BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC GIA ĐỊNH**

**KHOA: CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**TIỂU LUẬN**

**XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ HÓA ĐƠN TIỀN ĐIỆN CỦA KHÁCH HÀNG**

**MÔN: MẪU THIẾT KẾ CHO PHẦN MỀM**

|  |  |  |
| --- | --- | --- |
| Ngành | : | **KỸ THUẬT PHẦN MỀM** |
| Giảng viên hướng dẫn | : | **ThS. LÊ HUỲNH PHƯỚC** |
| Sinh viên thực hiện | : | **NGUYỄN NGỌC TRIỀU VỸ** |
| MSSV | : | **22140073** |
| Lớp | : | **221402** |

TP. Hồ Chí Minh, 29 tháng 7 năm 2024

**Khoa/Viện: Công Nghệ Thông Tin**

**NHẬN XÉT VÀ CHẤM ĐIỂM CỦA GIẢNG VIÊN**

**TIỂU LUẬN MÔN: MẪU THIẾT KẾ CHO PHẦN MỀM**

1. **Họ và tên sinh viên: Nguyễn Ngọc Triều Vỹ**
2. **Tên đề tài**: **XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ HÓA ĐƠN TIỀN ĐIỆN CỦA KHÁCH HÀNG**
3. **Nhận xét**:

***Những kết quả đạt được:***

***Những hạn chế:***

1. **Điểm đánh giá** *(theo thang điểm 10, làm tròn đến 0.5):*

Sinh viên:…………………………………………………………………….……....

Điểm số: ……….…… Điểm chữ: …………………………………………………..

|  |  |
| --- | --- |
|  | *TP. HCM, ngày … tháng … năm 20……*  **Giảng viên chấm thi**  *(Ký và ghi rõ họ tên)* |

**LỜI MỞ ĐẦU**

Trong bối cảnh công nghệ thông tin ngày càng phát triển mạnh mẽ, nhu cầu về việc quản lý tài chính và các dịch vụ liên quan đang trở thành một phần thiết yếu trong cuộc sống hàng ngày của người dân. Một trong những lĩnh vực quan trọng trong quản lý tài chính là hóa đơn tiền điện, đặc biệt là trong các đô thị lớn và khu vực có mức tiêu thụ điện năng cao, việc quản lý hóa đơn tiền điện không chỉ liên quan đến việc kiểm tra và thanh toán hóa đơn.

Với nhu cầu ngày càng cao về việc quản lý tài chính hiệu quả và chính xác, xây dựng một chương trình quản lý hóa đơn tiền điện cho khách hàng trở nên cấp thiết hơn bao giờ hết. Với giao diện đơn giản, trực quan và dễ sử dụng, chương trình sẽ cung cấp cho người dùng các tính năng thiết yếu như quản lý các hóa đơn thanh toán và thanh toán các hóa đơn tiền điện, tính toán chính xác số tiền phải trả, cũng như tạo ra các báo cáo thống kê. Chính vì những yếu tố trên nên nhóm chúng em quyết định chọn đề tài “Xây dựng chương trình quản lý hóa đơn tiền điện của khách hàng” để phát triển.

**LỜI CẢM ƠN**

Em xin chân thành gửi lời cảm ơn tới quý thầy cô trường Đại học Gia Định nói chung và các thầy cô bộ môn trong khoa Công nghệ thông tin nói riêng đã tạo điều kiện cho chúng em cơ hội thực hành và tiếp xúc thực tế, giúp chúng em tránh được những vướng mắc và bỡ ngỡ khi bước vào môi trường công việc trong tương lai.

Đặc biệt, chúng em xin gửi lời cảm ơn chân thành tới Thầy Lê Huỳnh Phước. Trong quá trình học tập và tìm hiểu về bộ môn Mẫu thiết kế cho phần mềm, đặc biệt là trong việc xây dựng chương trình quản lý hóa đơn tiền điện của khách hàng, chúng em đã nhận được sự quan tâm và hỗ trợ nhiệt tình từ Thầy. Thầy đã hướng dẫn và giúp chúng em tích lũy thêm nhiều kiến thức thực tiễn, làm hành trang quý báu cho sự nghiệp sau này.

Nhờ sự giúp đỡ tận tình và những chỉ bảo của Thầy từ những bước đầu cho đến khi hoàn thành tiểu luận, chúng em đã có thể hoàn thành đúng thời hạn quy định và tích lũy được một lượng nền tảng kiến thức quý giá.

Mặc dù vậy, trong quá trình hoàn thiện bài tiểu luận, không thể tránh khỏi những khó khăn và thiếu sót. Chúng em rất mong nhận được những góp ý và phản hồi từ Thầy để bài tiểu luận được hoàn thiện hơn, và từ đó, chúng em có thể rút kinh nghiệm cho các bài tiểu luận hoặc khóa luận trong tương lai.

Chúng em xin chân thành cảm ơn !

TP. Hồ Chí Minh, ngày 29 tháng 7 năm 2024

Sinh viên thực hiện

**Nguyễn Ngọc Triều Vỹ**

**DANH MỤC HÌNH ẢNH**

**Hình ảnh chương 1**

[Hình 1. 1 Three-Layer Pattern 3](#_Toc172972473)

[Hình 1. 2 Domain Layer 5](#_Toc172972474)

[Hình 1. 3 MVC Pattern 6](#_Toc172972475)

[Hình 1. 4 Publisher-Subcriber (Observer) Pattern 7](#_Toc172972476)

[Hình 1. 5 CommandProcessor Pattern 8](#_Toc172972477)

[Hình 1. 6 DAO 11](#_Toc172972478)

**Hình ảnh chương 2**

[Hình 2. 1 Sơ đồ hệ thống 13](#_Toc172972500)

[Hình 2. 2 Sơ đồ phân cấp chức năng 14](#_Toc172972501)

[Hình 2. 3 Sơ đồ USECASE Thêm hóa đơn 14](#_Toc172972502)

[Hình 2. 4 Sơ đồ USECASE Sửa hóa đơn 15](#_Toc172972503)

[Hình 2. 5 Sơ đồ USECASE Xóa hóa đơn 15](#_Toc172972504)

[Hình 2. 6 Sơ đồ USECASE Tìm kiếm hóa đơn 16](#_Toc172972505)

[Hình 2. 7 Sơ đồ USECASE Tính tiền hóa đơn 16](#_Toc172972506)

[Hình 2. 8 Sơ đồ tuần tự Thêm hóa đơn 17](#_Toc172972507)

[Hình 2. 9 Sơ đồ tuần tự Sửa hóa đơn 17](#_Toc172972508)

[Hình 2. 10 Sơ đồ tuần tự Xóa hóa đơn 18](#_Toc172972509)

[Hình 2. 11 Sơ đồ tuần tự Tìm kiếm hóa đơn 18](#_Toc172972510)

[Hình 2. 12 Tầng Presentation 19](#_Toc172972511)

[Hình 2. 13 Tầng Domain 20](#_Toc172972512)

[Hình 2. 14 Tầng Persistence 20](#_Toc172972513)

**Hình ảnh chương 3**

[Hình 3. 1 Giao diện phần mềm 21](#_Toc172972619)

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN VỀ MẪU THIẾT KẾ CHO PHẦN MỀM 1](#_Toc172973937)

[1.1. Khái niệm Design Pattern, tác dụng, lý do, phân loại Design Pattern 1](#_Toc172973938)

[1.1.1. Khái niệm 1](#_Toc172973939)

[1.1.2. Tác dụng của Design Patterns 1](#_Toc172973940)

[1.1.3. Lý do áp dụng Design Patterns 1](#_Toc172973941)

[1.1.4. Phân loại Design Patterns 2](#_Toc172973942)

[1.2. Mô hình 3 lớp (Three-layer) 3](#_Toc172973943)

[1.2.1. Khái niệm 3](#_Toc172973944)

[1.2.2. The Domain Layer 4](#_Toc172973945)

[1.2.3. The Presentation Layer 6](#_Toc172973946)

[1.2.4. The Persistence Layer 10](#_Toc172973947)

[CHƯƠNG 2: PHÂN TÍCH, THIẾT KẾ CHO PHẦN MỀM 12](#_Toc172973948)

[2.1. Mô hình nghiệp vụ 12](#_Toc172973949)

[2.1.1. Sơ đồ hệ thống 13](#_Toc172973950)

[2.1.2. Sơ đồ phân cấp chức năng 14](#_Toc172973951)

[2.1.3. Sơ đồ USECASE Thêm hóa đơn 14](#_Toc172973952)

[2.1.4. Sơ đồ USECASE Sửa hóa đơn 15](#_Toc172973953)

[2.1.5. Sơ đồ USECASE Xóa hóa đơn 15](#_Toc172973954)

[2.1.6. Sơ đồ USECASE Tìm kiếm hóa đơn 16](#_Toc172973955)

[2.1.7. Sơ đồ USECASE Tính tiền hóa đơn 16](#_Toc172973956)

[2.2. Sơ đồ tuần tự (Sequency Diagram) 17](#_Toc172973957)

[2.2.1. Sơ đồ tuần tự Thêm hóa đơn 17](#_Toc172973958)

[2.2.2. Sơ đồ tuần tự Sửa hóa đơn 17](#_Toc172973959)

[2.2.3. Sơ đồ tuần tự Xóa hóa đơn 18](#_Toc172973960)

[2.2.4. Sơ đồ tuần tự Tìm kiếm hóa đơn 18](#_Toc172973961)

[2.3. Sơ đồ thiết kế 19](#_Toc172973962)

[CHƯƠNG 3: CÀI ĐẶT THỰC NGHIỆM PHẦN MỀM 21](#_Toc172973963)

[3.1. Giao diện phần mềm 21](#_Toc172973964)

[3.2. Thực nghiệm các chức năng 21](#_Toc172973965)

[3.2.1. Chức năng thêm hóa đơn 21](#_Toc172973966)

[3.2.2. Chức năng sửa hóa đơn 22](#_Toc172973967)

[3.2.3. Chức năng xóa hóa đơn 23](#_Toc172973968)

[3.2.4. Chức năng tìm kiếm 23](#_Toc172973969)

[3.2.5. Chức năng tính tiền 24](#_Toc172973970)

[3.2.6. Chức năng tính tổng số lượng cho từng loại khách hàng 25](#_Toc172973971)

[3.2.7. Chức năng tính trung bình thành tiền cho khách nước ngoài 25](#_Toc172973972)

[3.2.8. Chức năng xuất hóa đơn trong tháng bất kì 25](#_Toc172973973)

[CHƯƠNG 4: KẾT LUẬN 27](#_Toc172973974)

[DANH MỤC TÀI LIỆU THAM KHẢO 28](#_Toc172973975)

# CHƯƠNG 1: TỔNG QUAN VỀ MẪU THIẾT KẾ CHO PHẦN MỀM

## 1.1. Khái niệm Design Pattern, tác dụng, lý do, phân loại Design Pattern

### 1.1.1. Khái niệm

Trong lĩnh vực phần mềm, Design Pattern là tên gọi chung để chỉ các giải pháp giúp giải quyết những vấn đề cụ thể trong thiết kế phần mềm. Design dịch ra là thiết kế, còn Pattern là khuôn mẫu –> Design Pattern có thể là bản mô tả, là template, concept,… phục vụ mục đích thiết kế phần mềm hợp lý và hiệu quả. Design Pattern được xây dựng thông qua mồ hôi và kinh nghiệm đúc kết của hàng tá lập trình viên, dựa trên bài toán và giải pháp thực tế của họ.

### 1.1.2. Tác dụng của Design Patterns

Nhờ có Design Patterns mà các dev có thể áp dụng để giải quyết nhiều cấn đề khác nhau một cách tương tự. Một số vấn đề mà các bạn gập phải, có thể nếu bạn đã có kinh nghiệm về lĩnh vực IT, bạn có thể nghĩ và đưa ra những giải pháp cho nó. Tuy nhiên đó chưa chắc đã là những phương án tối ưu nhất. Tốt hơn hết bạn nên áp dụng những Patterns để đem lại những phương án tối ưu và chất lượng.

Design Patterns giúp bạn tái sử dụng code và dễ dàng mở rộng code.

Design Patterns cung cấp giải pháp ở dạng tổng quát, giúp tăng tốc độ phát triển phần mềm bằng cách đưa ra các mô hình test, mô hình phát triển đã qua kiểm nghiệm.

Khi bảo trì những dự án nếu được code theo các pattern thì chắc hẳn việc đọc code và fix bug sẽ dễ dàng hơn cho lập trình viên cả về mặt chất lượng công việc cũng như thời gian.

### 1.1.3. Lý do áp dụng Design Patterns

Phần mềm giúp cho các lập trình viên có thể hiểu sâu thêm về các mã code của người khác một cách nhanh chóng hơn nhiều lần. Có thể nói cách khác là hiểu về những mối quan hệ giữa các Module với nhau. Các thành viên ở trong team sẽ trao đổi với nhau một cách dễ dàng và cùng nhau xây dựng các dự án mà không mất quá nhiều thời gian.

Các giải pháp do Design Pattern mang đến đều đã được tối ưu hóa một cách tối đa, hơn nữa kiểm chứng rõ ràng vài giải quyết tốt trong software engineering. Với dạng tổng quát này, phần mềm có thể tăng tốc độ phát triển bằng cách đưa ra nhiều mô hình khác nhau, bao gồm mô hình phát triển, mô hình test khi đã qua kiểm nghiệm.

Nhờ có Design Pattern mà sản phẩm được linh hoạt hơn qua nhiều khâu, có thể dễ dàng mang đi bảo trì và thay đổi nếu có rủi ro xảy ra

Việc phát triển phần mềm luôn gắn liền với những yêu cầu thay đổi. Hệ thống khi phình to ra sẽ có những tính năng mới được thêm vào.

### 1.1.4. Phân loại Design Patterns

Năm 1994, bốn tác giả Erich Gamma, Richard Helm, Ralph Johnson và John Vlissides đã cho xuất bản một cuốn sách với tiêu đề Design Patterns – Elements of Reusable Object-Oriented Software, đây là khởi nguồn của khái niệm design pattern trong lập trình phần mềm.

Hệ thống các mẫu Design pattern hiện có 23 mẫu được định nghĩa trong cuốn “Design patterns Elements of Reusable Object Oriented Software” và được chia thành 3 nhóm:

Creational Pattern (nhóm khởi tạo – 5 mẫu) gồm: Factory Method, Abstract Factory, Builder, Prototype, Singleton. Những Design pattern loại này cung cấp một giải pháp để tạo ra các object và che giấu được logic của việc tạo ra nó, thay vì tạo ra object một cách trực tiếp bằng cách sử dụng method new. Điều này giúp cho chương trình trở nên mềm dẻo hơn trong việc quyết định object nào cần được tạo ra trong những tình huống được đưa ra.

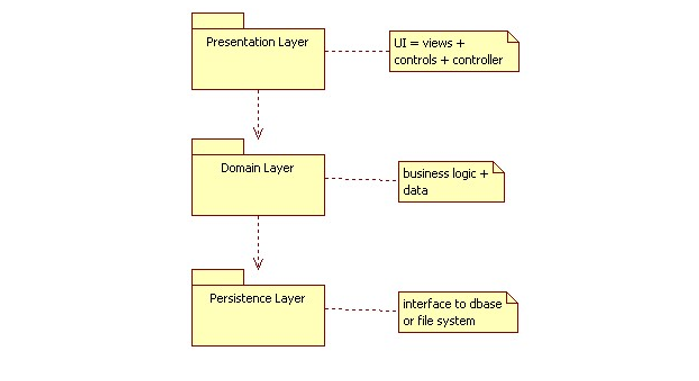
Structural Pattern (nhóm cấu trúc – 7 mẫu) gồm: Adapter, Bridge, Composite, Decorator, Facade, Flyweight và Proxy. Những Design pattern loại này liên quan tới class và các thành phần của object. Nó dùng để thiết lập, định nghĩa quan hệ giữa các đối tượng.

Behavioral Pattern (nhóm tương tác/ hành vi – 11 mẫu) gồm: Interpreter, Template Method, Chain of Responsibility, Command, Iterator, Mediator, Memento, Observer, State, Strategy và Visitor. Nhóm này dùng trong thực hiện các hành vi của đối tượng, sự giao tiếp giữa các object với nhau.

## 1.2. Mô hình 3 lớp (Three-layer)

### 1.2.1. Khái niệm

Mô hình 3 lớp hay còn được gọi là mô hình Three Layer (3-Layer), mô hình này ra đời nhằm phân chia các thành phần trong hệ thống, các thành phần cùng chức năng sẽ được nhóm lại với nhau và phân chia công việc cho từng nhóm để dữ liệu không bị chồng chéo và chạy lộn xộn. Mô hình này phát huy hiệu quả nhất khi bạn xây dựng một dự án(project) lớn, việc quản lý code và xử lý dữ liệu lỗi dễ dàng hơn.



Hình 1. 1 Three-Layer Pattern

**Ưu điểm :**

Phân loại rõ ràng các lớp có các nhiệm vụ khác nhau. Từ đó ta có thể quản lý và duy trì project tốt hơn.

- Dễ dàng phân loại các hành động tại Business.

- Dễ dàng phân loại các hàm truy xuất tại Database, phân loại hàm theo table,…

- Ứng dụng được cho các project lớn ở bên ngoài.

Mô hình 3-layer gồm có 3 phần chính:

Presentation Layer (GUI): lớp này có nhiệm vụ chính là giao tiếp với người dùng. Nó gồm các thành phần giao diện ( winform, webform, …) và thực hiện các công việc như nhập liệu, hiển thị dữ liệu, kiểm tra tính đúng đắn dữ liệu trước khi gọi lớp Business Logic Layer (BLL).

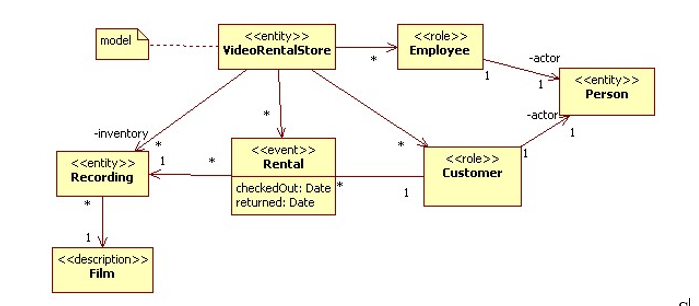
Business Logic Layer (BLL) Layer này phân ra 2 thành nhiệm vụ: Đây là nơi đáp ứng các yêu cầu thao tác dữ liệu của GUI layer, xử lý chính nguồn dữ liệu từ Presentation Layer trước khi truyền xuống Data Access Layer và lưu xuống hệ quản trị CSDL. Còn là nơi kiểm tra các ràng buộc, tính toàn vẹn và hợp lệ dữ liệu, thực hiện tính toán và xử lý các yêu cầu nghiệp vụ, trước khi trả kết quả về Presentation Layer.

Data Access Layer (DAL): Lớp này có chức năng giao tiếp với hệ quản trị CSDL như thực hiện các công việc liên quan đến lưu trữ và truy vấn dữ liệu ( tìm kiếm, thêm, xóa, sửa,…).

### 1.2.2. The Domain Layer

Lớp Domain layer được gọi là lớp nghiệp vụ , nó đóng góp logic và dữ liệu business or domain logic and data. Lớp này có thể được triển khai như một application server.

Domain Model là thuật ngữ cùng để chỉ là các class đại diện cho các domain. Trong Domain Model sẽ bao gồm các đối tượng thuộc business như Client, Report, Department,… Các class đại diện cho kết quả tính toán, ham số đầu vào cho service tính toán,… đều được coi là domain model. Domain Model cũng sẽ bao gồm cả entity, tuy nhiên nó sẽ tương ứng với table trong database, có thể map vào database được và chỉ có entity mới có thể đại diện cho data trong database mà thôi.



Hình 1. 2 Domain Layer

Trong Domain Model sẽ được phân thành 4 layer có vai trò đảm nhiệm những nhiệm vụ khác nhau như sau:

User Interface Layer: trong domain model class diagram sẽ có chức năng biểu diễn thông tin trực quan cho user. Sau đó sẽ dịch những user command xử lý ở những tầng dưới hoạt động tiếp theo.

Application Layer: được thiết kế khá mỏng làm nhiệm vụ quan trọng là coordinate những Activity của Application. Trong Application Layer không chứa Business Logic, đặc biệt là nó không chứa state của những Business Object mà chỉ chứa state của Application Task Progress mà thôi.

Domain Layer - domain model class diagram: Domain Layer chính là bộ phận quan trọng nhất. Đặc biệt là state của Business Object đều đã lưu trữ sẽ được chuyển giao cho tầng Infrastructure ở dưới.

Infrastructure Layer chính là supporting libraries cho những tầng khác hiệu quả tốt hơn. Nó vừa hỗ trợ sự giao tiếp giữa những Layer với nhau, cũng như như giúp lưu trữ những Business Object của tầng Domain model một cách hoàn thiện nhất.

### 1.2.3. The Presentation Layer

Lớp này có nhiệm vụ chính là giao tiếp với người dùng. Nó gồm các thành phần giao diện ( winform, webform, …) và thực hiện các công việc như nhập liệu, hiển thị dữ liệu, kiểm tra tính đúng đắn dữ liệu trước khi gọi lớp Business Logic Layer (BLL).

*1.2.3.1. Model-View-Controller Pattern*

Cấu trúc của lớp Presentation và domain layers thể hiện mô hình thiết kế Model-View-Controller.

A diagram of a system

Description automatically generated

Hình 1. 3 MVC Pattern

*1.2.3.2. View Notification Problem*

Khi dữ liệu trong lớp domain layer thay đổi, các chế độ xem của dữ liệu này trong lớp presentation layer sẽ biết cách tự vẽ lại như thế nào? Hãy nhớ rằng không có sự phụ thuộc nào từ domain layer đến lớp presentation layer. Đây được gọi là vấn đề view notification problem, có nhiều cơ chế khác nhau để giải quyết các vấn đề notification problems. Cơ chế nổi tiếng nhất là mô hình publisher-subsscriber

A diagram of a computer generated flowchart

Description automatically generated

Hình 1. 4 Publisher-Subcriber (Observer) Pattern

*1.2.3.3. Controllers*

Khi người dùng nhấp vào một nút, chọn một mục menu hoặc nhập văn bản tại dấu nhắc, một số loại lệnh được tạo và gửi đến bộ điều khiển. Mối quan hệ giữa views and controllers có thể là nhiều-một hoặc một-một (hoặc N-đến-M):

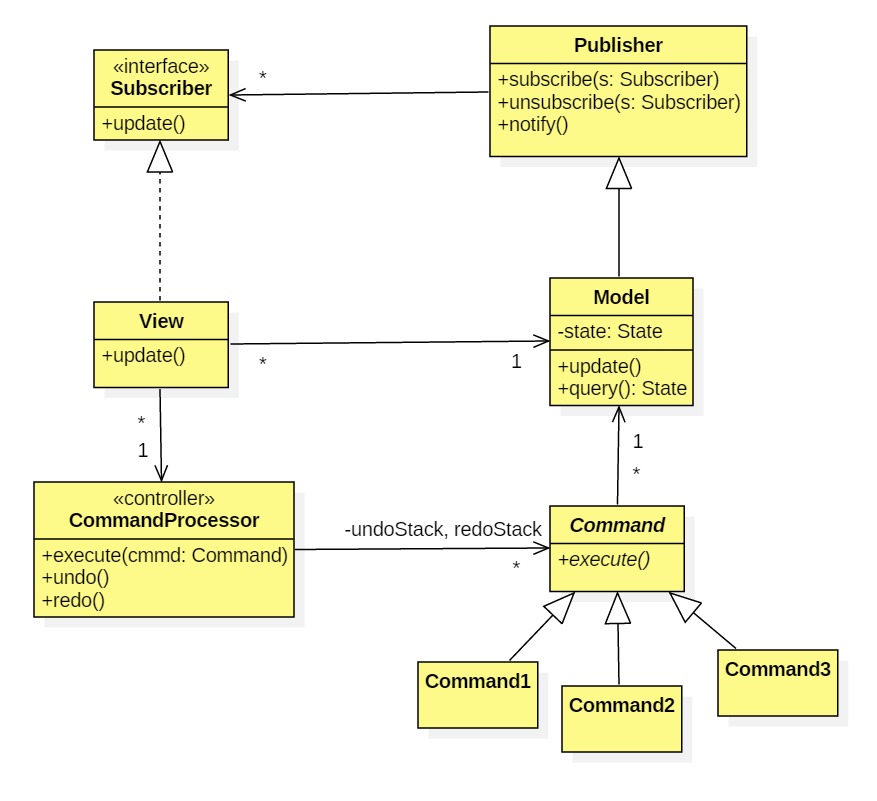
A diagram of a diagram

Description automatically generated

A diagram of a diagram

Description automatically generated

Nếu có một bộ controller tập trung, chúng ta gọi đó controller hoặc command processor. Ngoài ra, có thể có một bộ controller cho mỗi view (gọi là bộ cotroller page) hoặc một bộ controller cho mỗi trường hợp sử dụng gọi là bộ điều khiển trường hợp sử dụng.



Hình 1. 5 CommandProcessor Pattern

*1.2.3.4. Smart controller, dumb commands and views*

Trong trường hợp này, controller biết cách cập nhật mô hình. Lệnh chỉ là chuỗi hoặc danh sách mã thông báo. View chỉ biết cách vẽ.

A diagram of a computer program

Description automatically generated

*1.2.3.5. Smart commands, dumb controller and views.*

Trong trường hợp này, controller chỉ yêu cầu lệnh tự thực thi.

A diagram of a computer program

Description automatically generated

Một view xem thông minh biết cách hiển thị chính nó và cách cập nhật mô hình. Bộ controller chỉ chọn chế độ xem dựa trên lệnh, sau đó chuyển tiếp lệnh chưa thực hiện đến view.

A diagram of a computer program

Description automatically generated

### 1.2.4. The Persistence Layer

Không phải tất cả các đối tượng đều có thể được lưu vào bộ nhớ thứ cấp (tức là tệp hoặc cơ sở dữ liệu). Một đối tượng có thể được gọi là bền bỉ. Một đối tượng không thể được gọi là tạm thời. Thông thường, các đối tượng kinh doanh đại diện cho khách hàng, nhân viên, giao dịch, v.v. cần phải bền bỉ, trong khi các đối tượng kiến ​​trúc như thành phần UI có thể là tạm thời. Theo mặc định, tất cả các đối tượng Java đều là tạm thời.

Việc lưu các đối tượng vào tệp được gọi là tuần tự hóa.

Hầu hết các ứng dụng doanh nghiệp lưu các đối tượng bền bỉ vào cơ sở dữ liệu quan hệ.

**Relational Databases**

A diagram of a server

Description automatically generated

**Table Gateway (DAO)**

Ý tưởng của cổng bảng là cung cấp một lớp cổng cho mỗi bảng cơ sở dữ liệu.

A diagram of a computer

Description automatically generated

Hình 1. 6 DAO

# CHƯƠNG 2: PHÂN TÍCH, THIẾT KẾ CHO PHẦN MỀM

## 2.1. Mô hình nghiệp vụ

**- Mô tả các yêu cầu:**

*+ Thêm Hóa đơn:*

- Chức năng: Cho phép người dùng nhập thông tin hóa đơn mới vào hệ thống.

- Thông tin cần thiết: Mã hóa đơn, họ tên, ngày ra hóa đơn, đối tượng khách hàng, quốc tịch (Việt Nam/người nước ngoài), số lượng điện tiêu thụ, đơn giá, định mức và thành tiền.

*+ Xóa Hóa đơn:*

- Chức năng: Cho phép người dùng xóa một hóa đơn cụ thể khỏi hệ thống.

- Yêu cầu: Cung cấp mã hóa đơn nhận diện khác để xác định hóa đơn cần xóa.

*+ Sửa Hóa đơn:*

- Chức năng: Cho phép người dùng chỉnh sửa thông tin của một hóa đơn đã tồn tại.

- Thông tin cần sửa: Mã hóa đơn, họ tên, ngày ra hóa đơn, đối tượng khách hàng, quốc tịch (Việt Nam/người nước ngoài), sốlượng điện tiêu thụ, đơn giá, định mức và thành tiền.

*+ Tính tổng số lượng cho từng loại khách hàng:*

- Chức năng: Tính tổng số lượng điện tiêu thụ cho từng loại khách hàng (sản xuất, sinh hoạt, kinh doanh).

- Yêu cầu: Phân loại khách hàng và cộng dồn số lượng điện tiêu thụ tương ứng.

*+ Tính trung bình thành tiền của khách hàng người nước ngoài:*

- Chức năng: Tính giá trị trung bình của các hóa đơn cho khách hàng người nước ngoài.

- Yêu cầu: Phân lọai các hóa đơn của khách hàng người nước ngoài và tính trung bình thành tiền của những hóa đơn này.

*+ Xuất ra các hóa đơn trong tháng nào đó:*

- Chức năng: Hiển thị tất cả các hóa đơn trong một tháng cụ thể.

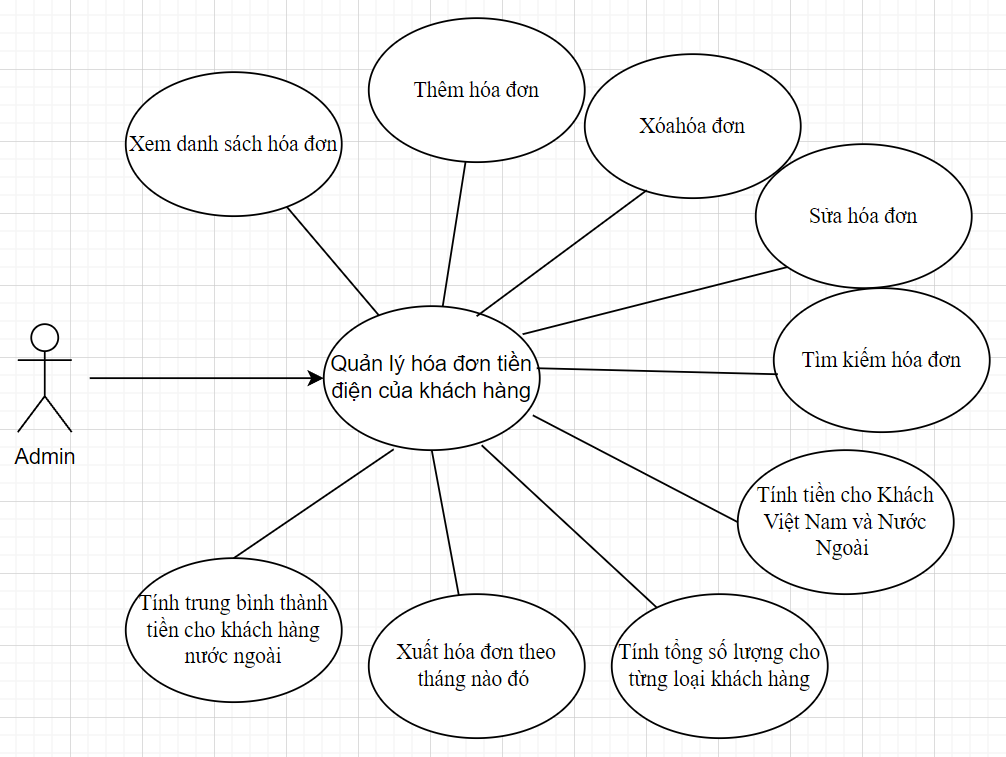
- Yêu cầu: Nhập tháng để lọc và xuất ra các hóa đơn tương ứng.

*+ Tìm kiếm Hóa đơn:*

- Chức năng: Tìm kiếm hóa đơn dựa trên tên khách hàng.

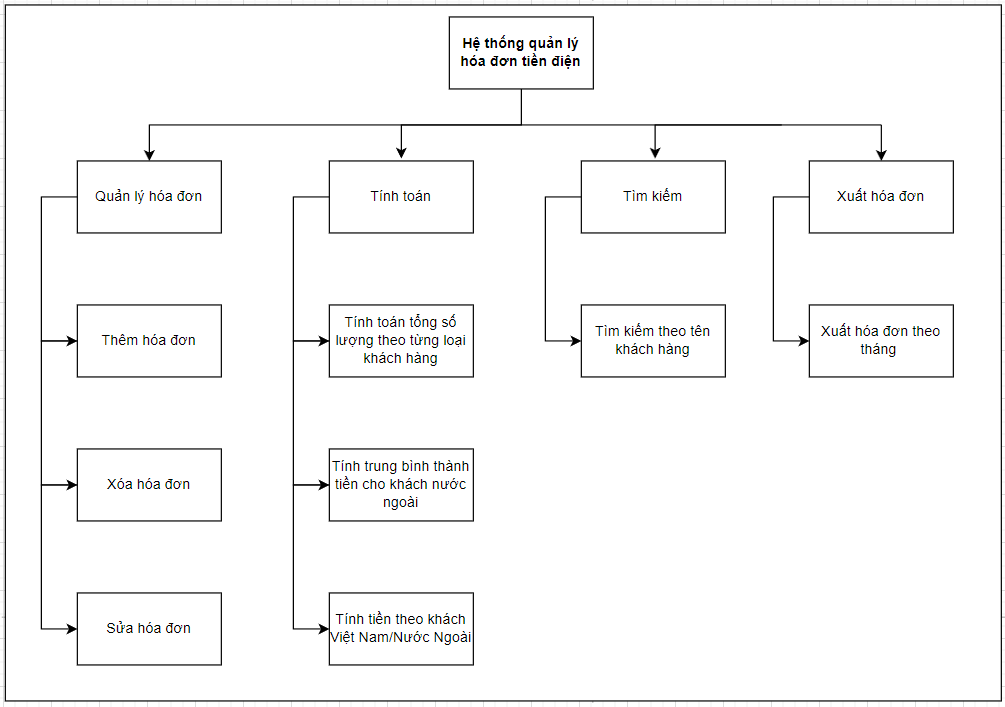
- Yêu cầu: Cung cấp tiêu chí tìm kiếm và hệ thống sẽ trả về các hóa đơn phù hợp.

### 2.1.1. Sơ đồ hệ thống



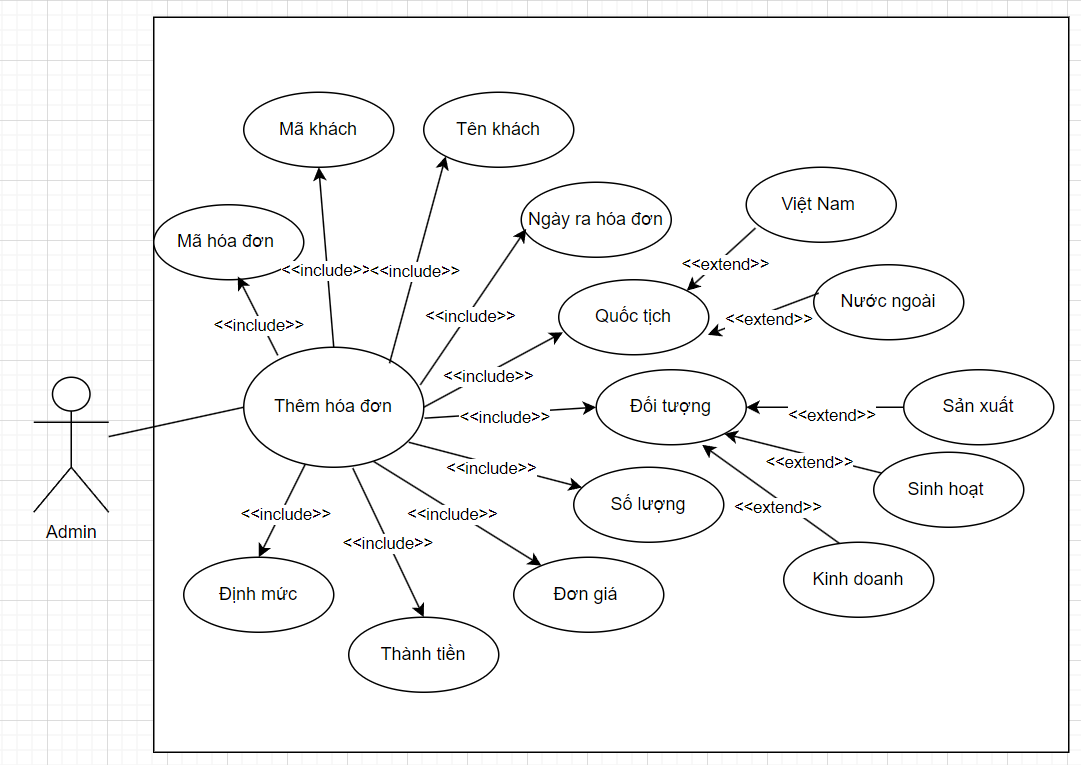
Hình 2. 1 Sơ đồ hệ thống

### 2.1.2. Sơ đồ phân cấp chức năng



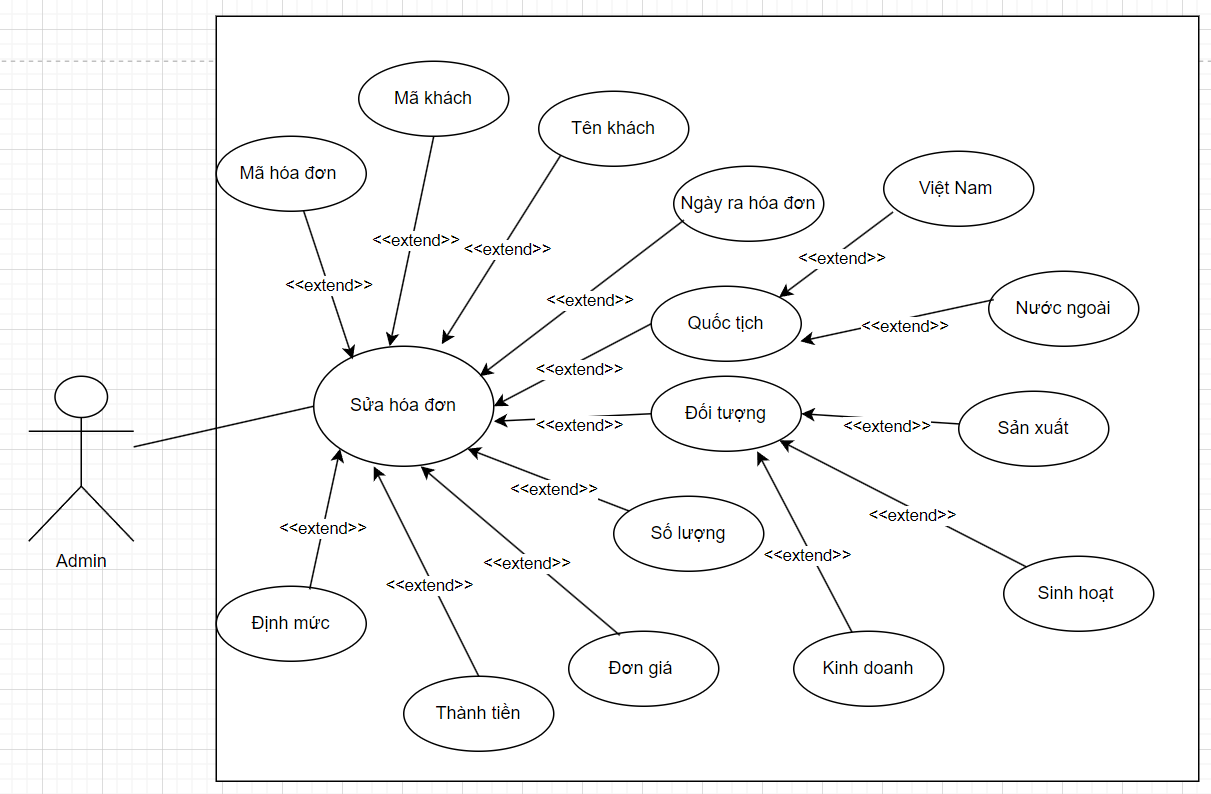
Hình 2. 2 Sơ đồ phân cấp chức năng

### 2.1.3. Sơ đồ USECASE Thêm hóa đơn



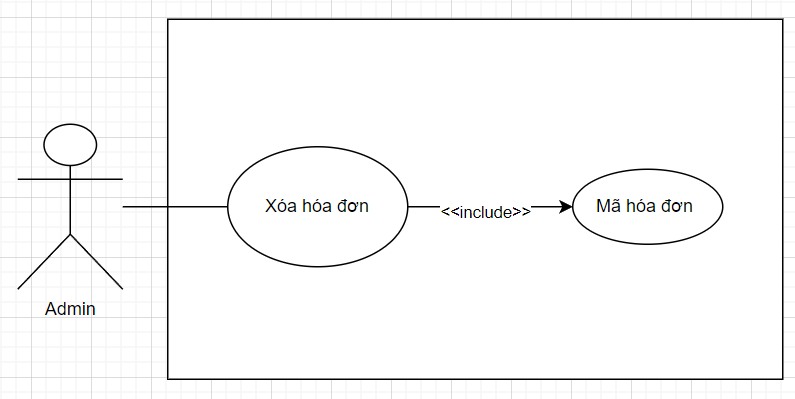
Hình 2. 3 Sơ đồ USECASE Thêm hóa đơn

### 2.1.4. Sơ đồ USECASE Sửa hóa đơn



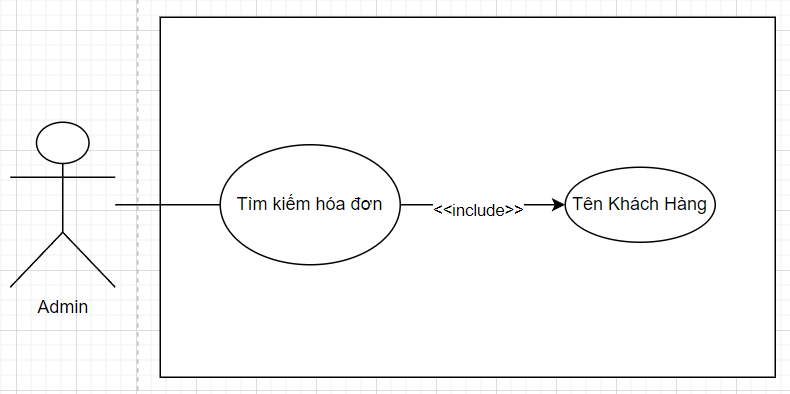
Hình 2. 4 Sơ đồ USECASE Sửa hóa đơn

### 2.1.5. Sơ đồ USECASE Xóa hóa đơn



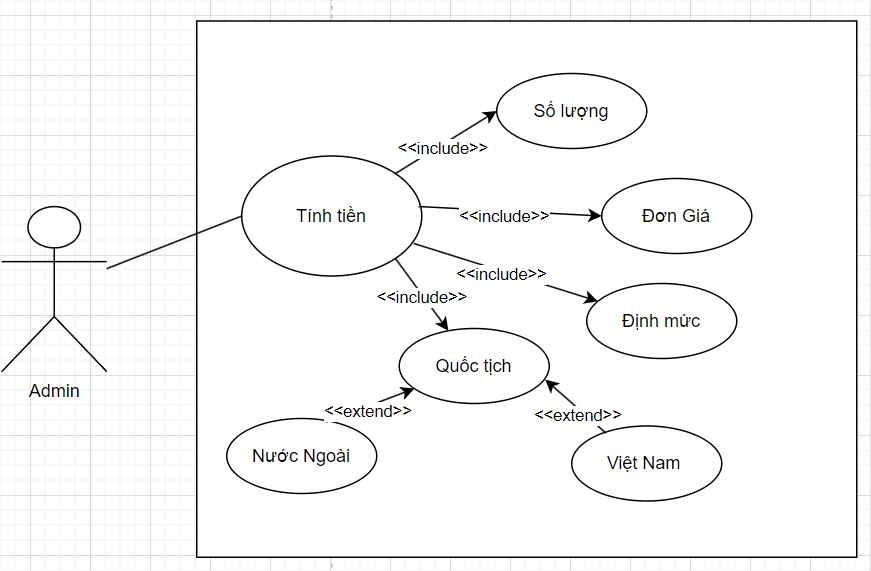
Hình 2. 5 Sơ đồ USECASE Xóa hóa đơn

### 2.1.6. Sơ đồ USECASE Tìm kiếm hóa đơn



Hình 2. 6 Sơ đồ USECASE Tìm kiếm hóa đơn

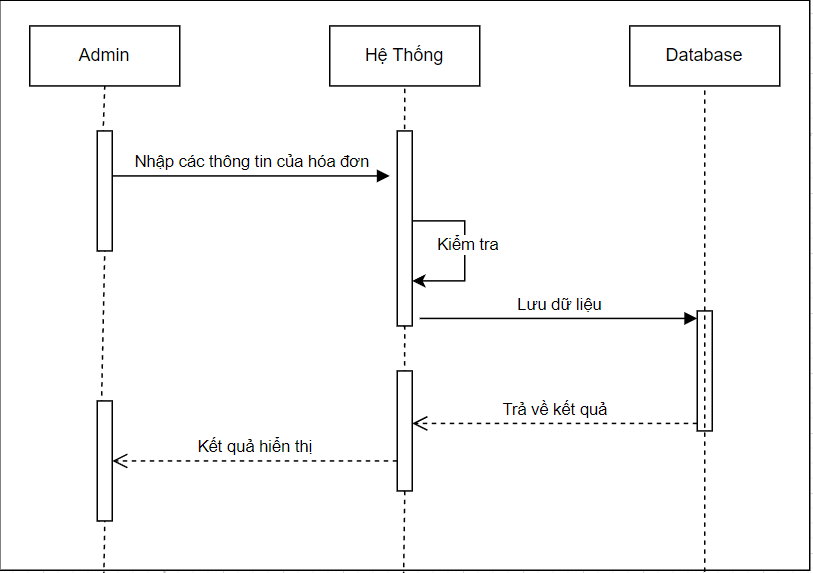
### 2.1.7. Sơ đồ USECASE Tính tiền hóa đơn



Hình 2. 7 Sơ đồ USECASE Tính tiền hóa đơn

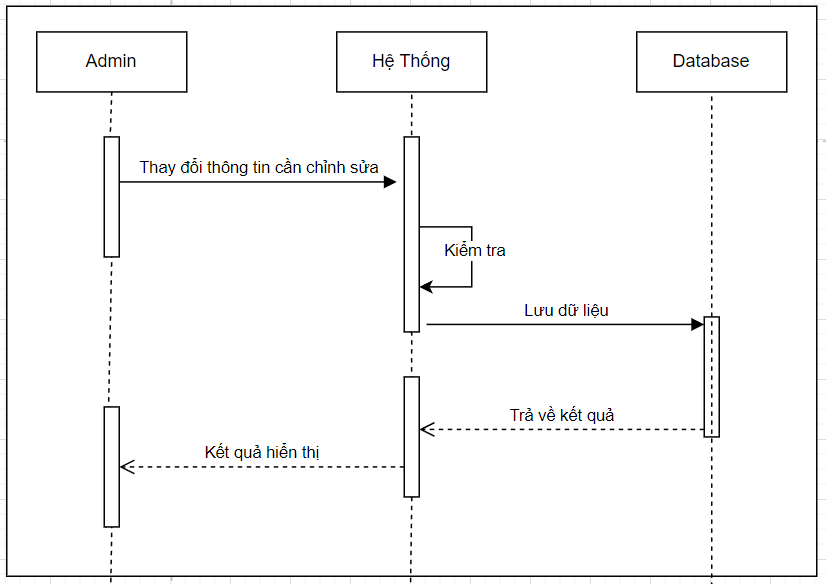
## **2.2. Sơ đồ tuần tự (Sequency Diagram)**

### 2.2.1. Sơ đồ tuần tự Thêm hóa đơn



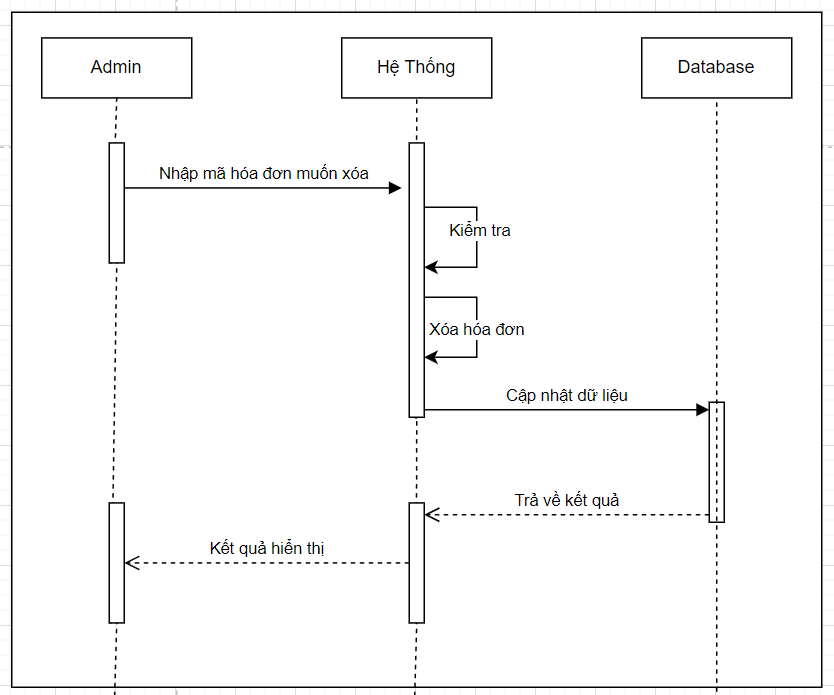
Hình 2. 8 Sơ đồ tuần tự Thêm hóa đơn

2.2.2. Sơ đồ tuần tự Sửa hóa đơn



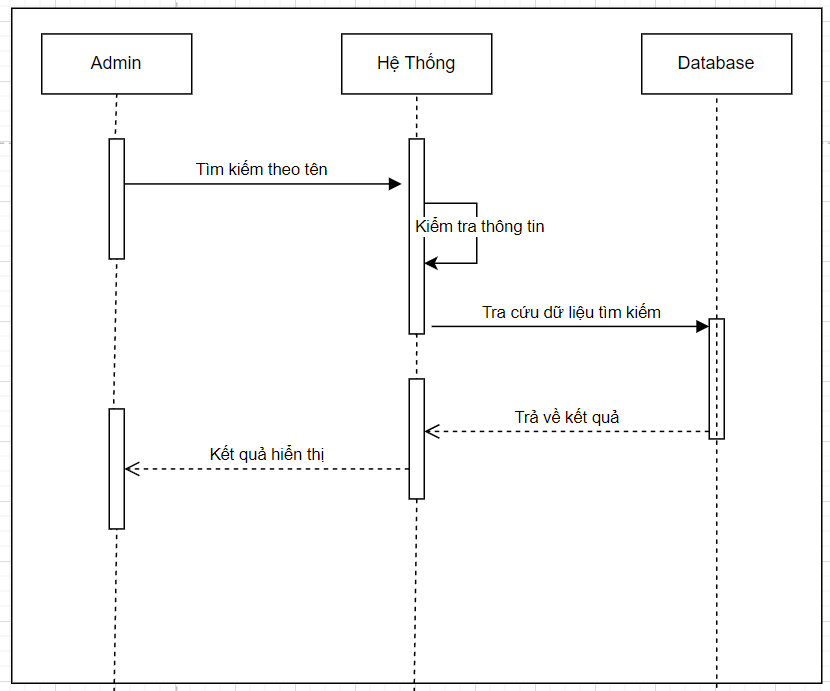
Hình 2. 9 Sơ đồ tuần tự Sửa hóa đơn

2.2.3. Sơ đồ tuần tự Xóa hóa đơn



Hình 2. 10 Sơ đồ tuần tự Xóa hóa đơn

### 2.2.4. Sơ đồ tuần tự Tìm kiếm hóa đơn



Hình 2. 11 Sơ đồ tuần tự Tìm kiếm hóa đơn

## 2.3. Sơ đồ thiết kế

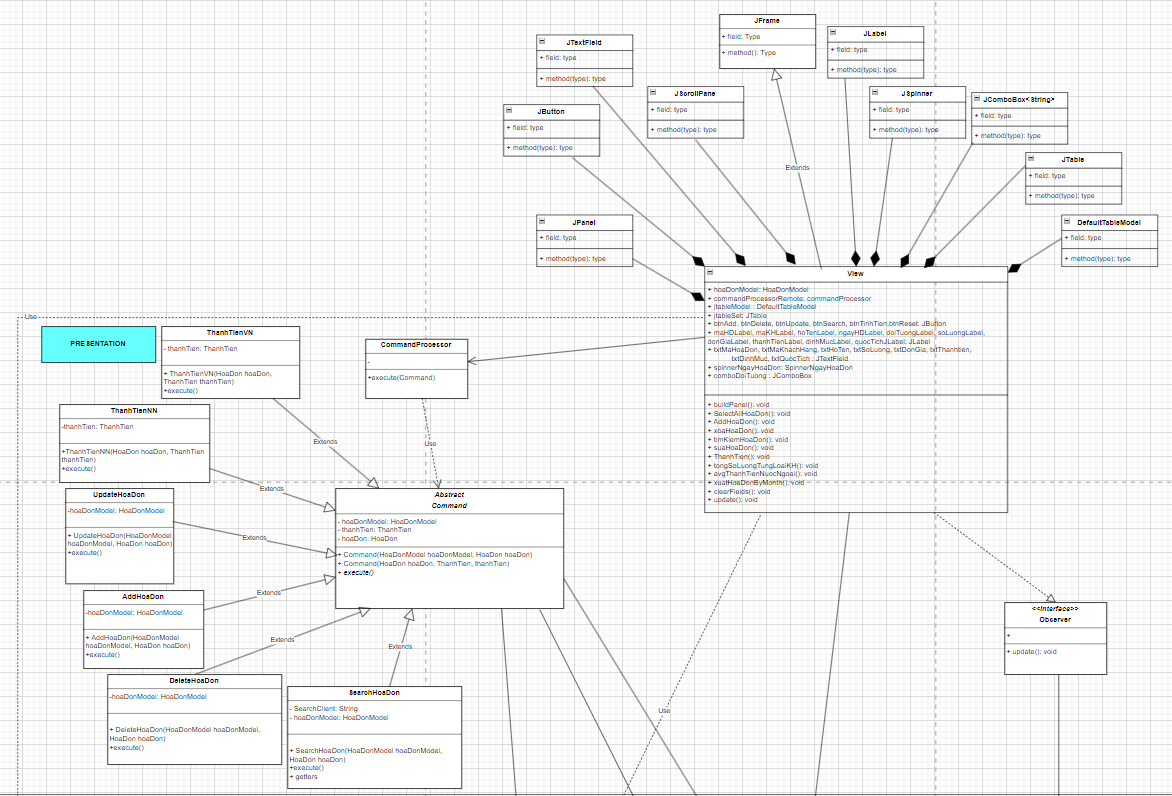
- Sử dụngmô hình Three-Layer đề chia chương trình thành 3 tầng:

+ Presentation Layer: Tầng này chịu trách nhiệm về giao diện người dùng (UI) và tương tác với người dùng.

+ Domain Layer: Thực hiện các Logic nghiệp vụ, xử lý dữ liệu và các quy tắc kinh doanh.

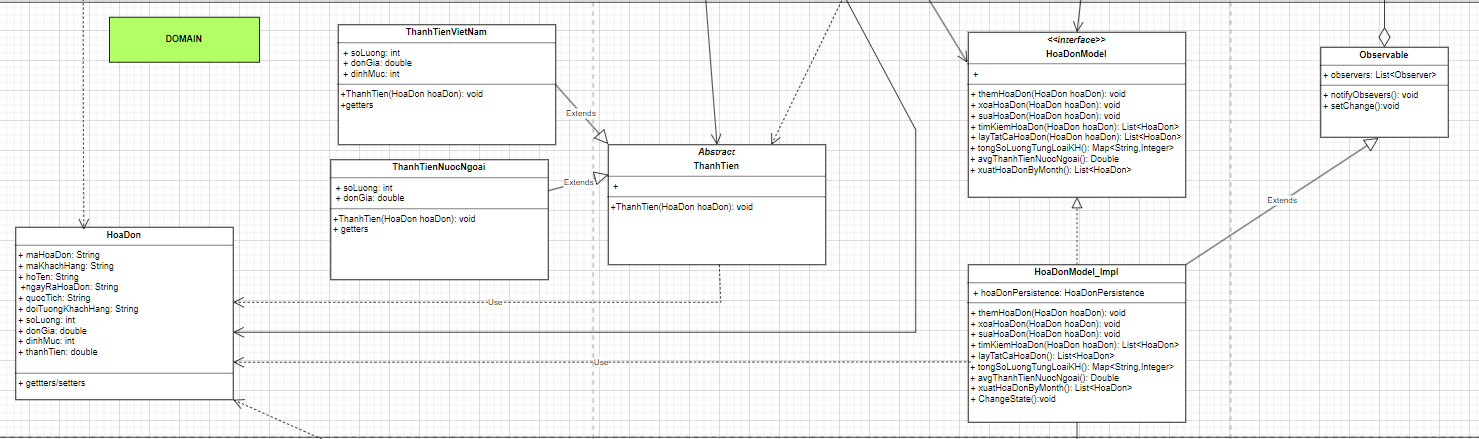
+ Persistence Layer: Tầng này chịu trách nhiệm về việc lưu trữ và truy xuất dữ liệu từ cơ sở dữ liệu hoặc các nguồn dữ liệu khác. Nó cung cấp các interface để giao tiếp với cơ sở dữ liệu và quản lý các hoạt động CRUD (Create, Read, Update, Delete).

*Tầng Presentation*



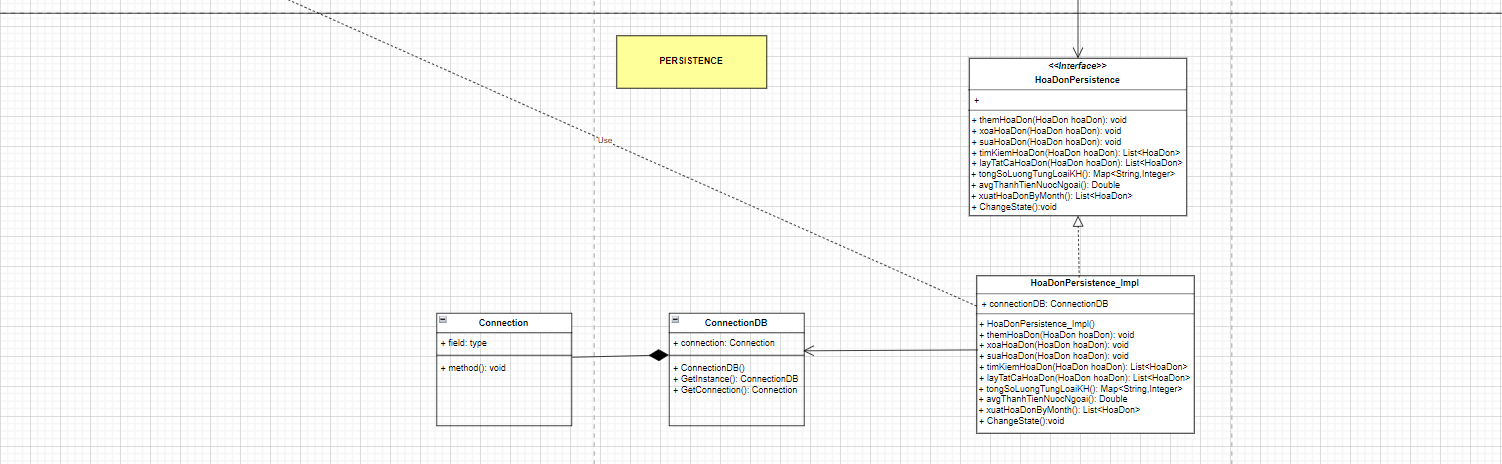
Hình 2. 12 Tầng Presentation

*Tầng Domain*

**

Hình 2. 13 Tầng Domain

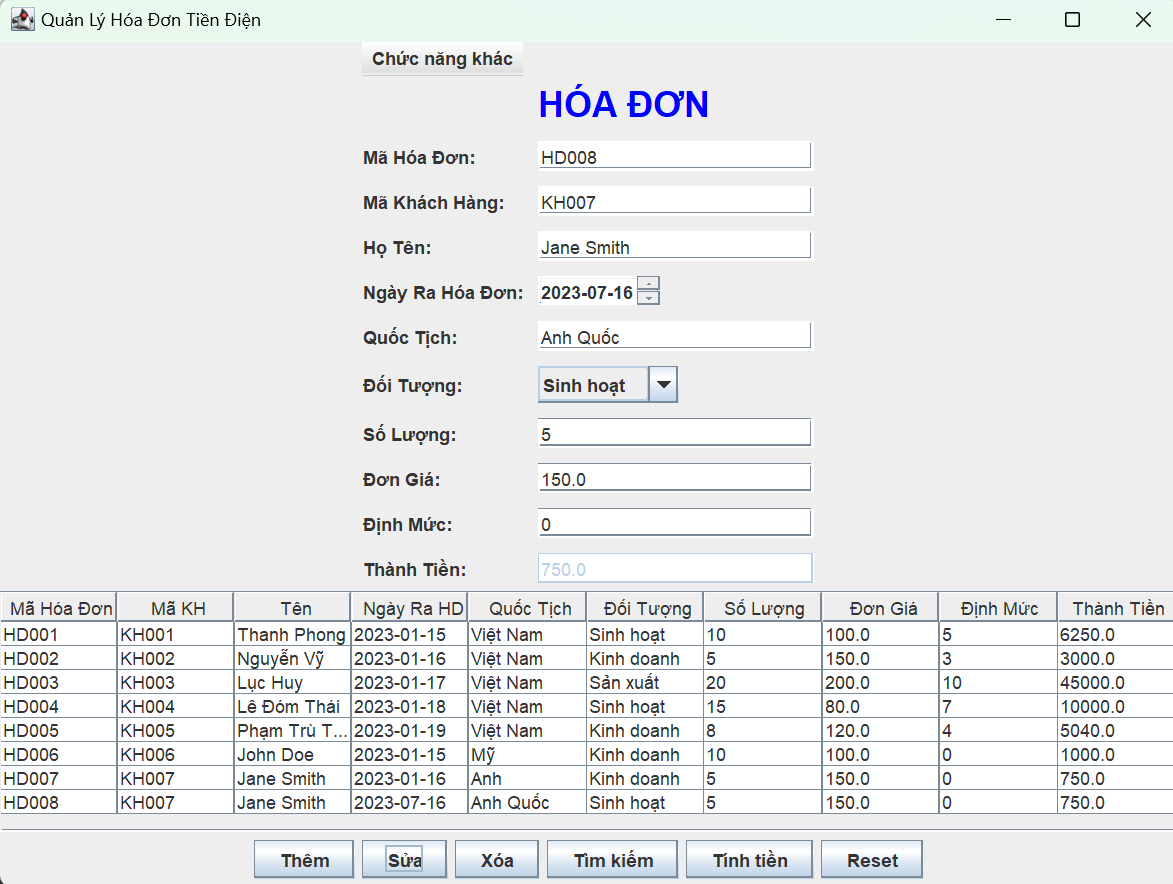
*Tầng Persistence*

**

Hình 2. 14 Tầng Persistence

# **CHƯƠNG 3: CÀI ĐẶT THỰC NGHIỆM PHẦN MỀM**

## 3.1. Giao diện phần mềm

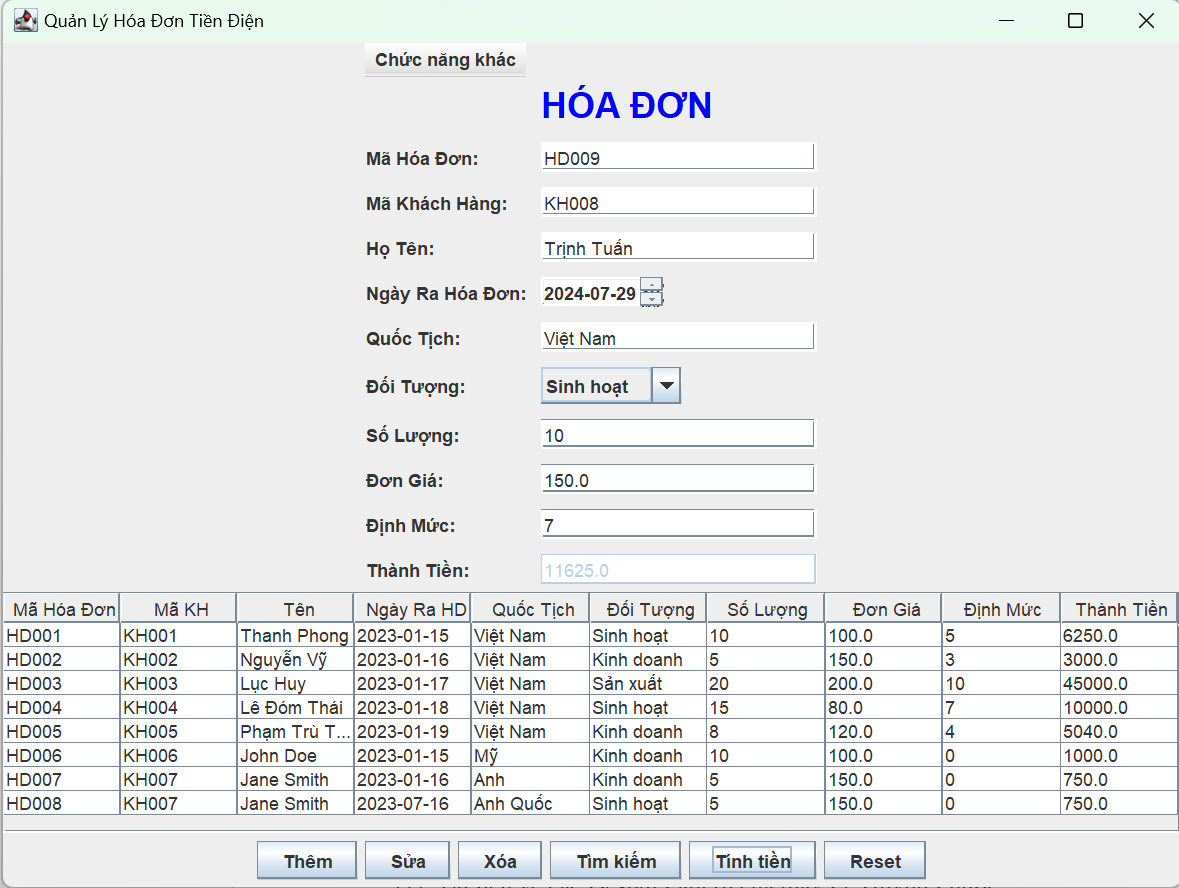
****

Hình 3. 1 Giao diện phần mềm

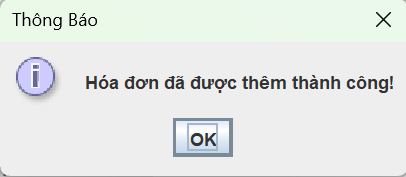
## 3.2. Thực nghiệm các chức năng

### 3.2.1. Chức năng thêm hóa đơn

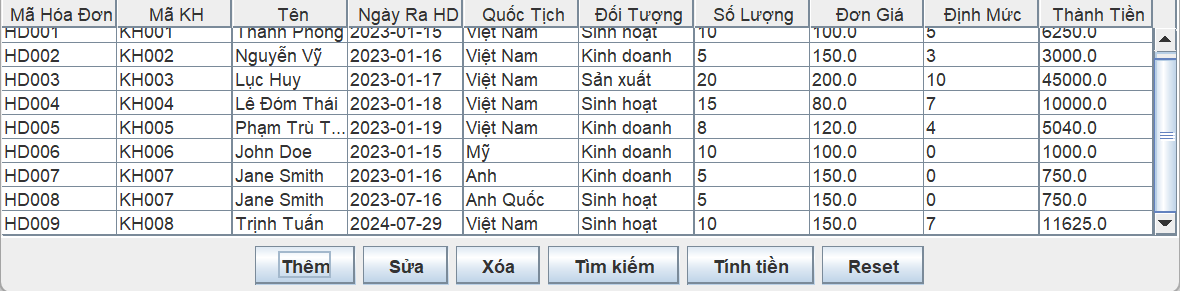
- Nhập các thông tin cho hóa đơn



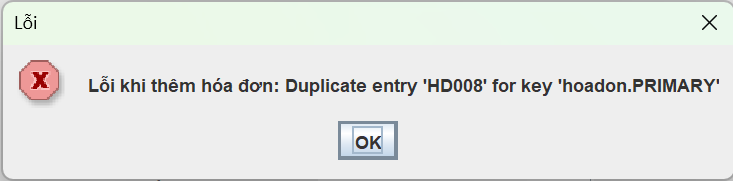
- Sau khi bấm nút Thêm



- Kết quả sau khi thêm

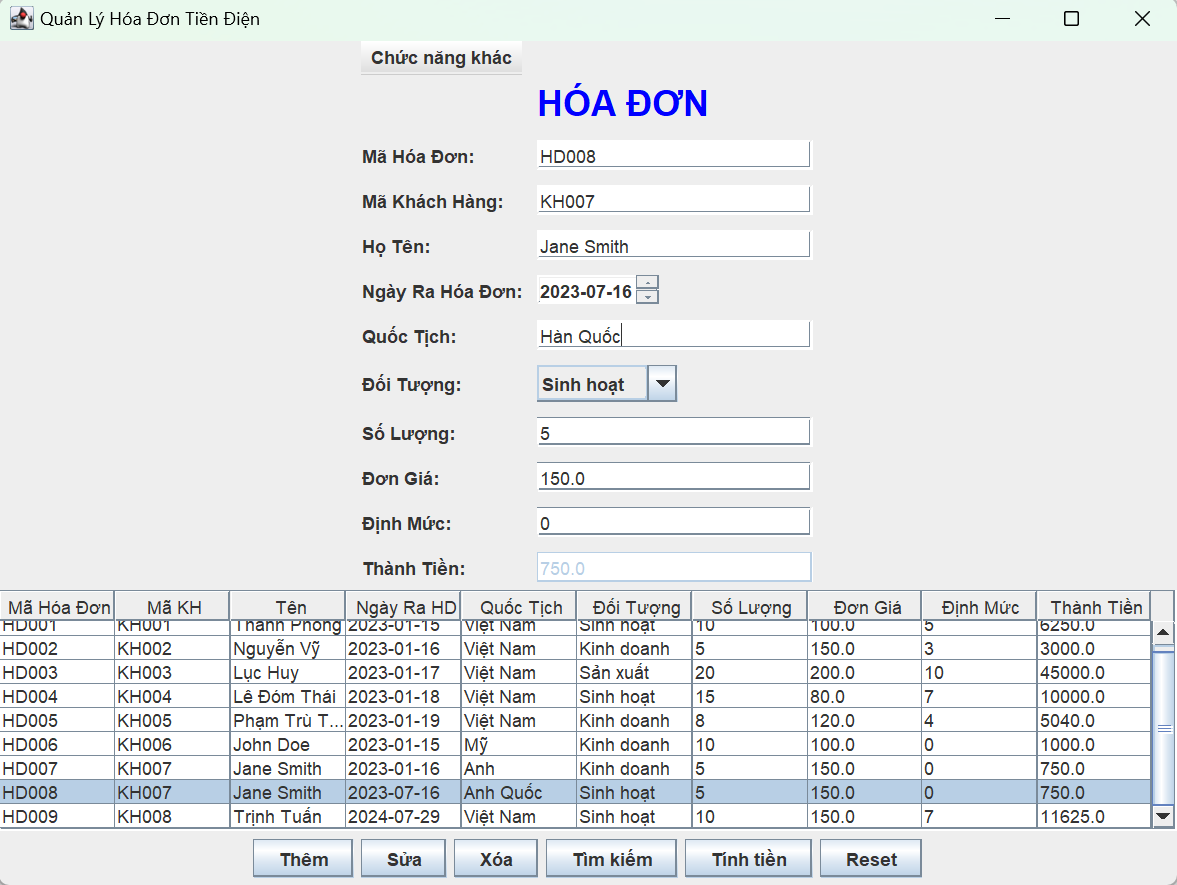


- Ngược lại nếu nhập trùng mã hóa đơn:

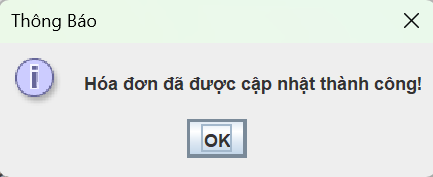


### 3.2.2. Chức năng sửa hóa đơn

- Thay đổi thông tin cần sửa của hóa đơn



- Sau khi bấm nút sửa



- Kết quả

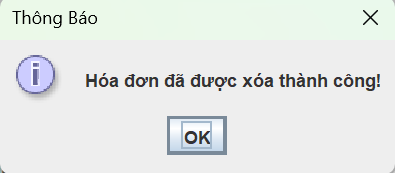


### 3.2.3. Chức năng xóa hóa đơn

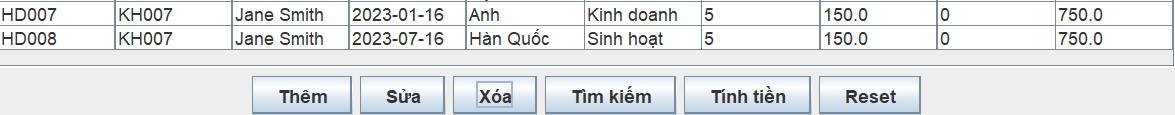
- Nhập mã hóa đơn muốn xóa



- Sau khi bấm nút xóa

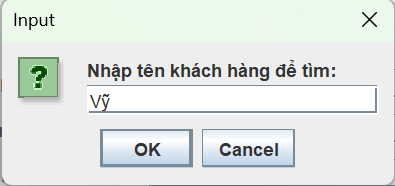


-Kết quả sau khi xóa

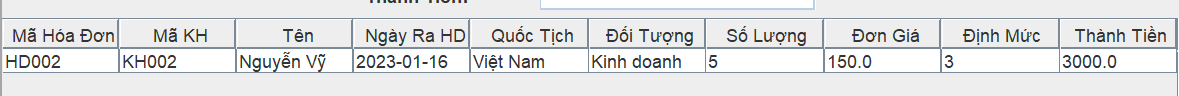


### 3.2.4. Chức năng tìm kiếm

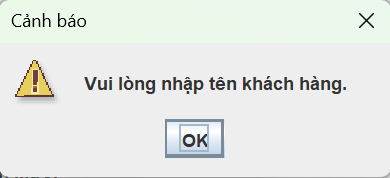
- Bấm nút tìm kiếm



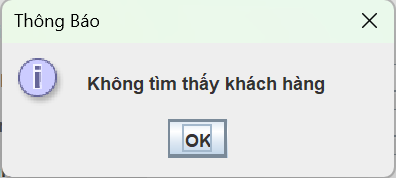
- Kết quả tìm kiếm



- Ngược lại, nếu không nhập tên khách hàng thì:



- Nếu không tìm thấy:

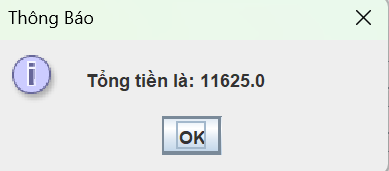


### 3.2.5. Chức năng tính tiền

- Nhập các thông tin cần thiết



- Sau khi bấm nút tính tiền



- Kết quả sẽ lưu vào textfeilds Thành Tiền

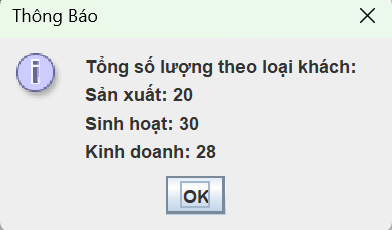


### 3.2.6. Chức năng tính tổng số lượng cho từng loại khách hàng

- Bấm chọn chức năng trên menu Chức năng

- Chọn chức năng tính tổng số lượng cho từng khách hàng

- Kết quả:

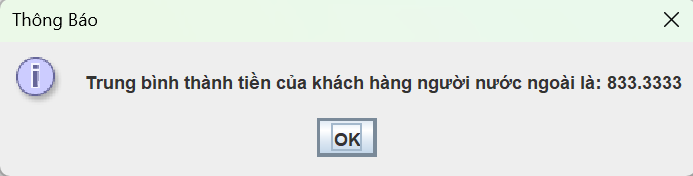


### 3.2.7. Chức năng tính trung bình thành tiền cho khách nước ngoài

- Bấm chọn chức năng trên menu Chức năng

- Chọn chức năng tính trung bình thành tiền cho khách nước ngoài

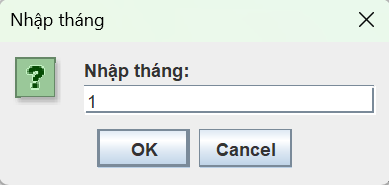
- Kết quả:



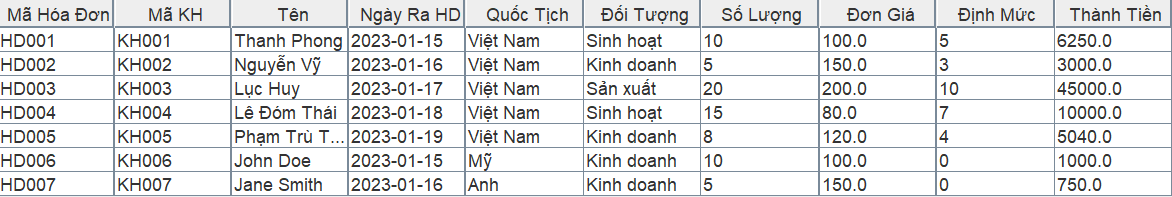
### 3.2.8. Chức năng xuất hóa đơn trong tháng bất kì

- Bấm chọn chức năng trên menu Chức năng

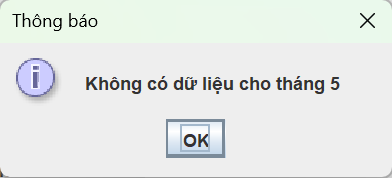
- Chọn chức năng xuất hóa đơn trong tháng



- Kết quả sau khi nhập tháng và bấm OK



- Nếu không tìm thấy dữ liệu của tháng bất kì:



# CHƯƠNG 4: KẾT LUẬN

**Kết luận:**

Trong bài tiểu luận này, chúng em đã phân tích và xây dựng một chương trình quản lý hóa đơn tiền điện dành cho khách hàng với mục tiêu cải thiện hiệu quả quản lý và cung cấp dịch vụ khách hàng tốt hơn. Qua quá trình nghiên cứu và phát triển, chúng em đã đạt được một số kết quả:

* Kết quả đạt được: Chương trình quản lý hóa đơn tiền điện được thiết kế đã thực hiện thành công các chức năng chính bao gồm theo dõi và quản lý hóa đơn và cung cấp thông tin chi tiết về mức tiêu thụ điện năng. Hệ thống này đã cải thiện rõ rệt sự minh bạch trong việc quản lý hóa đơn và giảm thiểu các lỗi liên quan đến việc thanh toán và ghi chép.
* Đánh giá ưu điểm: Chương trình không chỉ giúp tiết kiệm thời gian và công sức cho nhân viên quản lý mà còn nâng cao trải nghiệm thông qua việc cung cấp giao diện thân thiện và các công cụ hỗ trợ tự động. Đặc biệt, việc tích hợp các chức năng tự động hóa giúp giảm thiểu rủi ro từ lỗi con người và cải thiện độ chính xác của thông tin hóa đơn.
* Khuyết điểm: Mặc dù chương trình đã đạt được nhiều kết quả tích cực, nhưng vẫn còn một số hạn chế cần được cải thiện. Cụ thể, giao diện cần được tối ưu hóa thêm để phù hợp với nhu cầu và thói quen sử dụng của đa dạng đối tượng. Bên cạnh đó, hệ thống hiện tại chưa hỗ trợ đầy đủ các ngôn ngữ và phương thức thanh toán khác nhau, điều này có thể hạn chế khả năng quản lý.

**Hướng phát triển tương lai:**

Để nâng cao hơn nữa hiệu quả của chương trình, trong tương lai, cần tiếp tục cải tiến giao diện người dùng, mở rộng hỗ trợ đa ngôn ngữ và tích hợp thêm các phương thức thanh toán mới. Ngoài ra, việc ứng dụng công nghệ trí tuệ nhân tạo và phân tích dữ liệu để dự đoán xu hướng tiêu thụ và quản lý hóa đơn cũng là một hướng phát triển tiềm năng. Điều này không chỉ giúp tối ưu hóa quy trình quản lý mà còn mang lại lợi ích lâu dài cho cả khách hàng và nhà cung cấp dịch vụ.

# DANH MỤC TÀI LIỆU THAM KHẢO

[1]. Tài liệu về các Design Pattern của thầy Lê Huỳnh Phước.

[2]. Phong, T. T. Sử dụng hiệu quả ngôn ngữ đặc tả UML trong phát triển phần mềm.

[3]. Jade Morales (2023). Tìm hiểu các chi tiết quan trọng của sơ đồ tuần tự UML.