

HỌC VIỆN NGÂN HÀNG
KHOA CÔNG NGHỆ THÔNG TIN VÀ KINH TẾ SỐ



TIỂU LUẬN NHÓM
Học Phần: Lập Trình DotNet

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG QUẢN LÝ CỦA
HÀNG MÁY TÍNH DÀNH CHO NHÂN VIÊN BẢNG
ENTITY FRAMEWORK**

Giáo viên hướng dẫn: ThS. Lê Cẩm Tú

Lớp học phần: 232IS25A03 – Nhóm 14

Danh sách thành viên:

Họ và tên	Mã sinh viên
1. Nguyễn Đức Công	24A4042427
2. Lê Hương Giang	24A4042434
3. Nguyễn Cảnh Phong	24A4042605
4. Ngô Mạnh Thắng	24A4042611
5. Nguyễn Thị Thảo Trang	24A4041688

Hà Nội, 5/2024

HỌC VIỆN NGÂN HÀNG
KHOA CÔNG NGHỆ THÔNG TIN VÀ KINH TẾ SỐ



TIỂU LUẬN NHÓM

Học Phần: Lập Trình DotNet

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG QUẢN LÝ CỦA
HÀNG MÁY TÍNH DÀNH CHO NHÂN VIÊN BẰNG
ENTITY FRAMEWORK**

Giáo viên hướng dẫn: ThS. Lê Cẩm Tú

Lớp học phần: 232IS25A03 – Nhóm 14

Danh sách thành viên:

Họ và tên	Mã sinh viên	Phần trăm đóng góp
1. Nguyễn Đức Công (Nhóm trưởng)	24A4042427	22%
2. Lê Hương Giang	24A4042434	19.5%
3. Nguyễn Cảnh Phong	24A4042605	20%
4. Ngô Mạnh Thắng	24A4042611	18.5%
5. Nguyễn Thị Thảo Trang	24A4041688	20%

Hà Nội, 5/2024

LỜI CAM ĐOAN

Tôi xin cam đoan kết quả đạt được trong báo cáo là sản phẩm nghiên cứu, tìm hiểu của riêng nhóm chúng tôi. Trong toàn bộ nội dung của báo cáo, những điều được trình bày hoặc là của cá nhân tôi hoặc là được tổng hợp từ nhiều nguồn tài liệu. Tất cả các tài liệu tham khảo đều có xuất xứ rõ ràng và được trích dẫn hợp pháp.

Tôi xin hoàn chịu trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan của mình.

SINH VIÊN THỰC HIỆN

Nguyễn Đức Công

Lê Thị Hương Giang

Nguyễn Cảnh Phong

Ngô Mạnh Thắng

Nguyễn Thị Thảo Trang

LỜI CẢM ƠN

Trong suốt thời gian từ khi bắt đầu thực hiện bài thực tập đến nay, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ của Khoa và quý thầy cô. Với lòng biết ơn sâu sắc, chúng em xin gửi lời cảm ơn đầu tiên đến quý thầy cô của Khoa Công Nghệ Thông Tin Và Kinh Tế Số – Học viện Ngân Hàng đã tạo điều kiện cho chúng em được học tập bộ môn “Lập trình DotNet” trong điều kiện tốt nhất. Kiến thức từ môn học này đã giúp nhóm có cái nhìn tổng quát nhất về “Lập trình hướng đối tượng” cũng như “ngôn ngữ lập trình C#” và cách thức triển khai một ứng dụng Winform.

Chúng em cũng xin đặc biệt gửi lời cảm ơn đến cô Lê Cẩm Tú đã tận tâm hướng dẫn, giúp đỡ chúng em trong suốt quá trình thực hiện bài báo cáo, từ những ngày đầu tiên cho đến khi hoàn thành. Nếu không có sự chỉ bảo của cô thì bài thu hoạch này của chúng em rất khó có thể hoàn thiện được. Một lần nữa, chúng em xin chân thành cảm ơn cô! Trong quá trình giải quyết bài toán, do kiến thức, lý luận cũng như kinh nghiệm thực tiễn của chúng em còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót. Chúng em rất mong nhận được góp ý quý báu của quý thầy, cô để bài báo cáo của em được hoàn thiện. Từ đó chúng em có thể rút ra được nhiều kinh nghiệm để hoàn thành tốt hơn những bài báo cáo sau này!

Sau cùng, chúng em xin kính chúc quý thầy, cô trong Khoa Công Nghệ Thông Tin Và Kinh Tế Số thật dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh trồng người.

Chúng em xin chân thành cảm ơn!

MỤC LỤC

LỜI CAM ĐOAN.....	I
LỜI CẢM ƠN	II
MỤC LỤC	III
DANH MỤC HÌNH ẢNH	1
LỜI MỞ ĐẦU	2
CHƯƠNG 1: TỔNG QUAN VỀ DỰ ÁN.....	3
1.1. GIỚI THIỆU BÀI TOÁN.....	3
1.2. CÁC CÔNG CỤ SỬ DỤNG	4
1.3. DESIGN PATTERN.....	4
CHƯƠNG 2: TRIỂN KHAI DỰ ÁN	6
2.1. XÂY DỰNG VÀ THIẾT KẾ DATABASE	6
2.1.1. Mô tả các thực thể và thuộc tính.....	6
2.1.2. Mối quan hệ.....	6
2.1.3. Chuẩn hóa CSDL.....	7
2.1.4. Biểu đồ Diagram.....	9
2.2. XÂY DỰNG KHUNG CÂY THƯ MỤC CHO DỰ ÁN.....	10
2.2.1. Xây dựng theo mô hình MVP (Model – Presenter - View)	10
2.2.2. Cách chuyển phần code của mình cho người khác khi sử dụng EF	19
CHƯƠNG 3: XÂY DỰNG CÁC TÍNH NĂNG CHO PROJECT	23
3.1. CHỨC NĂNG ĐĂNG NHẬP	23
3.1.1. Phân lớp Presenter – Class PreLogin	23
3.1.2. Phân lớp View – Form Login	23
3.2. XÂY DỰNG MENUSTRIP	24
3.2.1. Phân lớp Presenter – Class PreMain	24

3.2.2. <i>Cách thức triệu gọi trong Form cần MenuStrip</i>	26
3.3. HÀNG HÓA	26
3.3.1. <i>Phân lớp View</i>	26
3.3.2. <i>Phân lớp Presenter – Class PreComputer</i>	31
3.3.3. <i>Phân lớp Model – Class MayVT</i>	32
3.4. GIAO DỊCH	33
3.4.1. <i>Phân lớp View</i>	33
3.4.2. <i>Phân lớp Presenter</i>	36
3.4.3. <i>Phân lớp Model</i>	39
3.5. ĐỐI TÁC	39
3.5.1. <i>Phân lớp View</i>	39
3.5.2. <i>Phân lớp Presenter – Customer – Employee – Supplier</i>	44
3.5.3. <i>Phân lớp Model</i>	45
3.6. PHÂN LỚP VIEW CỦA BÁN HÀNG VÀ NHẬP HÀNG	46
3.7. PHÂN LỚP VIEW – FORM TỔNG QUAN.....	50
KẾT LUẬN	52
TÀI LIỆU THAM KHẢO.....	53

DANH MỤC HÌNH ẢNH

Hình 1: Lược đồ quan hệ của database	9
Hình 2: Thêm thư mục Model	11
Hình 3: Tạo Entity Data Model	12
Hình 4: Chọn EF Designer From Database	13
Hình 5: Tạo Entity	14
Hình 6: Database Diagram trong Visual Studio.	15
Hình 7: Các Item trong thư mục Presenter	16
Hình 8: Dòng code mà ở class nào trong Presenter hầu như cũng có.....	17
Hình 9: Vùng code sự kiện.....	18
Hình 10: App.config trong Solution Explore	20
Hình 11: Đoạn code có chứa Data source.....	20
Hình 12: Tên server trong SSMS	21
Hình 13 :Thêm Item đã tồn tại.....	21
Hình 14: Chọn Form Login	22
Hình 15: Form Đăng nhập	24
Hình 16: Triệu gọi MenuStrip trong phương thức khởi tạo của Form bán hàng	26
Hình 17: Form Hàng Hóa	28
Hình 18: Form thêm hàng hóa được hiển thị dưới dạng popup	30
Hình 19: Form hóa đơn nhập.....	34
Hình 20: Form hóa đơn bán.....	36
Hình 21: Form nhân viên.....	40
Hình 22: Form khách hàng	41
Hình 23: Form nhà cung cấp	42
Hình 24: Form thêm khách hàng hiển thị dưới dạng popup	43
Hình 25: Form thêm nhân viên hiển thị dưới dạng popup.....	44
Hình 26: Form nhà cung cấp hiển thị dưới dạng popup	44
Hình 27 : Form bán hàng.....	49
Hình 28: Form nhập hàng.....	50

LỜI MỞ ĐẦU

Trong bối cảnh hiện nay khi mà công nghệ phát triển. Các cửa hàng bắt đầu có xu hướng nâng cao vấn đề quản lý doanh nghiệp của mình. Theo dòng phát triển các lập trình viên dần yêu cầu khả năng tư duy hướng đối tượng và các design pattern đang dần trở lên là yêu cầu không thể thiếu khi thiết kế một phần mềm. Bên cạnh đó, việc cập nhật các công nghệ mới để có thể xây dựng một phần mềm một cách hiệu quả cũng là yêu cầu bức thiết để trở thành một lập trình viên giỏi.

Theo lẽ đó, nhóm chúng mình đã quyết định thử sức triển khai những công nghệ mới và các design pattern theo hướng đơn giản hóa nhằm mang các bạn lại gần với công việc lập trình viên. Hi vọng các bạn đọc sẽ dõi theo tiến trình của chúng mình trong bài báo cáo này.

Bài báo cáo sẽ gồm có ba chương:

- Chương 1: Tổng quan về dự án
- Chương 2: Triển khai dự án
- Chương 3: Xây dựng cách tính năng cho project

CHƯƠNG 1: TỔNG QUAN VỀ DỰ ÁN

1.1. GIỚI THIỆU BÀI TOÁN

Xây dựng hệ thống quản lý cửa hàng máy tính dựa trên Entity Framework là một đề tài khá là ít ai làm về nó (trong phạm vi tài liệu tiếng việt). Chủ yếu là họ sẽ xây dựng hệ thống theo hướng cơ bản nhất để mọi người có thể đọc và triển khai chúng một cách dễ dàng, vì thế cho nên tài liệu hạn chế là điều tất yếu.

Xây dựng hệ thống theo cách này sẽ tận dụng được nguồn lực một cách tối đa vì không còn phải tạo các lớp thực thể một cách thủ công. Bên cạnh đó EF và Linq là combo tối thượng khi kết hợp với nhau.

Về mục đích, nhóm chúng em thực hiện đề tài này với mục đích giới thiệu cho mọi người một hướng tiếp cận mới cũng như hiệu quả của nó khi triển khai trong các dự án thực tế. Giúp cho nhân viên có thể bán hàng, nhập hàng, quan sát doanh thu theo ngày, theo thứ trong tuần, thêm sản phẩm, khách hàng, nhà cung cấp,...

Có một điều các bạn cần phải chú ý trong bài báo cáo của nhóm đó chính là: “trong bài báo cáo của nhóm không hề có chuyện sửa hay xóa dữ liệu”. Không phải là vì nhóm không làm được hay là quên mà vì nhóm có tư duy như sau:

- Một phần mềm quản lý dành cho nhân viên thì không thể xóa, sửa dữ liệu một cách tùy ý. Nếu có thể xóa, sửa một cách tùy ý như vậy thì rất có thể Database sẽ bị rối loạn. Rất khó để cho các nhân sự cấp cao dựa vào Database mà phân tích dữ liệu.
- Nếu có thể sửa được dữ liệu thì sẽ không đảm bảo được vấn đề bảo mật của hệ thống. Ví dụ khi mà nhân viên giảm số lượng hàng hóa nhập của một hóa đơn từ rất lâu trước đây thì nó có thể ảnh hưởng đến hệ thống hiện tại => Tạo ra lỗ hổng cực kỳ nghiêm trọng cho hệ thống

Từ hai lý do trên nhóm quyết định sẽ không cung cấp chức năng xóa, sửa các trường dữ liệu cho nhân viên mà sẽ để dành cho phiên bản dành cho quản lý ở những lần phát hành sau.

1.2. CÁC CÔNG CỤ SỬ DỤNG

Nhìn chung nhóm sẽ sử dụng các công cụ tương tự như cô giáo đề xuất. Đại khái gồm có các công cụ sau:

- ❖ IDE - môi trường tích hợp phát triển: Visual Studio 2022 Community – Miễn Phí¹
- ❖ Framework: .NetFramework, ADO.Net Entity Data Model²
- ❖ Database: SQL server 2022 và công cụ quản lý SSMS20v1³.
- ❖ Công cụ lưu trữ và quản lý mã nguồn: Github và Git (vì đây là thành quả của nhóm nên mình sẽ không để public repository này!)
- ❖ Ngoài ra còn có một số Packet cài thêm từ bên ngoài

1.3. DESIGN PATTERN

Sau khi tham khảo ở nhiều nguồn khác nhau về kiến trúc, cách tổ chức thư mục thì nhóm đã quyết định chọn mô hình MVP(Model – View – Presenter). Giới thiệu sơ lược về mô hình này thì mô hình MVP gồm 3 phân lớp đó là:

- ❖ Model: Phân lớp này sẽ chứa các class liên quan đến dữ liệu và các logic kinh doanh liên quan đến dữ liệu. Có thể nói phân lớp này sẽ nặng về mặt “hướng đối tượng”. Nhiệm vụ chính của phân lớp này là thực hiện các truy xuất đến cơ sở dữ liệu.
- ❖ View: Phân lớp này sẽ chứa các form giao diện người dùng của ứng dụng nó sẽ chứa các lớp cung cấp việc hiển thị dữ liệu và xác định những hành động của người dùng sau đó thông báo tới phân lớp Presenter để xử lý logic.
- ❖ Presenter: Phân lớp này sẽ chứa các lớp liên quan tới việc xử lý logic trong phân lớp View. Presenter sẽ lấy dữ liệu từ lớp Model và sau đó áp dụng các logic đã được thiết lập sẵn trong phân lớp này để hiển thị lên View.

Điểm đặc biệt của mô hình này chính là việc Model và View sẽ không biết về sự tồn tại của nhau. Qua đó, hệ thống sẽ được đảm bảo an toàn về vấn đề bảo mật. Hơn thế nữa, nếu làm việc theo mô hình này thì tốc độ triển khai dự án sẽ nhanh chóng hơn. Nếu

¹:<https://visualstudio.microsoft.com/vs/community/>

² Entity Framework

³ Phiên bản 20+ năm 2024

chẳng may có lỗi xảy ra thì chỉ có lỗi cục bộ chứ không hỏng luôn chương trình, điều này rất quan trọng trong việc rút ngắn thời gian debug.

Một khả năng tiếp theo cũng đáng để nhắc tới đó chính là khả năng mở rộng ứng dụng cực kỳ linh hoạt của model này. Đơn giản là bạn chỉ cần thêm các Form và chức năng tương ứng mà không ảnh hưởng đến phần cũ.

Phần Model kết hợp với Entity Framework sẽ cực kỳ mạnh mẽ trong việc truy vấn cũng như kết hợp với phân lớp phía trên, chưa kể nếu như trong Database có gì cập nhật thì chỉ cần một vài thao tác cũng có thể update thành công cũng như là không phải tạo Connection thủ công như phương pháp cũ nữa.

Chưa kể đến việc khi ta sử dụng được EF thì sẽ khắc phục được nhược điểm chí tử của lỗ hổng bảo mật SQL Injection do quá trình dùng các câu truy vấn SQL bằng string. Cụ thể, EF sẽ sử dụng kết hợp với Linq qua đó đưa người dùng tìm đến đúng đích, đúng dữ liệu mà không có hơn. Một phần cũng hạn chế được sự tấn công của các đối tượng khả khi vào hệ thống.

⇒ **Chú ý:** Bài tập của nhóm chỉ là mô phỏng dựa theo mô hình MVP chứ không hoàn toàn là MVP, một hệ thống MVP sẽ yêu cầu các interface và luồng dữ liệu lưu chuyển nghiêm ngặt hơn rất nhiều. Nhưng vì để bài làm của nhóm không bị khó hiểu thì nhóm em sẽ bỏ đi những phần nghiêm ngặt đó, nhưng vẫn tuân thủ theo luồng dữ liệu cơ bản của MVP đó là $V - P - M \Leftrightarrow M - P - V$ đồng thời cũng có một vài ngoại lệ.

CHƯƠNG 2: TRIỂN KHAI DỰ ÁN

2.1. XÂY DỰNG VÀ THIẾT KẾ DATABASE

2.1.1. Mô tả các thực thể và thuộc tính

- Loa (Mã loa, tên loa)
- Loại Máy (Mã loại máy, tên loại máy)
- Chip(Mã chip, tên chip)
- Ổ cứng (Mã Ổ cứng, tên ổ cứng)
- Dung lượng (Mã dung lượng, tên dung lượng)
- Tốc độ (Mã tốc độ, tên tốc độ)
- OCD (Mã ổ, tên ổ)
- RAM (Mã ram, tên ram)
- Màn hình (Mã màn hình, tên loại MH)
- Cỡ MH (Mã cỡ MH, tên cỡ MH)
- Chuột (Mã chuột, tên chuột)
- Bàn phím (Mã bàn phím, tên bàn phím)
- USB (Mã ổ USB, tên ổ usb, dung lượng)
- Hãng sản xuất (Mã hãng sx, tên hãng sx)
- Máy vi tính (Mã máy VT, tên máy VT, giá nhập, giá bán, thời gian BH, số lượng, ảnh, ghi chú)
- Nhà cung cấp (Mã NCC, tên NCC, địa chỉ, điện thoại)
- Nhân viên (Mã NV, tên NV, ngày sinh, giới tính, địa chỉ, điện thoại, mật khẩu)
- Khách hàng (Mã khách, tên khách, địa chỉ, điện thoại)

2.1.2. Mối quan hệ

- Máy vi tính <có> Loa (1:M)
- Máy vi tính <có> Loại máy (1:M)

- Máy vi tính <có> Chip (1:M)
- Máy vi tính <có> Ổ cứng (1:M)
- Máy vi tính <có> Dung lượng (1:M)
- Máy vi tính <có> Tốc độ (1:M)
- Máy vi tính <có> Ổ CD (1:M)
- Máy vi tính <có> RAM (1:M)
- Máy vi tính <có> Màn hình (1:1)
- Máy vi tính <có> Cỡ MH (1:M)
- Máy vi tính <có> Chuột (1:M)
- Máy vi tính <có> Bàn phím (1:M)
- Máy vi tính <có> USB (1:M)
- Máy vi tính <có> Hãng sản xuất (1:1)
- Nhân Viên <nhập> Máy vi tính(1:M): (Mã HDN, ngày nhập, số lượng, thành tiền, tổng tiền)
- Khách hàng <mua> Máy vi tính (1:M) (Mã HDB, Ngày bán, số lượng, thành tiền, tổng tiền)
- Máy vi tính <thuộc> Nhà cung cấp (1:M)

2.1.3. Chuẩn hóa CSDL

- Máy vi tính (Mã máy VT, tên máy VT, giá nhập, giá bán, thời gian BH, số lượng, ảnh, ghi chú, Mã loa, Mã loại máy, Mã chip, Mã màn hình, Mã cỡ MH, Mã ổ cứng, Mã USB, Mã dung lượng, Mã tốc độ, Mã ổ CD, Mã bàn phím, Mã chuột, Mã RAM, Mã hãng sản xuất)

Ta có bảng Hóa đơn nhập được sinh ra từ hoạt động dựa trên mối quan hệ giữa nhân viên và Máy vi tính, Hóa đơn bán dựa trên mối quan hệ giữa Khách hàng và Máy vi tính:

- Hóa đơn nhập (Mã máy vi tính, Mã HDN, ngày nhập, số lượng, đơn giá, thành tiền, tổng tiền, Mã NV, Mã NCC)

- Hóa đơn bán (Mã máy vi tính, Mã HDB, ngày bán, số lượng, tổng tiền, thành tiền, Mã NV, Mã khách)

Vì Thành tiền là thuộc tính dẫn xuất có thể tính được từ số lượng*đơn giá nên ta cần loại bỏ nó để không bị vi phạm chuẩn 1NF. Hóa đơn nhập và Hóa đơn bán chưa đạt chuẩn dạng chuẩn 2NF cần đưa về dạng 2NF thì cần tách thành các bảng sau đây:

- Chi tiết HDN (Mã máy vi tính, Mã HDN, số lượng, đơn giá)
- Hóa đơn nhập(MaHDN, ngày nhập, tổng tiền, MãNV, MaNCC)
- Chi tiết HDB(Mã máy vi tính, Ma HDB, số lượng, đơn giá)
- Hóa đơn bán (Ma HDB, ngày bán, tổng tiền, Mã NV, Mã khách)

Để đảm bảo vấn đề bảo mật, không cho nhân viên khác xem được thông tin tài khoản và mật khẩu. Nhóm sẽ tách trường thuộc bảng nhân viên thành một bảng riêng biệt là LoginAccount.

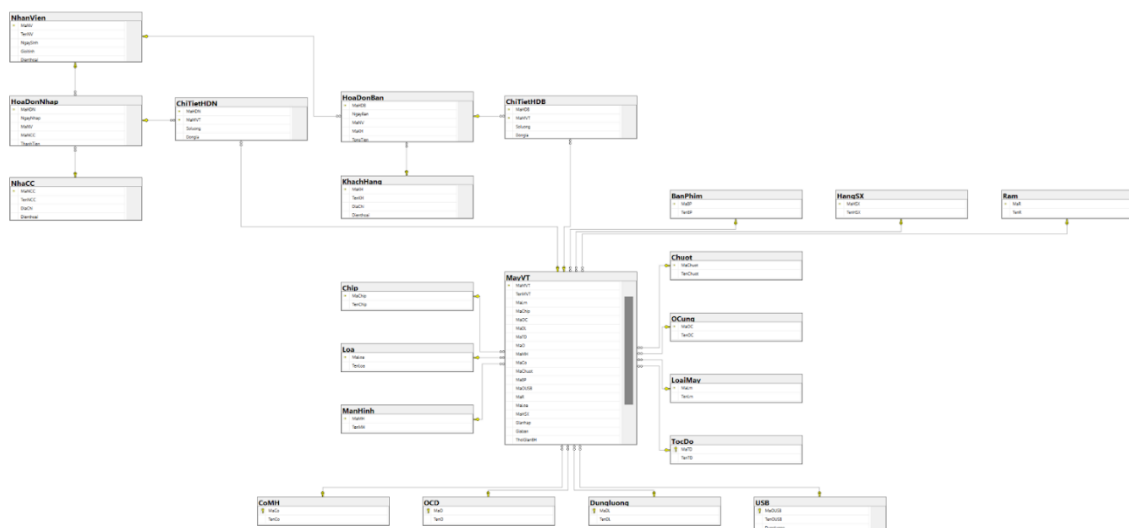
- LoginAccount(Mã NV , mật khẩu)

Dữ liệu trên đã ở dạng chuẩn 2NF và cũng đã đạt chuẩn 3NF. Vậy ta có dữ liệu đã được chuẩn hóa là:

- Loa (Mã loa, tên loa)
- Loại Máy (Mã loại máy, tên loại máy)
- Chip(Mã chip, tên chip)
- Ổ cứng (Mã Ổ cứng, tên ổ cứng)
- Dung lượng (Mã dung lượng, tên dung lượng)
- Tốc độ (Mã tốc độ, tên tốc độ)
- OCD (Mã ổ, tên ổ)
- RAM (Mã ram, tên ram)
- Màn hình (Mã màn hình, tên loại MH)
- Cỡ MH (Mã cỡ MH, tên cỡ MH)
- Chuột (Mã chuột, tên chuột)

- Bàn phím (Mã bàn phím, tên bàn phím)
- USB (Mã ổ USB, tên ổ usb, dung lượng)
- Hãng sản xuất (Mã hãng sx, tên hãng sx)
- Máy vi tính (Mã máy VT, tên máy VT, giá nhập, giá bán, thời gian BH, số lượng, ảnh, ghi chú, Mã loa, Mã loại máy, Mã chip, Mã màn hình, Mã cỡ MH, Mã ổ cứng, Mã USB, Mã dung lượng, Mã tốc độ, Mã ổ CD, Mã bàn phím, Mã chuột, Mã RAM, Mã hãng sản xuất)
- Nhà cung cấp (Mã NCC, tên NCC, địa chỉ, điện thoại)
- Nhân viên (Mã NV, tên NV, ngày sinh, giới tính, địa chỉ, điện thoại)
- LoginAccount(Mã NV, mật khẩu)
- Khách hàng (Mã khách, tên khách, địa chỉ, điện thoại)
- Hóa đơn nhập(Mã HDN, ngày nhập, tổng tiền, Mã NV, MaNCC)
- Chi tiết HDN (Mã máy vi tính, Mã HDN, số lượng, đơn giá)
- Hóa đơn bán (Ma HDB, ngày bán, tổng tiền, Mã NV, Mã khách)
- Chi tiết HDB(Mã máy vi tính, Ma HDB, số lượng, đơn giá)

2.1.4. Biểu đồ Diagram



Hình 1: Lược đồ quan hệ của database

2.2. XÂY DỰNG KHUNG CÂY THƯ MỤC CHO DỰ ÁN

2.2.1. Xây dựng theo mô hình MVP (Model – Presenter - View)

2.2.1.1. Model

Ở trong thư mục Model sẽ là các class liên quan tới các logic nghiệp vụ, mối quan hệ, ràng buộc hay các cách thức để lấy được dữ liệu từ database lên.

Bình thường, chúng ta sẽ phải áp dụng lập trình hướng đối tượng để tạo ra các hiện thực hóa các đối tượng đó, rồi sau đó mới sử dụng tới các hàm và phương thức để “hứng” dữ liệu từ database đổ vào.

Nhưng tại sao chúng ta phải làm quá “basic” như thế trong khi Visual Studio đã hỗ trợ sẵn một thứ để làm công việc như vậy. Đó chính là EntityFrameWork, tuy ở trên lớp không được giới thiệu qua nhưng công cụ mạnh mẽ như vậy mà không sử dụng thì lại hơi uổng phí công sức của các nhà thiết kế đã tạo ra chúng.

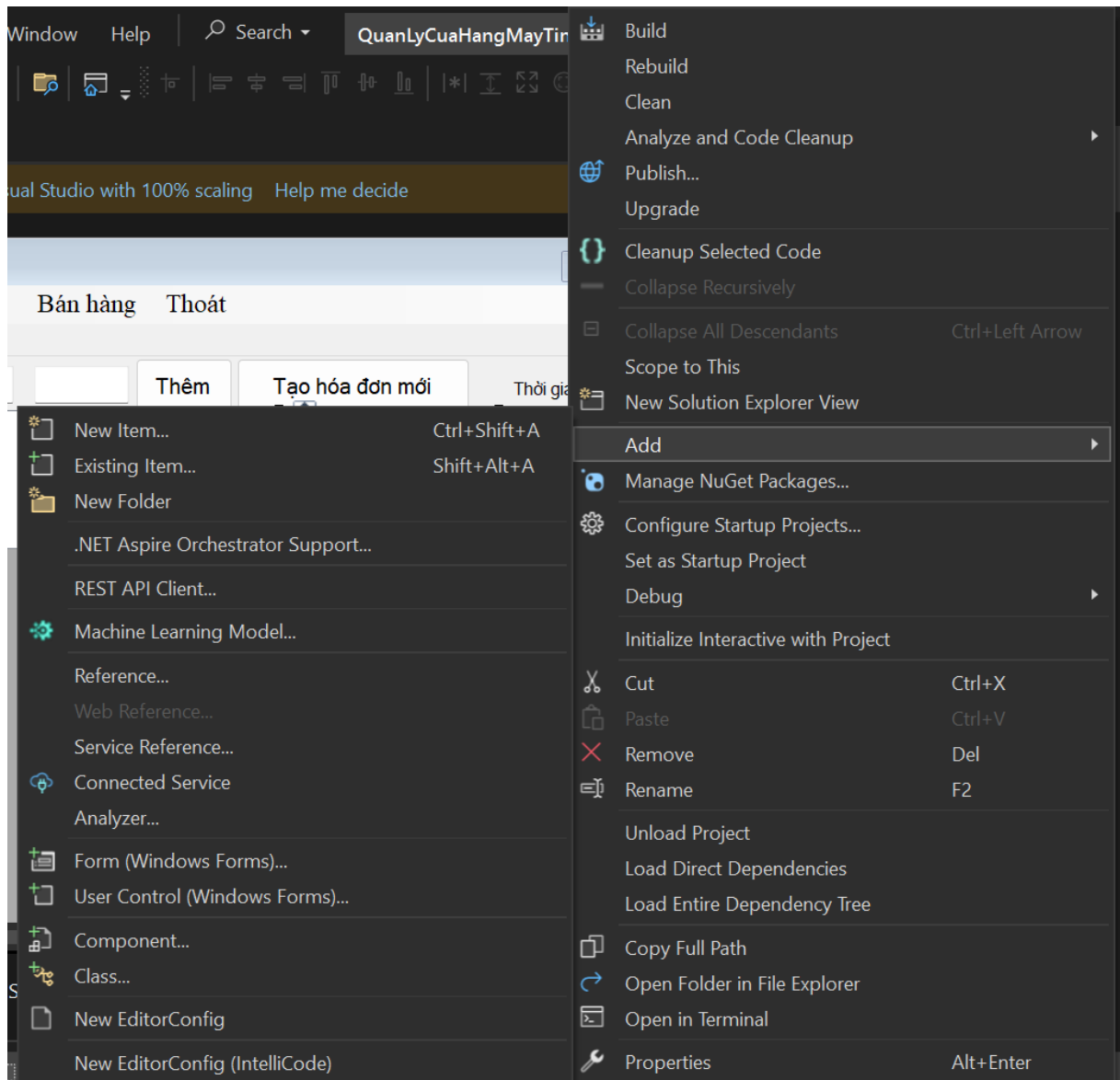
Khi ta sử dụng công cụ này, đồng nghĩa với việc Visual Studio sẽ giúp chúng ta sẽ tạo ra tất cả các thực thể và các bảng có liên quan trong database, giống y hệt như cách chúng ta tạo ra các class để “hứng” data từ database vậy, qua đó làm giảm một khối lượng công việc khổng lồ trong bài báo cáo.

Giới thiệu đến EF⁴ cũng đã dài, sau đây nhóm sẽ hướng dẫn cách để cài đặt EF làm Model trong mô hình MVP:

⁴ Entity FrameWork

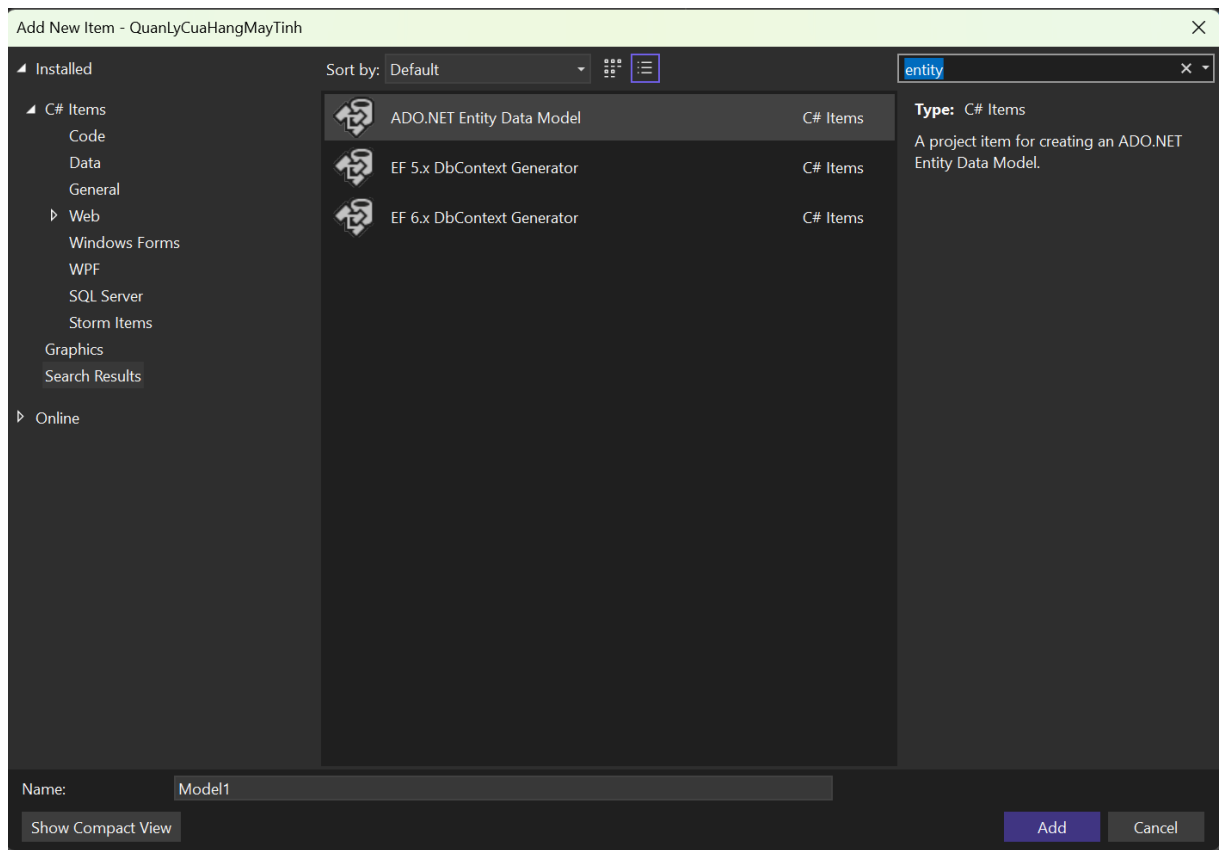
a. Cách cài đặt EF từ DB có sẵn.

- ✓ Bước 1: Tạo thư mục Model trong project. Trong cửa sổ solution Explore, bấm phải chuột vào dự án. Chọn Add=> chọn New Folder => Đặt tên cho thư mục là Model.



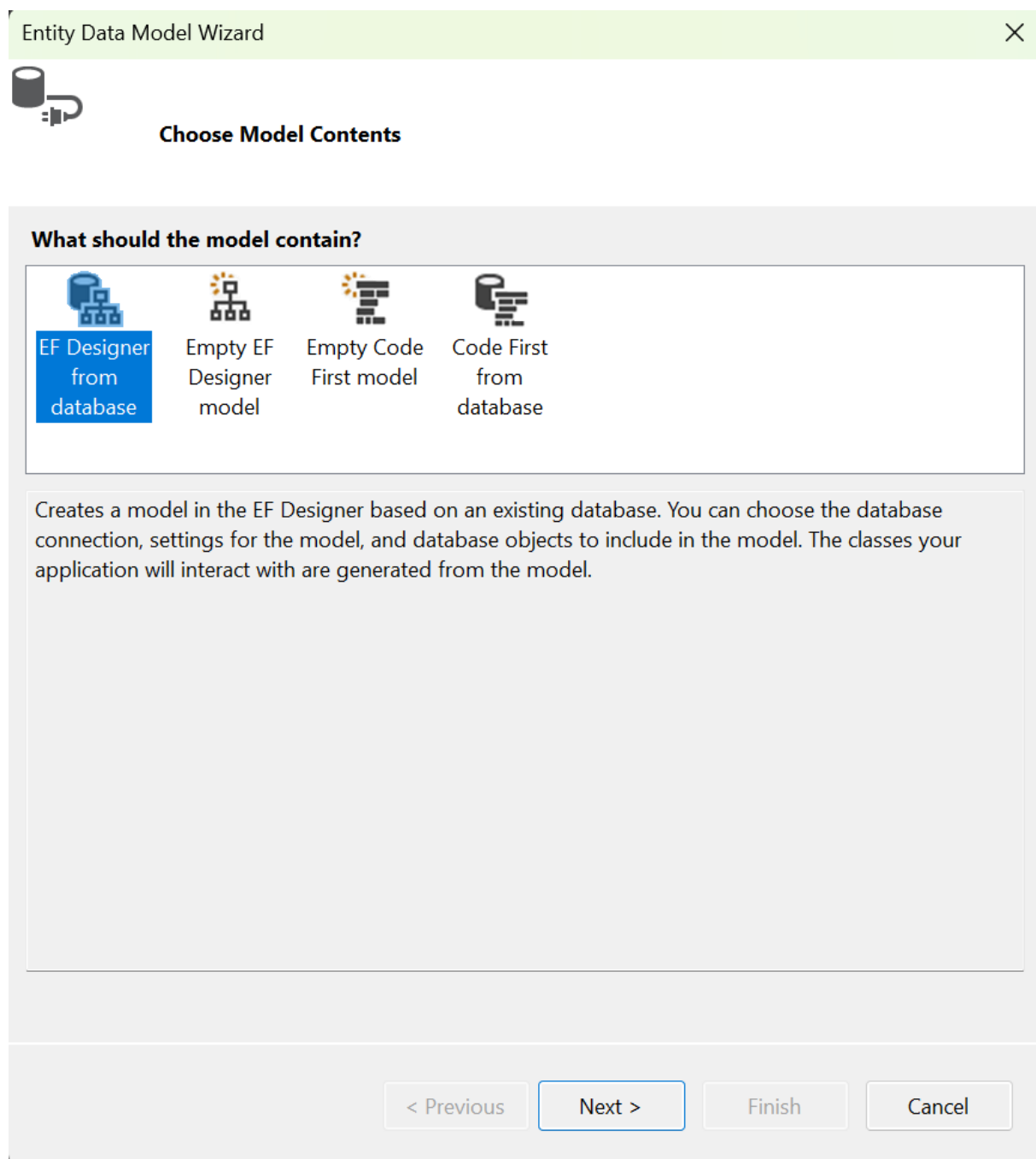
Hình 2: Thêm thư mục Model

- ✓ Bước 2: Nhấn phải chuột vào Folder=> chọn Add=> chọn New Item. Trong cửa sổ tìm kiếm phía trên cùng bên phải, các bạn hãy gõ Entity. Sau đó chọn ADO.NET Entity Data Model. Đặt tên cho nó rồi nhấn Enter.



Hình 3: Tạo Entity Data Model

✓ Bước 3: Chọn EF Designer from Database=> Next.




Hình 4: Chọn EF Designer From Database

✓ Bước 4: Chọn New Connection. Ở bước này các bạn làm giống hệt như khi các cách bạn kết nối tới cơ sở dữ liệu mà mình đã học ở trên lớp.

✓ Bước 5: Khi đã kết nối tới cơ sở dữ liệu xong thì các bạn hãy đặt tên cho EF này, ở đây mình sẽ đặt tên cho nó là Entity (lưu ý cái tên này sẽ được sử dụng để tương tác với dữ liệu và nó được lưu trữ trong App.config dưới dạng file XML). Sau đó nhấn Next.

Entity Data Model Wizard

 **Choose Your Data Connection**

Which data connection should your application use to connect to the database?

congsempai.btl.dbo New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set it in my application code.

☐ Yes, include the sensitive data in the connection string.

Connection string:

```
metadata=res://*/Model.Model.csdl|res://*/Model.Model.ssdl|
res://*/Model.Model.msl;provider=System.Data.SqlClient;provider connection string="data
source=CONGSEMPAI;initial catalog=btl;integrated
security=True;encrypt=False;MultipleActiveResultSets=True;App=EntityFramework"
```

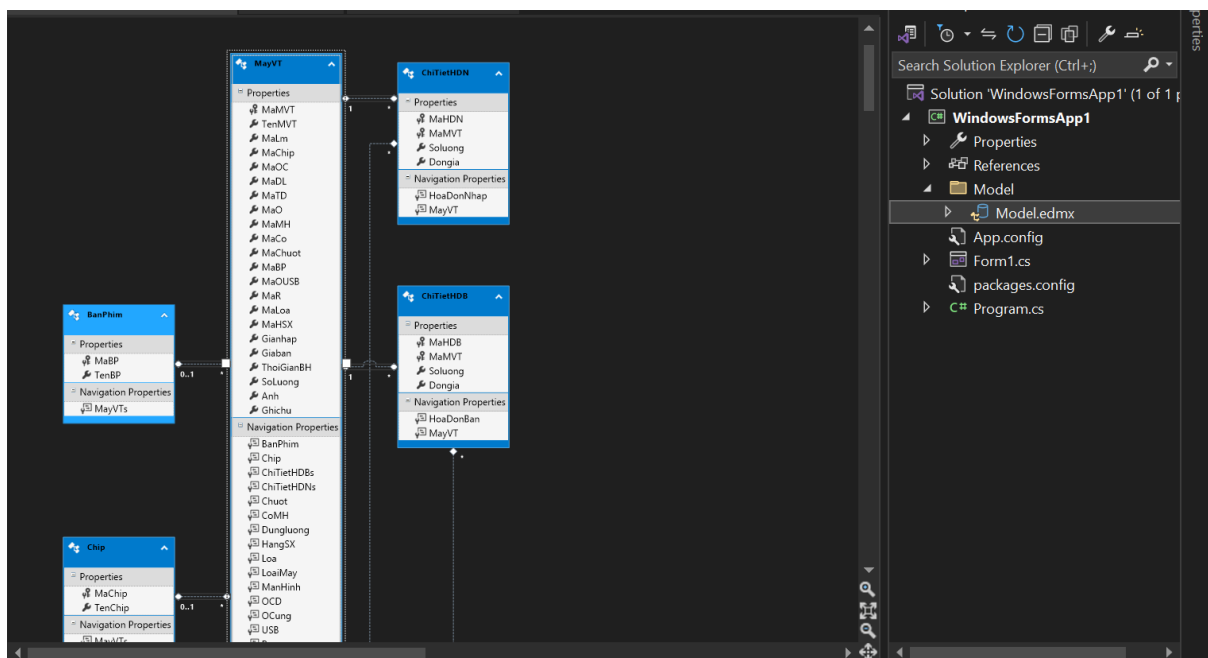
☒ Save connection settings in App.Config as:

Entity

< Previous Next > Finish Cancel

Hình 5: Tạo Entity

- ✓ Bước 6: Chọn EF 6x (mặc định). Nhấn Next, tích chọn vào Table rồi finish.
Sau khi tạo xong thì... Bùm!!!!



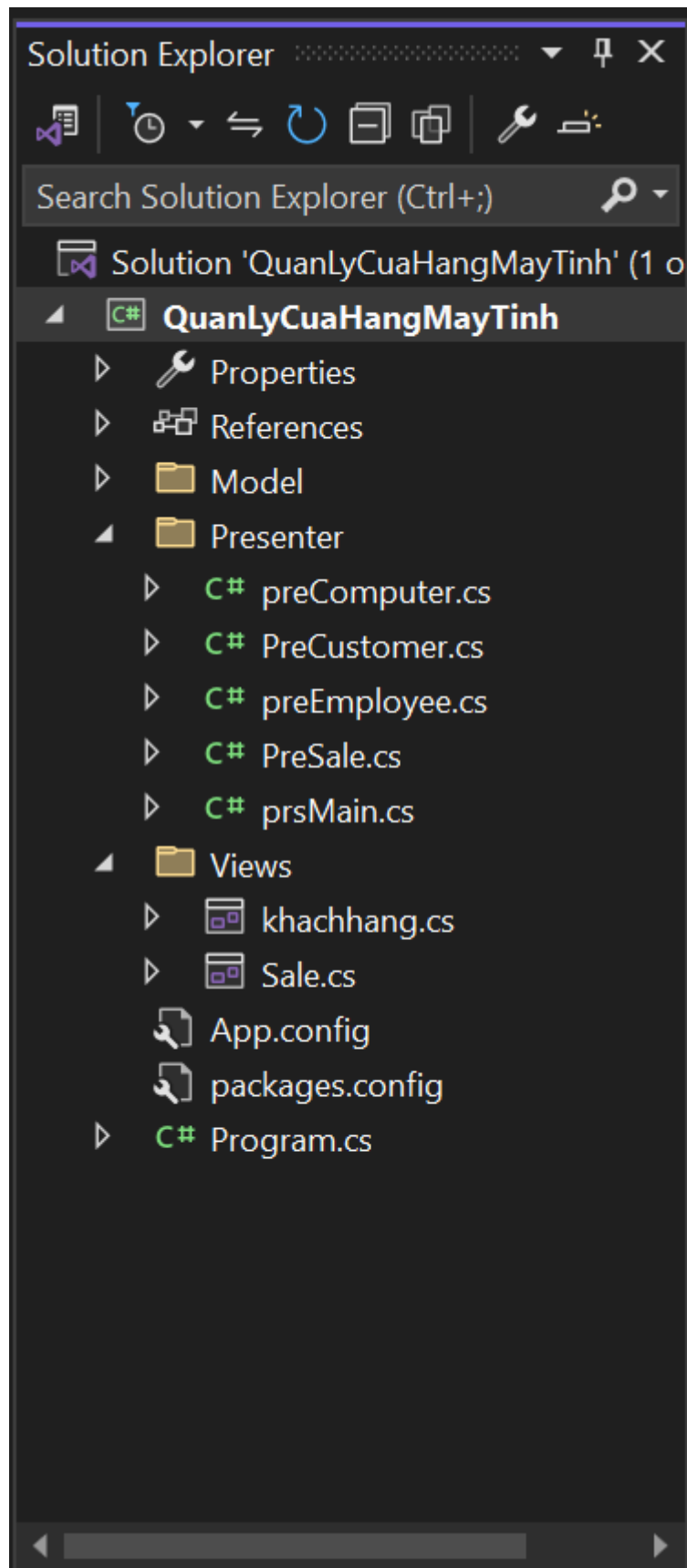
Hình 6: Database Diagram trong Visual Studio.

Vừa rồi là các bước thực hiện quá trình tạo ra EF ở trong thư mục Model. Về cách sử dụng nó trong thư mục này mình sẽ đề cập đến trong các chương sau.

2.2.1.2. Presenter

Như đã giới thiệu ở phần bên trên, phân lớp Presenter sẽ chứa các logic hoạt động mà người dùng có thể thực hiện ở trong View. Nhiệm vụ của nó là lấy dữ liệu từ lớp Model và trả lại cho phân lớp view. Vậy trong thư mục này có gì?

Đầu tiên, các bạn hãy tạo một thư mục có tên Presenter. Trong thư mục này sẽ là các class sử dụng để tương tác, gọi đến lớp Model để lấy dữ liệu của nó lên. Đặc điểm của phân lớp này là các Item chứa trong nó hầu hết là các class C# và các phương thức của nó đều là các phương thức trả về dữ liệu. Thường thì đối với mỗi Form mà phân lớp View có sẽ tương ứng có một class để diễn tả sự trình bày của phân lớp View. Ví dụ như hình sau:



Hình 7: Các Item trong thư mục Presenter

Ngoài ra, các class này còn có một đặc điểm chung đó chính là **Đều phải có** dòng code:

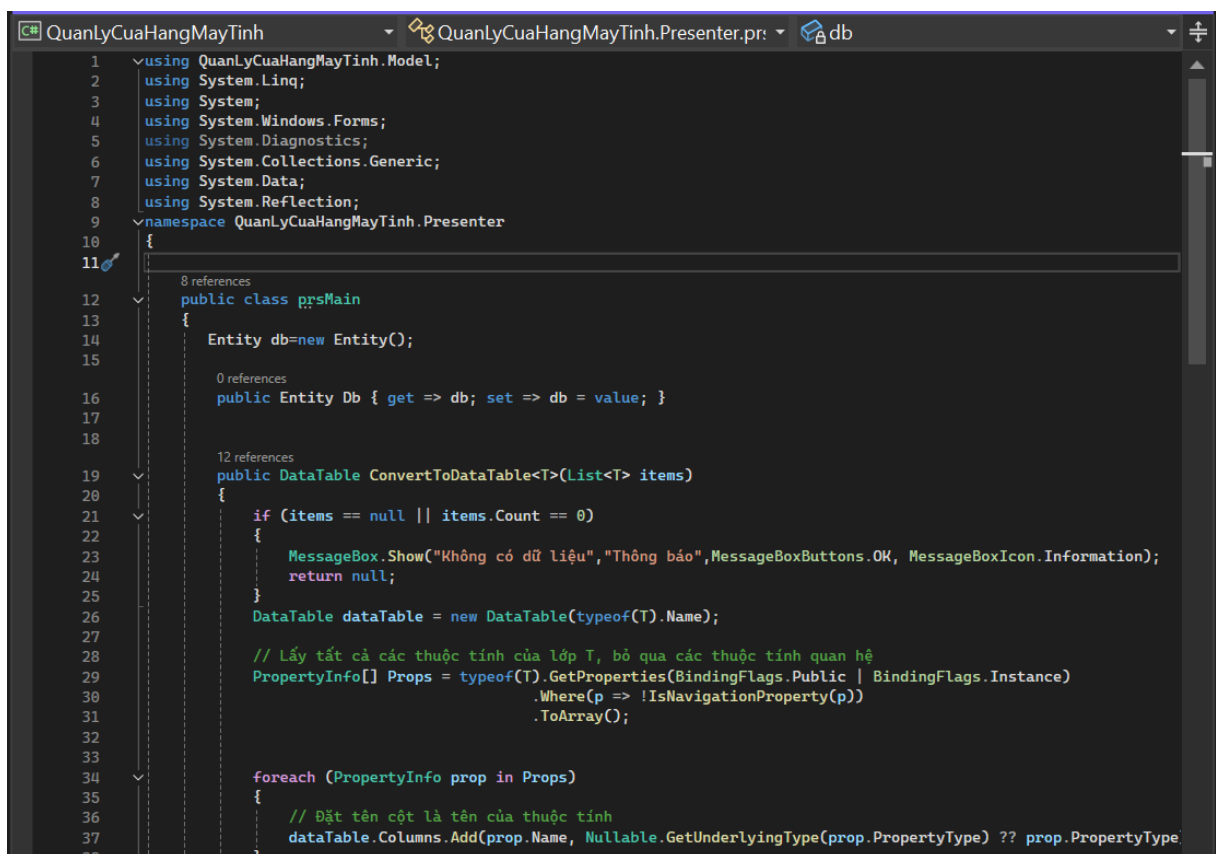
using [Tên Project].Model;

Và bên trong Class tương ứng của nó là dòng code

Entity db= new Entity();

Công dụng của hai dòng này là:

- Dòng thứ nhất: gọi thư mục mà chúng ta muốn sử dụng. Trong trường hợp này là thư mục Model nơi chứa EF của chúng ta.
- Dòng thứ 2: khởi tạo một biến Entity để tương tác với EF. “Entity” ở đây chính là tên mà mình đã đặt ở bước tạo EF trong Model, nếu các bạn đặt tên khác thì phải gọi đúng cái tên đó để đặt tên biến.



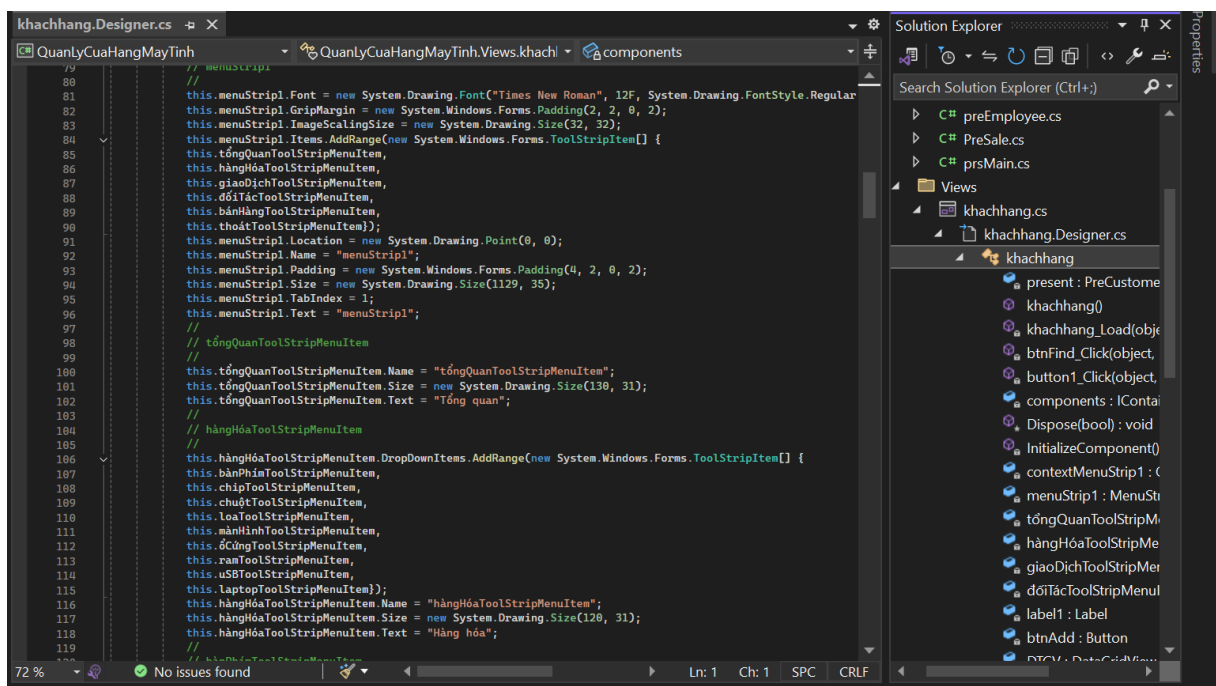
Hình 8: Dòng code mà ở class nào trong Presenter hầu như cũng có.

2.2.1.3. View

Phân lớp View thì có vẻ đơn giản hơn đối với các bạn khi mà nó chứa các Form giao diện như chúng ta đã học ở trên lớp.

Có thể các bạn chưa biết một Form giao diện sẽ chia thành 3 phân vùng gồm có các vùng sau:

- Vùng giao diện: đó chính là phần giao diện mà chúng ta code.
- Vùng code giao diện: đó chính là File code Design.cs, File code này chính là sự code hóa của vùng giao diện. Khi bạn làm bất cứ thứ gì liên quan tới vùng giao diện thì vùng code giao diện cũng sẽ thay đổi theo.
- Vùng Code sự kiện: đó chính là thứ mà khi chúng ta doubleclick vào bất kì một component nào trong form khi đang code thì nó sẽ hiện ra. Các bạn có thể tìm thấy nó ở dưới đây.



Hình 9: Vùng code sự kiện

- Khi bạn doubleclick vào các Item trong phần cửa sổ bên phải sổ xuống thì code của giao diện sẽ hiện ra. Vùng code này có mối liên hệ tương tác mật thiết với vùng code giao diện, vùng code giao diện sẽ sử dụng các method trong này để tiến hành thực thi sự kiện. Vì thế cho nên nhiều khi các bạn ấn doubleClick vào một Component mà lại xóa phương thức đó đi thì ngay lập tức bên vùng giao diện sẽ báo lỗi. Lỗi này là do vùng code giao diện đã lấy phương thức của vùng xử lý sự kiện để chạy nhưng lại không tìm thấy nó trong code sự kiện (vì bạn xóa đi rồi còn đâu mà tìm được). Để

sửa lỗi này thì cực kỳ đơn giản, các bạn hãy tìm đến dòng code lỗi trong phần code giao diện rồi xóa nó đi là xong.

2.2.2. Cách chuyển phần code của mình cho người khác khi sử dụng EF

2.2.2.1. Chuyển cả Project

Để chuyển cả Project cho các thành viên trong nhóm, nhóm mình quyết định sử dụng các công cụ quản lý phiên bản là Git và kho lưu trữ mã nguồn Github. Cách sử dụng Github và Git thì có lẽ các bạn sẽ biết rồi nên mình sẽ không nhắc tới trong bài này. Nếu các bạn chưa biết thì các bạn có thể xem [tại đây!](#)

2.2.2.2. Chuyển đổi EntityFrameWork

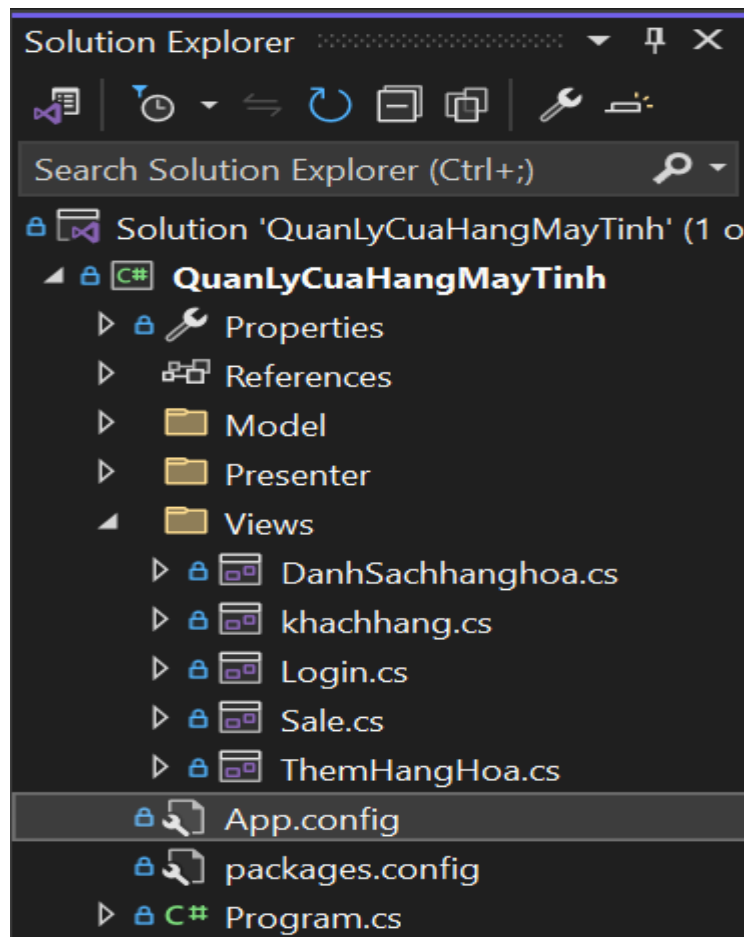
Nếu như bạn Clone Repository từ Github về thì chắc chắn các bạn sẽ không chạy được đâu, bởi vì đơn giản là Project này đang được thiết lập cho Database của các thành viên trong nhóm bạn chứ không phải là thiết lập cho máy của bạn, nếu bạn Clone lần đầu thì đây là giải pháp cho bạn để có thể chạy chương trình trên máy của mình một cách đơn giản nhất.

- Bước 1: Cài đặt Database tại máy của mình. Đây là phần mà khi học trên lớp ai cũng phải biết mình sẽ không nhắc lại⁵.
- Bước 2: mở Project trong VS⁶ trên máy local. Trong cửa sổ Solution Explore tìm đến file App.Config⁷. Sau đó mở nó ra.

⁵ Có 3 cách để cung cấp Database cho người khác

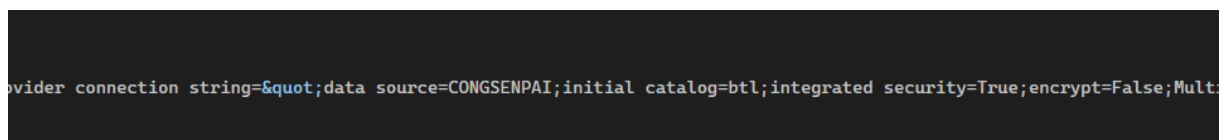
⁶ Visual Studio

⁷ Đây là File chứa cấu hình cho ứng dụng của bạn, được viết dưới dạng XML



Hình 10: App.config trong Solution Explore

- Bước 3: Nhìn xuống phía cuối File sẽ xuất hiện cặp thẻ `<connectionString></connectionString>`. Chúng ta sẽ kéo thanh cuộn của trang sang phía bên phải. Tìm đến đoạn data source =...



Hình 11: Đoạn code có chứa Data source

- Bước 4: Bật SSMS lên trong cửa sổ SSMS⁸ tại dòng Server name. Các bạn hãy copy đoạn text này của mình rồi sau đó paste vào phần data source trong File App.config vừa tìm là đã hoàn thành.

⁸ SQL Server Management Studio

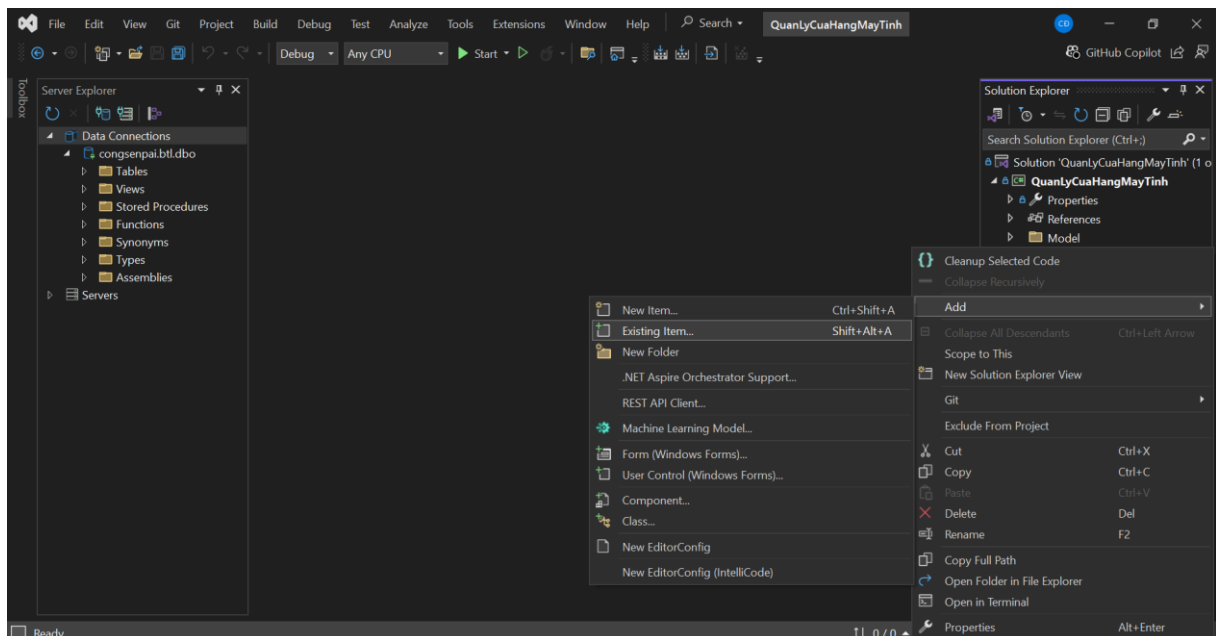
Server type:	Database Engine
Server name:	CONGSENPAI
Authentication:	Windows Authentication

Hình 12: Tên server trong SSMS

2.2.2.3. Cách gộp Form của các thành viên khác vào Project của mình.

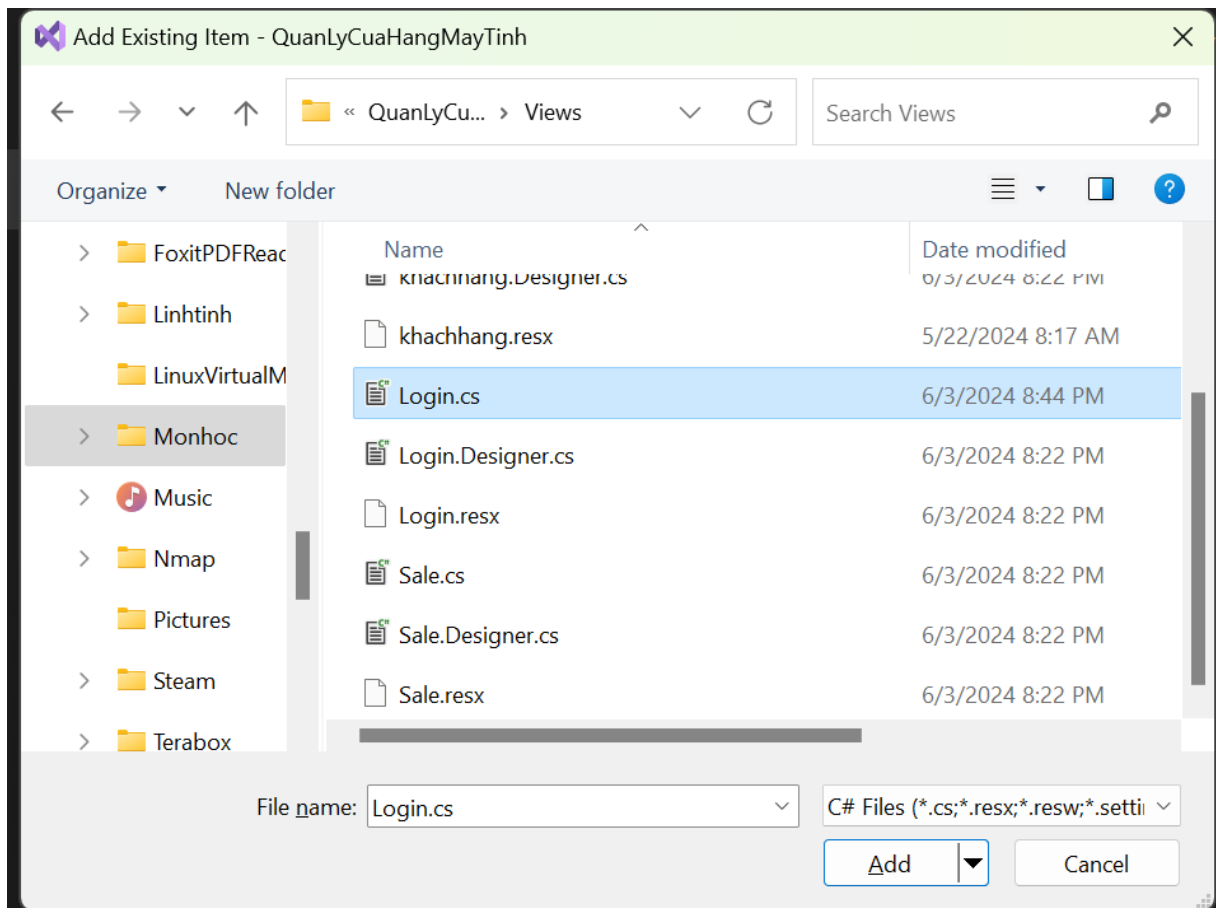
Nếu các bạn gặp phải lỗi liên quan đến Github mà không biết cách khắc phục thì bạn có thể sử dụng phương pháp thủ công là gửi File zip Project trực tiếp cho nhau. Tuy nhiên cần phải chú ý, VS chỉ hỗ trợ copy theo kiểu chỉ định tức là nếu bạn copy Form (theo cách thông thường) thì sẽ không copy được file thiết kế. Để thực hiện quá trình Copy thì các bạn hãy làm theo các bước sau:

- Bước 1: Mở Project trên máy của bạn. Trong cửa sổ Solution Explore, tìm đến thư mục View (vì mình xây dựng dự án theo MVP nên Form sẽ được lưu tại đây). Nhấn phải chuột chọn Add => Existing Item.



Hình 13 :Thêm Item đã tồn tại

- Bước 2: Tìm đến thư mục View của Project mà thành viên khác trong nhóm gửi cho bạn. Sau đó chọn Form bạn muốn thêm. Ví dụ ở đây mình sẽ chọn Form login. Nhớ là hãy chọn file Login.cs để không xuất hiện hiệu quả ngoài ý muốn.



Hình 14: Chọn Form Login

- Bước 3: Nếu chương trình của bạn chạy bị lỗi thì hãy nhấn vào Design của form vừa thêm. Nhấn F7 để vào phần code, theo dõi các dòng code xuất hiện trong bài, rồi sau đó kiểm tra xem các đoạn code đó đã có trong thư mục Presenter chưa. Nếu chưa có thì các bạn cũng làm tương tự các bước trên để copy các File Presenter tương ứng vào Project của mình hoặc chỉ Copy các đoạn code mà mình cần. Chúc các bạn thành công!

CHƯƠNG 3: XÂY DỰNG CÁC TÍNH NĂNG CHO PROJECT

3.1. Chức năng đăng nhập

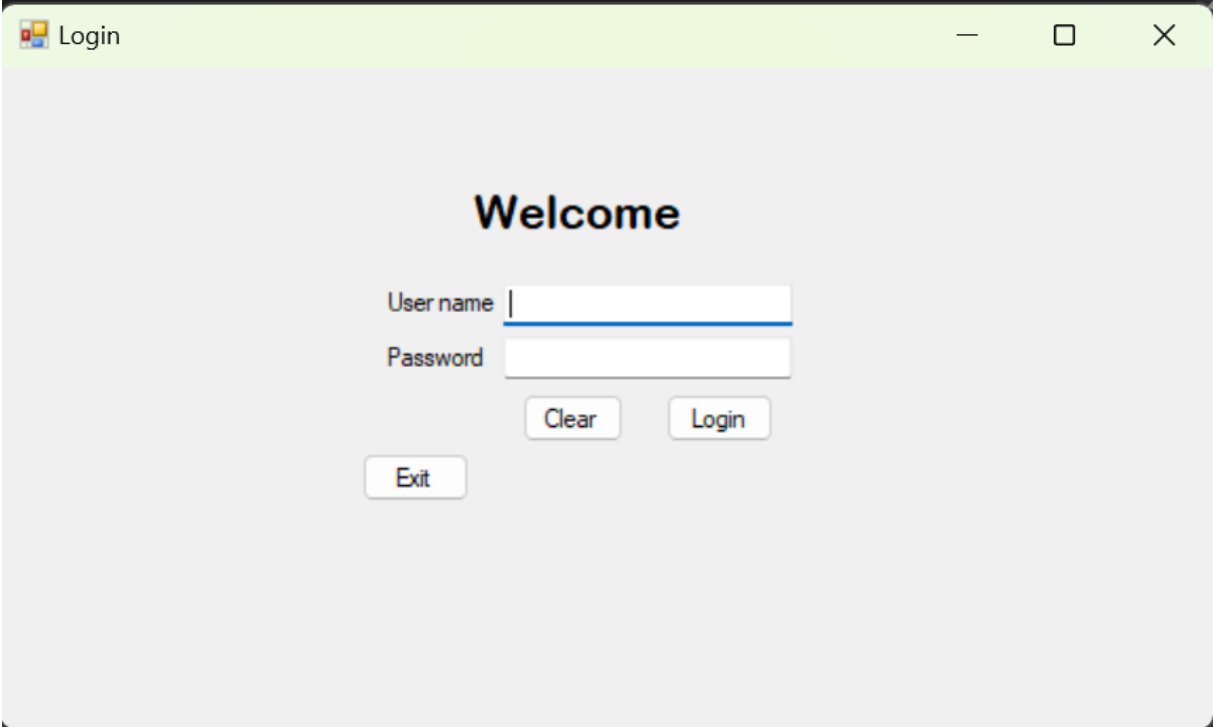
3.1.1. Phân lớp Presenter – Class PreLogin

- ❖ Đầu tiên, biến db để tương tác với cơ sở dữ liệu.
- ❖ Tiếp theo hàm findById(MaNV, password) có nhiệm vụ kiểm tra xem một tài khoản có tồn tại trong cơ sở dữ liệu hay không. Dựa vào thông tin đăng nhập (tên đăng nhập tk và mật khẩu mk), hàm sẽ kiểm tra xem có bất kỳ tài khoản nào trong bảng loginAccounts của đối tượng Db có mã nhân viên (MaNV) và mật khẩu (Password) khớp với thông tin đăng nhập được cung cấp hay không. Nếu có, hàm trả về giá trị true, ngược lại trả về false.

3.1.2. Phân lớp View – Form Login

- ❖ Đầu tiên, ta khởi tạo biến preLogin để truyền tương tác của người dùng với PreLogin.
- ❖ Hàm khởi tạo Form
- ❖ Hàm Login_Load(): Sử dụng để che giấu mật khẩu mà người dùng nhập vào bằng cách set phương thức Password char cho textBox nhập mật khẩu.
- ❖ Hàm btnlogin_Click()
 - Đây là hàm xử lý sự kiện khi người dùng nhấn nút “Đăng nhập”.
 - Trước tiên nó tạo 2 biến MaNV và password
 - Tiếp theo ta thực hiện việc gán giá trị của ô văn bản txtusername (mà người dùng đã nhập) cho biến MaNV và giá trị txtpassword cho biến password.
 - Sau đó, nó gọi phương thức findById(MaNV, password) từ prelogin:
 - Nếu true, chương trình hiển thị form BanHang và ẩn form hiện tại.
 - Nếu false, hiển thị thông báo lỗi và xóa nội dung trong ô nhập liệu.
- ❖ Hàm btnclear_Click(): Thực hiện việc xóa nội dung của hai ô văn bản và đặt trỏ chuột vào ô txtusername. Điều này giúp người dùng dễ dàng nhập thông tin mới sau khi đã xóa dữ liệu cũ.
- ❖ Hàm btnexit_Click(): Trong đoạn mã của bạn xử lý sự kiện khi người dùng nhấn vào nút “Thoát” (Exit). Hãy xem xét từng phần:
 - Hiển thị hộp thoại xác nhận với thông báo “Bạn có chắc muốn thoát” và hai lựa chọn: “Yes” và “No”. Người dùng có thể chọn “Yes” để thoát khỏi ứng dụng hoặc “No” để tiếp tục sử dụng.
 - Xử lý lựa chọn của người dùng:
 - Nếu người dùng chọn “Yes”, ứng dụng sẽ thoát bằng cách gọi Application.Exit();.

- Nếu người dùng chọn “No” , ứng dụng sẽ hiển thị lại cửa sổ hiện tại bằng cách gọi `this.Show()`.



Hình 15: Form Đăng nhập

3.2. Xây dựng MenuStrip

3.2.1. Phân lớp Presenter – Class PreMain

Lớp này được viết nhằm mục đích tổng quát hóa những phương thức chung nhất của các Presenter khác để không phải viết lại trong các lớp khác mà chỉ cần gọi ra. Tiêu biểu nhất là MenuStrip. Cụ thể để xây dựng MenuStrip cần có 2 phương thức sau:

- ❖ Phương thức `AddMenuStripToForm()`:
 - Tham số đầu vào là một Form
 - Khởi tạo một MenuStrip.
 - Tạo các ToolStripMenuItem là các chức năng chính của Form như: Tổng quan, hàng hóa, giao dịch, đối tác, bán hàng, nhập hàng, thoát.
 - Bên trong các chức năng chính đó sẽ là các chức năng phụ ví dụ như trong mục: Giao dịch sẽ có hóa đơn bán, hóa đơn nhập và trong mục Đối tác sẽ có nhà cung cấp, nhân viên và khách hàng.

- Sau đó thêm các sự kiện Open tương ứng. Ví dụ khi người dùng nhấn vào tổng quan thì sẽ gọi phương thức OpenForm để mở Form đích và đóng Form hiện tại lại và giải phóng bộ nhớ.
 - Thêm các item vào MenuStrip và gắn nó vào Form mà mình truyền vào (tham số).
- ❖ Phương thức OpenForm
- Yêu cầu tham số truyền vào là Form hiện tại và Form đích.
 - Form này gồm 3 nhiệm vụ tuần tự:
 - Ẩn Form hiện tại
 - Hiện thị Form đích
 - Đóng form hiện tại và giải phóng bộ nhớ về Form hiện tại.
- ❖ Phương thức ConvertToDataTable()
- Phương thức này nhận tham số đầu vào là một list có kiểu dữ liệu là T (T tùy biến) và trả về một DataTable.
 - Phương thức này sẽ lấy tất cả các “cột” có trong list mà bạn truyền vào, bỏ qua các cột là thuộc tính quan hệ.
 - Đặt tên cột là tên các thuộc tính trong list và thêm vào trong DataTable với kiểu dữ liệu là kiểu dữ liệu tương ứng của thuộc tính trong list.
 - Duyệt qua từng phần tử trong list và thêm vào DataTable
- ❖ Phương thức IsNavigationProperty
- Phương thức này dùng để kiểm tra một thuộc tính có phải là thuộc tính quan hệ hay không bằng cách sử dụng PropertyInfo.
- ❖ Phương thức lấy mã theo tên đã chọn – getCodeByNameChoice
- Tham số truyền vào là tên và kiểu: Ví dụ cần xác định mã Chip thì cần truyền vào 2 tham số là Tên Chip và kiểu “Chip”.
 - Hàm này lấy mã tương ứng từ tên đã chọn trong combobox. Nó tìm kiếm trong bảng dữ liệu và trả về mã nếu tìm thấy.
- ❖ Phương thức CheckExits

- Tham số truyền vào là mã của đối tượng muốn kiểm tra và kiểu kiểm tra. Ví dụ nếu muốn kiểm tra Nhân viên 1 có tồn tại hay không thì truyền tham số như sau: `checkExists("NV01", "NV")`.
- Phương thức này sẽ kiểm tra xem trong DataTable của lớp Pre tương ứng có tồn tại nhân viên có mã nhân viên như tham số truyền vào không. Nếu có trả về true nếu không trả về false

3.2.2. Cách thức triệu gọi trong Form cần MenuStrip

Để triệu gọi MenuStrip tại một Form thì cần làm theo các bước sau:

- ❖ Bước 1: Xác định Form cần triệu gọi.
- ❖ Bước 2: Tìm đến phương thức khởi tạo của Form đó.
- ❖ Bước 3: Gọi phương thức vào các Form cần tạo MenuStrip

```
public BanHang()
{
    InitializeComponent();
    loadCBEmployee();
    loadLBCustomer();
    loadLBComputers();
    prsMain.AddMenuStripToForm(this);
}
```

Hình 16: Triệu gọi MenuStrip trong phương thức khởi tạo của Form bán hàng

3.3. Hàng hóa

3.3.1. Phân lớp View

3.3.1.1. Form Hàng Hóa

- ❖ Khai báo namespace và lớp
 - Khai báo namespace 'Views' của dự án 'QuanLyCuaHangMayTinh'.
 - Khai báo lớp 'FormHangHoa' kế thừa từ lớp 'Form' của Windows Forms.
 - Khai báo một đối tượng 'present' thuộc lớp 'preComputer' - lớp trình bày dữ liệu theo mô hình MVP.
 - Constructor của lớp 'FormHangHoa', gọi phương thức 'InitializeComponent', thiết lập chế độ tự động thay đổi kích thước form 'AutoSizeMode', và gọi hàm

InitializeRadioButtons() để thêm sự kiện checkedChanged cho các rdButton⁹ ở hai FlowLayoutPanel.

➤ Gọi hàm thêm MenuStrip vào Form

❖ Phương thức 'hanghoa_Load'

➤ Hàm này được gọi khi form tải lên

- Nó thiết lập nguồn dữ liệu cho DataGridView 'DTGV' bằng cách gọi hàm 'loadComputer()' từ đối tượng 'present' để tải danh sách máy tính.
- hàm 'ComputerListHasName' để định dạng danh sách đó ví dụ thay đổi các trường dữ liệu tham chiếu để người dùng có cái nhìn trực quan nhất.

❖ Phương thức 'btnFind_Click'

➤ Hàm này được gọi khi người dùng nhấn nút tìm kiếm. Nó kiểm tra xem nút radio nào được chọn và sử dụng giá trị trong hộp tìm kiếm 'txtSearch' để tìm kiếm các máy tính theo tiêu chí tương ứng. Kết quả tìm kiếm sẽ được gán vào DataGridView 'DTGV'.

➤ Hàm này cũng sử dụng cách thức tương tự phương thức hanghoa_Load bên trên.

❖ Phương thức 'btnXem_Click'

➤ Hàm này được gọi khi người dùng nhấn nút "Xem". Nó tải và hiển thị toàn bộ danh sách máy tính trong DataGridView.

➤ Đây là Form ban đầu khi Form load để giúp người dùng xem lại dữ liệu khi chưa thực hiện lọc.

❖ Phương thức 'InitializeRadioButtons'

➤ Hàm này thiết lập sự kiện 'CheckedChanged' cho tất cả các nút radio trong hai FlowLayoutPanel 'flowLayoutPanel1' và 'flowLayoutPanel2'. Sự kiện này sẽ được xử lý bởi hàm 'RadioButton_CheckedChanged'.

⁹ Radio Button

❖ Phương thức 'RadioButton_CheckedChanged'

- Hàm này xử lý sự kiện khi một nút radio được chọn. Nó sẽ bỏ chọn tất cả các nút radio trong FlowLayoutPanel còn lại để đảm bảo chỉ có một nút radio được chọn tại một thời điểm.

❖ Phương thức 'btnAdd_Click'

- Hàm này được gọi khi người dùng nhấn nút "Thêm". Nó sẽ mở form 'ThemHangHoa' để thêm một máy tính mới vào danh sách.
- Form được hiện ra dưới dạng PopUp

❖ Phương thức 'DTGV_CellDoubleClick'

- Hàm này được gọi khi người dùng nhấp đúp vào một ô trong DataGridView. Nó sẽ mở form 'ThemHangHoa' và điền dữ liệu của hàng được chọn vào form này để người dùng có thể xem chi tiết thông tin của hàng hóa đó theo cách trực quan nhất.
- Gọi phương thức DisableAllItem để tắt tất cả textBox và cbBox không cho người dùng thao tác.

Mã máy	Tên máy	Loại máy	Chip	Ổ đĩa
MVT01	Lenovo Ideapad ...	Laptop	CPU Intel Core i5...	Ổ L
MVT02	HP Pavilion 14	Laptop	Intel Core i5-1235U	Ổ H
MVT03	MSI Gaming Brav...	Laptop	GPU AMD Rade...	Ổ M
MVT04	Macbook Air M2	Laptop	Apple M2	Ổ A
MVT05	Dell Inspiron 15	Laptop	CPU Intel core i5...	Ổ D

Hình 17: Form Hàng Hóa

3.3.1.2. Form Thêm Hàng Hóa

❖ Khai báo namespace và lớp

- Khai báo một namespace ‘QuanLyCuaHangMayTinh.Views’ và lớp ‘ThemHangHoa’, kế thừa từ lớp ‘Form’ của Windows Forms
- Biến ‘present’ là một đối tượng của lớp ‘preComputer’ để làm việc với dữ liệu máy tính.
- ‘imageData’ là một mảng byte để lưu trữ dữ liệu hình ảnh.

❖ Phương thức ‘ThemHangHoa’

- Hàm khởi tạo này được gọi khi đối tượng ‘ThemHangHoa’ được tạo ra. Nó khởi tạo các thành phần giao diện.
- Gọi hàm ‘loadCBInFormAdd()’ để tải các item vào combobox tương ứng
- Vô hiệu hóa các nút và ô nhập liệu không cần thiết ban đầu: Nút lưu, bảo hành và số lượng.

❖ Xử lý sự kiện nhấn nút bỏ qua - ‘btnBoQua_Click’

- Khi nút "Bỏ Qua" được nhấn, một hộp thoại xác nhận sẽ hiện lên. Nếu người dùng chọn "Yes", cửa sổ sẽ đóng lại, chọn “No” thì quay lại tiếp tục thêm.

❖ Phương thức ‘loadCBInFormAdd’

- Hàm này tải dữ liệu từ cơ sở dữ liệu và thêm vào các combobox tương ứng trong form. Các loại dữ liệu như loại máy tính, chip, ổ cứng, v.v., được tải qua các phương thức của đối tượng ‘present’.
- Hàm này sẽ lấy tên của các thực thể để truyền các ComboBox tương ứng.

❖ Xử lý sự kiện nhấn nút lưu - ‘btnLuu_Click’

- Khi nút "Lưu" được nhấn, hàm này kiểm tra các giá trị nhập vào, xác thực dữ liệu, và thêm dữ liệu mới vào cơ sở dữ liệu nếu hợp lệ.
 - Gọi hàm getCodeByNameChoice để lấy mã từ tên.
 - Kiểm tra xem máy tính này tồn tại trong DB chưa và trả về kết quả dạng bool.
 - Nếu tồn tại rồi thì kết thúc hàm

- Nếu chưa thì gọi hàm AddMayVT và truyền các tham số tương ứng. Nếu thêm thành công thì đưa ra thông báo và gọi phương thức ResetForm().
- ❖ Phương thức đặt lại form - 'reSetForm'
 - Hàm này đặt lại các giá trị trong form về trạng thái ban đầu sau khi thêm mới dữ liệu thành công.
 - ❖ Phương thức kiểm tra tính hợp lệ - 'checkInvalid'
 - Hàm này kiểm tra xem các ô nhập liệu và combobox đã được điền đầy đủ hay chưa. Nếu thiếu, nó sẽ hiển thị thông báo và yêu cầu người dùng nhập lại.
 - ❖ Xử lý sự kiện nhấn nút chọn ảnh - 'btnChoose_Click'
 - Khi nút "Chọn ảnh" được nhấn, một hộp thoại mở tệp sẽ hiện lên để người dùng chọn ảnh. Ảnh được chọn sẽ được hiển thị trên form và dữ liệu ảnh được lưu vào biến 'imageData'.
 - ❖ Xử lý sự kiện nhấn nút thêm mới - 'btnAdd_Click'
 - Khi nút "Thêm mới" được nhấn, form sẽ được đặt lại và chuyển sang chế độ thêm mới. Các nút và ô nhập liệu cần thiết sẽ được kích hoạt.
 - ❖ Đặt KeyPress cho giá bán và giá nhập.

The screenshot shows a Windows-style popup window titled 'Hàng Hóa'. Inside, there's a grid of input fields and dropdown menus for product information. On the left, there's a placeholder for an image with a 'Chọn' (Choose) button below it. The main area contains fields for 'Tên máy' (Machine name), 'Loại máy' (Machine type), 'Chip', 'Ổ cứng' (Hard drive), 'OCD', 'Cỡ màn hình' (Screen size), 'Ổ USB', 'Hãng' (Brand), 'Tốc độ' (Speed), 'Dung lượng' (Capacity), 'Màn hình' (Screen), 'Chuột' (Mouse), 'RAM', and 'Thời gian bảo hành' (Warranty time). To the right, there are fields for 'Bàn phím' (Keyboard), 'Loa' (Speaker), 'Giá bán' (Selling price), 'Giá nhập' (Buying price), and 'Số lượng' (Quantity). At the bottom left, there's a 'Thời gian hiện tại' (Current time) field showing '6/ 6/2024' and a 'Ghi chú' (Note) text area. At the bottom right, there are three buttons: 'Thêm mới' (Add new), 'Bỏ qua' (Skip), and 'Lưu' (Save). The 'Thêm mới' button is highlighted with a blue border.

Hình 18: Form thêm hàng hóa được hiển thị dưới dạng popup

3.3.2. Phân lớp Presenter – Class PreComputer

❖ Khai báo namespace và lớp

- Khai báo một namespace ‘QuanLyCuaHangMayTinh.Presenter’ và lớp ‘preComputer’ để xử lý dữ liệu.

❖ Thuộc tính và biến của lớp ‘preComputer’

- ‘PreMain prs’: Tạo một đối tượng mới để chuyển đổi danh sách dữ liệu thành bảng dữ liệu (DataTable).
- ‘Entity db’: Tạo một đối tượng mới để truy cập cơ sở dữ liệu.
- Tạo get and set cho ‘db’.

❖ Phương thức checkAmount

- Phương thức này sẽ kiểm tra số lượng của Máy vi tính theo tham số truyền vào là mã máy vi tính.

❖ Phương thức lấy dữ liệu từ cơ sở dữ liệu - ‘List<...>’

- Hàm này lấy danh sách các đối tượng từ cơ sở dữ liệu và trả về dưới dạng danh sách.

❖ Phương thức chuyển đổi danh sách dữ liệu thành DataTable - ‘DataTable load...’

- Các hàm này sử dụng đối tượng ‘prs’ để chuyển đổi danh sách dữ liệu thành bảng dữ liệu.

❖ Phương thức ‘ComputerListHasName’

- Khởi tạo một DataTable mới với các cột tương ứng, sau đó duyệt qua từng dòng trong DataTable đầu vào.
- Truy vấn cơ sở dữ liệu để lấy thông tin chi tiết của từng máy tính và thêm thông tin vào DataTable vừa tạo.
- Trả về DataTable mới chứa đầy đủ thông tin chi tiết của các máy tính (Theo dạng dễ đọc đối với người dùng).

- ❖ Phương thức tìm kiếm dữ liệu trong cơ sở dữ liệu - ‘DataTable find...’
 - Các hàm trong phần này được thiết kế để tìm kiếm thông tin về các máy tính trong cơ sở dữ liệu dựa trên các tiêu chí khác nhau, các tiêu chí này được nhập vào thông qua chuỗi tìm kiếm ‘searchString’
 - ‘Db.MayVTs.Where(...)’ sẽ lọc các máy tính trong cơ sở dữ liệu sao cho thông tin của chúng có chứa phần được nhập trong chuỗi tìm kiếm.
 - ‘ToList()’ chuyển đổi kết quả lọc thành danh sách.
 - ‘prs.ConvertToDataTable(computers)’ chuyển danh sách thành bảng dữ liệu (DataTable).
- ❖ Phương thức thêm máy tính mới vào cơ sở dữ liệu - ‘AddMayVT’
 - Hàm này tạo một đối tượng máy tính mới và gọi hàm addData để truyền dữ liệu xuống phân lớp Model để thêm dữ liệu vào DB.

3.3.3. Phân lớp Model – Class MayVT

Vì đây là EF nên class này là do IDE tự sinh ra, mình sẽ không nói quá nhiều về nó mà sẽ tập trung vào những hàm viết thêm vào class này.

- ❖ Phương thức autoGenCode
 - Phương thức này sẽ tạo một DbContext đến CSDL sau đó sử dụng Linq sắp xếp mã Máy vi tính theo kiểu tăng dần. Vì mã máy vi tính sẽ có dạng MVT___. Nên ta sẽ sử dụng phương thức SubString để lấy tất cả ký tự cuối cùng vì nó sẽ là dạng số.
 - Convert nó sang kiểu Int và cộng thêm 1. Nếu số này < 10 thì sẽ thêm số 0 vào đằng trước chuỗi, nếu > 10 thì giữ nguyên số đó.
 - Ghép MVT và mã số sau khi đã tăng bằng cách cộng chuỗi (sử dụng phương thức ToString và Convert để chuyển đổi qua lại).
- ❖ Phương thức addData
 - Phương thức này khởi tạo một đối tượng MayVT mới theo các tham số truyền vào và MãMVT là mã tự tạo (gọi phương thức autoGenCode).
 - Gọi phương thức Add để thêm Hàng hóa đó vào DB.
 - Gọi phương thức Db.SaveChanges() để lưu lại thay đổi.

3.4. Giao dịch

3.4.1. Phân lớp View

3.4.1.1. Form Hóa Đơn Nhập

- ❖ Khai báo namespace và Class kế thừa từ lớp Form
- ❖ Khai báo các thuộc tính cho Form.
 - Thuộc tính present là một đối tượng PreImport.
 - Thuộc tính employees là một đối tượng PreEmployee¹⁰.
 - Thuộc tính suppliers là một đối tượng PreSupplier¹¹.
- ❖ Phương thức khởi tạo
 - Hàm InitializeComponent dùng để khởi tạo phần tử trong Form
 - Thiết lập AutoSizeMode
 - Gọi hàm loadCB để tải dữ liệu của nhà cung cấp và nhân viên lên cbBox¹².
 - Gọi hàm addMenuStripToForm để thêm MenuStrip vào Form
- ❖ Phương thức loadCB
 - Khai báo các biến DataTable để hứng dữ liệu từ employees.loadEmployee và suppliers.loadSupplier đổ vào.
 - Duyệt qua từng phần tử tương ứng của mỗi dataTable, gán cột dữ liệu tên của nhân viên và nhà cung cấp vào các cbBox tương ứng.
- ❖ Phương thức Order_Load
 - Phương thức này được gọi đến khi Form load.
 - Khi Form load sẽ truyền dữ liệu vào DTGV¹³ được present.loadHoaDonNhap đổ vào.
- ❖ Hàm xử lý sự kiện ctbFind_Click
 - Hàm này sẽ lấy mã nhân viên và mã nhà cung cấp từ tên họ truyền vào trong cbBox thông qua hàm GetCodeByNameChoice.
 - Lấy hai giá trị thời gian từ DateTimePicker 1 và 2.
 - Đổ dữ liệu vào DTGV thông qua hàm FindHoaDon của PreImport
- ❖ Hàm xử lý sự kiện btnXem_click

¹⁰ Thông tin chi tiết vui lòng xem ở các chương sau

¹¹ Thông tin cũng xin vui lòng xem ở các chương sau.

¹² ComboBox

¹³ Data Grid View

- Khi nhấn vào hàm này DTGV sẽ được đổ dữ liệu vào từ hàm loadHoaDonNhap(Dự liệu hiển thị lúc đầu).
- ❖ Hàm xử lý sự kiện button1_Click
 - Khi nhấn vào nút nàyForm nhập hàng sẽ hiển thị và Form hiện tại sẽ được đóng và giải phóng bộ nhớ.

MaHDN	NgayNhap	MaNV	MaNCC	ThanhTien
HDN01	2/15/2024	NV01	NCC01	140000
HDN02	12/31/2023	NV02	NCC02	150000
HDN03	6/4/2024	NV01	NCC01	135000
HDN04	6/4/2024	NV03	NCC02	350000
HDN05	6/4/2024	NV01	NCC02	255000
HDN06	6/4/2024	NV01	NCC01	170000
HDN07	6/4/2024	NV02	NCC01	340000
HDN08	6/4/2024	NV03	NCC02	850000
HDN09	6/4/2024	NV01	NCC02	230000
HDN10	6/4/2024	NV02	NCC02	130000

Hình 19: Form hóa đơn nhập

3.4.1.2. Form Hóa Đơn Bán

- ❖ Khai báo nameSpace và Class kế thừa từ lớp Form
- ❖ Khai báo các thuộc tính cho Form.
 - Thuộc tính present là một đối tượng PreSale.
 - Thuộc tính employees là một đối tượng PreEmployee¹⁴.
 - Thuộc tính customers là một đối tượng PreCustomer¹⁵.
- ❖ Phương thức khởi tạo
 - Hàm InitializeComponent dùng để khởi tạo phần tử trong Form
 - Thiết lập AutosizeMode

¹⁴ Thông tin chi tiết vui lòng xem ở các chương sau

¹⁵ Thông tin cũng xin vui lòng xem ở các chương sau.

- Gọi hàm loadCB để tải dữ liệu của nhà cung cấp và nhân viên lên cbBox¹⁶.
- Gọi hàm addMenuStripToForm để thêm MenuStrip vào Form
- ❖ Phương thức loadCB
 - Khai báo các biến DataTable để hứng dữ liệu từ employees.loadEmployee và customers.loadComputer đổ vào.
 - Duyệt qua từng phần tử tương ứng của mỗi dataTable, gán cột dữ liệu tên của nhân viên và nhà cung cấp vào các cbBox tương ứng.
- ❖ Phương thức Order_Load
 - Phương thức này được gọi đến khi Form load.
 - Khi Form load sẽ truyền dữ liệu vào DTGV¹⁷ được present.loadHoaDonBan đổ vào.
- ❖ Hàm xử lý sự kiện ctbFind_Click
 - Hàm này sẽ lấy mã nhân viên và mã nhà cung cấp từ tên họ truyền vào trong cbBox thông qua hàm GetCodeByNameChoice.
 - Lấy hai giá trị thời gian từ DateTimePicker 1 và 2.
 - Đổ dữ liệu vào DTGV thông qua hàm FindHoaDon của PreSale
- ❖ Hàm xử lý sự kiện btnXem_click
 - Khi nhấn vào hàm này DTGV sẽ được đổ dữ liệu vào từ hàm loadHoaDonBan(Dự liệu hiển thị lúc đầu).
- ❖ Hàm xử lý sự kiện button1_Click
 - Khi nhấn vào nút nàyForm nhập hàng sẽ hiển thị và Form hiện tại sẽ được đóng và giải phóng bộ nhớ.

¹⁶ ComboBox

¹⁷ Data Grid View

	MaHDB	NgàyBan	MaNV	MaKH	TongTien
▶	HDB01	4/24/2024	NV01	KH01	15000
	HDB02	4/30/2024	NV02	KH02	17000
	HDB03	5/1/2024	NV03	KH03	23000
	HDB04	6/2/2024	NV02	KH02	34000
	HDB05	6/4/2024	NV02	KH03	170000
	HDB06	6/5/2024	NV02	KH02	23000
	HDB07	6/6/2024	NV02	KH02	23000

Hình 20: Form hóa đơn bán

3.4.2. Phân lớp Presenter

3.4.2.1. Class PreImport

- ❖ Khai báo namespace và class
 - Khai báo thuộc tính Entity để tương tác với phân lớp Model
 - Tạo getter and setter cho Entity
- ❖ Phương thức listHoaDon
 - Phương thức này cho phép lấy dữ liệu hóa đơn nhập từ Db và trả về dưới dạng list
- ❖ Phương thức loadHoaDonNhap
 - Phương thức này chuyển đổi list bên trên thành kiểu DataTable thông qua hàm convertToDataTable của PreMain.
- ❖ Phương thức FindHoaDon
 - Phương thức này cho phép tìm hóa đơn dựa theo các thông tin truyền vào cụ thể là chuỗi tìm kiếm (Mã hóa đơn), khoảng thời gian, mã nhân viên và mã nhà cung cấp.
 - Đầu tiên phương thức này tạo một list hóa đơn nhập rỗng để hứng dữ liệu tìm kiếm được.

- Tiếp theo thực hiện kiểm tra, nếu mã nhân viên hoặc mã nhà cung cấp không tồn tại thì khi dùng Linq sẽ không sử dụng và ngược lại. Kiểm tra tồn tại hay không thông qua hàm CheckExists.
- Dùng list vừa tạo để lấy kết quả truy vấn sau đó chuyển sang dạng datatable thông qua phương thức convertToDataTable của PreMain.
- Trả về DataTable
- ❖ Phương thức Import
 - Phương thức này sử dụng để khởi tạo một hóa đơn nhập từ các tham số truyền vào
 - Gọi hàm addData trong Model để truyền dữ liệu xuống dưới. Và thực hiện thêm dữ liệu vào DB ở phân lớp Model.
 - Trả về hóa đơn nhập vừa tạo.
- ❖ Phương thức AddImportDetails
 - Phương thức này sử dụng để thêm chi tiết hóa đơn nhập cho hóa đơn.
 - Phương thức này tiến hành duyệt từng dòng dữ liệu trong hóa đơn truyền vào. Với mỗi dòng dữ liệu nó sẽ tiến hành tạo một dòng dữ liệu chi tiết hóa đơn mới với mã MVT, số lượng và đơn giá lấy trong hóa đơn. Mã hóa đơn thì lấy của Hóa đơn nhập trả về khi gọi hàm Import để tạo hóa đơn nhập.
 - Gọi Db.SaveChanges() để lưu thông tin vào DB.
 - Sử dụng Linq để truy vấn đến bảng Máy VT và cộng thêm số lượng theo chi tiết hóa đơn vừa tạo.

3.4.2.2. Class PreSale

- ❖ Khai báo namespace và class
 - Khai báo thuộc tính Entity để tương tác với phân lớp Model
 - Tạo getter and setter cho Entity
- ❖ Phương thức listHoaDon
 - Phương thức này cho phép lấy dữ liệu hóa đơn nhập từ Db và trả về dưới dạng list
- ❖ Phương thức loadHoaDonBan
 - Phương thức này chuyển đổi list bên trên thành kiểu DataTable thông qua hàm convertToDataTable của PreMain.

❖ Phương thức FindHoaDon

- Phương thức này cho phép tìm hóa đơn dựa theo các thông tin truyền vào cụ thể là chuỗi tìm kiếm (Mã hóa đơn), khoảng thời gian, mã nhân viên và mã khách hàng.
- Đầu tiên phương thức này tạo một list hóa đơn bán rỗng để hứng dữ liệu tìm kiếm được.
- Tiếp theo thực hiện kiểm tra, nếu mã nhân viên hoặc mã khách hàng không tồn tại thì khi dùng Linq sẽ không sử dụng và ngược lại. Kiểm tra tồn tại hay không thông qua hàm CheckExists.
- Dùng list vừa tạo để lấy kết quả truy vấn sau đó chuyển sang dạng datatable thông qua phương thức convertToDataTable của PreMain.
- Trả về DataTable

❖ Phương thức Buy

- Phương thức này sử dụng để khởi tạo một hóa đơn bán từ các tham số truyền vào
- Gọi hàm addData trong Model để truyền dữ liệu xuống dưới. Và thực hiện thêm dữ liệu vào DB ở phân lớp Model.
- Trả về hóa đơn bán vừa tạo.

❖ Phương thức AddBuyDetails

- Phương thức này sử dụng để thêm chi tiết hóa đơn bán cho hóa đơn.
- Phương thức này tiến hành duyệt từng dòng dữ liệu trong hóa đơn truyền vào. Với mỗi dòng dữ liệu nó sẽ tiến hành tạo một dòng dữ liệu chi tiết hóa đơn mới với mã MVT, số lượng và đơn giá lấy trong hóa đơn. Mã hóa đơn thì lấy của Hóa đơn bán trả về khi gọi hàm Buy để tạo hóa đơn bán.
 - Gọi Db.SaveChanges() để lưu thông tin vào DB.
 - Sử dụng Linq để truy vấn đến bảng Máy VT và trừ đi số lượng theo chi tiết hóa đơn vừa bán.

❖ Phương thức SumOfOrder

- Phương thức này trả về tổng số lượng hóa đơn dựa theo số dòng của DataTable truyền vào.

❖ Phương thức GetOrderBy...

- Các phương thức này sử dụng Linq để truy vấn dựa trên một list được tạo bên trên thông qua phương thức listHoaDon.
- Khởi tạo một dataTable để lưu kết quả.
- Tạo cột cho DataTable: TotalAmount và Date
- Tính tổng các hóa đơn và gán vào TotalAmount, và ngày tương ứng
- Sử dụng foreach để duyệt qua từng phần tử trong biến đựng kết quả truy vấn và gán kết quả vào DataTable đã tạo ở trên.
- Trả về DataTable.
- ❖ Phương thức GetMostRecentMonday
 - Phương thức này trả về ngày thứ 2 gần nhất hiện tại.
- ❖ Phương thức GetOrderByCurrentWeek
 - Phương thức này sẽ trả về các hóa đơn kể từ ngày thứ hai gần nhất thời điểm hiện tại.
 - Cơ chế hoạt động của nó giống phương thức GetOrderBy... bên trên.

3.4.3. Phân lớp Model

Trong class HoaDonBan và HoaDonNhap cũng có 2 phương thức mà mình thêm vào đó là phương thức autoGenCode và phương thức AddData. Về cơ bản thì 2 phương thức này giống như 2 phương thức cùng tên trong class MayVT đã đề cập ở bên trên và chỉ khác ở tham số truyền vào nên mình sẽ không nhắc lại nó nữa.

3.5. Đối tác

3.5.1. Phân lớp View

3.5.1.1. Form Nhân Viên

- ❖ Khởi tạo NameSpace và class kế thừa từ class Form
- ❖ Phương thức khởi tạo
 - Gọi hàm khởi tạo các components trong Form.
 - Gọi hàm AddMenuStripToForm để thêm menuStrip vào Form.
- ❖ Hàm Nhanvien_Load
 - Được gọi khi Form tải.
 - Đổ dữ liệu tất cả nhân viên vào DTGV.
 - Đặt checked mặc định cho radioButton vào mã nhân viên và kiểu sắp xếp tăng dần.

- ❖ Hàm xử lý sự kiện btnFind_click
 - Thực hiện các kiểm tra sự checked của các radioButton để điều hướng sự tìm kiếm theo đúng luồng mà người dùng yêu cầu.
 - Đối với mỗi luồng đó sẽ gọi các hàm tìm kiếm tương ứng trong lớp PreEmployee.
- ❖ Hàm xử lý sự kiện button1_click (sự kiện khi click của nút xem)
 - Hàm này sẽ hiển thị DTGV ở trạng thái ban đầu.
- ❖ Hàm xử lý sự kiện btnAdd_click
 - Khởi tạo Form thêm nhân viên dưới dạng popup.

	MaNV	TenNV	NgaySinh	Giotinh	Dienthoai
▶	NV01	Chu Thị Anh	1/5/2003	Nữ	
	NV02	Đào Văn Lộc	2/19/2000	Nam	
	NV03	Nguyễn Tiến ...	12/31/1999	Nam	
	NV04	Long Ma Bắc ...	11/23/2003	Nam	0123456789
*					

Hình 21: Form nhân viên

3.5.1.2. Form Khách Hàng

- ❖ Khởi tạo NameSpace và class kế thừa từ class Form
- ❖ Phương thức khởi tạo
 - Gọi hàm khởi tạo các components trong Form.
 - Gọi hàm AddMenuStripToForm để thêm menuStrip vào Form.
- ❖ Hàm Khachhang_Load
 - Được gọi khi Form tải.
 - Đồ dữ liệu tất cả khách hàng vào DTGV.
 - Đặt checked mặc định cho radioButton vào mã nhân viên và kiểu sắp xếp tăng dần.

- ❖ Hàm xử lý sự kiện btnFind_click
 - Thực hiện các kiểm tra sự checked của các radioButton để điều hướng sự tìm kiếm theo đúng luồng mà người dùng yêu cầu.
 - Đối với mỗi luồng đó sẽ gọi các hàm tìm kiếm tương ứng trong lớp PreCustomer.
- ❖ Hàm xử lý sự kiện button1_click (sự kiện khi click của nút xem)
 - Hàm này sẽ hiển thị DTGV ở trạng thái ban đầu.
- ❖ Hàm xử lý sự kiện btnAdd_click
 - Khởi tạo Form thêm khách hàng dưới dạng popup.

	MaKH	TenKH	DiaChi	Dienthoai
▶	KH01	Chị An	Hà Nội	0977005092
	KH02	Anh Thắng	Thanh Hóa	0987654312
	KH03	Cô Linh	Hải Phòng	0876543323
	KH04	Chú Tuấn	Nghệ An	0947357455
	KH05	Em Trang	Hải Dương	0374573474
	KH06	Long Ma Bắc Giang	Hà Nội	0147258369
*				

Hình 22: Form khách hàng

3.5.1.3. Form Nhà Cung Cấp

Form nhà cung cấp được xây dựng tương tự như hai Form bên trên, chỉ khác ở dữ liệu truyền vào và các tham số nên mình cũng sẽ không nhắc lại nữa.

	MaNCC	TenNCC	DiaChi	Dienthoai
▶	NCC01	Công ty TNHH T...	Hà Nội	036666888
	NCC02	Công ty CP Thể ...		0977009092
	NCC03	Long Môn tiêu cục	Hà Nội	0123456789
*				

Hình 23: Form nhà cung cấp

3.5.1.4. Form Thêm Nhân Viên, Thêm nhà cung cấp và Thêm khách hàng.

Vì ba Form này khá giống nhau, nên mình sẽ tiếp tục tổng hợp lại thành một mục. Nguyên do cũng vì câu nói: “Nếu muốn tối ưu code thì đừng bao giờ lặp lại đoạn code của mình”.

❖ Khai báo namespace và class

- Khởi tạo một đối tượng PreEmployee và các class tương ứng để thực hiện việc gọi hàm thêm dữ liệu sau này.

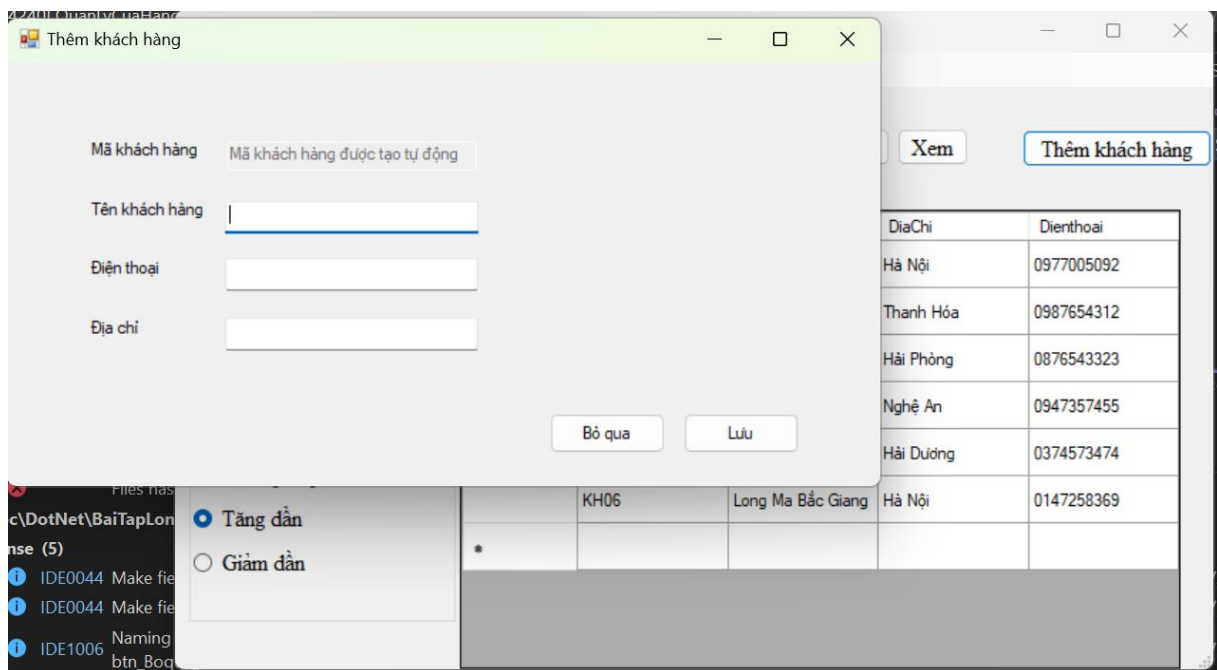
❖ Phương thức khởi tạo

- Tham số đầu vào là một form
- Khởi tạo các component trong form hiện tại.

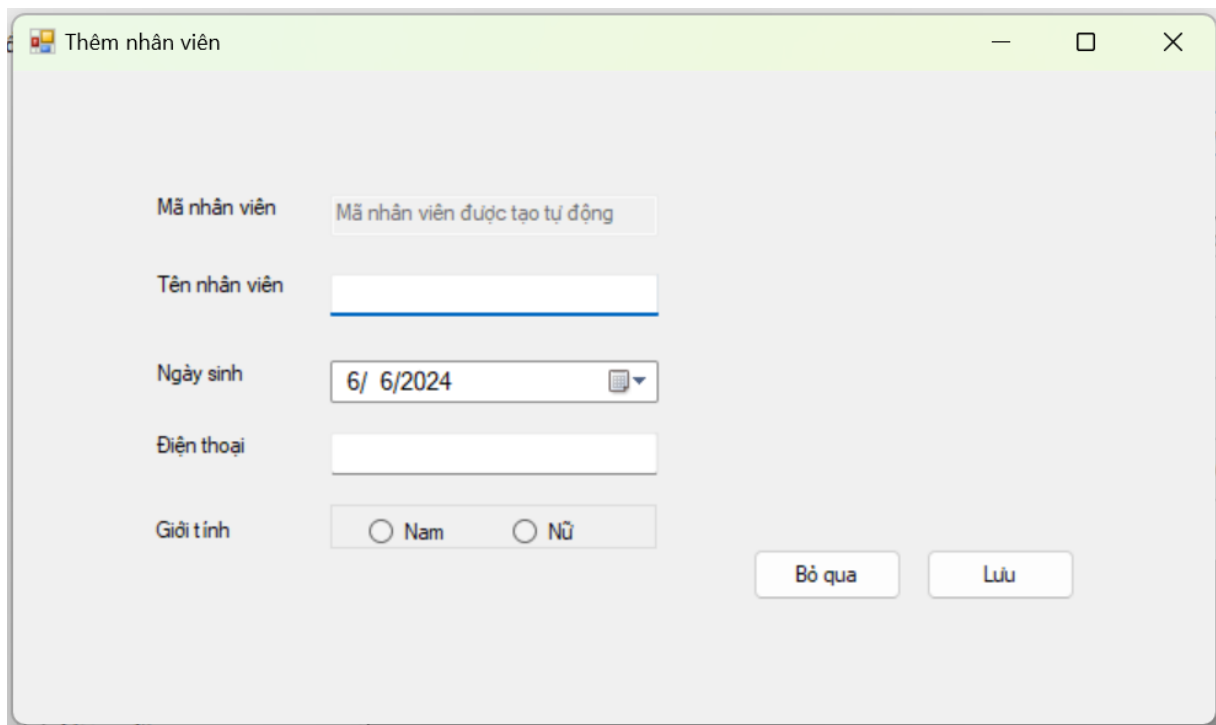
❖ Xử lý sự kiện Add.._load

- Hàm này được gọi khi Form load
 - Đặt text cho mã ...
 - Tắt Enable cho mã.....
 - Enable nút Lưu và nút Bỏ qua.

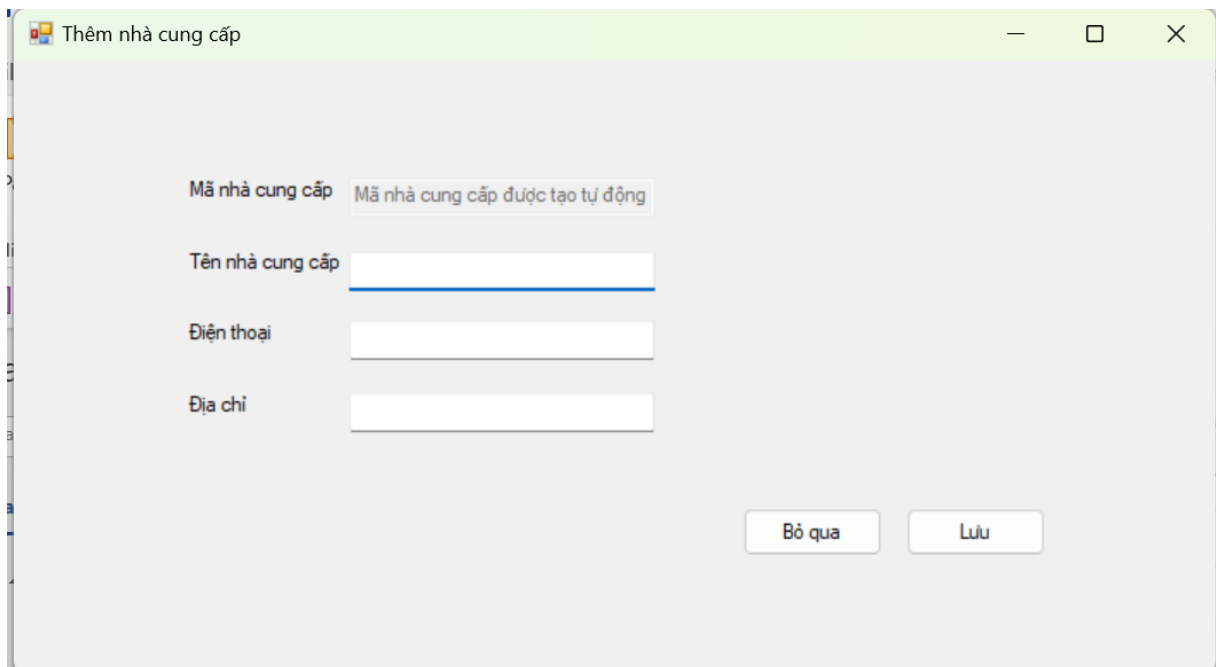
- ❖ Hàm checkKeyPress
 - Hàm này được sử dụng để đặt Keypress cho trường chỉ chứa số.
- ❖ Hàm xử lý sự kiện btnBoqua_click
 - Tắt Form hiện tại và hiển thị Form gốc.
- ❖ Hàm btnLuu_click
 - Hàm này lấy thông tin từ các trường dữ liệu nhập vào sau đó gọi ra hàm Add... ở các phân lớp Pre tương ứng sau đó xuất ra thông báo popUp báo hiệu chèn dữ liệu thành công.
- ❖ Hàm xử lý sự kiện add...FormClose
 - Hàm này xử lý sự kiện khi đóng Form hiện tại thì Form gốc sẽ hiển thị lên màn hình.



Hình 24: Form thêm khách hàng hiển thị dưới dạng popup



Hình 25: Form thêm nhân viên hiển thị dưới dạng popup



Hình 26: Form nhà cung cấp hiển thị dưới dạng popup

3.5.2. Phân lớp Presenter – Customer – Employee – Supplier

Mặc dù có cấu trúc khá là giống nhau. Nhưng điểm nổi bật nhất của 3 Class này là góp mặt cực kỳ nhiều trong các phân vùng khác của chương trình. Nó sẽ cung cấp các phương thức cơ bản để tương tác với các đối tượng ở trong nó. Mình sẽ tóm gọn lại cấu trúc của các class này như sau.

- ❖ Khởi tạo namespace và class
 - Khởi tạo đối tượng Entity để tương tác với phân lớp Model
 - Khởi tạo đối tượng PreMain để sử dụng các phương thức trong đó.
 - Tạo getter and setter cho Entity
- ❖ Phương thức List<T> list...
 - Phương thức này thực hiện quá trình lấy toàn bộ dữ liệu của thực thể T và trả về dưới dạng danh sách.
- ❖ Phương thức load...
 - Phương thức này thực hiện quá trình biến đổi T từ kiểu List về kiểu DataTable
- ❖ Phương thức Add
 - Phương thức này sẽ khởi tạo một đối tượng T mới dựa theo những thông tin truyền vào.
 - Sau đó sẽ gọi hàm addData để truyền dữ liệu xuống phân lớp Model và lưu lại.
- ❖ Phương thức FindBy...
 - Phương thức này sẽ dựa tên hàm và dữ liệu truyền vào để trả về các DataTable khác nhau.
 - Ví dụ nếu sử dụng FindByID thì sẽ yêu cầu người dùng truyền vào ID và kiểu sắp xếp sau đó hàm sẽ sử dụng Linq để lấy dữ liệu dạng list lên sau đó trả về cho người dùng dưới dạng datatable nhờ có hàm convertToDataTable của PreMain.

3.5.3. Phân lớp Model

Các class ở phân lớp Model thuộc vùng đối tác này là: NhanVien, KhachHang và NCC.

Về cơ bản các phương thức được viết thêm cũng tương tự như các phương thức đã viết ở các lớp model khác. Khi tạo các bạn phải chú ý thay đổi các tham số truyền vào cho hợp lý cũng như quy luật của phương thức autoGenCode. Ví dụ: nếu bạn đang thao tác với lớp NhanVien thì phải chỉnh sửa SubString thành 2 thay vì 3 như ở các lớp trên.

3.6. Phân lớp View của Bán hàng và Nhập hàng

Vì phân lớp này sử dụng các class trong phân lớp Present là PreSale, PreImport, PreEmployee, PreCustomer, PreComputer, PreSupplier mà mình đã phân tích bên trên rồi nên mình sẽ không nhắc lại nó ở đây nữa.

❖ Khai báo lớp và các trường:

- Lớp BanHang và NhapHang là lớp thừa kế từ Form.
- Khởi tạo nhiều đối tượng presenter (PreSale, PreEmployee, PreCustomer, PreComputer, PreMain, PreSupplier) được khởi tạo.
- originalItemsCus, OriginalItemsSup, originalItemsCom là các danh sách chứa dữ liệu tên của khách hàng và máy tính.
- hoadon là một DataTable để lưu trữ chi tiết hóa đơn.

❖ Phương thức khởi tạo

- Constructor khởi tạo form, tải dữ liệu nhân viên, khách hàng, nhà cung cấp và máy tính vào các control tương ứng và thêm menu strip vào form.

❖ Phương thức FocusChanged sẽ xác định tính ẩn hiện của các listBox là lbComputer, lbSupplier và lbCustomer.

- Ta sẽ sử dụng hàm này để kiểm soát sự xuất hiện của các listBox trên dựa trên các tham số truyền vào.

❖ Phương thức xử lý sự kiện Sale_load

- Phương thức này sẽ được gọi khi Form tải.
- Gán Text cho nhãn lbTime là thời gian hiện tại.
- Gọi hàm Start của control Timer1.
- Ẩn các listBox

❖ Phương thức xử lý sự kiện timer1_Tick

- Phương thức này sẽ tự động cập nhật lại lbTime.Text sau mỗi lần tick thành thời gian hiện tại.

❖ Phương thức LoadCBEmployee

- Phương thức này có tác dụng đổ tên nhân viên vào ComboBox.

❖ Phương thức LoadLBComputer, LoadLBSupplier và LoadLBCustomer

- Các phương thức này được sử dụng để truyền dữ liệu tên của máy tính, nhà cung cấp và khách hàng vào trong listBox cũng như là truyền dữ liệu vào các list OriginalItem đã khai báo ở bên trên.

❖ Phương thức xử lý sự kiện khi ô tìm kiếm (ô tìm kiếm sản phẩm, nhà cung cấp và tìm kiếm khách hàng) thay đổi

- Hiện ô listBox lên.
- Kiểm tra nếu ô này Null thì thực hiện tìm kiếm trong danh sách các tên gần giống và gán vào một danh sách khác.
- Xóa tất cả Item đang có trong ListBox.
- Thêm vào các item vừa tìm được.
- Đặt vị trí con trỏ bắt đầu sau mỗi lần ô text thay đổi thành vị trí cuối của chuỗi.
- Hàm này sẽ tạo ra hiệu ứng gõ tới đâu thì sản phẩm sẽ xuất hiện đến đấy.

❖ Hàm CheckExit

- Hàm này có tác dụng kiểm tra xem trong hóa đơn (DTGV) đã tồn tại sản phẩm đó hay chưa.
- Nếu tồn tại rồi thì trả về vị trí còn nếu không thì trả về -1.

❖ Hàm xử lý sự kiện btnAdd_click

- Hàm này có tác dụng thêm sản phẩm vào hóa đơn.
- Gọi hàm FocusChanged(true,true) để ẩn cả lbComputerItem và lbCustomer.
- Kiểm tra xem nhập số lượng chưa.
- Kiểm tra xem nhập sản phẩm chưa.
- Kiểm tra xem số lượng còn đủ trong kho hàng không. Nếu không đủ số lượng thì đưa ra thông báo.
- Nếu đủ rồi thì kiểm tra xem sản phẩm đó có trong hóa đơn chưa.

- Nếu chưa có thì tạo một dòng dữ liệu mới sau đó thêm vào Datatable hóa đơn.
- Nếu có rồi thì tăng số lượng lên bằng số lượng ban đầu cộng với số lượng lúc sau.
- Cập nhật lại DTGV bằng thông tin trong hóa đơn vừa tạo.
- Khi thêm thì cập nhật lại tổng tiền vào lbTotalCost, lbCuspay và PayBack.
- ❖ Hàm xử lý sự kiện Cuspay_KeyPress
 - Hàm này chỉ cho phép người sử dụng nhập vào số
- ❖ Hàm xử lý sự kiện Cuspay_TextChanged
 - Hàm này có tác dụng là mỗi khi txtCuspay thay đổi thì payback cũng sẽ thay đổi theo. Tức là mỗi khi người dùng nhập số tiền thanh toán vào thì số tiền cần trả lại cũng sẽ thay đổi.
- ❖ Hàm xử lý sự kiện btnBuy_click
 - Hàm này được gọi khi người dùng nhấn vào nút thanh toán
 - Nó sẽ tiến hành kiểm tra xem trong hóa đơn có hàng hóa không. Nếu không có thì đưa ra thông báo.
 - Khai báo biến dateTime chứa ngày giờ bán, biến string chứa mã nhân viên, mã khách hàng, biến double chứa tổng tiền.
 - Kiểm tra label tổng tiền. Nếu chiều dài tổng tiền bằng 0 thì đưa ra thông báo chưa có sản phẩm. Nếu có thì Convert sang dạng double rồi gán vào biến phía trên.
 - Kiểm tra xem cbBox nhân viên có giá trị chưa. Nếu chưa thì đưa ra thông báo và Focus vào đó. Nếu có thì gọi hàm getCodeByNameChoice của PreMain. Tương tự với textBox khách hàng và nhà cung cấp.
 - Gọi hàm addBuyDetails của PreSale và truyền các tham số vào. Tương tự với addImportDetails của PreImport.
 - Gọi hàm ResetOrder để reset lại hóa đơn.

❖ Hàm ResetOrder

- Hàm này có tác dụng xóa hết text ở các ô trong hóa đơn và DTGV trừ ô Nhân viên vì có thể vẫn là nhân viên đó bán.

❖ Hàm xử lý sự kiện khi người dùng chọn Item trong listBox (lbComputerItem, lbSupplier và lbCustomer).

- Các hàm này có tác dụng gán thông tin của item người dùng chọn vào ô text tương ứng.

- Ẩn ListBox này sau khi đã chọn.

❖ Các sự kiện để điều hướng sự xuất hiện và ẩn của các listBox

- Các hàm này sử dụng hàm FocusChanged theo tham số đầu vào để điều hướng.

❖ Xử lý sự kiện khi người dùng nhấn vào nút thêm hóa đơn mới

- Gọi hàm resetOrder.

- Xóa Employee, Customer hiện tại ra khỏi cbBox.

❖ Xử lý sự kiện khi người dùng nhấn vào nút “dấu cộng” bên cạnh textbox khách hàng và nhà cung cấp

- Mở Form thêm mới khách hàng hoặc nhà cung cấp

Hình 27 : Form bán hàng

Hình 28: Form nhập hàng

3.7. Phân lớp View – Form Tổng quan

❖ Khai báo nameSpace và khởi tạo

- Tongquan: Đây là lớp kế thừa từ Form, tạo ra giao diện cho ứng dụng.
- PreSale order = new PreSale(): Tạo một đối tượng PreSale để truy xuất dữ liệu đơn hàng.
- PreMain prsMain = new PreMain(): Tạo một đối tượng PreMain.
- Tongquan(): Hàm khởi tạo, gọi phương thức InitializeComponent để khởi tạo các thành phần của form và thêm MenuStrip vào form bằng PreMain.AddMenuStripToForm(this).
- Form1_Load: Phương thức xử lý sự kiện khi form được tải, gọi các phương thức để hiển thị biểu đồ và tải thông tin doanh số bán hàng.

❖ Phương thức khởi tạo biểu đồ

- InitializeChart: Phương thức để khởi tạo biểu đồ. Nó xóa các chuỗi và khu vực biểu đồ hiện có, sau đó thêm chuỗi mới và thiết lập tiêu đề cho các trục X và Y. Loại biểu đồ được đặt là cột (Column).

❖ Các phương thức hiển thị biểu đồ

- **showChartByMonth:** Phương thức để hiển thị biểu đồ doanh thu theo tháng. Nó khởi tạo biểu đồ và lấy dữ liệu đơn hàng trong tháng hiện tại, sau đó thêm các điểm dữ liệu vào biểu đồ.
- **showChartByDayOfWeek:** Phương thức để hiển thị biểu đồ doanh thu theo ngày trong tuần. Nó lấy dữ liệu đơn hàng trong tuần hiện tại và thêm các điểm dữ liệu vào biểu đồ theo từng ngày trong tuần.

❖ Xử lý các sự kiện nút bấm

- **btn_Theongay_Click:** Phương thức xử lý sự kiện khi người dùng nhấn nút để hiển thị biểu đồ theo ngày.
- **btn_Theothu_Click:** Phương thức xử lý sự kiện khi người dùng nhấn nút để hiển thị biểu đồ theo ngày trong tuần.

❖ Phương thức tải tổng doanh số ngày

- **LoadDayTotalSale:** Phương thức để tải tổng doanh số của ngày hiện tại và hiển thị chúng lên các nhãn **lbDailySale1**, **lbDailySale2** và **lbDailyOrderCount**.

❖ Các hàm so sánh doanh số

- **LoadComparePrevDay:** Phương thức để so sánh doanh số của ngày hiện tại với ngày hôm qua và hiển thị phần trăm tăng trưởng hoặc giảm trên nhãn **lbComparePrevDay**.
- **LoadComparePrevMonth:** Phương thức để so sánh doanh số của tháng hiện tại với tháng trước và hiển thị phần trăm tăng trưởng hoặc giảm trên nhãn **lbComparePrevMonth**.

KẾT LUẬN

Qua bài báo cáo trên nhóm đã đạt được những thành tựu của học phần như: tư duy lập trình hướng đối tượng, cách xây dựng một hệ thống winform bằng .Net Framework, cách thức kết nối cơ sở dữ liệu,... Bên cạnh đó nhóm cũng đã chủ động tiếp thu những kiến thức mới mạnh mẽ - bảo mật – dễ thao tác – dễ bảo trì để triển khai cho hệ thống mà nhóm đang nhắm đến.

Theo quan điểm của nhóm, bài báo cáo đã đạt được mức cao trong quá trình triển khai và phân tích bài toán. Hi vọng bạn đọc có thể tiếp nhận bài báo cáo này một cách toàn vẹn, hiểu được những tâm tư của nhóm. Nhóm mình sẽ rất vui nếu nhận được những lời nhận xét tích cực từ bạn đọc, nếu có gì sai sót trong quá trình triển khai dự án hay là tìm ra lỗ hổng nào đó, nhóm cũng xin được nhận những phản hồi để có thể cải thiện hệ thống tốt hơn nữa trong những phiên bản sau này.

TÀI LIỆU THAM KHẢO

HowKteam. (2017). *Youtube*. Được truy lục từ Youtube.com:

<https://www.youtube.com/watch?v=dtYVRWfGhzI&list=PL33lvabfss1y2T7yK--YZJHCsU7LZVzBS>

Nhung, T. B. (2023). *Slide bài giảng bộ môn phân tích thiết kế hệ thống*. Hà Nội: Học Viện Ngân Hàng.

Trang, N. T. (2023). *Slide bài giảng môn thiết kế cơ sở dữ liệu*. Hà Nội: Học Viện Ngân Hàng.

Trang, N. T. (2024). *Slide bài giảng môn Hệ quản trị cơ sở dữ liệu*. Hà Nội: Học Viện Ngân Hàng.

Tú, T. L. (2024). *Giáo trình C#*. Học viện Ngân Hàng.