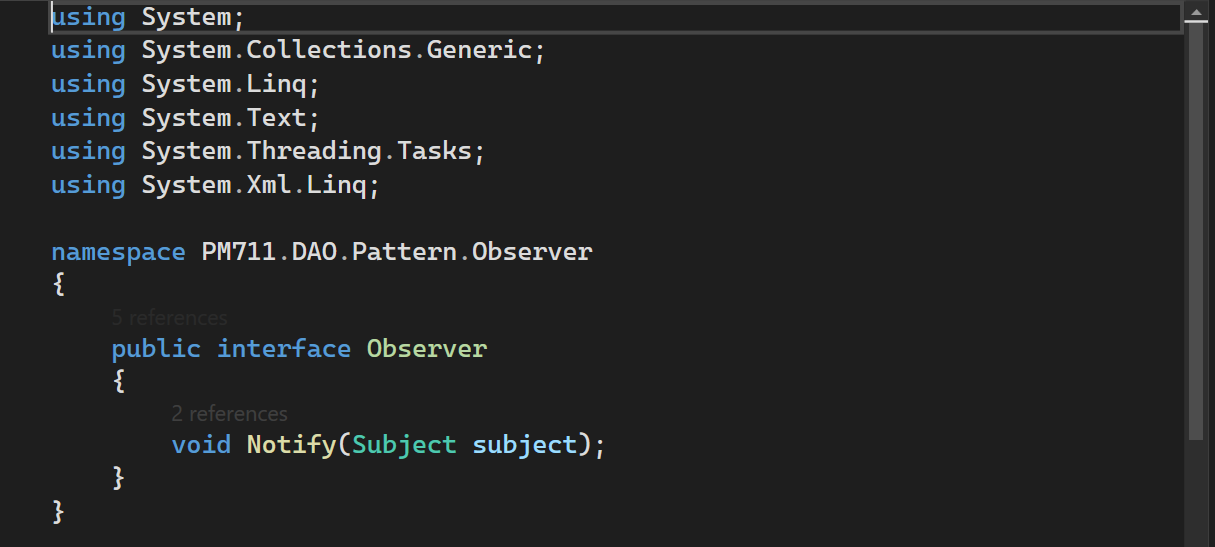


Hình 2. Sơ đồ Observer pattern

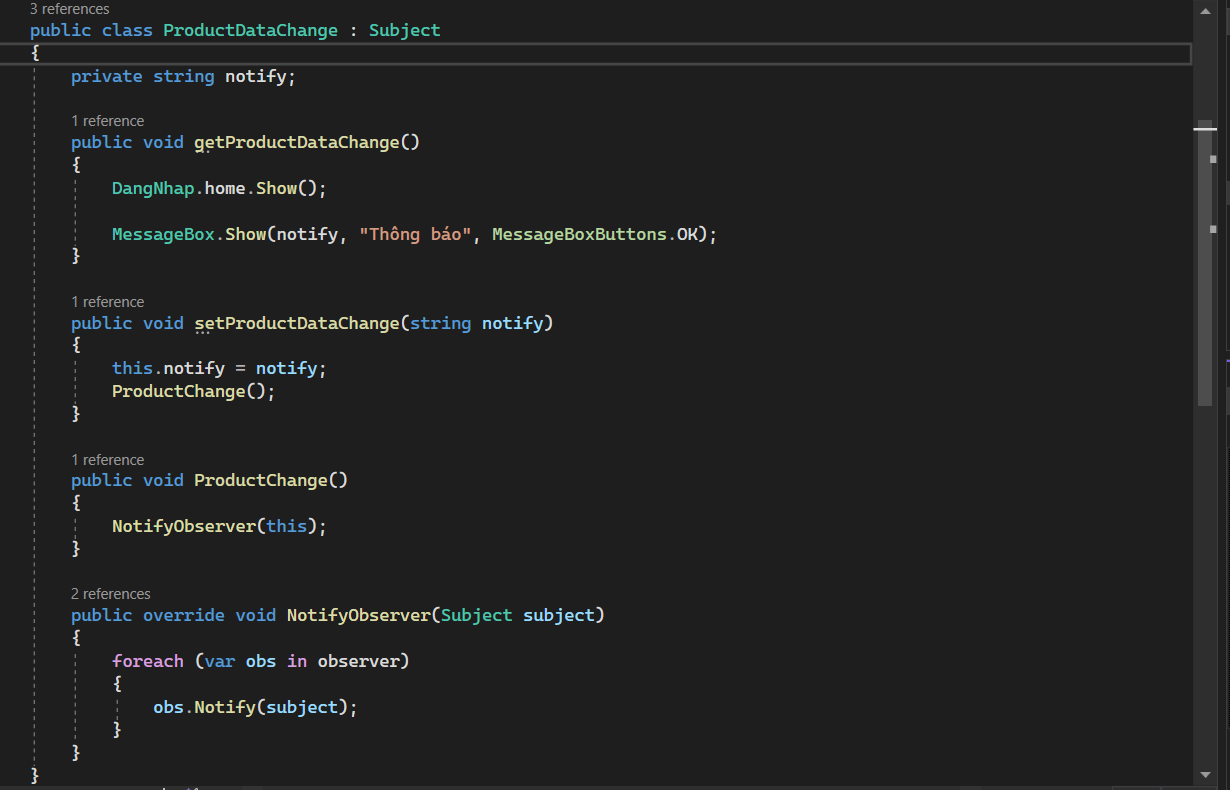
2.7.3 Code áp dụng mẫu thiết kế:



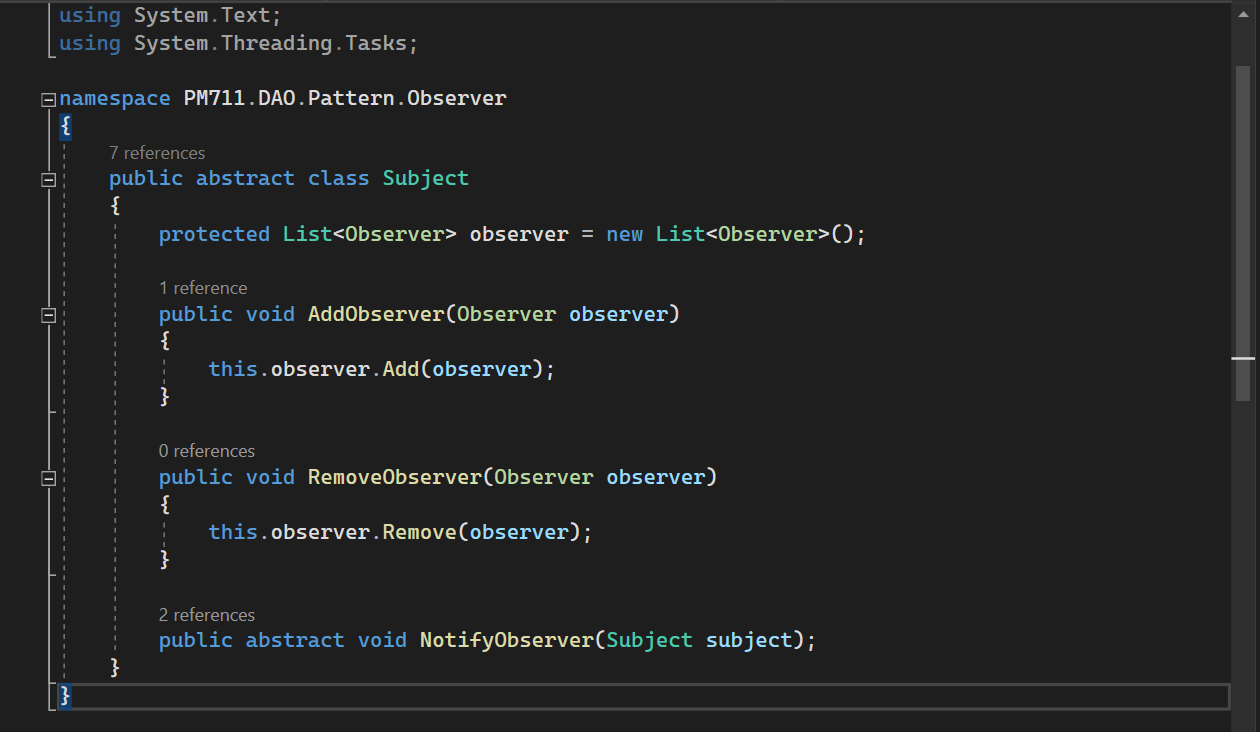
Hình 2. Code áp dụng Observer pattern (1)



Hình 2. Code áp dụng Observer pattern (2)



Hình 2. Code áp dụng Observer pattern (3)



Hình 2. Code áp dụng Observer pattern (4)

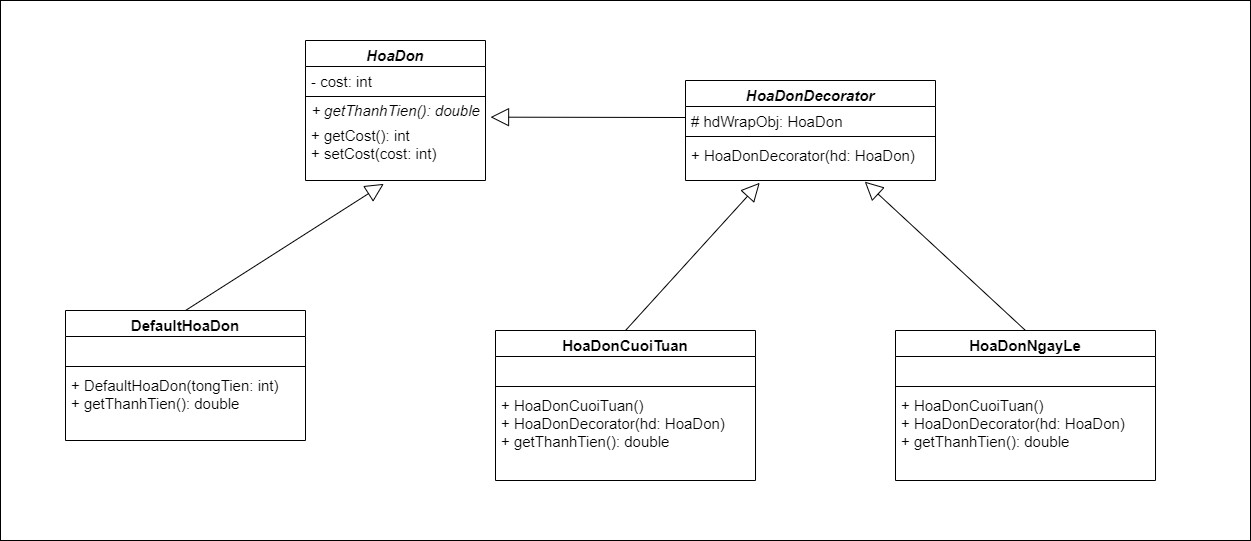
2.8 Decorator Pattern

2.8.1 Lý do áp dụng

Trong chức năng thanh toán của hệ thống, đôi khi chúng ta cần thêm các đối tượng khuyến mãi vào hóa đơn, nếu không sử dụng Decorator pattern, việc thêm các đối tượng đó vào hóa đơn sẽ phức tạp, tốn nhiều công sức hay khó mở rộng.

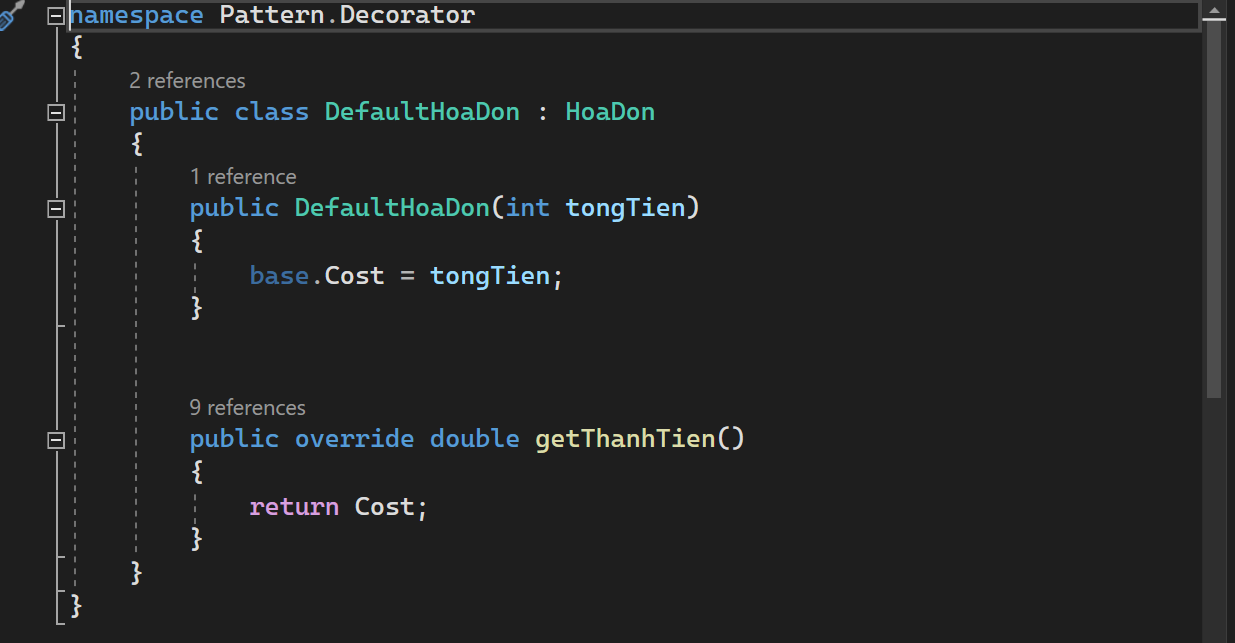
Vì vậy chúng em áp dụng mẫu thiết kế Decorator để giúp cho các đối tượng khuyến mãi có thể thêm vào hóa đơn linh hoạt hơn, đa dạng và dễ dàng mở rộng hơn.

2.8.2 Sơ đồ lớp

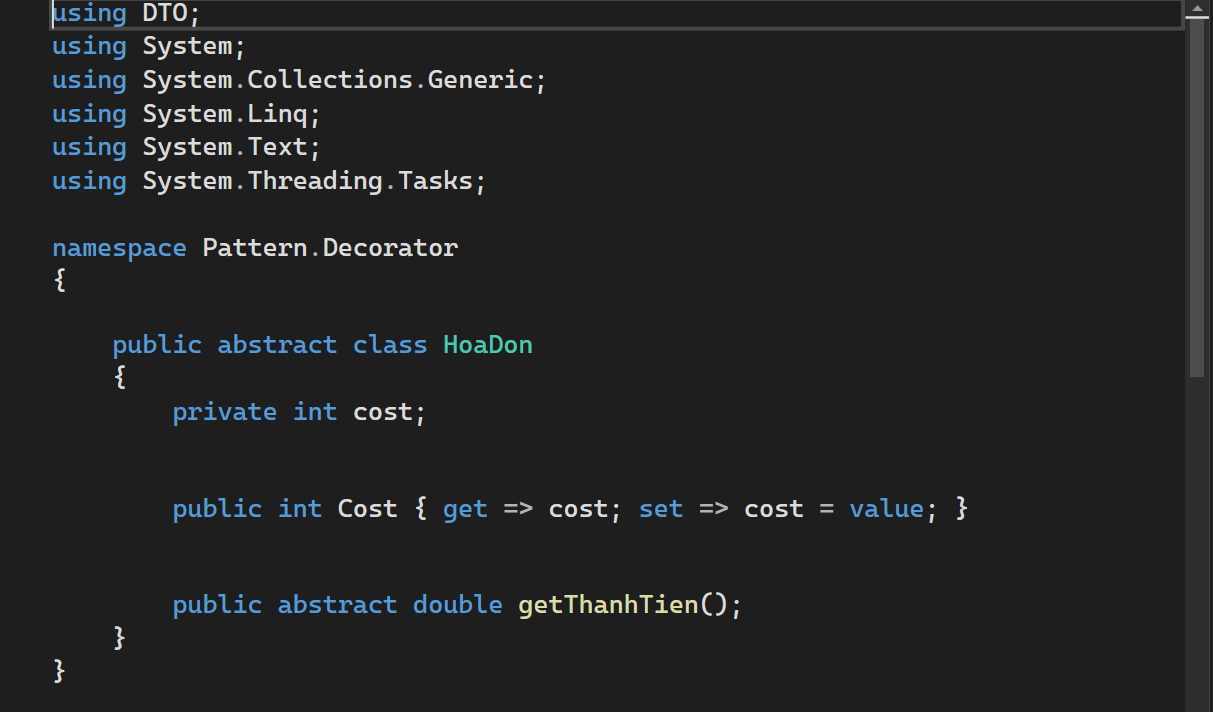


Hình 2. Sơ đồ lớp Decorator pattern

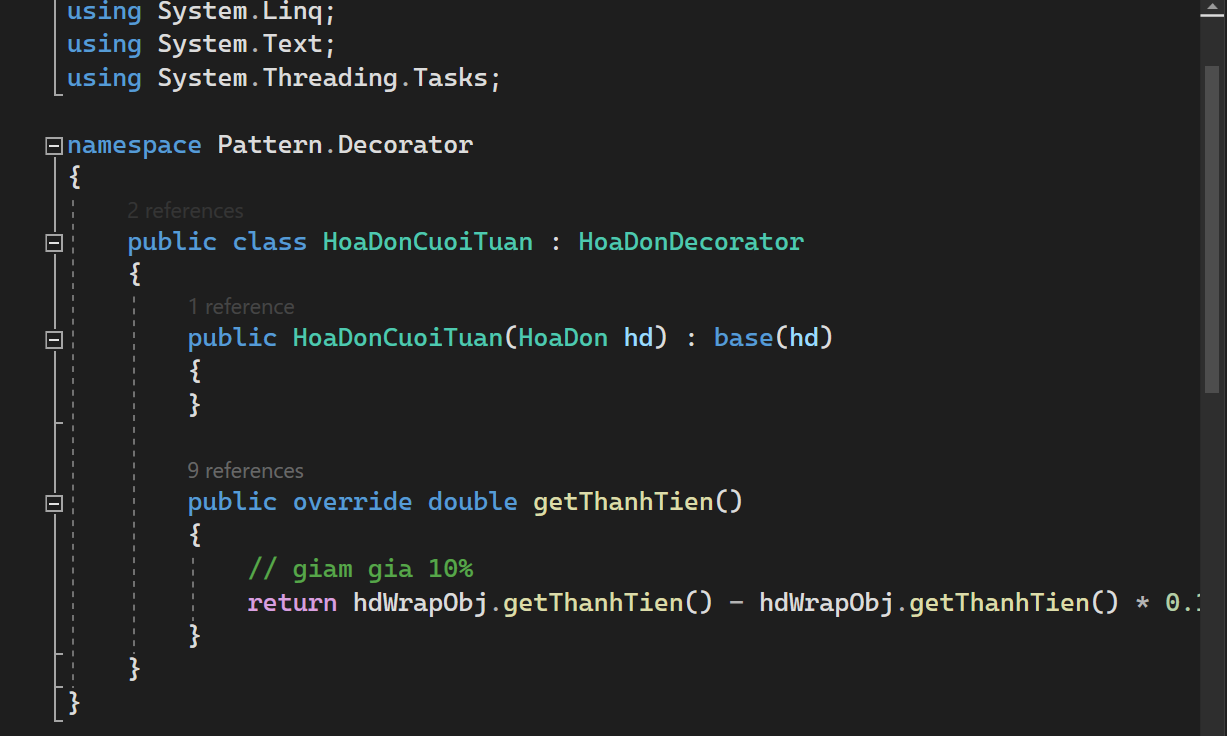
2.8.3 Code áp dụng mẫu thiết kế:



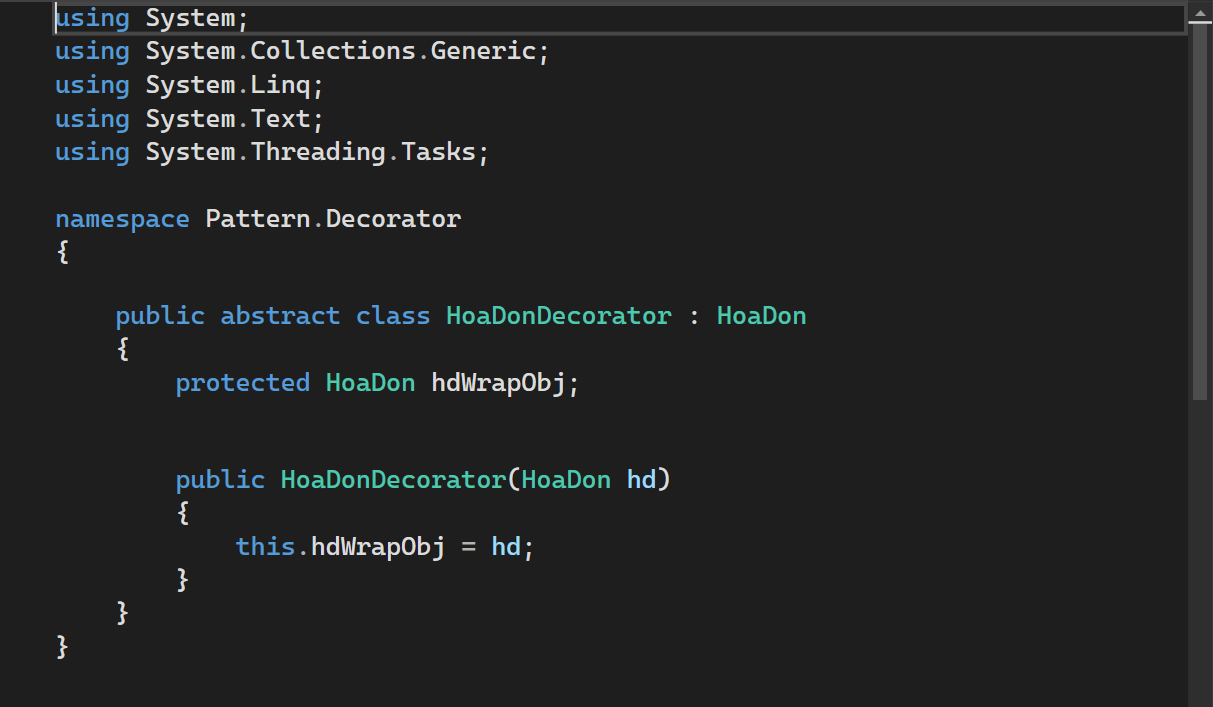
Hình 2. Code áp dụng Decorator pattern (1)



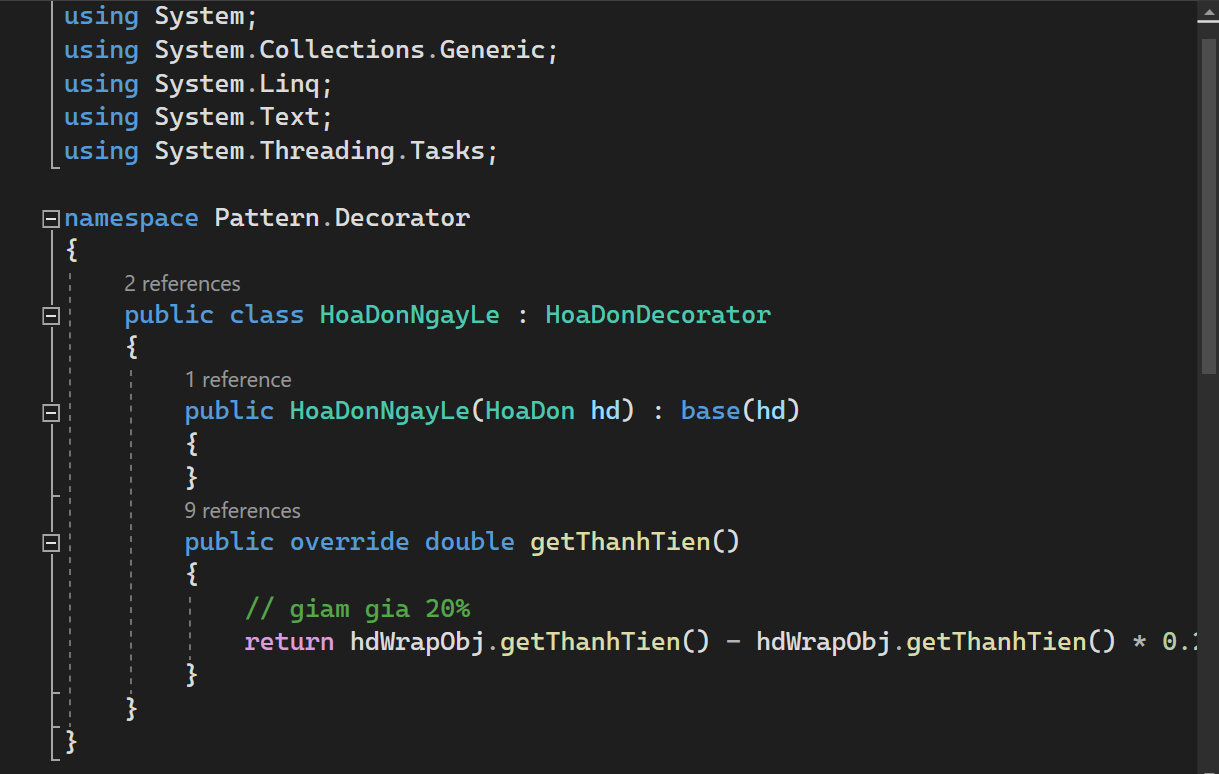
Hình 2. Code áp dụng Decorator pattern (2)



Hình 2. Code áp dụng Decorator pattern (3)



Hình 2. Code áp dụng Decorator pattern (4)



Hình 2. Code áp dụng Decorator pattern (5)

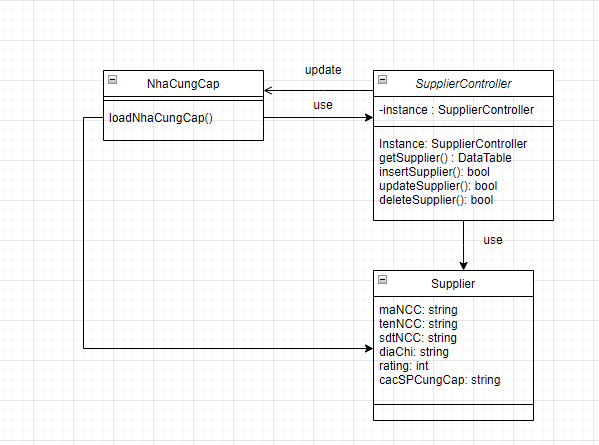
2.9 MVC Pattern

2.9.1 Lý do áp dụng

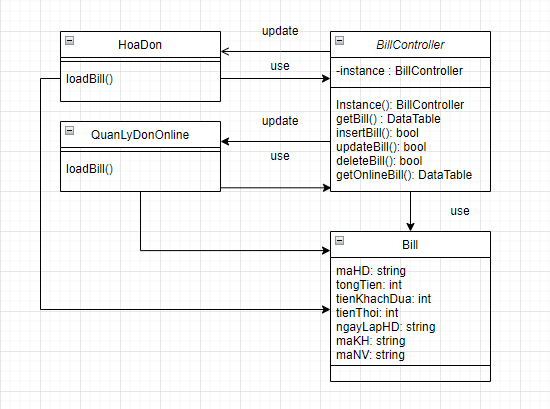
Dự án ban đầu không sử dụng mô hình MVC, khiến cho các thành phần của phần mềm không được phân tách rõ ràng, mã nguồn phức tạp, khó bảo trì và mở rộng phần mềm, làm giảm hiệu suất hoạt động của ứng dụng và dễ dẫn đến các lỗi không mong muốn.

Vì vậy chúng em đã áp dụng mô hình MVC vào dự án để giúp phân tách rõ ràng các thành phần của phần mềm, đơn giản hóa mã nguồn, tăng hiệu suất hoạt động phần mềm, giảm các lỗi ngoài ý muốn và làm cho phần mềm trở nên dễ bảo trì và mở rộng

2.9.2 Sơ đồ lớp

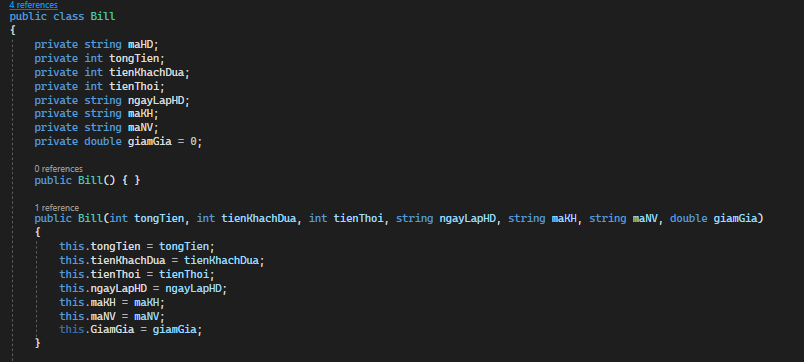


Hình 2. Một trong những sơ đồ lớp MVC (1)

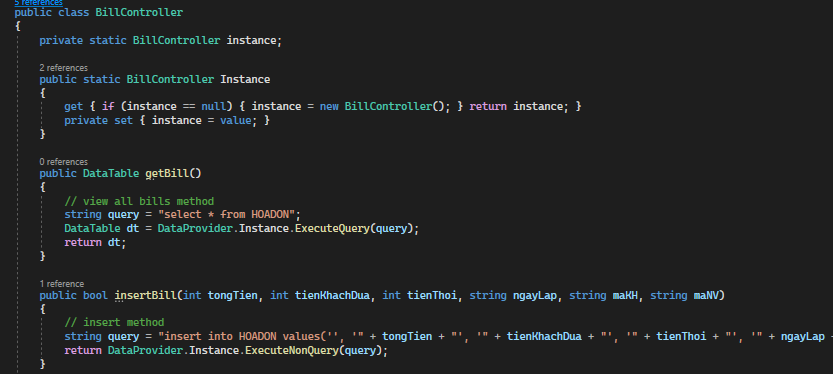


Hình 2. Một trong những sơ đồ lớp MVC (2)

2.9.3 Code áp dụng mẫu thiết kế:



Hình 2. Code áp dụng MVC (1)



Hình 2. Code áp dụng MVC (2)

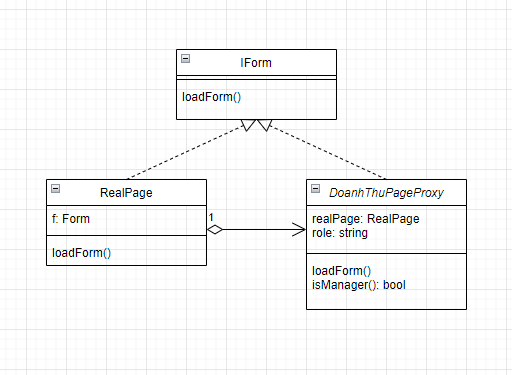
2.10 Proxy pattern

2.10.1 Lý do áp dụng

Trong phần mềm ban đầu, việc phân quyền giữa nhân viên và quản lý chưa được thực hiện rõ ràng, nên đôi khi xảy ra các lỗi về bảo mật.

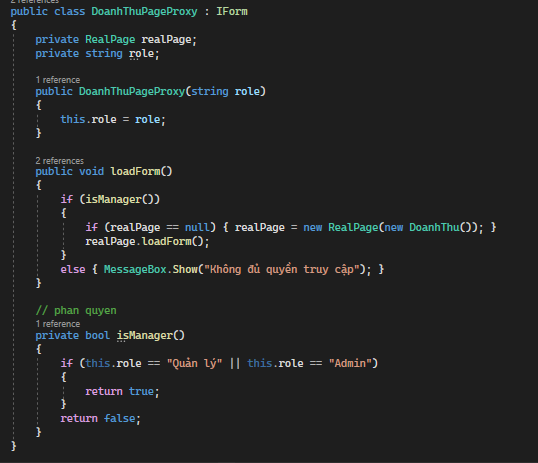
Vì vậy chúng em sử dụng Proxy để hiện thực chức năng phân quyền cho hệ thống, chỉ cho phép quản lý được phép xem doanh thu của cửa hàng. Giúp cho phần mềm tăng tính bảo mật đồng thời sẵn sàng cho các chức năng cần tới phân quyền sau này.

2.10.2 Sơ đồ lớp

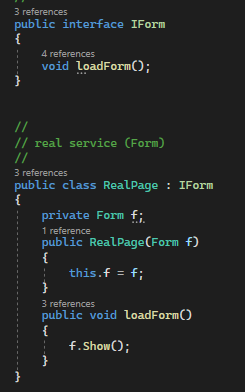


Hình 2. Sơ đồ lớp Proxy pattern

2.10.3 Code áp dụng mẫu thiết kế:



Hình 2. Code áp dụng Proxy pattern (1)



Hình 2. Code áp dụng Proxy pattern (2)

TÀI LIỆU THAM KHẢO

* Strategy Design Pattern: Áp dụng vào code (C#)

<https://www.youtube.com/watch?v=rXS6q0pG9Ow&list=PLoaAbmGPgTSOrVuxwbnDJ14U9J6CXJXUk&index=4>

* Observer Design Pattern: Áp dụng vào Code (C#)

<https://www.youtube.com/watch?v=zc2smrSk5I0&list=PLoaAbmGPgTSOrVuxwbnDJ14U9J6CXJXUk&index=7>

* Decorator Design Pattern: Áp dụng vào code, thử vài trường hợp (C#)

<https://www.youtube.com/watch?v=U2XUrAefHME&list=PLoaAbmGPgTSOrVuxwbnDJ14U9J6CXJXUk&index=13>

* Proxy Design Pattern - Trợ thủ đắc lực của Developers

<https://viblo.asia/p/proxy-design-pattern-tro-thu-dac-luc-cua-developers-RQqKLB2bl7z>

* Design Patterns - Singleton pattern

<https://viblo.asia/p/design-patterns-singleton-pattern-maGK7zra5j2>