

Thành viên hội đồng
(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Trước tiên, em xin gửi lời cảm ơn chân thành sâu sắc tới quý thầy, cô Trường Đại học Trà Vinh nói chung và quý thầy, cô giảng viên Khoa Kỹ thuật và Công nghệ, bộ môn Công nghệ thông tin nói riêng đã tận tình giảng dạy, truyền đạt cho em những kiến thức, kinh nghiệm quý báu trong suốt thời gian qua.

Đặc biệt, em xin gửi lời cảm ơn đến thầy **Phạm Minh Dương** đã tận tình giúp đỡ, trực tiếp chỉ bảo và hướng dẫn trong suốt quá trình làm đồ án chuyên ngành.

Trong thời gian làm việc với thầy, em không những tiếp thu thêm nhiều kiến thức bổ ích mà còn học tập được tinh thần làm việc, thái độ nghiên cứu khoa học nghiêm túc, hiệu quả, đây là những điều rất cần thiết cho em trong quá trình học tập cũng như làm việc sau này.

Sau cùng xin gửi lời cảm ơn chân thành tới gia đình, bạn bè đã động viên giúp đỡ em trong quá trình học tập, nghiên cứu và hoàn thành đồ án chuyên ngành này.

Chân thành cảm ơn!

.

Lê Bùi Minh Nhẫn

MỤC LỤC

MỞ ĐẦU	1
1. Lý do chọn đề tài:	1
2. Mục tiêu nghiên cứu:	1
3. Đối tượng và phạm vi nghiên cứu:	1
4. Phương pháp nghiên cứu:	1
5. Cấu trúc báo cáo:	2
CHƯƠNG 1: TỔNG QUAN	3
CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT.....	4
2.1 Tổng quan về WPF.....	4
2.1.1 Khái niệm.....	4
2.1.2 Nền tảng thống nhất để xây dựng giao diện người dùng.....	4
2.1.3 Khả năng làm việc chung giữa người thiết kế giao diện và lập trình viên	6
2.1.4 Công nghệ chung cho giao diện trên Windows và trên trình duyệt Web	8
2.1.5 Điểm nổi bật	10
2.2 Mô hình MVVM với WPF.....	11
2.2.1 Tổng quan	11
2.2.2 Cấu trúc và nguyên lý hoạt động	11
2.2.3 Ưu điểm và khuyết điểm của MVVM	12
2.3 Tổng quan về Entity Framework.....	12
2.3.1 Tổng quan	12
2.3.2 Lịch sử	13
2.3.3 Kiến trúc	14
2.3.4 Các tình huống sử dụng Entity Framework	16
CHƯƠNG 3: ĐÁNH GIÁ KẾT QUẢ.....	18
3.1 Mô tả đề tài.....	18
3.2 Yêu cầu.....	18
3.3 Mô hình dữ liệu.....	19
3.4 Mô hình xử lý.....	20
3.5 Giao diện	25
3.5.1 Thiết kế màn hình đăng nhập.....	25
3.5.2 Thiết kế giao diện chính.....	26
3.5.3 Thiết kế màn hình nhập dữ liệu	26
3.6 Kết quả thử nghiệm	27
CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	29

4.1	Kết luận	29
4.1.1	Kết quả đạt được	29
4.1.2	Hạn chế	29
4.2	Hướng phát triển	29
DANH MỤC TÀI LIỆU THAM KHẢO.....		30

DANH MỤC HÌNH ẢNH – BẢNG BIỂU

Bảng biểu

<i>Bảng 1 Công nghệ tích hợp vào WPF.....</i>	<i>5</i>
<i>Bảng 2: Đặc tả Use case đăng nhập.....</i>	<i>20</i>
<i>Bảng 3: Đặc tả Use case tìm kiếm</i>	<i>21</i>
<i>Bảng 4: Đặc tả Use case Thêm thông tin Khách hàng</i>	<i>22</i>
<i>Bảng 5: Đặc tả Use case Xóa thông tin Khách hàng.....</i>	<i>23</i>
<i>Bảng 6: Đặc tả Use case Sửa thông tin Khách hàng.....</i>	<i>24</i>

Hình ảnh

<i>Hình 1: Công dụng của mã Xaml.....</i>	<i>7</i>
<i>Hình 2:Minh họa một ứng dụng dịch vụ tài chính trên WPF</i>	<i>9</i>
<i>Hình 3:Minh họa trên IE.....</i>	<i>9</i>
<i>Hình 4: Mô hình MVVM</i>	<i>12</i>
<i>Hình 5: Entity Framework</i>	<i>13</i>
<i>Hình 6: Mô hình Entity Framework.....</i>	<i>16</i>
<i>Hình 7: DatabaseFirst</i>	<i>17</i>
<i>Hình 8: CodeFirst</i>	<i>17</i>
<i>Hình 9: ModelFirst</i>	<i>17</i>
<i>Hình 10: Mô hình dữ liệu quản lý khách hàng.....</i>	<i>19</i>
<i>Hình 11: Mô hình Use Case quản lý khách hàng.....</i>	<i>20</i>
<i>Hình 12: Màn hình đăng nhập.....</i>	<i>25</i>
<i>Hình 13: Giao diện chính.....</i>	<i>26</i>
<i>Hình 14: Giao diện nhập liệu thông tin khách hàng.....</i>	<i>26</i>
<i>Hình 15: Thông báo lỗi đăng nhập.....</i>	<i>27</i>
<i>Hình 16: Chỉnh sửa thông tin khách hàng.....</i>	<i>28</i>

<i>Hình 17: Thêm mới giao dịch</i>	28
--	----

TÓM TẮT ĐỒ ÁN CHUYÊN NGÀNH

- **Vấn đề nghiên cứu**

Tập trung nghiên cứu WPF , mô hình MVVM trong WPF cùng với Entity Framework và các kiến thức cơ sở, nghiệp vụ để xây dựng chương trình quản lý khách hàng cho siêu thị VinMart Trà Vinh.

- **Các hướng tiếp cận**

- Tìm kiếm tài liệu, phân tích và tổng hợp lý thuyết: WPF, mô hình MVVM, Entity Framework...

- Tìm hiểu quy trình quản lý khách hàng của siêu thị VinMart Trà Vinh.

- **Cách giải quyết vấn đề**

- Tìm hiểu về WPF, điểm mạnh và tính ứng dụng như thế nào so với Window Form để xây dựng ứng dụng. Cùng với đó là việc ứng dụng mô hình MVVM trong phát triển ứng dụng, tạo sự chuyên nghiệp và quản lý code cũng như bảo trì sửa chữa sao này. Đi cùng với đó là Entity Framework để tương tác với dữ liệu đi cùng với phát triển lâu dài tiết kiệm được nhiều thời gian.

- Tiếp theo tìm hiểu quy trình quản lý khách hàng, thiết kế một giao diện phù hợp để dùng đi đôi với đầy đủ chức năng yêu cầu nghiệp vụ cho việc quản lý như quản lý thông tin khách hàng, quản lý giao dịch...

- **Các kết quả đạt được**

- Xây dựng được chương trình quản lý khách hàng.

- Quản lý được thông tin các nhân khách hàng

- Quản lý thông tin giao dịch của khách hàng.

- Quản lý điểm tích lũy và ưu đãi dành cho khách hàng.

- Giao diện hiện đại dễ sử dụng.

- Thống kê được số lượng khách hàng, số lượt giao dịch và tổng doanh thu.

- Xuất danh sách khách hàng ra file excel.

MỞ ĐẦU

1. Lý do chọn đề tài:

Ngày nay cùng với đà phát triển kinh tế của đất nước, đi cùng với sự phát triển về mọi mặt trong đời sống, thì việc ứng dụng công nghệ thông tin vào các lĩnh vực đời sống hằng ngày đang ngày càng phổ biến và cần thiết.

Ngành bán lẻ ở Việt Nam phát triển nhanh đi cùng với lượng khách hàng vô cùng lớn chính vì điều đó xây dựng một ứng dụng quản lý khách hàng cho siêu thị nói chung và siêu thị VinMart Trà Vinh nói riêng cho nên việc quản lý khách hàng là hết sức cần thiết, bằng kiến thức đã học và tìm hiểu em quyết định chọn đề tài chuyên ngành “ *Ứng dụng WPF để xây dựng chương trình quản lý khách hàng*” nhằm xây dựng một ứng dụng quản lý khách hàng cho siêu thị VinMart Trà Vinh.

2. Mục tiêu nghiên cứu:

Xây dựng ứng dụng quản lý khách hàng cho siêu thị VinMart Trà Vinh giúp cho việc quản lý thông tin khách hàng, thông tin giao dịch.. dễ dàng hơn, truy xuất nhanh chóng hiệu quả hơn, giảm thời gian làm việc cũng như chi phí so với làm thủ công.

3. Đối tượng và phạm vi nghiên cứu:

- Đối tượng nghiên cứu:

-Công nghệ WPF: Khái niệm, đặc điểm nổi bật, ưu khuyết điểm,...

- Hệ thống quản lý khách hàng: Tìm hiểu quy trình hoạt động quản lý khách hàng của siêu thị VinMart Trà Vinh để xây dựng ứng dụng.

- Phạm vi nghiên cứu:

Nghiên cứu thiết kế và xây dựng hệ thống quản lý dịch vụ khách hàng cho siêu thị VinMart Trà Vinh.

4. Phương pháp nghiên cứu:

- Tìm kiếm tài liệu, phân tích và tổng hợp lý thuyết: WPF, mô hình MVVM, Entity Framework...
- Khảo sát : Quy trình quản lý khách hàng của siêu thị VinMart Trà Vinh.
- Thực nghiệm: Xây dựng hệ thống minh họa cho vấn đề nghiên cứu.

5. Cấu trúc báo cáo:

Cấu trúc báo cáo gồm các chương :

Chương 1: Tổng quan

Chương 2: Nghiên cứu lý thuyết

Chương 3: Đánh giá kết quả

Chương 4: Kết luận và hướng phát triển

CHƯƠNG 1: TỔNG QUAN

Tài liệu giới thiệu tổng quan về WPF , mô hình MVVM trong WPF cùng với Entity Framework và các kiến thức cơ sở để thực hiện xây dựng chương trình quản lý khách hàng cho siêu thị VinMart Trà Vinh.

Ở các chương đầu tiên là phần giới thiệu tổng quan về WPF, điểm mạnh và tính ứng dụng như thế nào so với Window Form để xây dựng ứng dụng. Cùng với đó là việc ứng dụng mô hình MVVM trong phát triển ứng dụng, tạo sự chuyên nghiệp và quản lý code cũng như bảo trì sửa chữa sao này. Đi cùng với đó là Entity Framework để tương tác với dữ liệu đi cùng với phát triển lâu dài tiết kiệm được nhiều thời gian.

Tiếp theo đến với quy trình quản lý khách hàng, thiết kế một giao diện phù hợp để dùng đi đôi với đầy đủ chức năng yêu cầu nghiệp vụ cho việc quản lý như quản lý thông tin khách hàng, quản lý giao dịch...

Ở phần cuối sẽ là phần đánh giá kết quả, nêu ra những kết quả đạt được cũng như những khó khăn gặp phải khi xây dựng chương trình.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1 Tổng quan về WPF

2.1.1 Khái niệm

WPF, viết tắt của Windows Presentation Foundation, là hệ thống API mới hỗ trợ việc xây dựng giao diện đồ họa trên nền Windows. Được xem như thế hệ kế tiếp của WinForms, WPF tăng cường khả năng lập trình giao diện của lập trình viên bằng cách cung cấp các API cho phép tận dụng những lợi thế về đa phương tiện hiện đại. Là một bộ phận của .NET Framework 3.0, WPF sẵn có trong Windows Vista và Windows Server 2008. Đồng thời, WPF cũng có thể hoạt động trên nền Windows XP Service Pack 2 hoặc mới hơn, và cả Windows Server 2003.

WPF được xây dựng nhằm vào ba mục tiêu cơ bản:

- Cung cấp một nền tảng thống nhất để xây dựng giao diện người dùng;
- Cho phép người lập trình và người thiết kế giao diện làm việc cùng nhau một cách dễ dàng;
- Cung cấp một công nghệ chung để xây dựng giao diện người dùng trên cả Windows và trình duyệt Web.

2.1.2 Nền tảng thống nhất để xây dựng giao diện người dùng

Trước khi WPF ra đời, việc tạo giao diện người dùng theo những yêu cầu mô tả ở ví dụ trên đòi hỏi sử dụng rất nhiều công nghệ khác nhau. Để tạo form, các control và các tính năng kinh điển khác của một giao diện đồ họa Windows, thông thường lập trình viên sẽ chọn Windows Forms, một phần của .NET Framework. Nếu cần hiển thị văn bản, Windows Forms có một số tính năng hỗ trợ văn bản trực tiếp hoặc có thể sử dụng Adobe's PDF để hiển thị văn bản có khuôn dạng cố định.

Đối với hình ảnh và đồ họa 2 chiều, lập trình viên sẽ dùng GDI+, một mô hình lập trình riêng biệt có thể truy nhập qua Windows Forms. Để hiển thị video hay phát âm thanh, lập trình viên lại phải sử dụng Windows Media Player, và với đồ họa 3 chiều, anh ta lại phải dùng Direct3D, một thành phần chuẩn khác của Windows. Tóm lại, quá trình phát triển giao diện người dùng theo yêu cầu trở nên phức tạp, đòi hỏi lập trình viên quá nhiều kỹ năng công nghệ.

Bảng 1 Công nghệ tích hợp vào WPF

	Windows Forms	PDF	Windows Forms/ GDI+	Windows Media Player	Direct3D	WPF
Giao diện đồ họa (form và các control)	x					x
On-screen văn bản	x					x
Fixed-format văn bản		x				x
Hình ảnh			x			x
Video và âm thanh				x		x
Đồ họa 2 chiều			x			x
Đồ họa 3 chiều					x	x

WPF là giải pháp hợp nhất nhằm giải quyết tất cả những vấn đề công nghệ nêu trên, hay nói cách khác, WPF cung cấp nhiều tính năng lập trình giao diện trong cùng một công nghệ đơn nhất. Điều này giúp cho quá trình tạo giao diện người dùng trở nên dễ dàng hơn đáng kể.

Hình bên dưới cho thấy một giao diện quản lý và theo dõi bệnh nhân có sự kết hợp của hình ảnh, text, đồ họa 2 chiều/3 chiều và nhiều thông tin trực quan khác. Tất cả đều được tạo ra bằng WPF – lập trình viên không cần viết code để sử dụng các công nghệ chuyên biệt như GDI+ hay Direct3D.

Tuy nhiên, WPF ra đời không có nghĩa là tất cả những công nghệ nêu trên bị thay thế. Windows Forms vẫn có giá trị, thậm chí trong WPF, một số ứng dụng mới vẫn sẽ sử dụng Windows Forms. Windows Media Player vẫn đóng một vai trò công cụ độc lập để chơi nhạc và trình chiếu video. PDF cho văn bản vẫn tiếp tục được sử dụng. Direct3D vẫn là công nghệ quan trọng trong games và các dạng ứng dụng khác (Trong thực tế, bản thân WPF dựa trên Direct3D để thực hiện mọi biểu diễn đồ họa).

Việc tạo ra một giao diện người dùng hiện đại không chỉ là việc hợp nhất các công nghệ sẵn có khác nhau. Nó còn thể hiện ở việc tận dụng lợi điểm của card đồ

họa hiện đại. Để giải phóng những hạn chế của đồ họa bitmap, WPF dựa hoàn toàn trên đồ họa vector, cho phép hình ảnh tự động thay đổi kích thước để phù hợp với kích thước và độ phân giải của màn hình mà nó được hiển thị.

Bằng việc hợp nhất tất cả các công nghệ cần thiết để tạo ra một giao diện người dùng vào một nền tảng đơn nhất, WPF đơn giản hóa đáng kể công việc của lập trình viên giao diện. Với việc yêu cầu lập trình viên học một môi trường phát triển duy nhất, WPF góp phần làm giảm chi phí cho việc xây dựng và bảo trì ứng dụng. Và bằng việc cho phép tích hợp đa dạng nhiều cách biểu diễn thông tin trên giao diện người dùng, WPF góp phần nâng cao chất lượng, và theo đó là giá trị công việc, của cách thức người dùng tương tác với ứng dụng trên Windows.

2.1.3 Khả năng làm việc chung giữa người thiết kế giao diện và lập trình viên

Trong thực tế, việc xây dựng một giao diện người dùng phức hợp như trong ví dụ về ứng dụng quản lý bệnh nhân trên đòi hỏi những kỹ năng ít thấy ở những lập trình viên đơn thuần, mà chỉ có thể tìm thấy ở những người thiết kế giao diện chuyên nghiệp.

Thông thường, người thiết kế giao diện sử dụng một công cụ đồ họa để tạo ra những ảnh tĩnh về cách bố trí giao diện trên màn hình. Những hình ảnh này sau đó được chuyển tới lập trình viên với nhiệm vụ tạo ra mã trình để hiện thực hóa giao diện đã thiết kế. Đôi lúc vẽ ra một giao diện thì đơn giản với người thiết kế, nhưng để biến nó thành hiện thực có thể là khó khăn hoặc bất khả thi với lập trình viên.

Hạn chế về công nghệ, sức ép tiến độ, thiếu kỹ năng, hiểu nhầm hoặc đơn giản là bất đồng quan điểm có thể khiến lập trình viên không đáp ứng được đầy đủ yêu cầu từ người thiết kế. Do vậy, điều cần thiết ở đây là một cách thức để hai nhóm công tác độc lập này có thể làm việc với nhau mà không làm thay đổi chất lượng của giao diện đã thiết kế.

Để thực hiện được điều này, WPF đưa ra ngôn ngữ đặc tả eXtensible Application Markup Language (XAML). XAML định ra một tập các phần tử XML như Button, TextBox, Label,... Nhằm định nghĩa các đối tượng đồ họa tương ứng như nút bấm, hộp thoại, nhãn... Và nhờ đó cho phép mô tả chính xác diện mạo của

giao diện người dùng. Các phần tử XAML cũng chứa các thuộc tính, cho phép thiết lập nhiều tính chất khác nhau của đối tượng đồ họa tương ứng.

Ví dụ, đoạn mã sau sẽ tạo ra một nút bấm màu đỏ “Click me”:

```
<Button x:Name=”btn” Content=”Click me”/>
```

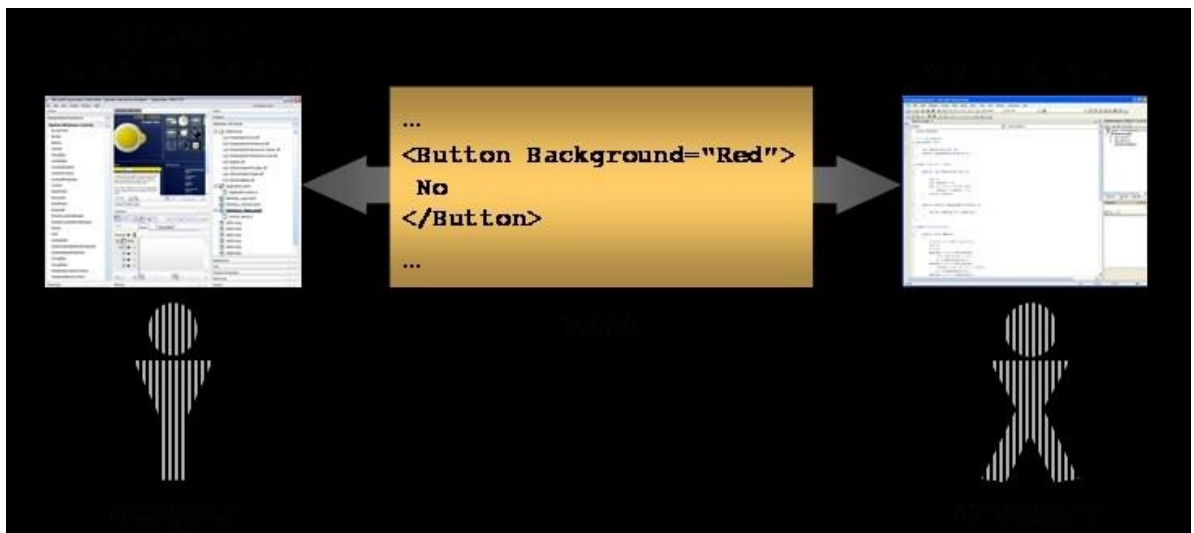
Mỗi phần tử XAML lại tương ứng với một lớp WPF, và mỗi thuộc tính của phần tử đó lại tương ứng với thuộc tính hay sự kiện của lớp này. Chẳng hạn, nút bấm màu đỏ trong ví dụ trên có thể tạo bằng C# code như sau:

```
Button btn = new Button();
```

```
btn.Background = Brushes.Red;
```

```
btn.Content = "Click me";
```

Nếu như mọi thứ có thể biểu diễn bằng XAML thì cũng có thể biểu diễn bằng đoạn mã, thì câu hỏi đặt ra là XAML có ý nghĩa gì? Câu trả lời là việc xây dựng các công cụ sinh và sử dụng các đặc tả bằng XML dễ dàng hơn nhiều so với xây dựng một công cụ tương tự làm việc với đoạn mã. Bởi vậy, XAML mở ra một cách thức tốt hơn để lập trình viên và người thiết kế làm việc với nhau. Hình dưới đây minh họa quá trình này.



Hình 1: Công dụng của mã Xaml

Người thiết kế có thể mô tả giao diện người dùng và tương tác với nó thông qua một công cụ, chẳng hạn như Microsoft Expression Interactive Designer. Chỉ tập trung vào việc định ra diện mạo và cảm quan cho giao diện đồ họa WPF, công cụ này sinh các đoạn mô tả giao diện thể hiện qua ngôn ngữ XAML. Lập trình viên

sau đó sẽ nhập đoạn mô tả XAML đó vào môi trường lập trình, chẳng hạn như Microsoft Visual Studio.

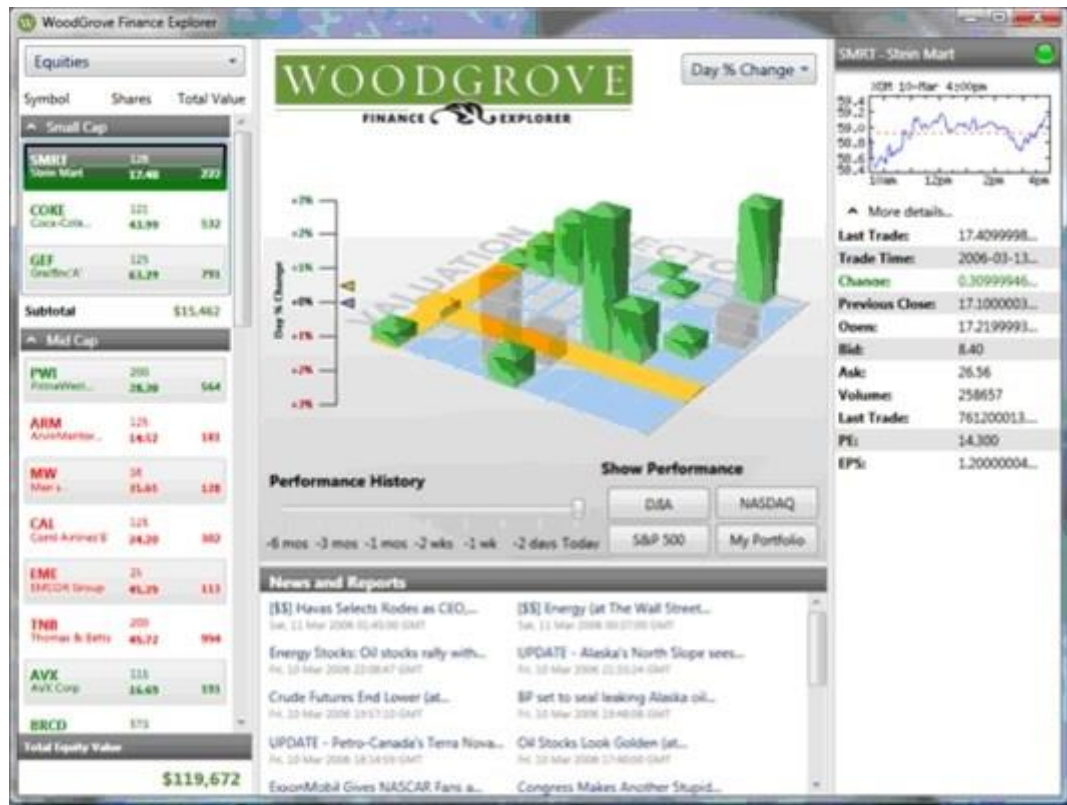
Thay vì lập trình viên phải tái tạo lại giao diện từ đầu dựa trên một ảnh tĩnh mà người thiết kế cung cấp, bản thân các đoạn XAML này sẽ được Microsoft Visual Studio biên dịch để tái tạo thành giao diện đồ họa đúng theo mô tả. Lập trình viên chỉ tập trung vào việc viết mã trình cho giao diện được sinh ra, chẳng hạn như xử lý các sự kiện, theo những chức năng đề ra của ứng dụng.

Việc cho phép người thiết kế và lập trình viên làm việc chung như vậy sẽ hạn chế những lỗi phát sinh khi hiện thực hóa giao diện từ thiết kế. Thêm vào đó, nó còn cho phép hai nhóm công tác này làm việc song song, khiến mỗi bước lặp trong quy trình phát triển phần mềm ngắn đi và việc phản hồi được tốt hơn. Vì cả hai môi trường đều có khả năng hiểu và sử dụng XAML, ứng dụng WPF có thể chuyển qua lại giữa hai môi trường phát triển để sửa đổi hay bổ sung giao diện. Với tất cả những lợi điểm này, vai trò của người thiết kế trong việc xây dựng giao diện được đặt lên hàng đầu.

2.1.4 Công nghệ chung cho giao diện trên Windows và trên trình duyệt Web

. Trong thời đại bùng nổ của Internet, các ứng dụng Web ngày một phát triển. Việc trang bị giao diện người dùng với đầy đủ tính năng như một ứng dụng desktop sẽ thu hút nhiều người sử dụng, và do đó góp phần làm tăng giá trị doanh nghiệp. Tuy nhiên, như đã nêu trong phần đầu, với những công nghệ truyền thống, để phát triển một giao diện đồ họa vừa hoạt động trên desktop vừa trên trình duyệt Web, đòi hỏi phải sử dụng những công nghệ hoàn toàn khác nhau, giống như việc xây dựng hai giao diện hoàn toàn độc lập. Điều này tạo ra chi phí không cần thiết để phát triển giao diện.

WPF là một giải pháp cho vấn đề này. Lập trình viên có thể tạo ra một ứng dụng trình duyệt XAML (XBAP) sử dụng WPF chạy trên Internet Explore. Trên thực tế, cùng đoạn code này có thể được dùng để sinh ứng dụng WPF chạy độc lập trên Windows. *Hình 2* minh họa một ứng dụng dịch vụ tài chính hoạt động như một ứng dụng WPF độc lập. Trong khi đó, *Hình 3* minh họa giao diện của cùng ứng dụng chạy trên Internet Explore dưới dạng XBAP.



Hình 2: Minh họa một ứng dụng dịch vụ tài chính trên WPF



Hình 3: Minh họa trên IE

Như đã thấy trong Hình 3, phần giao diện của ứng dụng dạng XBAP được trình duyệt chia thành các frame thay vì chạy trên các cửa sổ riêng, ngoài ra, các chức năng đều được bảo toàn. Cùng một đoạn mã được sử dụng chung cho cả hai

trường hợp sẽ làm giảm khối lượng công việc cần thiết để phát triển hai dạng giao diện. Thêm vào đó, sử dụng cùng một đoạn mã cũng có nghĩa là sử dụng cùng kỹ năng của lập trình viên.

Do đó, lập trình viên chỉ cần có học một kiến thức chung là có thể sử dụng trong cả hai trường hợp. Một lợi điểm nữa của việc dùng chung công nghệ cho cả giao diện Windows và giao diện Web là người xây dựng ứng dụng không nhất thiết phải quyết định trước loại giao diện nào được sử dụng. Miễn là máy client đáp ứng được những yêu cầu hệ thống để chạy XBAP, một ứng dụng có thể cung cấp cả giao diện Windows và giao diện Web, mà chỉ sử dụng phần lớn những đoạn mã giống nhau.

Mỗi ứng dụng XBAP được download khi cần từ một Web server, nên nó phải tuân theo những yêu cầu về an ninh khắt khe hơn đối với một ứng dụng Windows độc lập. Theo đó, XBAP chạy trong phạm vi sandbox an ninh do hệ thống an ninh truy nhập mã của .NET Framework cung cấp. XBAP chỉ chạy với các hệ thống Windows có cài đặt WPF và chỉ với Internet Explore phiên bản 6 và 7 trở lên.

2.1.5 Điểm nổi bật

- Nó mới hơn và do đó phù hợp hơn với các tiêu chuẩn hiện tại.
- Microsoft đang sử dụng nó cho rất nhiều ứng dụng mới, ví dụ: Visual Studio
- Nó linh hoạt hơn, vì vậy bạn có thể làm nhiều việc hơn mà không phải viết hoặc mua các control mới.
- Khi bạn cần sử dụng các control của bên thứ 3, các nhà phát triển các control này có thể sẽ tập trung hơn vào WPF vì nó mới hơn.
- XAML giúp dễ dàng tạo và chỉnh sửa GUI của bạn và cho phép công việc được phân chia giữa một nhà thiết kế (XAML) và một lập trình viên (C #, VB.NET, v.v.).
- Databinding, cho phép bạn có được một sự tách biệt hơn giữa data và layout.
- Sử dụng tăng tốc phần cứng để vẽ GUI, để có hiệu suất tốt hơn.

- Nó cho phép bạn tạo giao diện người dùng cho cả ứng dụng Windows và các ứng dụng web (Silverlight / XBAP).

2.2 Mô hình MVVM với WPF

2.2.1 Tổng quan

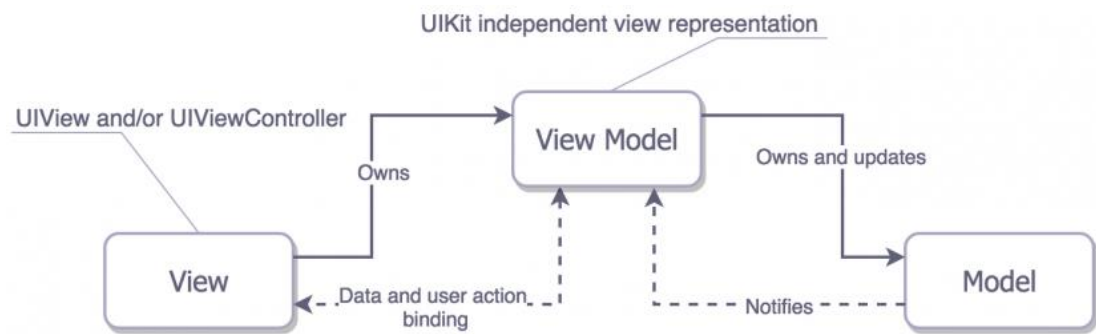
Kể từ khi Microsoft giới thiệu hai nền tảng phát triển ứng dụng mới là WPF và Silverlight, đã có nhiều thay đổi trong việc xử lý sự kiện và binding dữ liệu, giữa các tầng của ứng dụng với nhau. Qua đó, hầu hết các công việc của tầng kết hợp với lớp presentation. Điều này làm nảy sinh ra nhu cầu phải có một mô hình phát triển ứng dụng mới phù hợp hơn. Và do đó, Model – View – ViewModel (MVVM) pattern ra đời và ngày càng trở nên phổ biến.

Đa số các ứng dụng thuộc bất kì nền tảng nào cũng có thể chia thành hai phần: giao diện (View) và dữ liệu (Model). Vì việc tách riêng các phần này, cần phải có một phần trung gian nào đó nối kết hai phần này lại, và chúng tạo nên một mô hình (pattern).

2.2.2 Cấu trúc và nguyên lý hoạt động

- **View:** Tương tự như trong mô hình MVC, View là phần giao diện của ứng dụng để hiển thị dữ liệu và nhận tương tác của người dùng. Một điểm khác biệt so với các ứng dụng truyền thống là View trong mô hình này tích cực hơn. Nó có khả năng thực hiện các hành vi và phản hồi lại người dùng thông qua tính năng binding, command.
- **Model:** Cũng tương tự như trong mô hình MVC. Model là các đối tượng giúp truy xuất và thao tác trên dữ liệu thực sự.
- **ViewModel:** Lớp trung gian giữa View và Model. ViewModel có thể được xem là thành phần thay thế cho Controller trong mô hình MVC. Nó chứa các mã lệnh cần thiết để thực hiện data binding, command.

Một điểm cần lưu ý là trong mô hình MVVM, các tầng bên dưới sẽ không biết được các thông tin gì về tầng bên trên nó. Như hình minh họa dưới đây:



Hình 4: Mô hình MVVM

2.2.3 Ưu điểm và khuyết điểm của MVVM

- Ưu điểm
 - Tốt cho kiểm thử.
 - Tốt cho bảo trì.
 - Khả năng mở rộng, tái sử dụng.
- Khuyết điểm
 - Thiết kế ViewModel mất thời gian và khó khăn hơn.
 - Debug khó khi dữ liệu phức tạp.

2.3 Tổng quan về Entity Framework

2.3.1 Tổng quan

Entity Framework là một tập hợp các công nghệ trong ADO.NET hỗ trợ phát triển các ứng dụng phần mềm hướng dữ liệu. Các kiến trúc sư và nhà phát triển các ứng dụng định hướng dữ liệu thường phải vật lộn với nhu cầu đạt được hai mục tiêu rất khác nhau. Họ phải mô hình hóa các thực thể, mối quan hệ và logic của các vấn đề kinh doanh mà họ đang giải quyết và họ cũng phải làm việc với các công cụ dữ liệu được sử dụng để lưu trữ và truy xuất dữ liệu. Dữ liệu có thể trải rộng trên nhiều hệ thống lưu trữ, mỗi hệ thống có các giao thức riêng; ngay cả các ứng dụng hoạt động với một hệ thống lưu trữ duy nhất cũng phải cân bằng các yêu cầu của hệ thống lưu trữ so với các yêu cầu viết mã ứng dụng hiệu quả và có thể bảo trì.

Entity Framework cho phép các nhà phát triển làm việc với dữ liệu dưới dạng các đối tượng và thuộc tính của miền cụ thể, chẳng hạn như khách hàng và địa chỉ khách hàng, mà không phải quan tâm đến các bảng và cột cơ sở dữ liệu cơ bản

nơi dữ liệu này được lưu trữ. Với Entity Framework, các nhà phát triển có thể làm việc ở mức độ trừu tượng cao hơn khi họ xử lý dữ liệu và có thể tạo và duy trì các ứng dụng hướng dữ liệu với ít mã hơn trong các ứng dụng truyền thống.



Hình 5: Entity Framework

2.3.2 Lịch sử

Phiên bản đầu tiên của Entity Framework (EFv1) được bao gồm với .NET Framework 3.5 Service Pack 1 và Visual Studio 2008 Service Pack 1, được phát hành vào ngày 11 tháng 8 năm 2008. Phiên bản này đã bị chỉ trích rộng rãi, thậm chí còn thu hút một 'phiếu bầu không tin cậy' được ký bởi khoảng một nghìn nhà phát triển.

Phiên bản thứ hai của Entity Framework, được đặt tên là Entity Framework 4.0 (EFv4), đã được phát hành như một phần của .NET 4.0 vào ngày 12 tháng 4 năm 2010 và giải quyết nhiều lời chỉ trích về phiên bản 1.

Phiên bản thứ ba của Entity Framework, phiên bản 4.1, được phát hành vào ngày 12 tháng 4 năm 2011, với sự hỗ trợ của Code First .

Bản làm mới của phiên bản 4.1, có tên Entity Framework 4.1 Update 1, được phát hành vào ngày 25 tháng 7 năm 2011. Nó bao gồm sửa lỗi và các loại được hỗ trợ mới.

Phiên bản 4.3.1 được phát hành vào ngày 29 tháng 2 năm 2012. Có một vài cập nhật, như hỗ trợ cho việc di chuyển.

Phiên bản 5.0.0 được phát hành vào ngày 11 tháng 8 năm 2012 và được nhắm mục tiêu tại .NET framework 4.5. Ngoài ra, phiên bản này có sẵn cho .Net framework 4, nhưng không có bất kỳ lợi thế thời gian chạy nào so với phiên bản 4.

Phiên bản 6.0 được phát hành vào ngày 17 tháng 10 năm 2013 và hiện là một dự án nguồn mở được cấp phép theo Giấy phép Apache v2. Giống như ASP.NET MVC, mã nguồn của nó được lưu trữ tại GitHub bằng Git. Phiên bản này có một số cải tiến cho hỗ trợ mã đầu tiên.

Microsoft sau đó đã quyết định hiện đại hóa, thành phần hóa và đưa nền tảng chéo .NET lên Linux, OSX và các nơi khác, có nghĩa là phiên bản tiếp theo của Entity Framework sẽ được viết lại hoàn chỉnh. Vào ngày 27 tháng 6 năm 2016, nó được phát hành dưới dạng Entity Framework Core 1.0, cùng với ASP.Net Core 1.0 và .Net Core 1.0. Ban đầu nó được đặt tên là Entity Framework 7, nhưng được đổi tên để làm nổi bật rằng nó là một bản viết lại hoàn chỉnh thay vì nâng cấp gia tăng và nó không thay thế EF6.

EF Core 1.0 được cấp phép theo Giấy phép Apache v2 và đang được xây dựng hoàn toàn mở trên GitHub. Mặc dù EF Core 1.0 có một số điểm tương đồng về mặt khái niệm với các phiên bản trước của Entity Framework, nhưng đây là một cơ sở mã hoàn toàn mới được thiết kế để hiệu quả hơn, mạnh mẽ hơn, linh hoạt hơn và có thể mở rộng, sẽ chạy trên Windows, Linux và OSX và sẽ hỗ trợ một phạm vi mới lưu trữ dữ liệu quan hệ và NOSQL.

EF Core 2.0 được phát hành vào ngày 14 tháng 8 năm 2017 cùng với Visual Studio 2017 15.3 và ASP.NET Core 2.0.

2.3.3 Kiến trúc

Kiến trúc của Khung thực thể ADO.NET, từ dưới lên, bao gồm:

- **Application**

Application (ứng dụng) là tầng chứa giao diện trang Web (HTML, CSS, Javascript, hình ảnh, ...) và các đoạn mã nguồn (C#, VB) để tương tác dữ liệu với các tầng khác trong mô hình thông qua Object Services.

- **Object Services**

Object Services (tạm dịch là các dịch vụ đối tượng) là tầng chứa quá trình tương tác giữa ứng dụng và database, hay nói cách khác nó là nơi chủ yếu để truy cập dữ liệu từ database và trả ngược kết quả về giao diện. Object

Services cung cấp các tiện ích để truy vết các thay đổi và quản lý nhận dạng, đồng thời là các quan hệ và thay đổi ở database.

- EntityClient Data Provider

Đây là tầng cung cấp các kết nối, diễn dịch các truy vấn thực thể thành truy vấn nguồn dữ liệu (chuyển L2E – LINQ to Entity hay các truy vấn thực thể SQL thành truy vấn SQL), trả về data reader để EF dùng chuyển dữ liệu thực thể thành các đối tượng. Phần này kết nối ADO.NET Data Providers để gửi hoặc lấy dữ liệu từ database. Tầng này hoàn toàn khác với EDM (Entity Data Model) khi thực thi các truy vấn tương tự như cách thực hiện ở ADO.NET Provider. EntityClient Data Provider sử dụng ESQL (Entity SQL), một ngôn ngữ truy vấn độc lập dạng văn bản, tương tự như SQL.

- ADO.NET Data Providers

Đây là tầng thấp nhất để dịch các truy vấn L2E (LINQ to Entity) thông qua câu lệnh thành các câu lệnh SQL và thực thi các câu lệnh trong hệ thống DBMS (database management system – hệ quản lý dữ liệu) nào đó. Tầng này kết với database sử dụng ADO.NET.

- EDM (Entity Data Model)

EDM (tạm dịch là mô hình dữ liệu thực thể) chứa 3 phần chính: mô hình khái niệm (CSDL – Conceptual schema definition language), mô hình ánh xạ (MSL – mapping specification language) và mô hình lưu trữ (SSDL – store schema definition language). EDM khác với EntityClient Data Provider ở chỗ EDM sử dụng LINQ là ngôn ngữ truy vấn tương tác với database.

Lưu ý: Trong 1 dự án thực tế, bạn có thể sử dụng LINQ và ESQL (Entity SQL) cùng 1 lúc để đa dạng hóa cách tương tác cơ sở dữ liệu và tối ưu tốc độ lấy cơ sở dữ liệu trong 1 số trường hợp.

- Mô hình khái niệm (CSDL – Conceptual schema definition language)

Mô hình khái niệm chứa các lớp mô hình và mối quan hệ giữa các lớp này. Điều này để độc lập với mô hình quan hệ các bảng trong database.

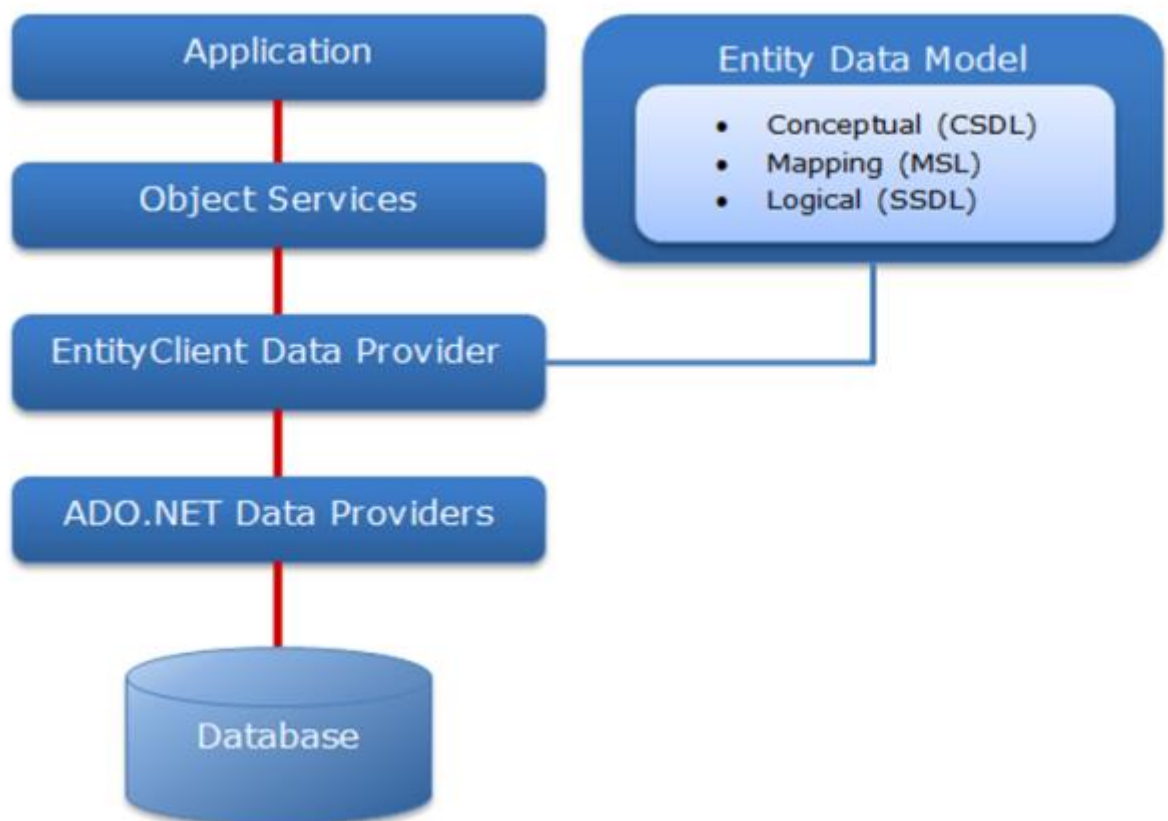
- Mô hình lưu trữ (SSDL – store schema definition language)

Mô hình lưu trữ là 1 mô hình thiết kế database bao gồm các bảng, view, stored procedure (thủ tục), và mối quan hệ giữa chúng và các khóa. Mô hình này thể hiện gần giống mô hình quan hệ các bảng trong database.

- Mô hình ánh xạ (MSL – mapping specification language)

Mô hình ánh xạ gồm thông tin về cách mô hình khái niệm được ánh xạ đến mô hình lưu trữ.

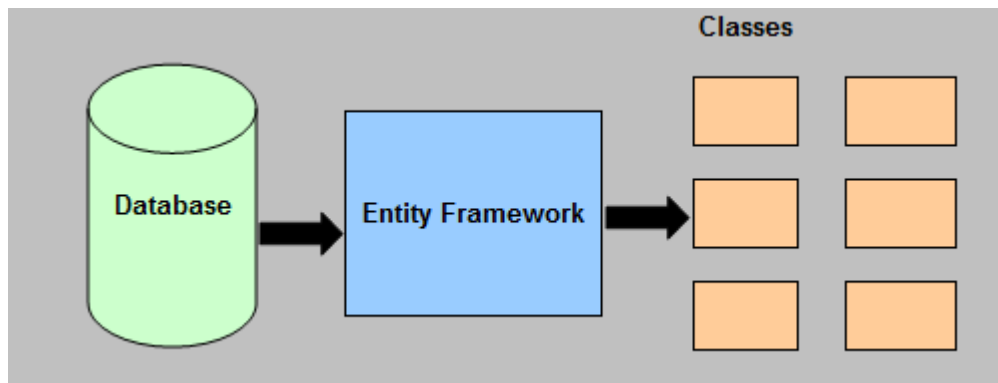
- L2E (LINQ to Entities): là 1 ngôn ngữ truy vấn được dùng để viết các truy vấn trái với mô hình đối tượng. L2E trả về các thực thể, được định nghĩa bởi mô hình khái niệm. Chúng ta có thể dùng LINQ trong ngôn ngữ này.



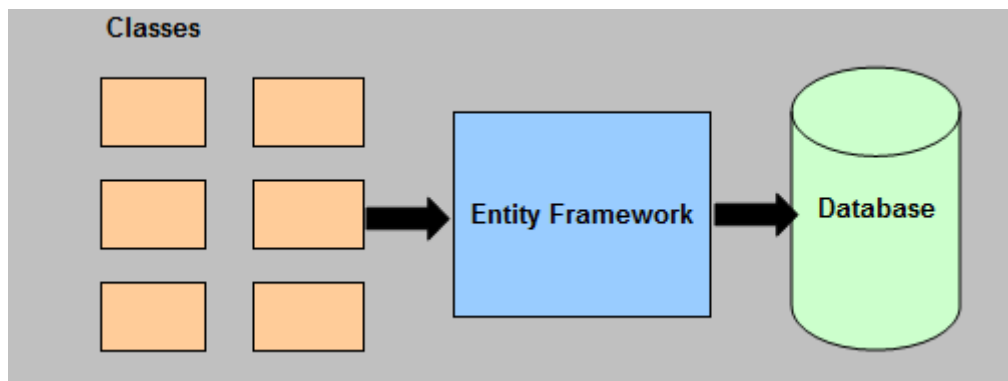
Hình 6: Mô hình Entity Framework

2.3.4 Các tình huống sử dụng Entity Framework

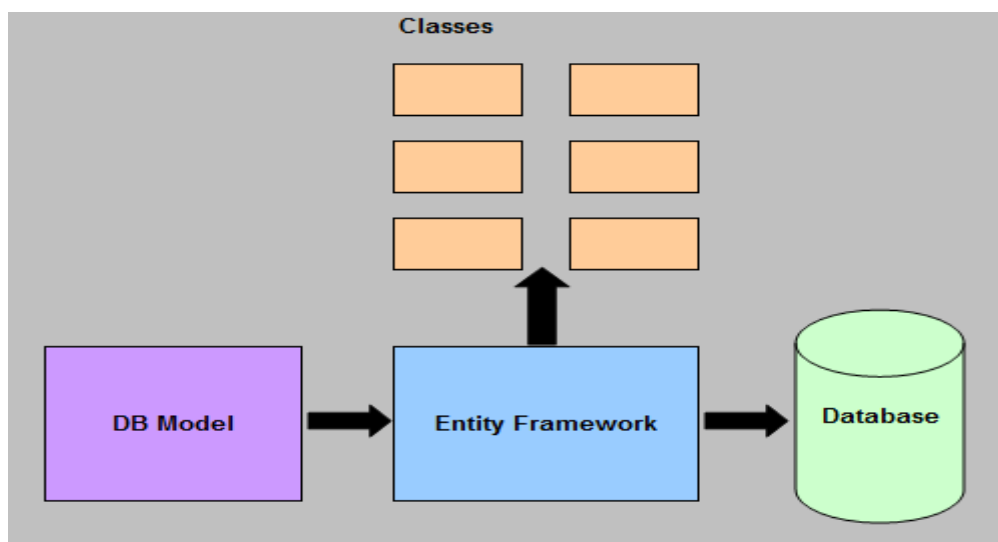
- Database First: Có sẵn một database, rồi sau đó sử dụng Entity Framework để tự động sinh ra các domain classes.

*Hình 7: DatabaseFirst*

- Code First: tạo domain classes, rồi sau đó dùng Entity Framework để tạo ra database.

*Hình 8: CodeFirst*

- Model First: Bạn tạo database schema trên visual designer, rồi sau đó dùng Entity Framework để tạo ra cả classes và database.

*Hình 9: ModelFirst*

CHƯƠNG 3: ĐÁNH GIÁ KẾT QUẢ

3.1 Mô tả đề tài

Quản lý khách hàng siêu thị VinMart Trà Vinh chỉ phát triển ở một module nhỏ trong hệ thống quản lý siêu thị, cho nên việc quản lý sẽ được thực hiện bởi người quản trị (nhân viên).

Khách hàng sẽ được lưu trữ thông tin vào hệ thống như thông tin cá nhân (họ tên, năm sinh, giới tính, số cmnd, địa chỉ, số điện thoại...), thông tin khách hàng đã giao dịch, thông tin điểm tích lũy của khách hàng, các ưu đãi mà khách hàng có.

Người sử dụng hệ thống có thể in danh sách khách hàng, thông tin chi tiết khách hàng và nhập thông tin giao dịch của khách hàng lên hệ thống bằng thông tin trên hóa đơn khách hàng đã mua.

Ngoài ra còn có các chức năng như thông kê số lượng khách hàng, số lượt giao dịch và tổng doanh thu.

3.2 Yêu cầu

- Yêu cầu chức năng:

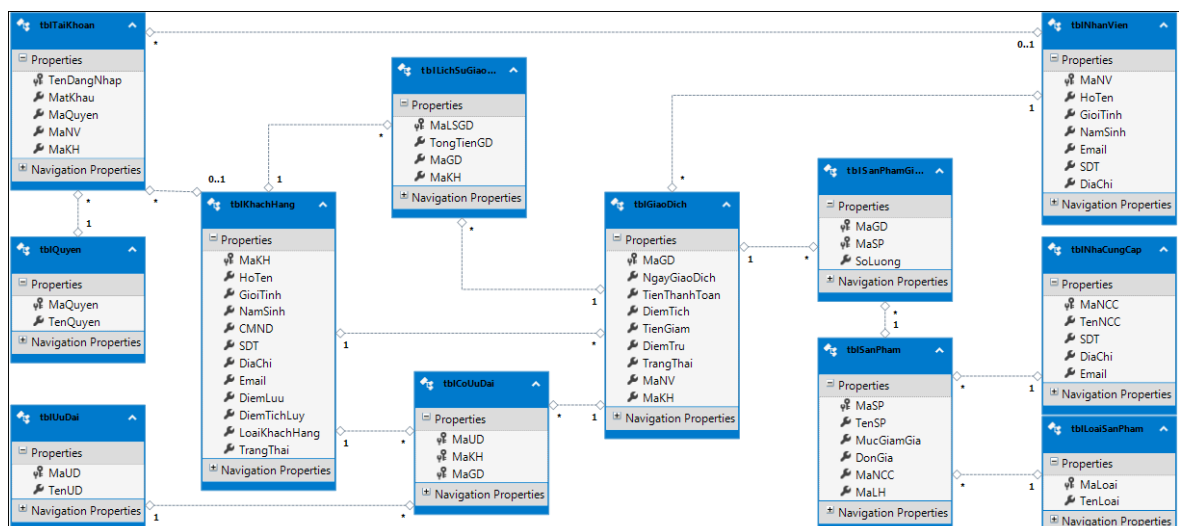
- Yêu cầu lưu trữ: lưu trữ thông tin khách hàng, thông tin điểm, thông tin giao dịch...
- Yêu cầu tra cứu: thông tin khách hàng, thông tin giao dịch,..
- Yêu cầu tìm kiếm: tìm kiếm khách hàng theo tên, số điện thoại, mã khách hàng..
- Yêu cầu kết xuất: xuất ra danh sách khách hàng dưới dạng file excel.

- Yêu cầu phi chức năng:

- Giao diện rõ ràng, dễ sử dụng.
- Hỗ trợ khách hàng và doanh nghiệp hiệu quả.
- Hạn chế chi phí về thời gian và nhân lực.

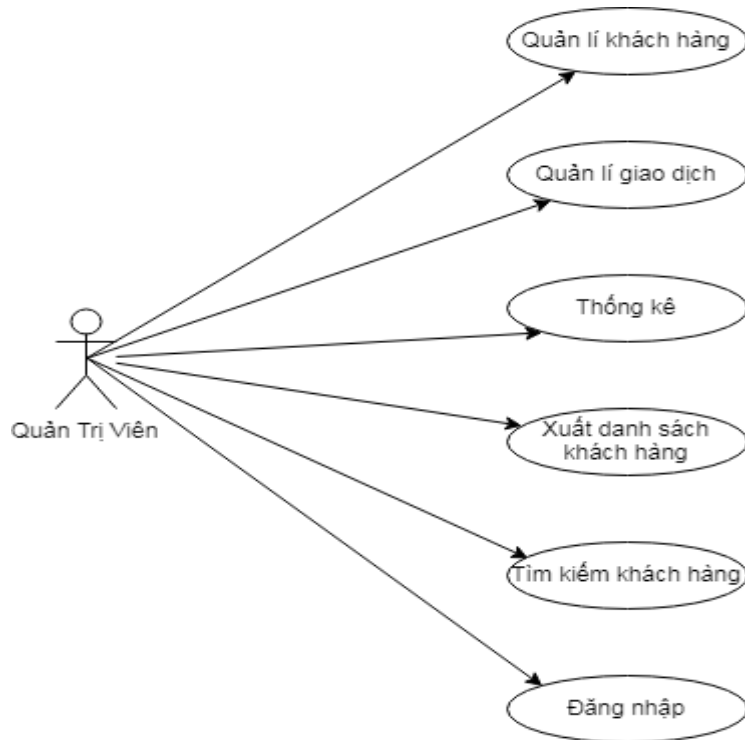
3.3 Mô hình dữ liệu

- **KHACHHANG**(MaKH, HoTen, GioiTinh, NamSinh, CMND, SDT, Email, DiaChi, DiemTichLuy, DiemTon, LoaiKhachHang, TrangThai).
- **NHANVIEN**(MaNV, HoTen, GioiTinh, NamSinh, CMND, SDT, Email, DiaChi).
- **GIAODICH**(MaGD, NgayGiaoDich, DiemTich, TienThanhToan, TienGiam, DiemTru, DiemTich, TrangThai, MaKH, MaNV).
- **LICHSUGIAODICH**(MaLSGD, TongTienGiaoDich, MaKH, MaNV).
- **SANPHAMGIAODICH**(MaGD, MaSP, SoLuong).
- **SANPHAM**(MaSP, TenSP, DonGia, MucGiamGia, MaLoai, MaNCC).
- **LOAISANPHAM**(MaLoai, TenLoai).
- **NHACUNGCAP**(MaNCC, TenNCC, DiaChi, SDT).
- **UUDAI**(MaUD, TenUuDai).
- **COUUDAI**(MaUD, MaKH, MaGD).
- **TAIKHOAN**(TenDangNhap, MatKhai, MaQuyen, MaKH, MaNV).
- **QUYEN**(MaQuyen, TenQuyen).



Hình 10: Mô hình dữ liệu quản lý khách hàng

3.4 Mô hình xử lí



Hình 11: Mô hình Use Case quản lí khách hàng

Đặc tả Use Case quản lí khách hàng:

- Use case Đăng nhập
 - Tên use case: Đăng nhập
 - Tác nhân: Người quản trị
 - Mục tiêu: Đăng nhập để sử dụng hệ thống
 - Mô tả: Người quản trị cần đăng nhập vào hệ thống để sử dụng các chức năng của hệ thống hỗ trợ.
 - Kịch bản:

Bảng 2: Đặc tả Use case đăng nhập

Hành động Actor	Phản ứng hệ thống
1. Mở form đăng nhập	2. Form đăng nhập được mở
3. Nhập thông tin tài khoản (username &	

password)	
4. Click button đăng nhập	5. Kiểm tra thông tin đăng nhập 6. Thông báo đăng nhập thành công.

Tình huống thay thế:

- Bước 5-6: Kiểm tra thông tin đăng nhập không chính xác, thông báo đăng nhập không thành công.
- **Use case tìm kiếm thông tin Khách hàng**
 - Tên use case: Tìm kiếm thông tin
 - Tác nhân: Khách hàng, Người quản trị
 - Mục tiêu: Tìm kiếm thông tin Khách hàng, Tìm kiếm hợp đồng.
 - Mô tả: Sử dụng chức năng tìm kiếm để tìm kiếm những thông tin cần thiết.
 - Kịch bản:

Bảng 3: Đặc tả Use case tìm kiếm

Hành động Actor	Phản ứng hệ thống
1. Mở form tìm kiếm 3. Nhập thông tin cần tìm 4. Nhấn nút tìm	2. Form tìm kiếm được mở 5. Kiểm tra thông tin trong CSDL 6. Hiện thị kết quả tìm kiếm.

Tình huống thay thế:

- Bước 5-6: Kiểm tra thông tin cần tìm kiếm không có trong CSDL, thông báo không có thông tin cần tìm.

5.1.3 Use case Thêm khách hàng

- Tên use case: Thêm Khách hàng.
- Tác nhân: Người quản trị
- Mục tiêu: Thêm thông tin của Khách hàng đến thuê xe
- Mô tả: Người quản trị thêm vào cơ sở dữ liệu thông tin Khách hàng đến thuê xe.
- Kịch bản:

Bảng 4: Đặc tả Use case Thêm thông tin Khách hàng

Hành động Actor	Phản ứng hệ thống
1. Mở form Thêm thông tin	2. Form thêm thông tin được mở
3. Nhập thông tin Khách hàng	
	4. Kiểm tra thông tin hợp lệ
	5. Thông báo kết quả thêm thành công.

Tình huống thay thế:

- Bước 4-5: Kiểm tra thông tin Khách hàng không hợp lệ, thông báo thêm không thành công!

5.1.4 Use case Xóa khách hàng

- Tên use case: Xóa Khách hàng.
- Tác nhân: Người quản trị
- Mục tiêu: Xóa thông tin của Khách hàng nhập sai

- Mô tả: Khi Người quản trị nhập sai thông tin khách hàng thì tiến hành xóa thông tin khách hàng.
- Kịch bản:

Bảng 5: Đặc tả Use case Xóa thông tin Khách hàng

Hành động Actor	Phản ứng hệ thống
1. Mở form Xóa thông tin	2. Form Xóa thông tin được mở
3. Sử dụng lại Use case tìm kiếm mã Khách hàng muốn xóa.	4. Kiểm tra mã Khách hàng
6. Nhấn nút Xóa	5. Hiện thị thông tin Khách hàng tương ứng.
	7. Thông báo Xóa thành công!

Tình huống thay thế:

- Bước 4-5: Kiểm tra thông tin Khách hàng không hợp lệ, thông báo không tìm thấy!

5.1.5 Use case Sửa thông tin khách hàng

- Tên use case: Sửa thông tin Khách hàng.
- Tác nhân: Người quản trị
- Mục tiêu: Sửa thông tin của Khách hàng.
- Mô tả: Khi Khách hàng thấy thông tin của mình bị sai thì liên hệ với Người quản trị để sửa cơ sở dữ liệu thông tin Khách hàng.
- Kịch bản:

Bảng 6: Đặc tả Use case Sửa thông tin Khách hàng

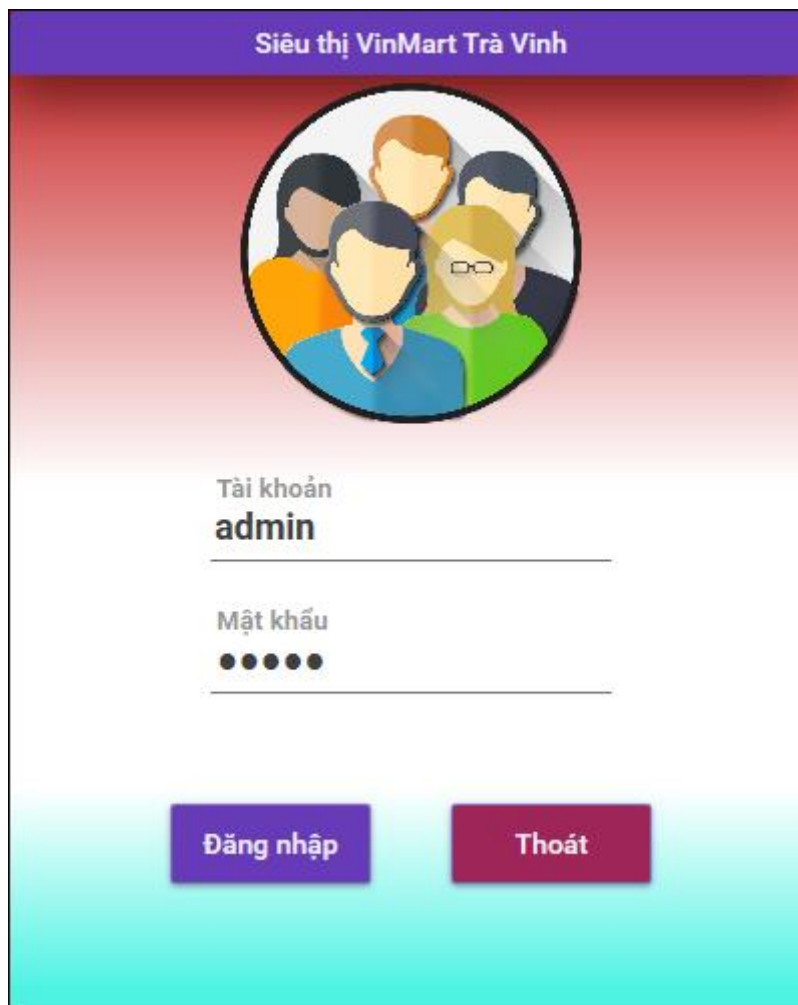
Hành động Actor	Phản ứng hệ thống
1. Mở form Sửa thông tin 3. Sử dụng lại Use case tìm kiếm mã Khách hàng 6. Nhấn nút Edit 7. Nhập các thông tin cần sửa 8. Nhấn nút Update	2. Form Sửa thông tin được mở 4. Kiểm tra mã Khách hàng 5. Hiện thị thông tin Khách hàng 9. Thông báo sửa thành công!

Tình huống thay thế:

- Bước 4-5: Kiểm tra thông tin Khách hàng thông báo không tìm thấy!

3.5 Giao diện

3.5.1 Thiết kế màn hình đăng nhập



Siêu thị VinMart Trà Vinh

Tài khoản
admin

Mật khẩu
●●●●●

Đăng nhập Thoát

Hình 12: Màn hình đăng nhập

3.5.2 Thiết kế giao diện chính

The screenshot displays the main application window with a top navigation bar containing buttons: "Quản Lý Thông Tin Khách Hàng", "Quản Lý Giao Dịch", "In Mã Thẻ", "Tìm Kiếm", "Nhà cung cấp", and "Thống Kê". Below this, a summary section shows three key metrics: "Số Lượng Khách Hàng" (368 Người), "Số Lượt Giao Dịch" (9 Lượt), and "Tổng Doanh Thu" (15910000 VND). A filter section includes "Giới tính" (dropdown), "Loại Khách Hàng" (dropdown), and buttons for "Xuất Danh Sách", "Lọc", and "Refresh". The main area features a table titled "DANH SÁCH KHÁCH HÀNG SIÊU THỊ VINMART TRÀ VINH" with columns: ID Khách Hàng, Tên Khách Hàng, Giới Tính, Năm Sinh, CMND, Điện Thoại, Địa Chỉ, Email, Điểm Lưu, Điểm Tích Lũy, and Loại Khách Hàng. The table lists 15 customers, including Lê Bùi Minh Nhân, Ngô Hương Loan, Nguyễn Thu Hà, etc.

Hình 13: Giao diện chính

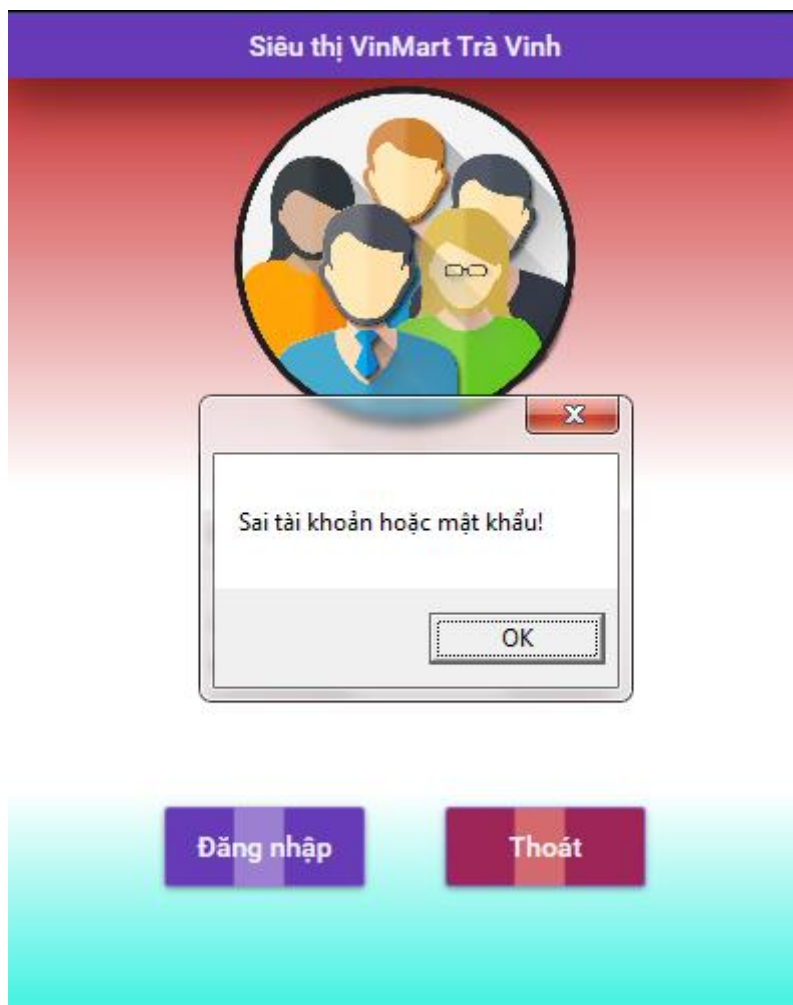
3.5.3 Thiết kế màn hình nhập dữ liệu

The screenshot shows the data entry screen. At the top, there are input fields for "Tên Khách Hàng" (Lê Bùi Minh Nhân), "Giới Tính" (Nam), "CMND" (334953508), "Địa chỉ" (Cảng Long, Trà Vinh), "Điện thoại" (0368495171), "Email" (lebuiminhnhan@gmail.com), and "Năm Sinh" (15/08/1997). Below these are buttons for "Thêm", "Sửa", and "Xóa". The main area contains a table with the same columns as in Hình 13, listing 15 customers. The table is scrollable, showing a list of customers with their details.

Hình 14: Giao diện nhập liệu thông tin khách hàng

3.6 Kết quả thử nghiệm

- Thông báo đăng nhập thất bại do sai tài khoản mật khẩu



Hình 15: Thông báo lỗi đăng nhập

- Chọn một khách hàng trong danh sách để chỉnh sửa thông tin hoặc xóa

ID Khách Hàng	Tên Khách Hàng	Giới Tính	Năm Sinh	CMND	Điện Thoại	Địa Chỉ
100408	Tô Lâm	Nam	8/15/1995 12:00:00 AM	3324548454	0985475444	Trà Vinh
100407	Lê Bùi Minh Nhân	Nam	8/15/1997 12:00:00 AM	334953508	0368495171	Càng Long , Trà Vinh
100367	NGÔ HƯƠNG LOAN	Nữ	7/3/1975 12:00:00 AM	023282148	0905326894	Huyện Trùng Khánh,
100366	NGUYỄN THU HÀ	Nữ	1/8/1974 12:00:00 AM	191514084	0982711883	Thành phố Trà Vinh,
100364	TÔ TRUNG HIẾU	Nam	7/2/1975 12:00:00 AM	024444724	0903727645	Thành phố Trà Vinh,
100363	ĐOÀN THỊ LAN ANH	Nữ	2/11/1971 12:00:00 AM	171287757	0917472003	Huyện Duyên Hải, Tr
100362	NGÔ THỊ THẢO	Nữ	7/19/1975 12:00:00 AM	150417010	0988383971	Huyện Trà Cú, Trà Vi
100361	NINH THỊ HẢI YẾN	Nữ	1/1/1978 12:00:00 AM	022618000	0905326894	Huyện Nghĩa Hưng,
100360	LÝ NGỌC HÙNG	Nam	1/1/1978 12:00:00 AM	173071944	0905326894	Huyện Ngã Sơn, Bắ

Hình 16: Chỉnh sửa thông tin khách hàng

- Thêm một giao dịch mới

Mã Giao Dịch	Khách Hàng	Nhân Viên	Tiền Thanh Toán	Điểm Tích	Điểm Trừ	Tiền Giảm	Ngày Giao Dịch
4000002	LÝ NGỌC HÙNG	Ngô Diệp Lan	1500000	0	0	0	1/10/2019 12:00:00 AM
4000003	TRẦN THUY THẢO	Trần Tuấn Anh	200000	0	0	0	1/4/2019 12:00:00 AM
4000004	NINH THỊ HẢI YẾN	Ngô Diệp Lan	0	6	0	0	1/10/2019 12:00:00 AM
4000005	NGÔ HƯƠNG LOAN	Trần Tuấn Anh	4000000	0	0	0	1/10/2019 12:00:00 AM
4000006	Lê Bùi Minh Nhân	Trần Tuấn Anh	10000000	0	0	0	1/10/2019 12:00:00 AM
4000007	Lê Bùi Minh Nhân	Trần Tuấn Anh	20000	0	0	0	1/11/2019 12:00:00 AM
4000008	Lê Bùi Minh Nhân	Ngô Diệp Lan	20000	0	0	0	1/15/2019 12:00:00 AM
4000009	Lê Bùi Minh Nhân	Ngô Diệp Lan	20000	0	0	0	8/15/1997 12:00:00 AM
4000010	NGÔ HƯƠNG LOAN	Trần Tuấn Anh	1500000	0	0	0	1/10/2019 12:00:00 AM

Hình 17: Thêm mới giao dịch

CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

4.1 Kết luận

4.1.1 Kết quả đạt được

- Xây dựng được chương trình quản lý khách hàng.
- Quản lý được thông tin các nhân khách hàng
- Quản lý thông tin giao dịch của khách hàng.
- Quản lý điểm tích lũy và ưu đãi dành cho khách hàng.
- Giao diện hiện đại dễ sử dụng.
- Thống kê được số lượng khách hàng, số lượt giao dịch và tổng doanh thu
- Xuất danh sách khách hàng ra file excel.

4.1.2 Hạn chế

- Chưa đồng bộ cùng lúc khi xử lý giao dịch với module bán hàng, còn thiết lập thông tin giao dịch với khách hàng riêng lẻ.
- Quản lý còn một chiều, khách hàng muốn tra cứu thông tin phải liên hệ dịch vụ khách hàng.

4.2 Hướng phát triển

- Phát triển hệ thống trên Webservice.
- Xây dựng website cho khách hàng tra cứu thông tin giao dịch, điểm, ưu đãi và cập nhật thông tin cá nhân của chính mình.
- Xây dựng thêm nhiều tính năng mới cho ứng dụng trên Window.
- Đồng bộ hóa với module bán hàng của siêu thị.
- Tích hợp Crystal Report in barcode mã thẻ cho khách hàng khi giao dịch nếu khách hàng quên mang thẻ.
- Phát triển ứng dụng trên nền tảng Android và IOS.

DANH MỤC TÀI LIỆU THAM KHẢO

1. Tìm hiểu về WPF, https://vi.wikipedia.org/wiki/Windows_Presentation_Foundation, ngày truy cập 10/01/2019.
2. Tìm hiểu và ứng dụng mô hình MVVM, <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93viewmodel>, ngày truy cập 10/01/2019.
3. Tìm hiểu Entity Framework, https://en.wikipedia.org/wiki/Entity_Framework, ngày truy cập 10/01/2019.
4. Matthew MacDonald , Pro WPF in C# 2010(2010).
5. Adam Nathan,Windows Presentation Foundation Unleashed(21/12/2006)