**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

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



**BÁO CÁO ĐỒ ÁN**

**LẬP TRÌNH TRÊN WINDOWS**

**QUẢN LÝ RẠP PHIM**

**SVTH: Võ Minh Hiếu**

**MSSV: 17110136**

**SVTH: Lê Minh Tiến**

**MSSV: 17110236**

**SVTH: Vũ Hoàng Hiệp**

**MSSV: 17110138**

**GVHD: Lê Thanh Vinh**

**TP. Hồ Chí Minh, tháng 5 năm 2018**

**LỜI NÓI ĐẦU**

Lời nói đầu tiên, nhóm xin cảm ơn quý thầy cô đã đọc bài báo cáo và chấm điểm đồ án cho nhóm. Nói sơ lược về nội dung báo cáo, bài báo cáo là tổng hợp những nội dung quan trọng mà nhóm sử dụng để làm đồ án. Nội dung được dựa trên những gì đã học của môn “Lập trình trên Windows”. Như tiêu đề ở trang bìa, đề tài nghiên cứu của nhóm là “Quản lý rạp phim”. Đây là lần đầu nhóm thực hiện một đồ án như vậy, nên không thể tránh khỏi những sai sót. Kính mong quý thầy cô và bạn đọc thông cảm bỏ qua. Chân thành cảm ơn.

MỤC LỤC

[I. Lý do chọn đề tài 8](#_Toc9105489)

[II. Giới thiệu phần mềm 8](#_Toc9105490)

[III. Nội dung 8](#_Toc9105491)

[VERSION 1: ADO.NET Library 8](#_Toc9105492)

[1. ADO.NET là gì? 8](#_Toc9105493)

[2. Software architecture 9](#_Toc9105494)

[a) DBMain 9](#_Toc9105495)

[b) BSLayer 11](#_Toc9105496)

[c) renderSeat 16](#_Toc9105497)

[3. Giao diện người dùng 19](#_Toc9105498)

[a) Sử dụng User Controls. 19](#_Toc9105499)

[b) QuanLyNhanVien.cs 22](#_Toc9105500)

[c) QuanLyKhachHang.cs 28](#_Toc9105501)

[d) QuanLyRap.cs 30](#_Toc9105502)

[e) PhimUC.cs 32](#_Toc9105503)

[f) DatVe.cs 33](#_Toc9105504)

[g) QuanLySuatChieu.cs 39](#_Toc9105505)

[h) QuanLyChucVu 39](#_Toc9105506)

[4. Database Diagram 40](#_Toc9105507)

[i) Bảng NhanVien 40](#_Toc9105508)

[j) Bảng DatVe 41](#_Toc9105509)

[k) Bảng KhachHang 41](#_Toc9105510)

[l) Bảng Rap 42](#_Toc9105511)

[m) Bảng Phim 42](#_Toc9105512)

[VERSION 2: LinQ to SQL Framework 43](#_Toc9105513)

[1. LinQ là gì? 43](#_Toc9105514)

[2. Vì sao LinQ ra đời? 43](#_Toc9105515)

[3. Lý do tại sao sử dụng SQL 44](#_Toc9105516)

[4. Software architecture 45](#_Toc9105517)

[a) BLChucVu 46](#_Toc9105518)

[b) BLKhachHang 48](#_Toc9105519)

[c) BLNhanVien 49](#_Toc9105520)

[d) BLPhim 50](#_Toc9105521)

[e) BLRap 51](#_Toc9105522)

[f) BLSuatChieu 52](#_Toc9105523)

[g) BLDatVe 53](#_Toc9105524)

[IV. Kết luận 54](#_Toc9105525)

[1. Ưu điểm: 54](#_Toc9105526)

[2. Nhược điểm: 54](#_Toc9105527)

[3. Hướng phát triển: 55](#_Toc9105528)

DANH MỤC CÁC HÌNH

[2. Software architecture 9](#_Toc9093627)

[2.1. Tiền khai báo các biến 11](#_Toc9093628)

[2.2. Code BDMain 11](#_Toc9093629)

[2.3. Code BLDatVe 13](#_Toc9093630)

[2.4. Code BLKhachHang 14](#_Toc9093631)

[2.5. Code BLNhanVien 15](#_Toc9093632)

[2.6. Code BLPhim 15](#_Toc9093633)

[2.7. Code BLRap 16](#_Toc9093634)

[2.8. Code BLChucVu 17](#_Toc9093635)

[2.9. Code BLSuatChieu 17](#_Toc9093636)

[2.10. Các biến cơ bản của renderSeat 18](#_Toc9093637)

[2.11. Hàm drawTable 18](#_Toc9093638)

[2.12. Hàm CellTable 19](#_Toc9093639)

[2.13. Hàm renderSeat 20](#_Toc9093640)

[2.14. Class buttonObject 20](#_Toc9093641)

[3. Giao diện người dùng 20](#_Toc9093642)

[3.1. User Control 21](#_Toc9093643)

[3.2. Form1.cs 21](#_Toc9093644)

[3.3. Giao diện chính 22](#_Toc9093645)

[3.4. Giao diện QLNhanVien 23](#_Toc9093646)

[3.5. Tiền khai báo 23](#_Toc9093647)

[3.6. LoadData 24](#_Toc9093648)

[3.7. Button Add 25](#_Toc9093649)

[3.8. Button Edit 25](#_Toc9093650)

[3.9. DataGridView\_CellClick 26](#_Toc9093651)

[3.10. Button Deleted 26](#_Toc9093652)

[3.11. Button Save 27](#_Toc9093653)

[3.12. Combobox SelectecdIndexChanged 27](#_Toc9093654)

[3.13. Button Quản Lý 27](#_Toc9093655)

[3.14. Các hàm đếm ngày 28](#_Toc9093656)

[3.15. Event Textbox\_KeyUp 28](#_Toc9093657)

[3.16. Button Thống kê 29](#_Toc9093658)

[3.17. Giai diên QLKhachHang 29](#_Toc9093659)

[3.18. LoadData 30](#_Toc9093660)

[3.19. Button Sửa và Thêm 30](#_Toc9093661)

[3.20. Button Lưu và Xoá 31](#_Toc9093662)

[3.21.Giao diện QLRap 31](#_Toc9093663)

[3.22. Code quản lý rạp 33](#_Toc9093664)

[3.23.Giao diện phim đang chiếu 33](#_Toc9093665)

[3.24. Button Brownse 34](#_Toc9093666)

[3.25. Biến chung Đặt vé 34](file:///D:\Temp\clone4\FinalWindows\Document\BaoCao.docx#_Toc9093667)

[3.26. Hàm load UC 35](#_Toc9093668)

[3.27. Hàm set Combobox 35](#_Toc9093669)

[3.28. Hàm setRaptxt 36](#_Toc9093670)

[3.29. Hàm seatpanelPaint 36](#_Toc9093671)

[3.30. Hàm getBookedSeat 36](#_Toc9093672)

[3.31. Sự kiện SelectedIndexChanged của ComboBox 37](#_Toc9093673)

[3.32. Event suất chiếu 37](#_Toc9093674)

[3.33. Set combobox suất chiếu 38](#_Toc9093675)

[3.34. Set combobox nhân viên 38](#_Toc9093676)

[3.35. Sự kiện click ghế 38](#_Toc9093677)

[3.36. Rút reload 39](#_Toc9093678)

[3.37. Hàm checkMaKH 39](#_Toc9093679)

[3.38.Hàm tìm mã phim 39](#_Toc9093680)

[3.39. Hàm set MaKH 40](#_Toc9093681)

[3.40. Giao diện QLSuatChieu 40](#_Toc9093682)

[3.41. Giao diện QLChucVu 40](#_Toc9093683)

[4. Database Diagram 41](#_Toc9093684)

[4.1. Database Diagram 41](#_Toc9093685)

[4.2. Design NhanVien 41](#_Toc9093686)

[4.3. Data NhanVien 42](#_Toc9093687)

[4.4. Design DatVe 42](#_Toc9093688)

[4.5. Data DatVe 42](#_Toc9093689)

[4.6. Design KhachHang 43](#_Toc9093690)

[4.7. Data KhachHang 43](#_Toc9093691)

[4.8. Design Rap 43](#_Toc9093692)

[4.9. Data Rap 43](#_Toc9093693)

[4.10. Design của Phim 44](#_Toc9093694)

[4.11. Data của Phim 44](#_Toc9093695)

[3. Lý do tại sao sử dụng SQL 45](#_Toc9093698)

[3.1. DataModel Object. 45](#_Toc9093699)

[4. Software architecture 46](#_Toc9093700)

[4.1. QuanLyRapPhimDataContext 47](#_Toc9093701)

[4.2. Lấy thông tin Object 47](#_Toc9093702)

[4.3. Thêm thông tin Object 48](#_Toc9093703)

[4.4. Xoá thông tin Object 48](#_Toc9093704)

[4.5. Cập nhật thông tin Object. 49](#_Toc9093705)

[4.6. BLKhachHangLinQ1 50](#_Toc9093706)

[4.7. BLKhachHangLinQ2 50](#_Toc9093707)

[4.8. BLNhanVienLinQ 51](#_Toc9093708)

[4.9. BLPhimLinQ 52](#_Toc9093709)

[4.10. BLRapLinQ 53](#_Toc9093710)

[4.11.BLSuatChieuLinQ 54](#_Toc9093711)

## Lý do chọn đề tài

Là sinh viên, chúng ta có khá nhiều thời gian để giải trí sau những giờ học tập căng thẳng. Có rất nhiều các để giải trí, nhưng trong số vô vàng lựa chọn, sinh viên của chúng ta thường lựa chọn đó là đi xem phim ở các rạp phim. Lý do có vẻ đơn giản, nhưng đó chính là nguồn động lực để thúc đẩy nhóm chọn đề tài liên quan đến lĩnh vực này – “Phần mềm quản lý rạp phim”

## Giới thiệu phần mềm

Phần mềm là 1 chuỗi các hệ thống quản lý bao gồm: Quản lý nhân viên, quản lý các khách hàng, quản lý lịch phát các phim trong rạp. Các tính năng quản lý cao cho người dùng. Các tính năng này sẽ được nói rõ hơn ở phần nội dung.

Phần mềm dùng ADO.NET để kết nối với Database và thao tác trên nó.

## Nội dung

# VERSION 1: ADO.NET Library

### ADO.NET là gì?

ADO.NET là một bộ các thư viện hướng đối tượng (OOP) cho phép bạn tương tác với dữ liệu nguồn. Thông thường thì dữ liệu nguồn là một cơ sở dữ liệu (database), nhưng nó cũng có thể là file text, exel hoặc XML. Theo những mục tiêu của hướng dẫn này, chúng ta sẽ chỉ xem xét tới cách ADO.NET làm việc với database.

Như bạn có thể biết rằng, có rất nhiều loại database hiện nay như Microsoft SQL Server, Microsoft Access, Oracle, Borland Interbase, và IBM DB2,… Để làm rõ hơn phạm vi của loạt bài này, tất cả ví dụ sẽ sử dụng SQL Server.

### Software architecture

Code chính của chương trình được chia làm 3 phần với BSLayer (Các query tương tác với database), DBLayer (Thiết lập các lênh truy vấn), renderSeat (tạo các ghế của rạp).

Đa số code được thực hiện qua DataSet. Một DataSet tương tự như một tập tin database vật lý hoàn chỉnh nhưng được lưu trong bộ nhớ. DataSet bao gồm các DataTable, DataTable bao gồm các DataColumn, DataRow, các constraint.

#### DBMain

Là tầng kết nối với database. Là trung gian để mở các kết nối dẫn các câu truy vấn từ client tới database. Được thực hiện qua DataSet. Với ExecuteNonQuery: kết quả trả về là số dòng bị ảnh hưởng.

Khởi tạo các đối tượng ban đầu để kết nối với database:

* SqlConnection

Để tương tác với database, bạn phải có một kết nối tới nó. Kết nối giúp xác định database server, database name, user name, password, và các tham số cần thiết để kết nối tới database. Một đối tượng connection được dùng bởi đối tượng command vì thế chúng sẽ biết database nào để thực thi lệnh.

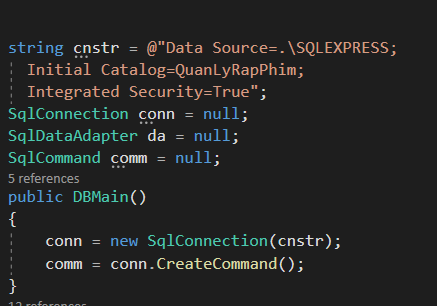
* SqlCommand

Quá trình tương tác với database cần phải biết hành động nào bạn muốn xảy ra. Điều này được thực hiện bởi đối tượng command. Bạn dùng đối tượng command để gửi một câu lệnh SQL tới database. Một đối tượng command dùng một đối tượng connection để xác định database nào sẽ được truy xuất. Bạn có thể dùng một đối tượng command riêng lẻ để thực thi lệnh trực tiếp, hoặc để gắn một tham chiếu của đối tượng command cho một SqlDataAdapter – đối tượng giữ các command sẽ làm việc trên một nhóm dữ liệu như sẽ đề cập tới trong phần dưới.

* SqlDataAdapter

Đôi lúc dữ liệu mà bạn làm việc là read-only và bạn ít khi cần thay đổi dữ liệu nguồn. Vài trường hợp cần lưu trữ tạm dữ liệu trong bộ nhớ để hạn chế truy xuất đến database. Data adapter làm điều này dễ dàng bằng cách giúp bạn quản lý dữ liệu trong chế độ ngắt kết nối. Data adapter sẽ đổ vào DataSet khi đọc dữ liệu và thực hiện thay đổi dữ liệu một lượt vào database.

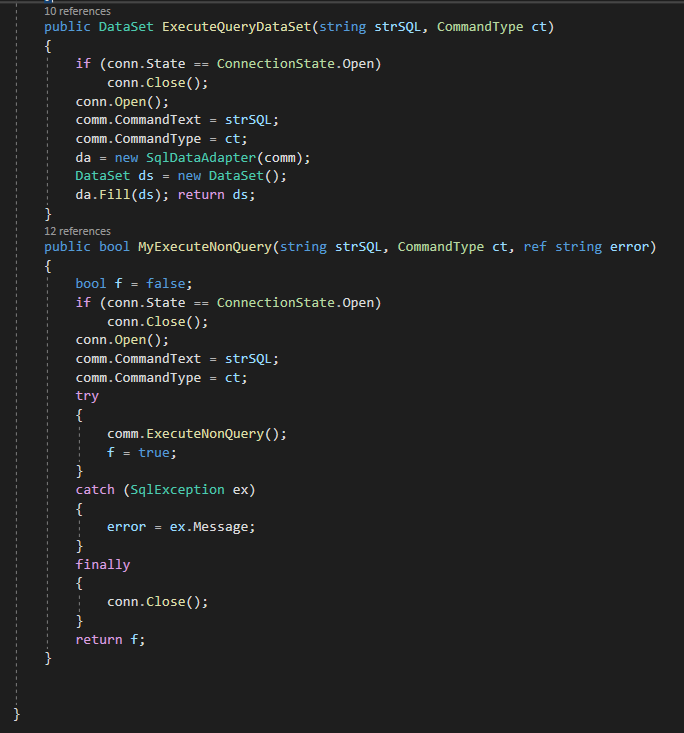
Data adapter chứa một tham chiếu đến đối tượng connection và mở/đóng kết nối tự động khi đọc và ghi dữ liệu vào database. Hơn nữa, data adapter chứa đối tượng command cho những thao tác SELECT, INSERT, UPDATE và DELETE trên dữ liệu. Bạn sẽ có một data adapter được định nghĩa cho mỗi table trong một DataSet và nó sẽ quản lý các giao tiếp với database cho bạn. Tất cả những gì bạn cần làm là chỉ cho data adapter khi nào nạp hoặc ghi vào database.[1]



##### 2.1. Tiền khai báo các biến

* DataSet

Đối tượng DataSet là một thể hiện của dữ liệu trong bộ nhớ. Chúng chứa nhiều đối tượng DataTable, bên trong DataTable lại có nhiều column và row, giống như các database table thông thường. Bạn thậm chí có thể định nghĩa dữ liệu giữa các table để tạo các quan hệ parent-child. DataSet được thiết kế đặc biệt để giúp quản lý dữ liệu trong bộ nhớ và để hỗ trợ các thao tác không cần kết nối (disconnected) trên dữ liệu. DataSet là một đối tượng được dùng bởi tất cả Data Provider, đó là lý do tại sao nó không có một Data Provider prefix trong tên gọi.



##### 2.. Code BDMain

* ExecuteQueryDataSet: Đổ dữ liệu từ database thông qua câu truy vấn strSQL ( câu truy vấn SELECT Object)
* MyExcuteNonQuery: Thực thi thao tác là thay đổi database (INSERT, UPDATE và DELETE của Object). Nếu thông tin chính xác và đầy đủ thì trả về true và tiến hành thay đổi database, nếu thông tin không chính xác thì trả về false và hiện thông báo lỗi. Sau khi thực hiện hết các bước trên ta sẽ đóng database lại.

#### BSLayer

Ở tầng BSLeyer sẽ là các query truy vấn can thiệp vào thêm, lưu sửa xoá các dữ liệu có trong database. Mỗi một bảng của database sẽ là 1 class trong BSLayer: BLDatVe.cs, BLKhachHang.cs, BLNhanVien.cs, BLPhim.cs, BLRap.cs

Đa số các code của mỗi class đều giống nhau. Nên chúng ta sẽ giải thích chung cho tất cả các Class. Ta gọi Obj thay cho các đối tượng cụ thể của các Class

+ DataSet LayThongTinObj(): Dùng câu truy vấn “Select \* from Obj” và ExecuteQueryDataSet đã được định nghĩa ở DBMain để lấy tất cả thông tin của Obj trong database.

+ DataSet findObj(): Dùng câu truy vấn “$"select column from DatVe where Col= +

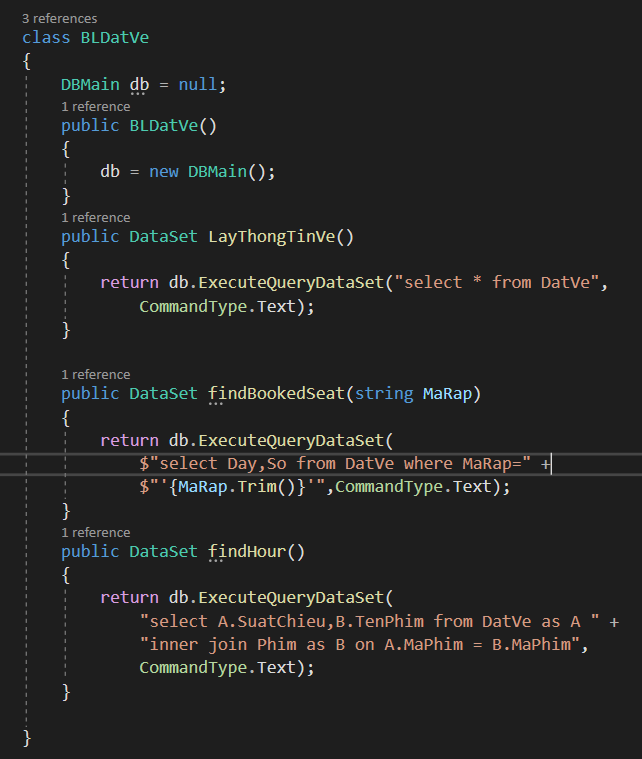
$"'{Row’s value}'"” và ExecuteQueryDataSet đã được định nghĩa ở DBMain để lấy thông tin cụ thể cần tìm của Obj trong database. Với “column” là tên các cột có trong database, “Col” là các hàng ứng với Row’s value, “Row’s value” giá trị của hàng đó ứng với Col.

+ Bool ThemObj(): Dùng câu lệnh “ Insert Into Obj Values + ‘infor’ “ và MyExcuteNonQuery đã được định nghĩa ở DBMain để thêm thông tin vào Database. Với infor là tất cả các thông tin mới ứng với dòng cột cần thêm vào database.

+ Bool XoaObj(): Dùng câu lệnh " Delete From Obj Where ‘Col = Row’s Value’ “. Với “Col” là các hàng ứng với Row’s value, “Row’s value” giá trị của hàng đó ứng với Col. Để xoá các thông tin cụ thể mà chúng ta cần xoá trong database.

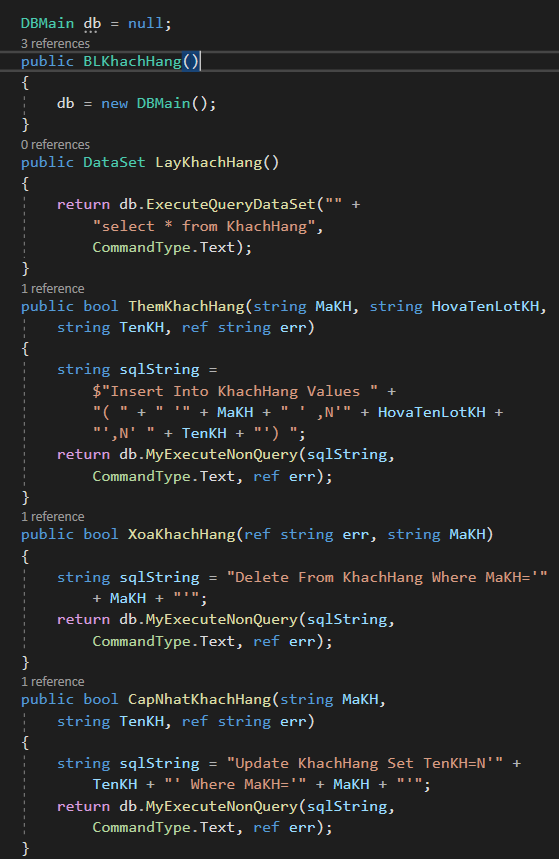
+ Bool CapNhatObj: Dùng câu lệnh “ Update Obj Set [Thông tin cần thay đổi] where ‘Col = Row’s Value’ “. “. Với “Col” là các hàng ứng với Row’s value, “Row’s value” giá trị của hàng đó ứng với Col. Để cập nhật lại các thông tin cụ thể mà chúng ta muốn thay đổi.

* Lưu ý: các cậu truy vấn trên là mã giả để người xem dể hiểu. Code thật sẽ được trích dẫn theo từng Class.
* Class BLDatVe: các câu truy vấn bao gồm LayThongTinVe(); findBookSeat()\_Tìm ghế book; findHour()\_tìm suất chiếu;



##### 2.. Code BLDatVe

* Class BLKhachHang: chứa các câu truy vấn bao giồm LayKhachHang()\_Lấy thông tin khách hàng; ThemKhachHang()\_Thêm thông tin khách hàng mới vào database; XoaKhachHang()\_Xoá khách hàng đã có trong database; CapNhatKhachHang()\_Thay đổi thông tin đã có trong database.



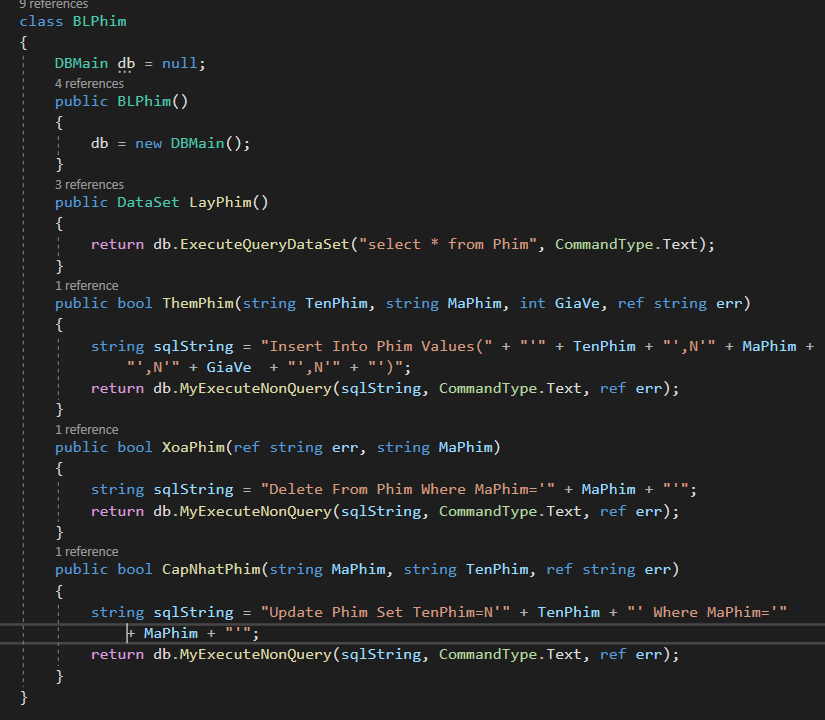
##### 2.. Code BLKhachHang

* Class BLNhanVien: chứa các câu truy vấn bao giồm LayNhanVien()\_Lấy thông tin nhân viên; ThemNhanVien()\_Thêm thông tin nhân viên mới vào database; XoaNhanVien()\_Xoá nhân viên đã có trong database; CapNhatNhanVien()\_Thay đổi thông tin đã có trong database. TimKiemNhanVien()\_Tìm kiếm nhân viên trong database bằng tên của nhân viên đó.



##### 2.. Code BLNhanVien

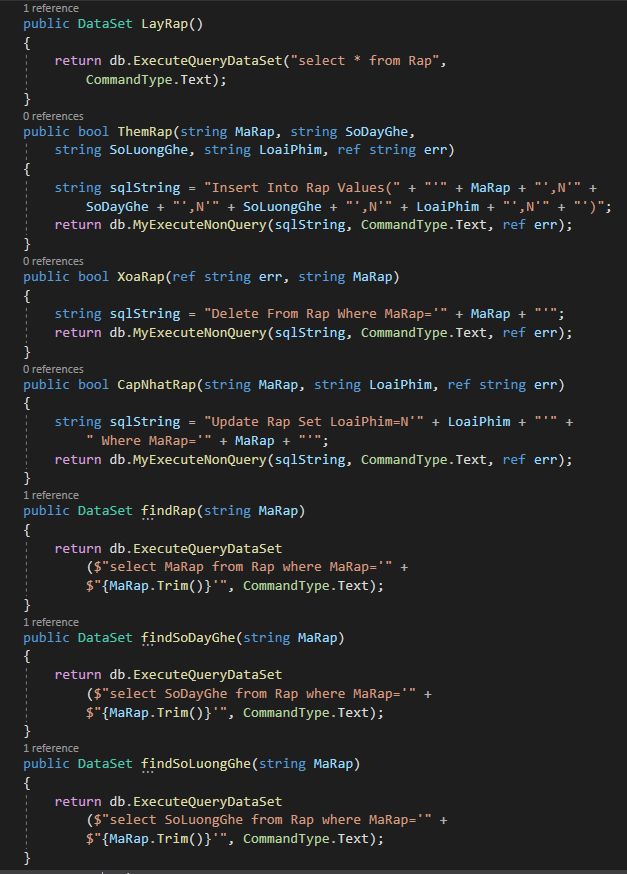
* Class BLPhim: chứa các câu truy vấn bao giồm LayPhim()\_Lấy thông tin phim; ThemPhim()\_Thêm thông tin phim mới vào database; XoaPhim()\_Xoá phim đã có trong database; CapNhatPhim()\_Thay đổi thông tin đã có trong database.



##### 2.. Code BLPhim

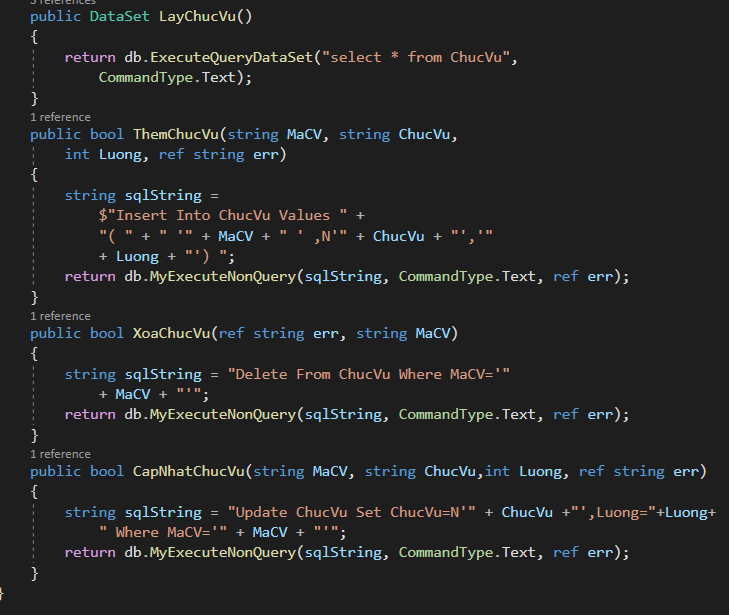
* Class BLRap: chứa các câu truy vấn bao giồm LayRap()\_Lấy thông tin Rap; ThemRap()\_Thêm thông tin Rap mới vào database; XoaRap()\_Xoá Rap đã có trong database; CapNhatRap()\_Thay đổi thông tin đã có trong database; findRap()\_tìm rạp trong database; findSoDayGhe()\_tìm số dãy ghế trong database; findSoLuongGhe()\_

tìm số dãy ghế trong database.



##### 2.. Code BLRap

* Class BLChucVu: chứa các câu truy vấn bao gồm LayChucVu()\_Lấy thông tin chức vụ; ThemChucVu()\_Thêm thông tin chức vụ mới vào database; XoaChucVu()\_Xoá chức vụ đã có trong database; CapNhatChucVu()\_Thay đổi thông tin đã có trong database.



##### .. Code BLChucVu

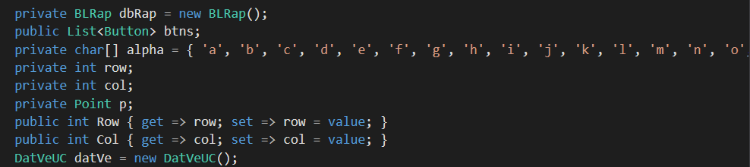
* Class BLSuatChieu: chứa các câu truy vấn bao giồm LaySuatChieu()\_Lấy thông tin của xuất chiếu; ThemSuatChieu()\_Thêm thông tin suất chiếu mới vào database; XoaSuatChieu()\_Xoá suất chiếu đã có trong database; CapNhatSuatChieu()\_Thay đổi thông tin đã có trong database.



##### .. Code BLSuatChieu

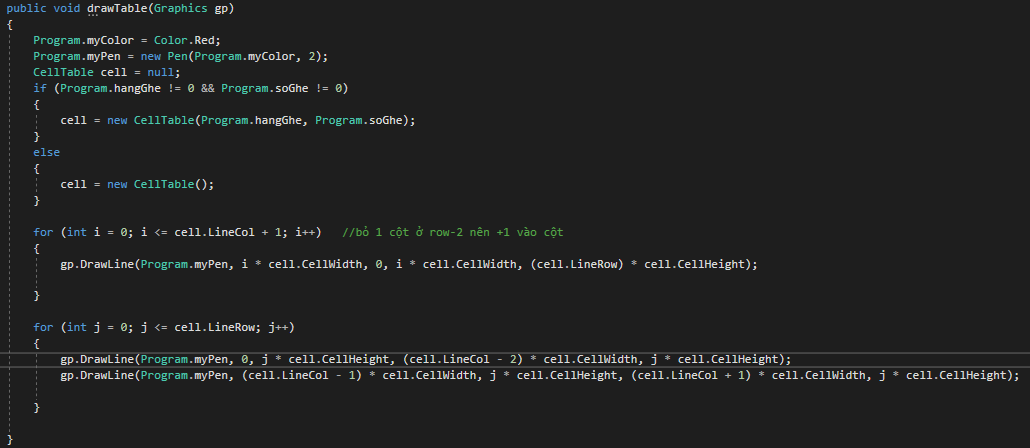
#### renderSeat

* + - * Render.cs



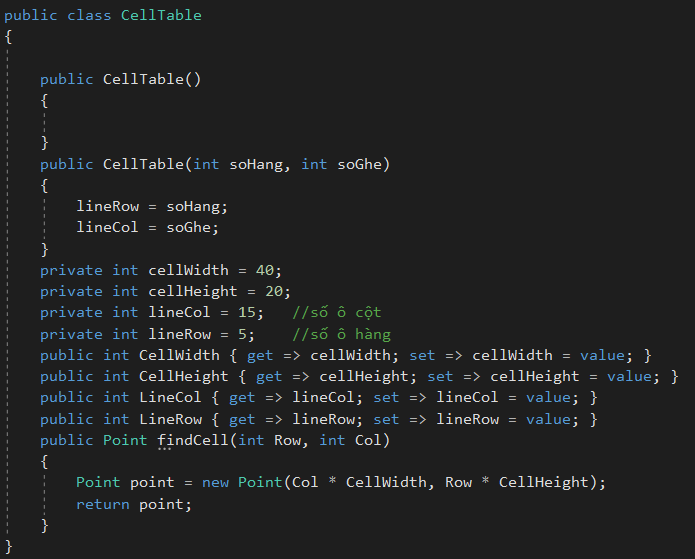
##### .. Các biến cơ bản của renderSeat

* Các biến cơ bản:
* Một list các button để chứa button cho chỗ ngồi.
* Alpha là dãy kí tự từ a đến z
* P là biến vị trí để đặt chỗ cho button.
* Hàm drawTable



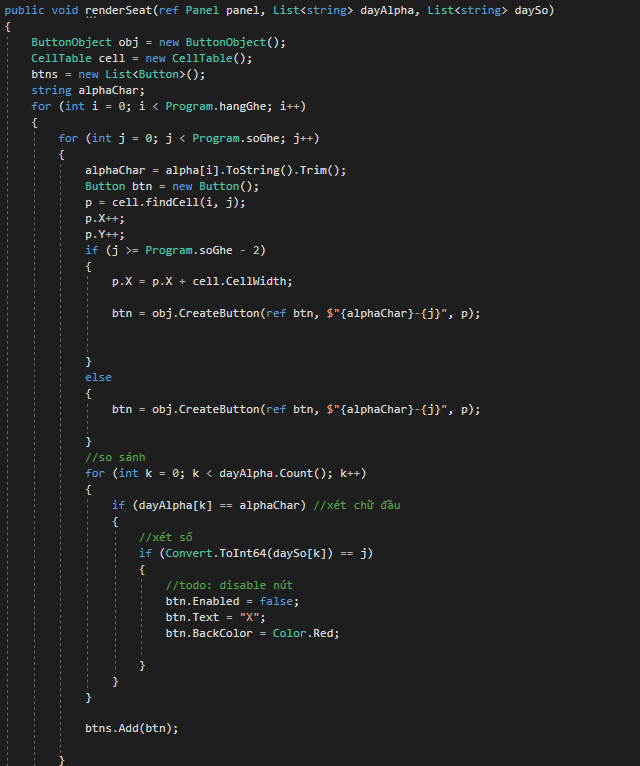
##### .. Hàm drawTable

* Hàm này dùng để vẽ các đường màu đỏ để định hình khung cho rạp.
* Dòng for đầu là vẽ cột. dòng for sau vẽ hàng(vẽ hàng sẽ phức tạp hơn tí).
* Ngoài ra có 2 biến static là hangGhe và soGhe: hàng ghế là số hàng ghế còn số ghế là số ghế trên 1 hàng ghế. Dựa 2 biến này để vẽ ra khung.
* Class CellTable



##### .. Hàm CellTable

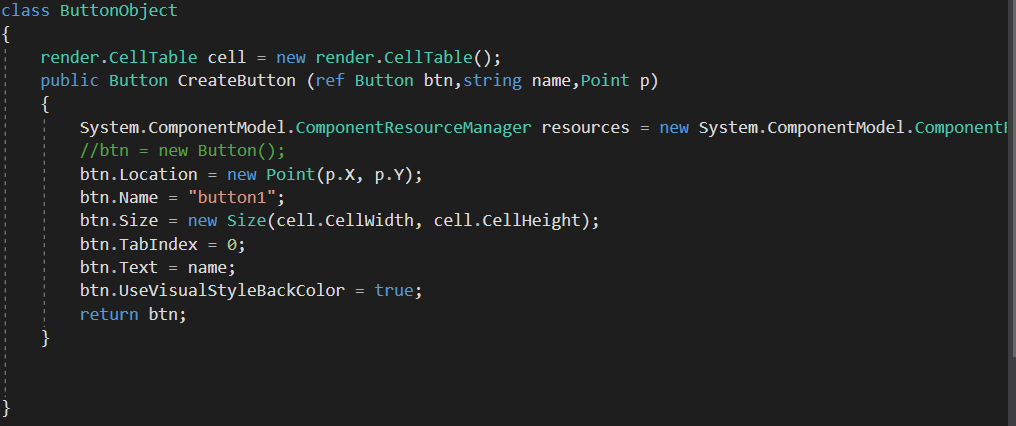
* Là class định hình cho chiều dài chiều rộng của nút và cũng như là ô của khung. Nếu không có số hàng và số ghế thì mặc định hàng sẽ là 5 và ghế sẽ là 15.
* findCell là hàm trả về vị trí của 1 ô trên bảng.
* hàm renderSeat





##### .. Hàm renderSeat

* Hàm này là hàm quan trọng nhất. hàm này cần truyền vào panel cần vẽ, dãy alpha là 1 list chữ lấy từ database. Và daySo cũng là số ghế đã được đặt trong database. Dùng 2 biến này để đánh dấu những ghế đã có người đặt
* Dùng 2 vòng lặp. dòng lặp đầu để render hàng và vòng lặp sau render cột tương ứng với ghế.. trong khi render dòng thì gắn chữ cái cho ghế. Vừa render vừa add vào list<button>
* Vòng lặp cuối cùng là dùng để đánh dấu nút nào vừa viết trùng với nút trên database để đánh dấu là đã có người đặt.
* Class buttonObject (createObject.cs)



##### .. Class buttonObject

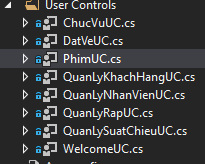
Là class dùng để tạo nút bằng code thay vì kéo thả. Làm vậy sẽ tiện để rtender hơn thay vì phải kéo thả. Chỉ cần truyền nút, tên và vị trí

### Giao diện người dùng

#### Sử dụng User Controls.

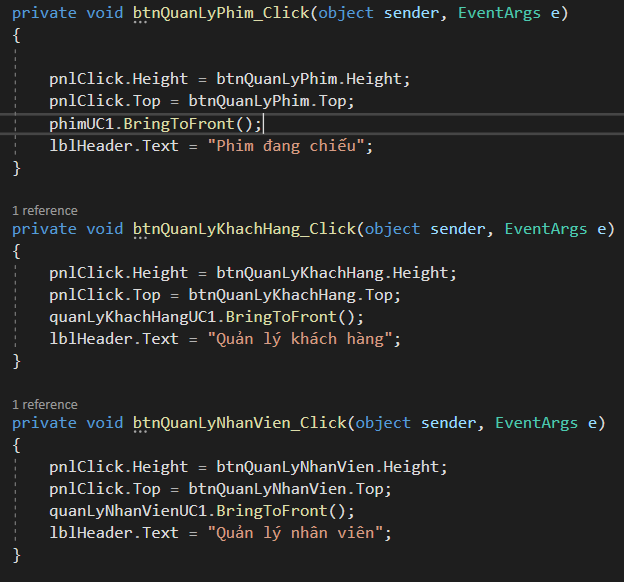
Thông thường khi chúng ta muốn mở một form mới, ta sẽ tạo 1 form mới xuất hiện cùng lúc với form cũ. Việc làm như vậy sẽ dẫn tới việc tiêu tốn tài nguyên của máy, logic của phần mềm cũng giảm đi. Vì vậy, để tất cả mọi thứ đều đổ vào 1 form ta sẽ sử dụng User Controls.

Tất các các User Controls sẽ được đặc vào thư mục User Controls.



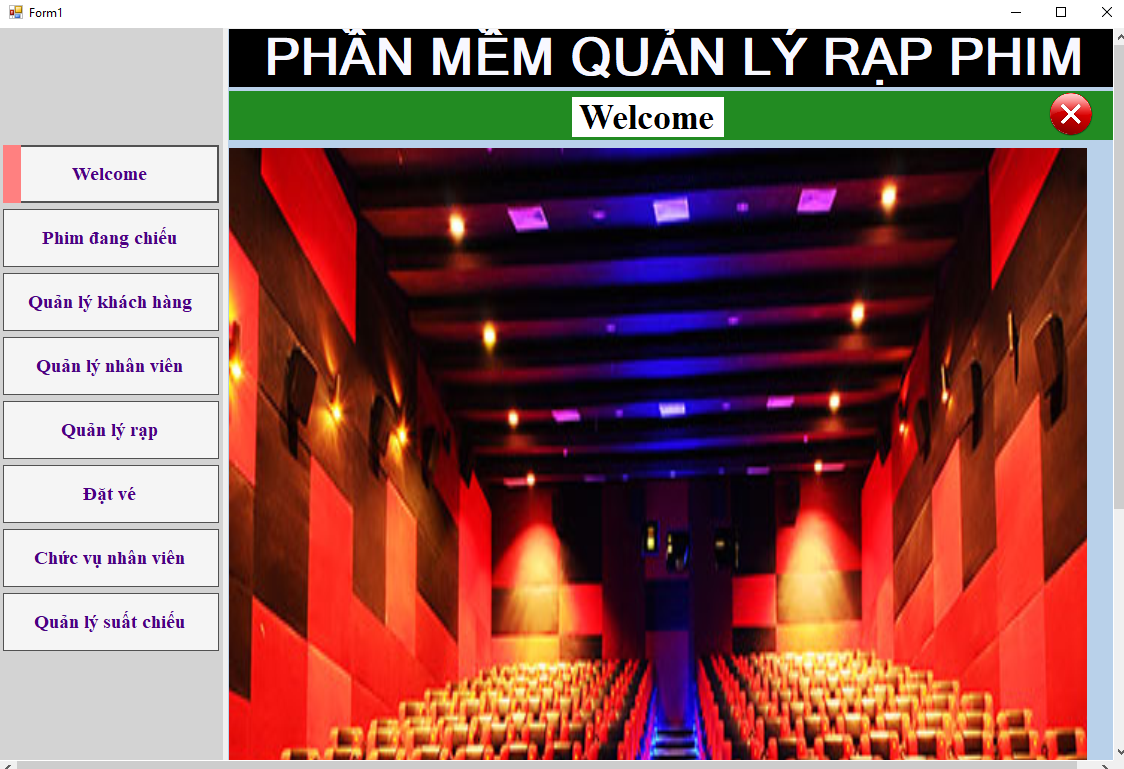
##### .. User Control

Ở From chính (form1.cs) ta sẽ dùng lệnh BringToFront() để đẩy từng User Controls và Form chính.



##### .. Form1.cs

Giao diện form chính.

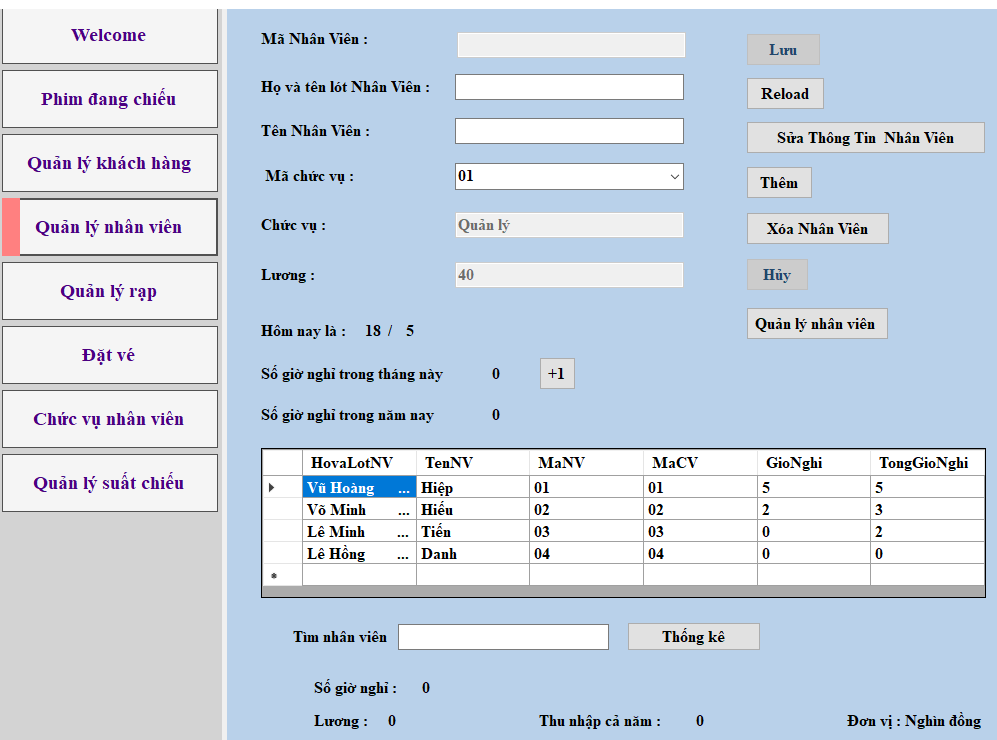


##### .. Giao diện chính

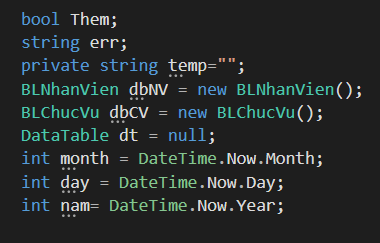
Mỗi User Controls sẽ là đại diện quản lý 1 table trong Database. Nó sẽ bao gồm các nút Thêm, Sửa, Xoá, Huỷ, Lưu, các textbox tương ứng với từng thông số trong table đó và một DataGridView chứa các thông tin của Table trong Database. Ví dụ như user controls quản lý nhân viên.

Đa số các chứa năng được đề cập ở trên giữa các User Controls là hoàn toàn giống nhau, chỉ khác mỗi giao diện. Vì vậy ta sẽ giải thích 1 User Control “quản lý nhân viên”. Source code mỗi control khác và những tính năng đặc thù của mỗi User Controls sẽ được đề cập bên dưới.

#### QuanLyNhanVien.cs

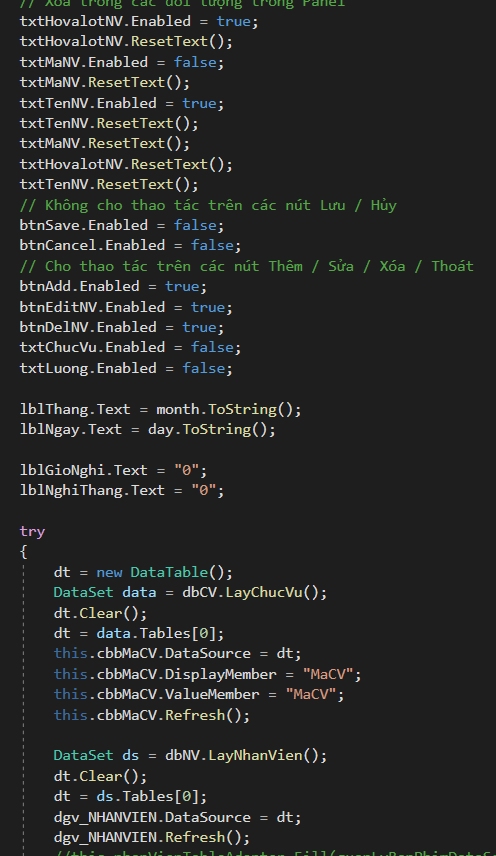


##### .. Giao diện QLNhanVien



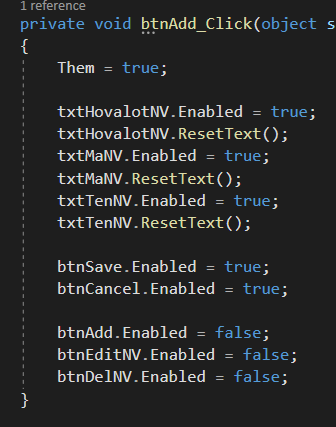
##### .. Tiền khai báo

* Đầu tiên khai báo đối tượng chính của User Control (NhanVien)



##### .. LoadData

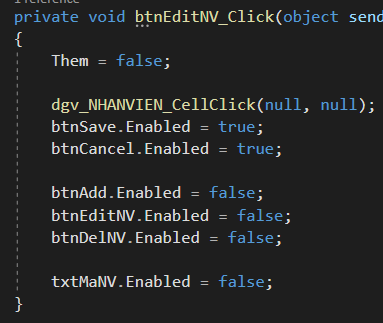
* Định nghĩa hàm LoadData():
  + - * Hàm LoadData là hàm tiền khởi đầu cho mỗi user controls. Mỗi khi sự kiên Form\_Load được bật lên. Thì hàm này sẽ được thực hiện. Trong LoadData, ta sẽ bật/khoá các chứng năng của các nút cần/không cần thiết. Ta sẽ không dùng các nút Lưu/Huỷ khi loadform. Nên ta sẽ tắt nó bằng button.Enable = false. Các nút Thêm, Sửa, Xoá, Thoát, các textbox chứa các giá trị thuộc tính của database sẽ được phép thao tác (Enable = True).
      * Ta tạo 1 DataTable để chứa thông tin từ bảng chức vụ. Sao đó lấy thông tin từ bảng chức vụ cho vào combobox.
      * Tạo mới 1 DataSet để và dùng hàm LayNhanVien để lấy các thông tin của bảng Nhân viên. Sao đó cho dữ liệu vào datagridview.
* Sự kiện click của button add: Khi click vào nút add thì sự kiện click sẽ được bật lên. Đầu tiên là bật cờ them ở chế độ true. Sau đó, ta sẽ mở khoá tất các textbox ( textbox.Enable = true) đồng thời ResetText các textbox. Tiếp đến là mở khoá chức năng Lưu và Huỷ (button.Enable = true), tắt các chức năng Thêm, Sửa, Xoá ( button.Enable = false).



##### .. Button Add

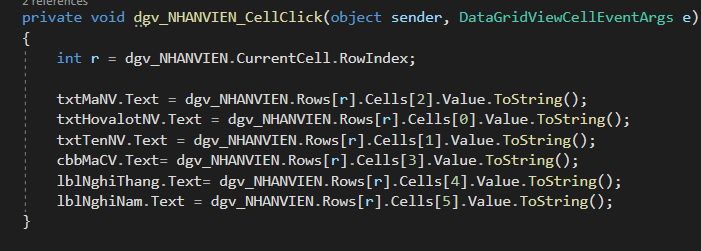
* Sự kiện click của button Edit: Khi click vào nút Sửa thì sự kiện click sẽ được bật lên. Đầu tiên là bật cờ them ở chế độ false. Sau đó, ta sẽ mở khoá các textbox liên quan (textbox.Enable = true), trừ textbox của khoá chính\_khoá chính chỉ được phép thêm, không được phép sửa ( key’s textbox = false), Tiếp đến là mở khoá chức năng Lưu và Huỷ (button.Enable = true), tắt các chức năng Thêm, Sửa, Xoá ( button.Enable = false).

Ngoài ra ta còn có sự kiện CellClick của DataGridView. Sự kiện này giúp ta lấy thông số từ datagridview ra ngoài textbox.



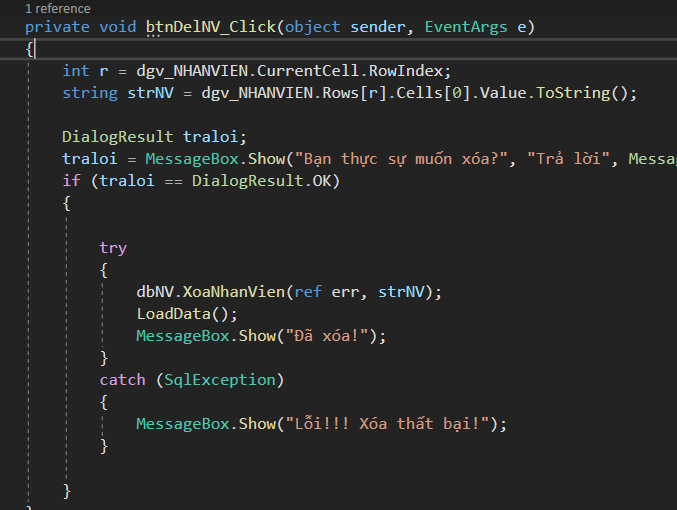
##### .. Button Edit

* Sự kiện CellClick của DataGridView: Là sự kiện mà chúng ta sẽ lấy thông của 1 dòng khi chúng ta click chuột vô dòng đó. Và thông tin sẽ được lưu vào trong textbox phía trên. Vd: MaMV(key) sẽ được lấy từ cột 2, HovaLot sẽ được lấy từ cột 0, Ten sẽ được lấy từ cột 1, v.v.



##### .. DataGridView\_CellClick

* Sự kiện click của button Xoá: ta sẽ thấy thông tin cần xoá bằng cách lấy khoá chính từ datagridview qua dòng strNV. Sau đó dùng hàm XoaNhanVien đã được định nghĩa ở tầng BSLayer để tiến hành xoá nó khỏi database. Để chương trình không bị lỗi khi đang chạy ta cùng try catch và SqlExeption để bắt lỗi.

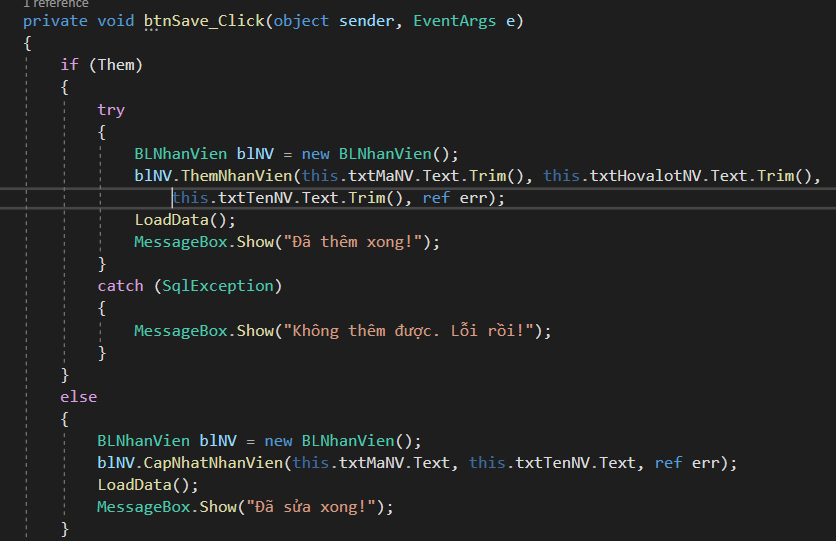


##### .. Button Deleted

* Sự kiện click của button Lưu: Lúc này, khi lúc vào nút Lưu. Thì nó sẽ kiểm tra xem cờ them (cho phép thêm) đã được bật hay chưa?

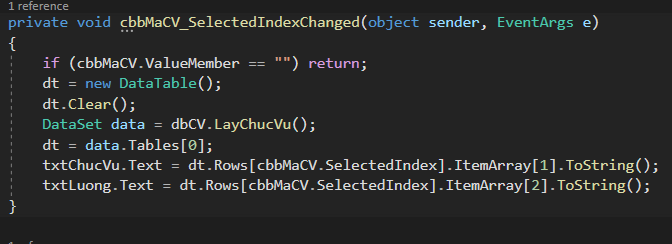
+ Nếu đã bật: ta sẽ khởi tạo mới một BLNhanVien và dùng hàm ThemNhanVien đã định nghĩa ở tầng BSLayer để tiến hành lưu. Khi lưu xong thì sẽ xuất hiện 1 thông báo đã thêm xong, còn nếu ko lưu được SqlExeption bắt lỗi và thông báo lỗi cho người dùng.

+ Nếu cờ them không bật: ta sẽ hiểu đó là cập nhật database. Thay vì dùng hàm ThemNhanVien ta sẽ dùng hàm CapNhatNhanVien. Các thao tác logic còn lại giống như khi bật cờ them.



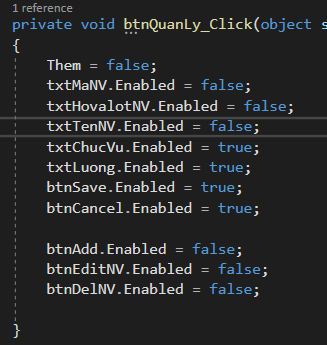
##### .. Button Save

* Sự kiện SelectecdIndexChanged của combobox: dữ liệu được lấy ở bảng Chức Vụ, sao đó được đổ vào combobox và txtxchucvu, txtluong.



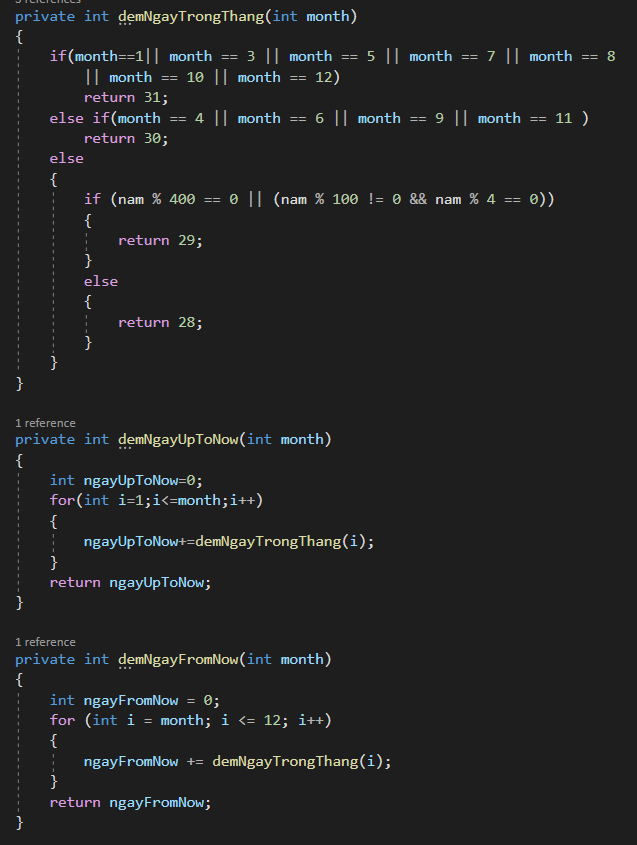
##### .. Combobox SelectecdIndexChanged

* Sự kiện click của button Quản Lý Nhân Viên: buton này cho phép thay đổi các chức vụ mà mức lương của nhân viên. Ta sẽ Enable = false tất cả, chỉ để lại combobox chức vụ và mã chức vụ mức lương.



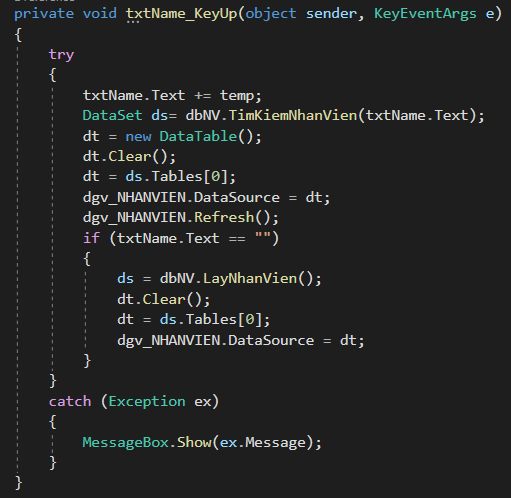
##### .. Button Quản Lý

* Các hàm đếm ngày: các hàm này phụ trợ cho phần thống kê nhân viên bên dưới:
  + - * Đếm ngày trong tháng ứng với năm.
      * Đếm ngày từ tháng 1 đến thời điểm hiện tại.
      * Đếm ngày từ thời điểm hiện tại đến cuối năm.



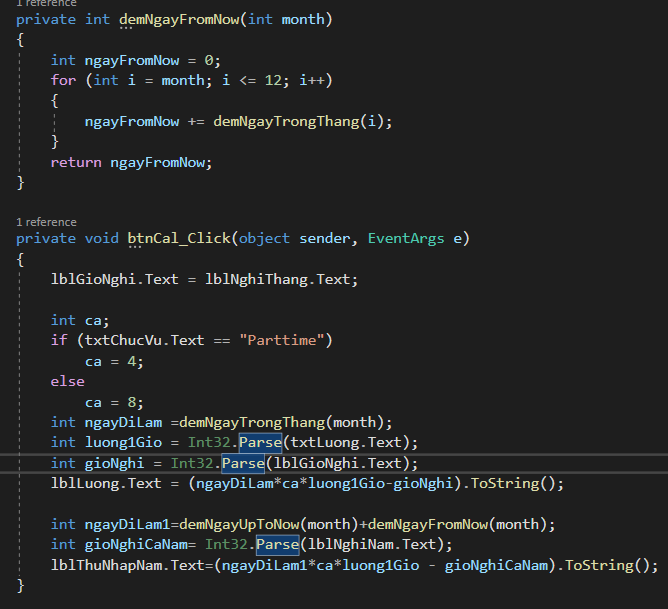
##### .. Các hàm đếm ngày

* Sự kiện KeyUp của textbox tìm kiếm(txtName): Hàm sẽ tìm kiếm tên của nhân viên là hiện lên DataGridView.



##### .. Event Textbox\_KeyUp

* Sự kiện click của Button thống kê: Sao khi tìm được nhân viên trong database bằng sự kiện KeyUp của textbox. Ta sẽ phân thời gian làm của nhân viên đó. Nếu parttime thì set 4 tiếng/ngày, còn lại là 8 tiếng/ ngày. Sau đó xét số giờ nghĩ, số ngày đi làm để tiến hành tính tiền lương cho nhân viên theo tháng và theo cả năm.
  + - * Nếu theo tháng thì chỉ trừ số ngày nghĩ rồi tính lương.
      * Nếu theo năm thì từ ngày hiện tại đến cuối năm sẽ coi như là không nghĩ ngày nào.



##### .. Button Thống kê

#### QuanLyKhachHang.cs

Đây là giao diện của tầng quản lý khách hàng. Ở tầng quản lý khách hàng. Ta chỉ có các nút cơ bản như Thêm, Sửa, Huỷ, Xoá và Lưu. Kết cấu thì hệt như QuanLyNhanVien



##### .. Giai diên QLKhachHang

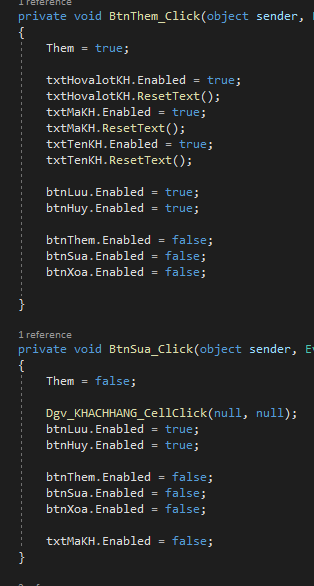
Đây là Source code: Code của Class này bao gồm hàm LoadData(), các sự kiện click của Thêm, Xoá, Sửa, Huỷ, Lưu, sự kiện CellClick của DataGridView.

* Hàm LoadDatat();



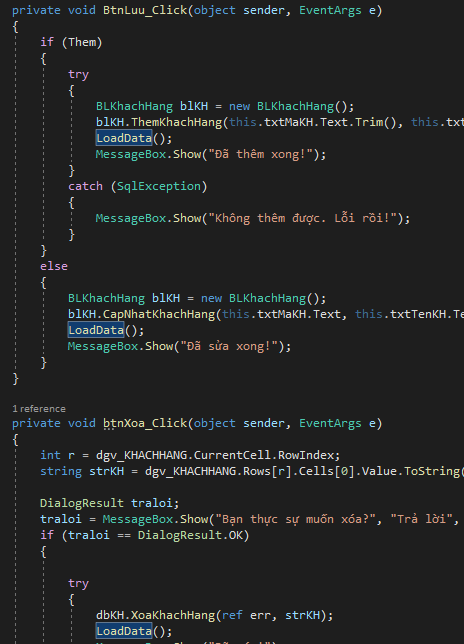
##### .. LoadData

* Sự kiện click của Button Thêm và Button Sửa.



##### .. Button Sửa và Thêm

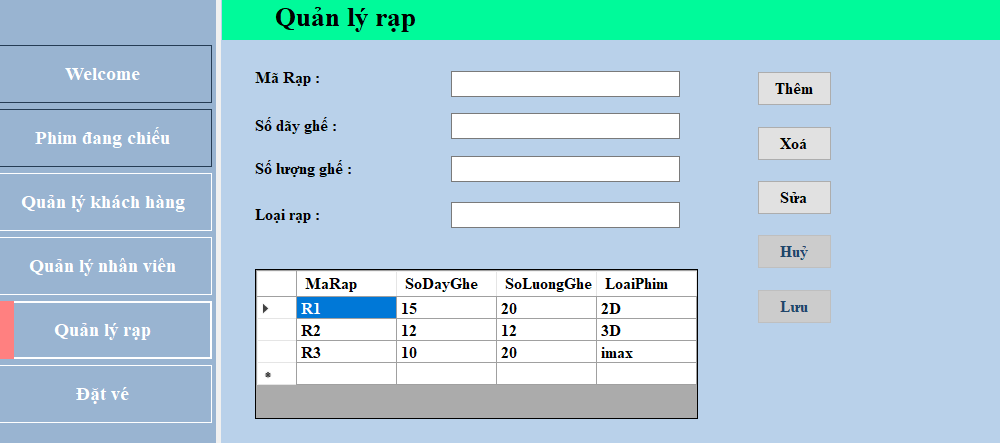
* Sự kiện click của Button Lưu và button Xoá.



##### .. Button Lưu và Xoá

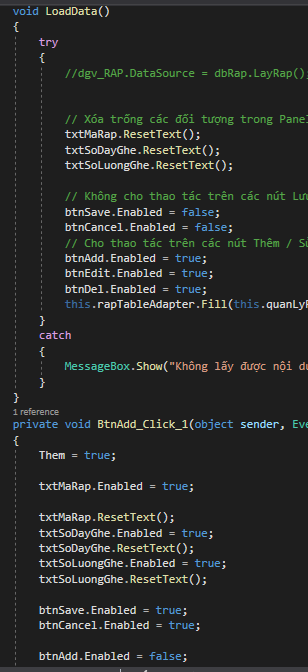
#### QuanLyRap.cs

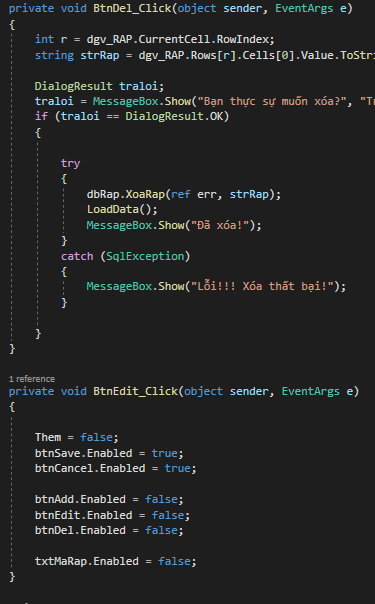
Giao diện của tầng QuanLyRap

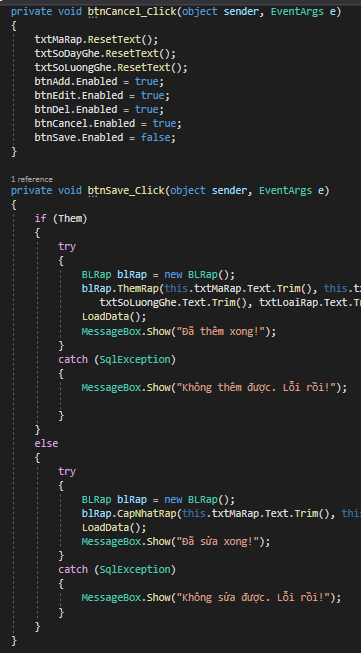


##### ..Giao diện QLRap

Code của Class này cũng bao gồm hàm LoadData(), các sự kiện click của Thêm, Xoá, Sửa, Huỷ, Lưu, sự kiện CellClick của DataGridView.

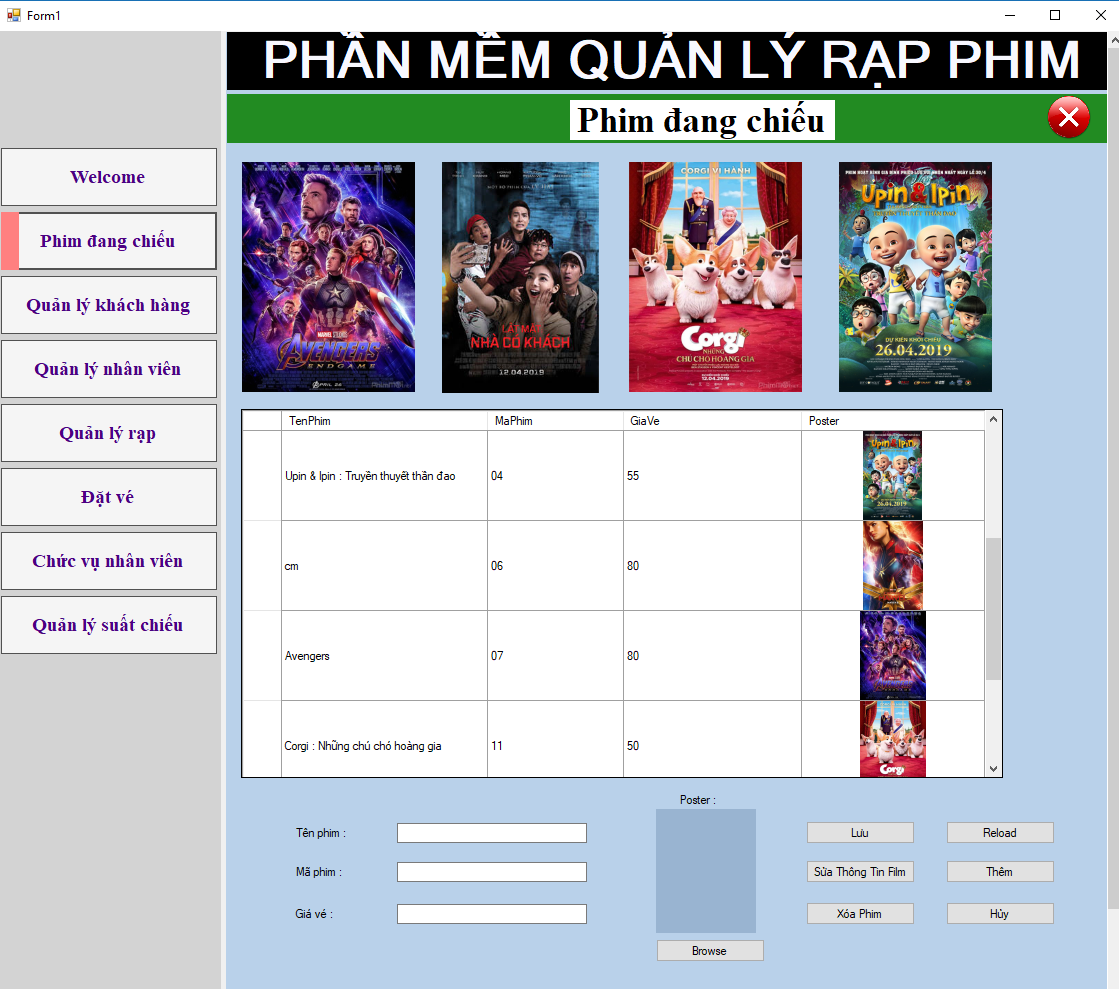






##### .. Code quản lý rạp

#### PhimUC.cs



##### ..Giao diện phim đang chiếu

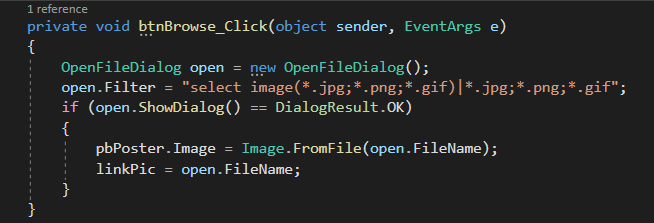
Ở bảng phim, các chức năng cơ bản thêm sửa xoá đều giống với các bản khác.

Điểm khác biệt ở đây là hàm thêm ảnh, và xử lý ảnh ở database và client.

* Khai báo biến string chưa text của hình

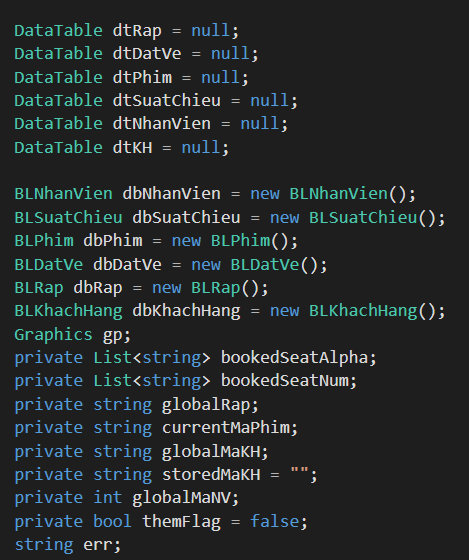


* Sự kiện click của Botton Browse: ta dùng OpenFileDialog để mở các forlder trong máy. Open.Filter để lọc các định dạng hình. Sau khi lấy hình, ta dùng hàm FileName để phân giải hình ảnh thành text để lưu vào database.



##### .. Button Brownse

#### DatVe.cs

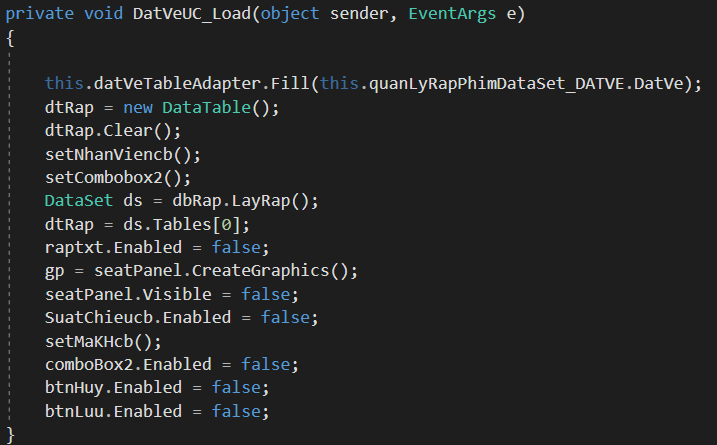
* Biến chung

##### 3.25. Biến chung Đặt vé

Tạo các datatable liên quan đến đặt vé và các dataset.

Ngoài ra tạo 2 list ghế và số ghế, biến cục rạp. mã phim, mã khách hàng (xóa cái stored makh đi). Cờ thêm, biến error

* Hàm load UC



##### .. Hàm load UC

Tạo rạp và khởi tạo combobo các combobox

* Combobox2

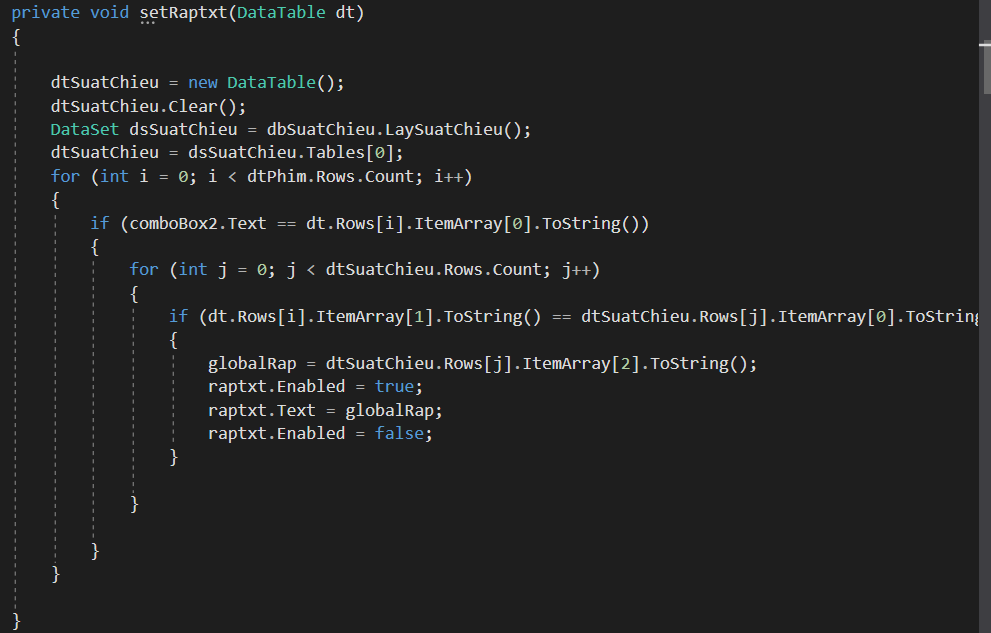


##### .. Hàm set Combobox



Đổ dữ liệu tên phim vào combobox.

* setRaptxt

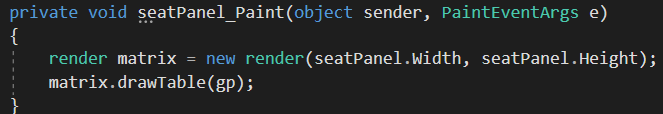


##### .. Hàm setRaptxt



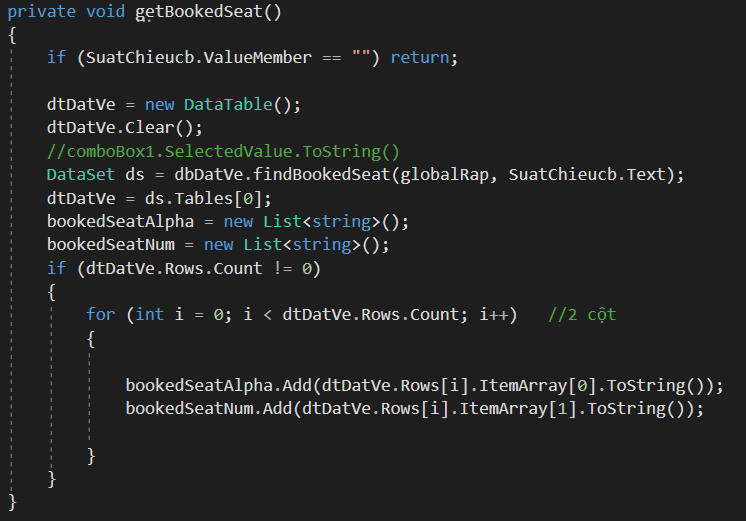
Đổ dữ liệu khi đã chọn tên phim và chọn suất chiếu.

* seatpanelPaint



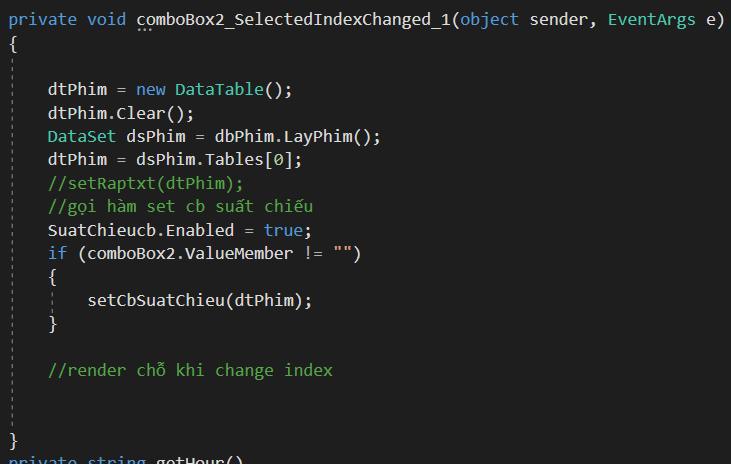
##### .. Hàm seatpanelPaint

Hàm vẽ trên panel



##### .. Hàm getBookedSeat

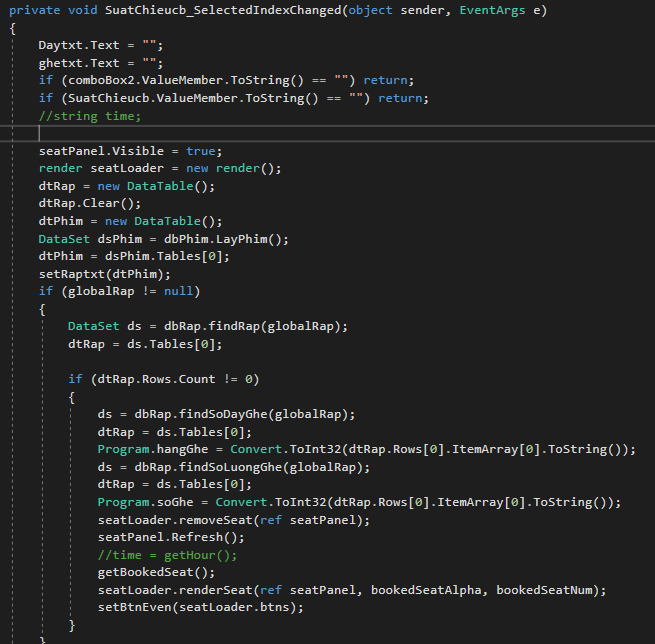
Hàm lấy dữ liệu từ đặt vé để render lên bảng ghế.



##### .. Sự kiện SelectedIndexChanged của ComboBox

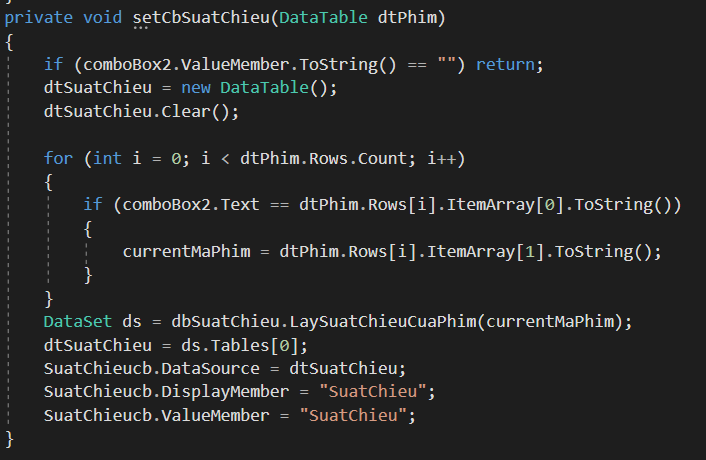
Set suất chiếu. bỏ cái dòng xanh thứ 3 nha

* Event suất chiếu



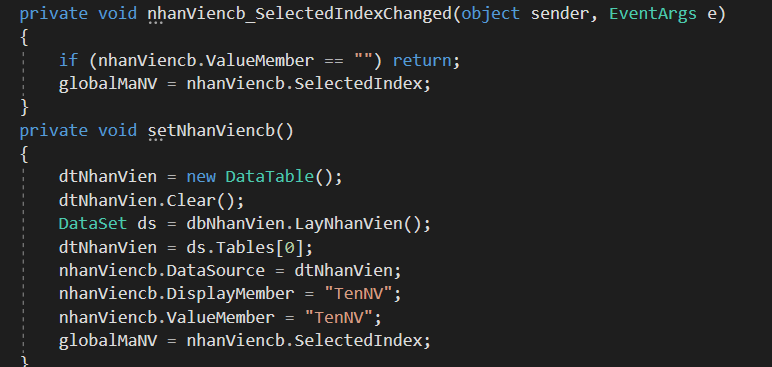
##### .. Event suất chiếu

* Set combobox suất chiếu:



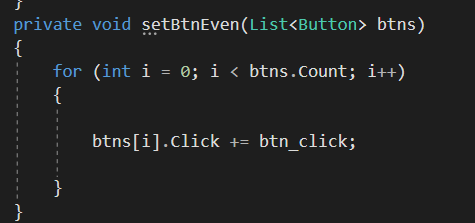
##### .. Set combobox suất chiếu

* Set combobox nhân viên:



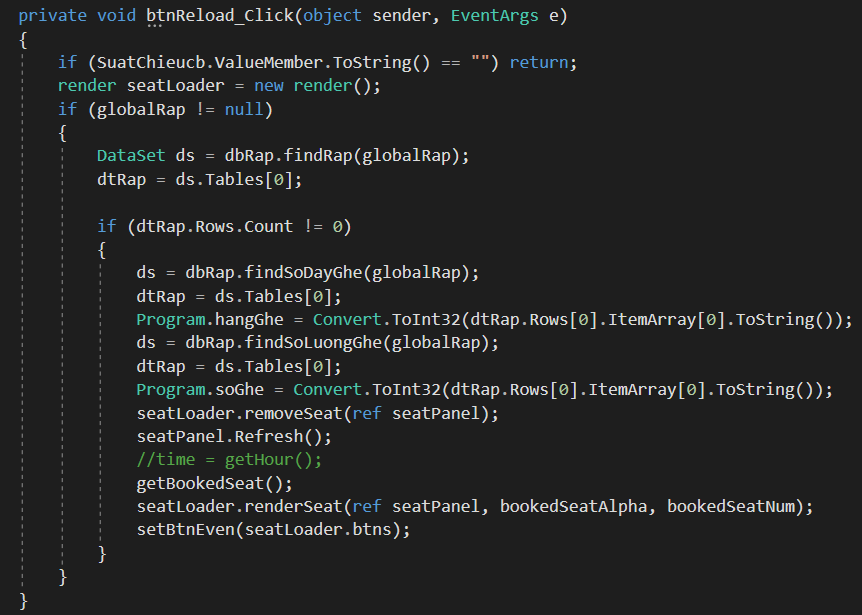
##### .. Set combobox nhân viên

* Event click button



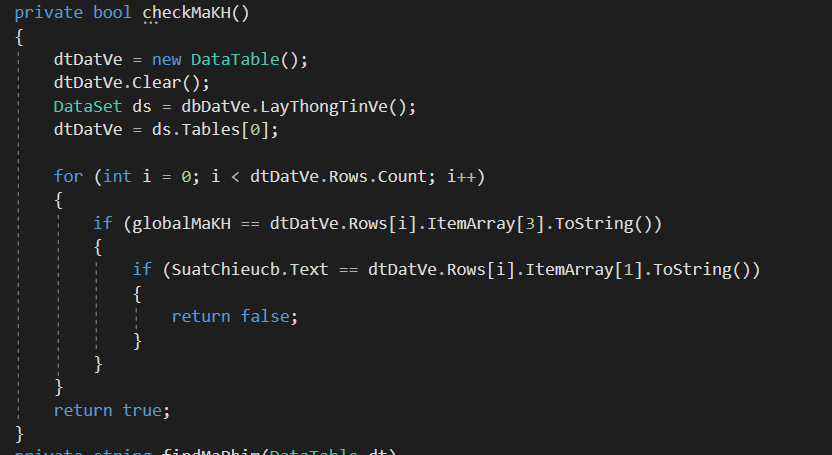
##### .. Sự kiện click ghế

* Set nút reload



##### .. Rút reload

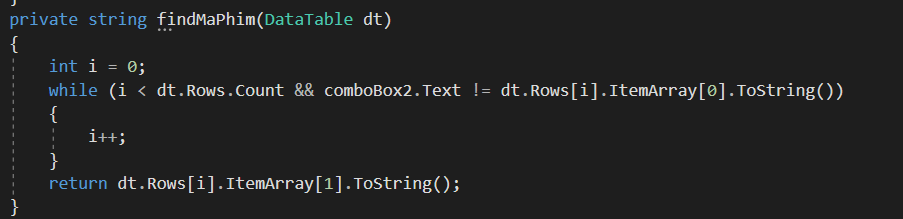
* Hàm checkMaKH



##### .. Hàm checkMaKH

Kiểm tra xem khách hàng đã đăng kí chưa.

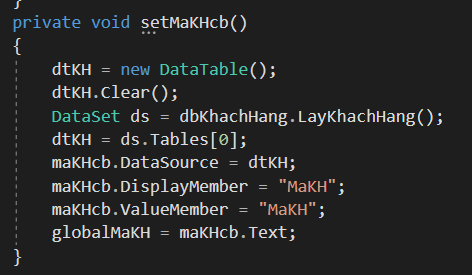
* Tìm mã phim



##### ..Hàm tìm mã phim

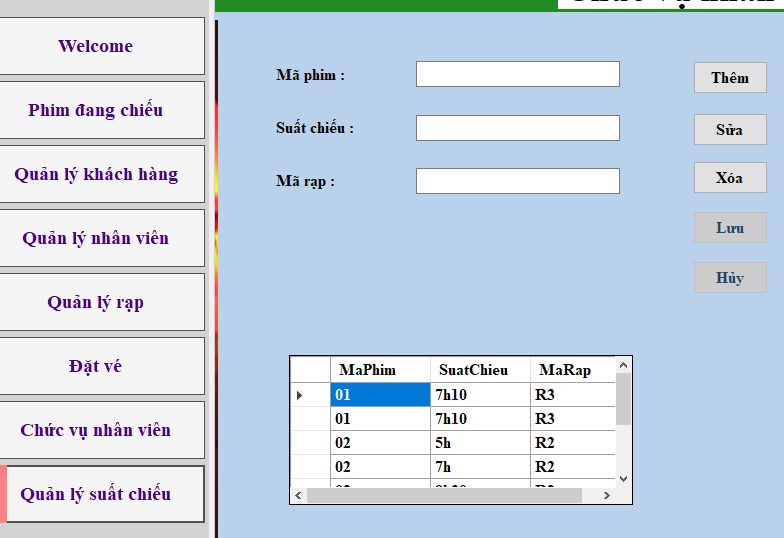
Hàm tìm mã phim

* Hàm set combobox mã khách hàng



##### .. Hàm set MaKH

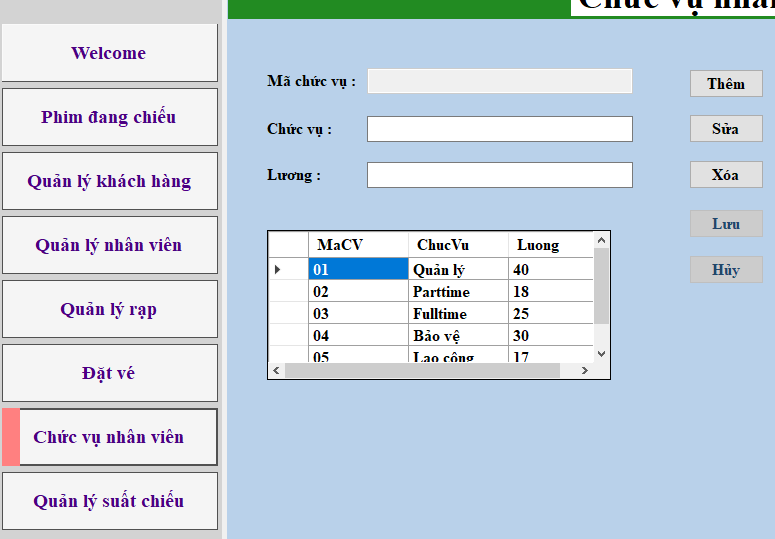
#### QuanLySuatChieu.cs



##### .. Giao diện QLSuatChieu

Phần này chỉ bao gồm các chức năng cơ bản như Thêm, Sửa, Xoá, Lưu và Huỷ.

#### QuanLyChucVu

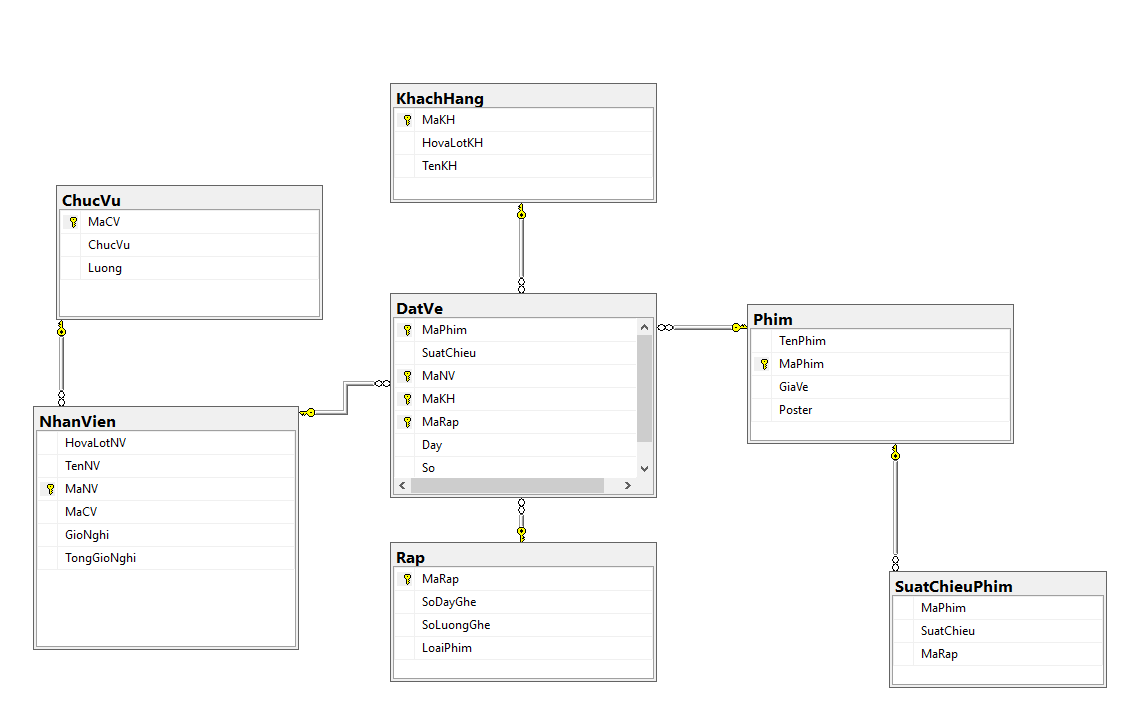


##### .. Giao diện QLChucVu

Chức năng của chức vụ chủ yếu là truy xuất phụ cho bảng NhanVien. Dùng ChucVu để phân lương và thống kê lương cho nhân viên.

### Database Diagram

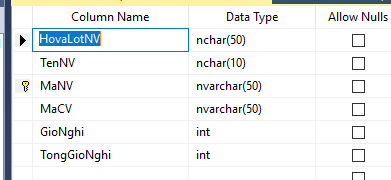
Đây là database diagram của đề tài. Bao gồm 6 bảng: Rap, NhanVien, DatVe, Phim, KhachHang, ChucVu, SuatChieu



##### 4.. Database Diagram

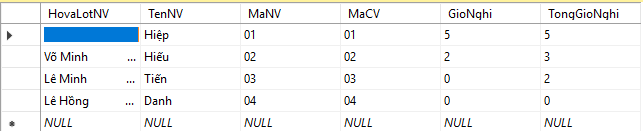
#### Bảng NhanVien

Phần design kiểu dữ liệu của bảng NhanVien, được thiết lập với khoá chính là MaNV.



##### 4.. Design NhanVien

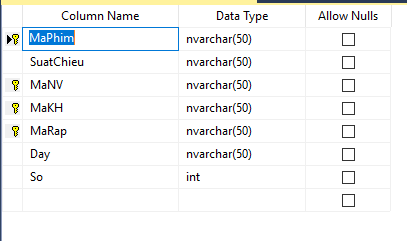
Phần data của NhanVien.



##### 4.. Data NhanVien

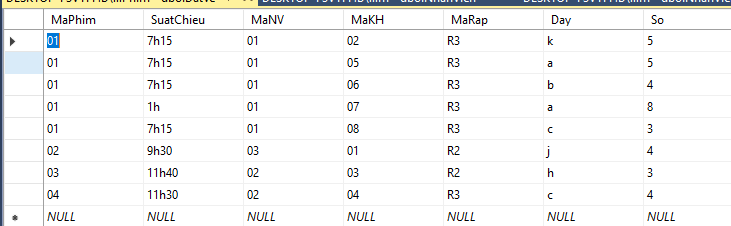
#### Bảng DatVe

Phần design kiểu dữ liệu của bảng DatVe với các khoá chính gồm có: MaPhim, MaNV, MaKH, MaRap.



##### 4.. Design DatVe

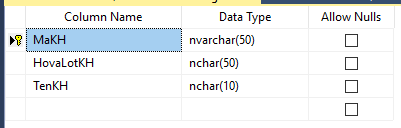
Phần data của DatVe



##### 4.. Data DatVe

#### Bảng KhachHang

Phần design kiểu dữ liệu của bảng KhachHang với khoá chính là MaKH.



##### 4.. Design KhachHang

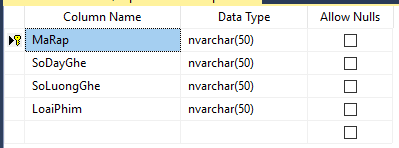
Phần data của KhachHang.



##### 4.. Data KhachHang

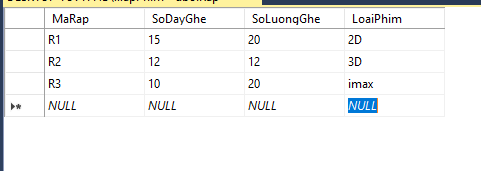
#### Bảng Rap

Phần design kiểu dữ liệu của bảng Rap với khoá chính là MaRap.



##### 4.. Design Rap

Phần data của Rap



##### 4.. Data Rap

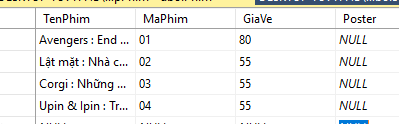
#### Bảng Phim

Phần design kiểu dữ liệu của bảng Phim với khoá chính là MaPhim.



##### 4.. Design của Phim

Phần data của Phim



##### .. Data của Phim

# VERSION 2: LinQ to SQL Framework

### LinQ là gì?

LinQ (Language Integrated Query) là 1 thư viện được tích hợp trong .NET Framework 3.5 dành cho C# lẫn VB (nhưng ở đây mình sẽ xoáy sâu vào C#). LinQ hiểu đơn giản là “mang SQL vào C#”, nó cung cấp khả năng truy vấn dữ liệu trực tiếp ngay trên chính C#. Mặc dù hiện tại có rất nhiều ngôn ngữ đã “mang LinQ về với bản làng” (PHP, JavaScript, TypeScript lẫn ActionScript), nhưng tất cả đều chỉ là những library bên ngoài, không được hỗ trợ tận răng như C#.[2]

### Vì sao LinQ ra đời?

Mục đích ra đời của LINQ hay LINQ to SQL là để phục vụ cho hệ quản trị cơ sở dữ liệu SQL Server và nền tảng .NET nói chung hay ngôn ngữ lập trình C#/VB.NET nói riêng chứ không phải mục đích đại trà cho các ngôn ngữ lập trình hay các hệ quản trị cơ sở dữ liệu khác. Tôi thường sử dụng C# nên sử dụng C# làm ví dụ, ta biết rằng:

C# là một ngôn ngữ lập trình hướng đối tượng hoàn toàn.

SQL Server là một hệ quản trị cơ sở dữ liệu (DBMS) theo mô hình quan hệ, mô hình CSDL quan hệ ghi các dữ liệu theo dòng trong các bảng dữ liệu.

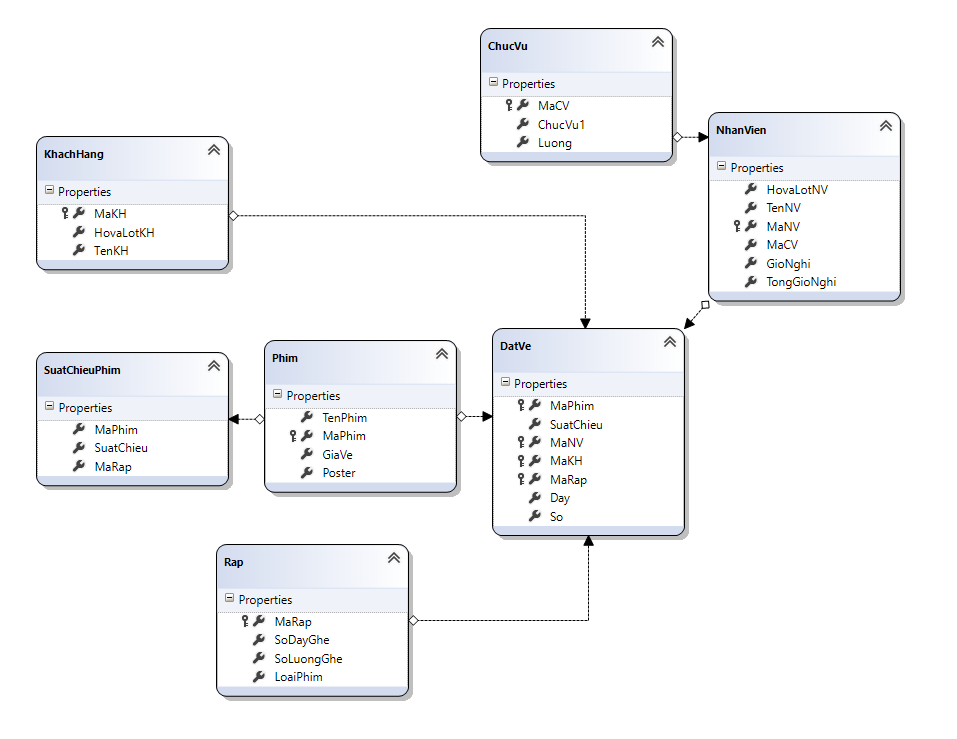
Tôi muốn dùng C# viết mã để lấy dữ liệu từ SQL Server nhưng tổ chức mô hình dữ liệu của C# và SQL Server là khác nhau. Để giải quyết tình trạng trên, một kỹ thuật gọi là ORM (Object Relational Mapping) ra đời nhằm mục đích chuyển đổi dữ liệu giữa các hệ thống khác (không phải là mô hình hướng đối tượng) sang các đối tượng trong ngôn ngữ lập trình hướng đối tượng. Tiếp đó LINQ to SQL ra đời dựa trên kỹ thuật ORM xóa bỏ khoảng cách giữa mô hình lập trình hướng đối tượng C#/VB.NET với hệ quản trị cơ sở dữ liệu SQL Server khi đã mô hình hóa theo hướng đối tượng các bảng trong Database thành các lớp tương ứng khi lập trình.

Sự ra đời của LINQ to SQL giúp các lập trình viên .NET bớt đi gánh nặng phụ thuộc bên thứ 3 (dùng SQL để truy vấn). LINQ to SQL đồng thời đồng bộ hóa dữ liệu lấy ra và trả về khi truy xuất dữ liệu bằng việc các Data Model hứng dữ liệu trả về được tạo tự động sao cho tương thích với kiểu dữ liệu tương ứng của chúng khi ánh xạ vào cơ sở dữ liệu. Điều này làm tránh tình trạng mất hoặc sai lệch dữ liệu khi truy xuất và thao tác với Database. Trên hết, một công cụ được phát hành cho nền tảng duy nhất là .NET với những anh em trong gia đình .NET sử dụng thì hiệu suất LINQ to SQL hằn là phải tốt hơn so với những công cụ bên thứ 3.[3]

### Lý do tại sao sử dụng SQL

* Hỗ trợ công cụ:

Khi sử dụng LINQ to SQL với hệ quản trị cơ sở dữ liệu SQL Server, tôi không phải tạo các lớp Data Model để hứng dữ liệu trả về khi truy vấn dữ liệu vì LINQ to SQL đã tạo sẵn những lớp này với đầy đủ các thuộc tính và kiểu dữ liệu phù hợp với kiểu dữ liệu các cột bạn qui định trong Database (các thuộc tính của mỗi lớp ánh xạ vào các cột của bảng tương ứng trong CSDL).



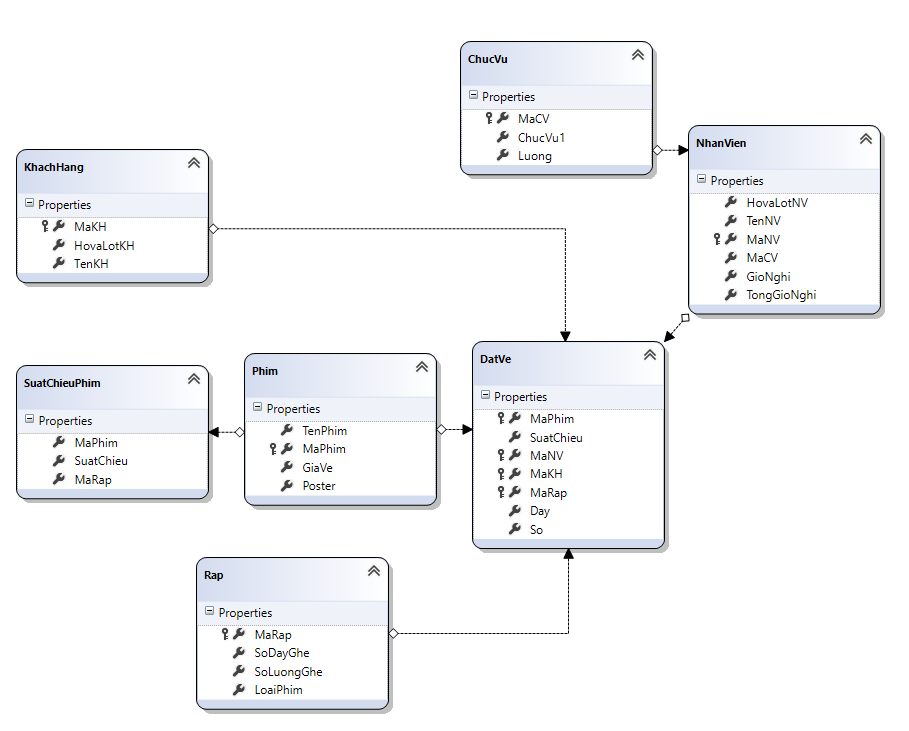
##### .. DataModel Object.

* Giao diện trực quan và tự động:

LINQ to SQL cung cấp giao diện trực quan về mối quan hệ các bảng dữ liệu sau khi được mô hình hóa. Các lớp DataContext sẽ được tạo ra tự động khi bạn Import file LINQ to SQL vào Project. Các lớp DataContext nhận nhiệm vụ mở kết nối đến cơ sở dữ liệu, thực hiện truy vấn hay thay đổi dữ liệu. Các lớp thuộc tính được mô hình hóa từ các bảng dữ liệu trong hệ quản trị cơ sở dữ liệu được truy cập thông qua các lớp DataContext. Lớp DataContext này gần như là một lớp bao (Wrapper Class), những thay đổi nếu có từ các bảng dữ liệu trong cơ sở dữ liệu thì lớp này sẽ cập nhật và thay đổi tương ứng (chúng cũng sẽ cập nhật vào các lớp Data Modal được tạo tự động). Điều này khiến cho việc thay đổi thuộc tính dữ liệu trong database diễn ra dễ dàng và người lập trình không mất quá nhiều công sức để chỉnh sửa lại code (do những thay đổi được tự động cập nhật lại).

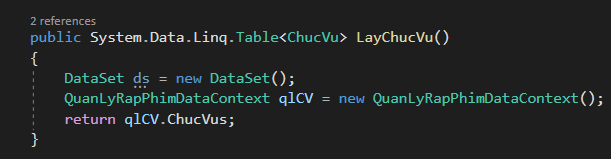
### Software architecture

Ở phiên bảng ADO.NET ta sẽ có thư mục DBMain để thực thi các truy vấn. Tuy nhiên ở LinQ to SQL đã hỗ trợ phần đó. Chúng ta chỉ việc thêm items SQL to LinQ Classes, sau đó thêm Database Diagram. LinQ sẽ làm gần như mọi thứ cho ta thao tác trên Database dễ dàng hơn. Tất cả các thao tác đều thông qua DataContext này.



##### .. QuanLyRapPhimDataContext

#### BLChucVu



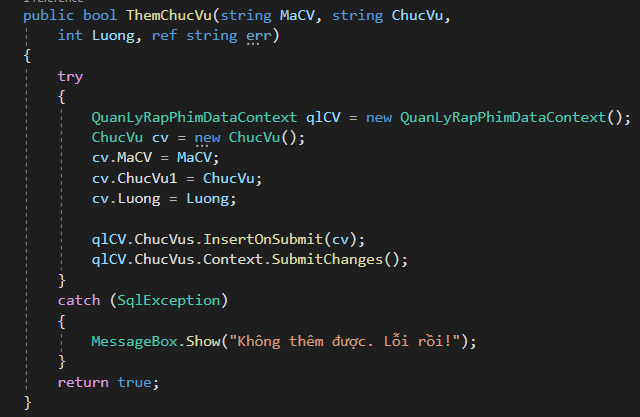
##### .. Lấy thông tin Object

* Hàm LayChucVu:

Do LinQ đã tạo cho ta sẳn các Table trong QuanLyRapPhimDataContext. Nên khi muốn lấy chức vụ. Ta chỉ cần tạo mới đối tượng DataContext để lấy thông tin từ bất kì bảng nào.

* Hàm thêm chức vụ:

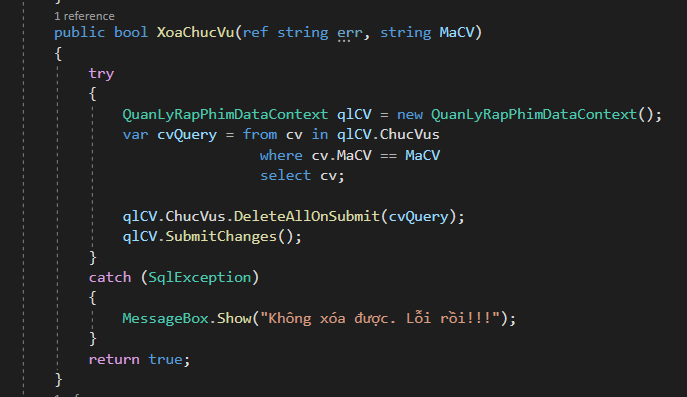
Khi thêm một dữ liệu mới và database. Ở ADO.NET ta sẽ phải viết các câu truy vấn rườm rà. Thì ở LinQ, ta sẽ viết có 1 cách tường minh hơn, “C# hơn”. Chỉ cần cho các cộng trong bảng bằng với các thuộc tính được thêm vào. Sau đó InserOnSubmit (thực thi thêm) và SubmitChanges(thực thi thay đổi database) để thêm dữ liệu mới.



##### .. Thêm thông tin Object

* Hàm xoá chức vụ:

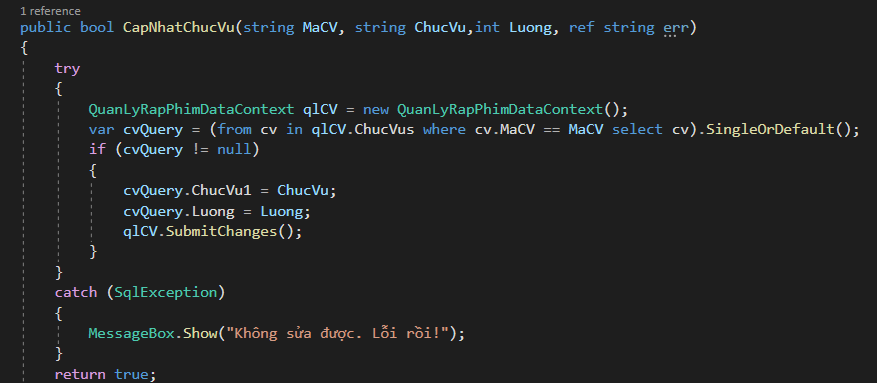
Để xoá chức vụ, ta cần phải chọn đối tượng dữ liệu cần xoá. Đến đây ta cần phải viết các câu truy vấn để xác định đối tượng. Cách viết đơn giản, gần như thuần truy vấn SQL, nên tạo được cảm giác thân thiện. Ta thực hiện xác định đối tượng bằng khoá chính của bảng ChucVu ( from cv in qlCV.ChucVus where cv.MaVC = MaCV select cv). Sau khi viết xong, ta sẽ bỏ lệnh truy vấn ấy vào trong hàm DelectedAllOnSubmit. Hàm sẽ thực thi xoá dữ liệu đc trỏ, sau đó xác nhận thay đổi bằng SubmitChanges.



##### .. Xoá thông tin Object

* Hàm Cập nhật chức vụ:

Để cập nhật thông tin chức vụ. Ta cần phải xác định dữ liệu cần trỏ qua khoá chính MaCV. Bằng câu truy vấn ta có thể lấy được thông tin cần trỏ. Nếu có thông tin trong database thì tiến hành sửa các thuộc tính có trong bảng (trừ khoá chính). Sau khi thay đổi, ta dùng SubmitChanges để thử thi thay đổi thông tin.

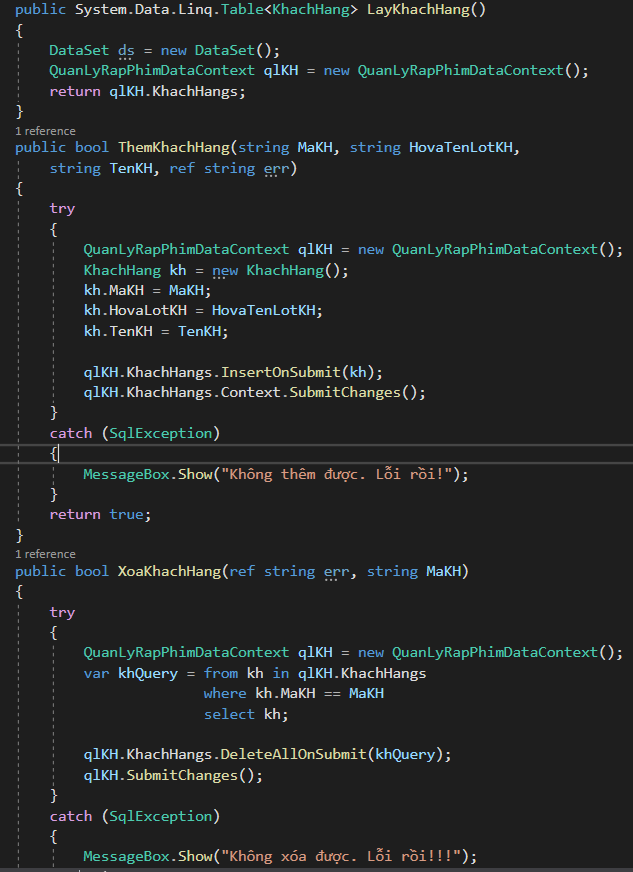


##### .. Cập nhật thông tin Object.

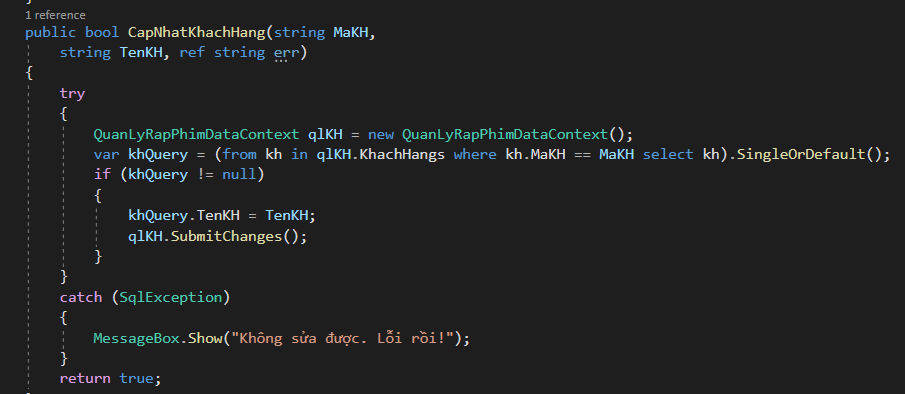
* Tất cả các hàm trên đều sẽ dùng SqlException để bắt lỗi.

#### BLKhachHang

Chứa các hàm được thao tác qua DataContext và Table LinQ bao giồm LayKhachHang()\_Lấy thông tin khách hàng; ThemKhachHang()\_Thêm thông tin khách hàng mới vào database; XoaKhachHang()\_Xoá khách hàng đã có trong database; CapNhatKhachHang()\_Thay đổi thông tin đã có trong database.



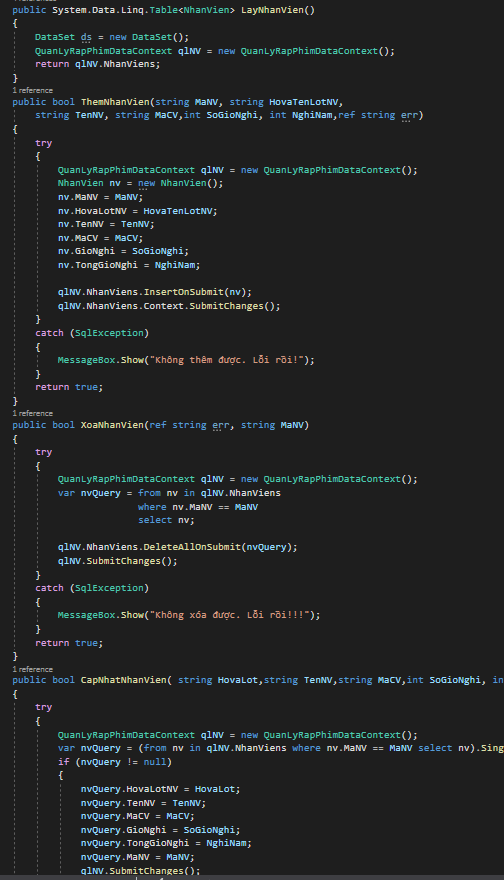
##### .. BLKhachHangLinQ1



##### .. BLKhachHangLinQ2

#### BLNhanVien

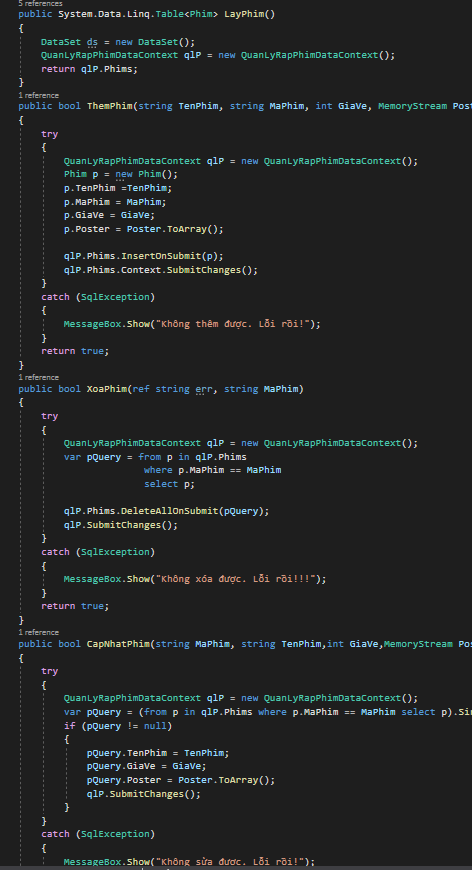
Chứa các hàm được thao tác qua DataContext và Table LinQ bao gồm LayNhanVien()\_Lấy thông tin nhân viên; ThemNhanVien()\_Thêm thông tin nhân viên mới vào database; XoaNhanVien()\_Xoá nhân viên đã có trong database; CapNhatNhanVien()\_Thay đổi thông tin đã có trong database. TimKiemNhanVien()\_Tìm kiếm nhân viên trong database bằng tên của nhân viên đó.



##### .. BLNhanVienLinQ

#### BLPhim

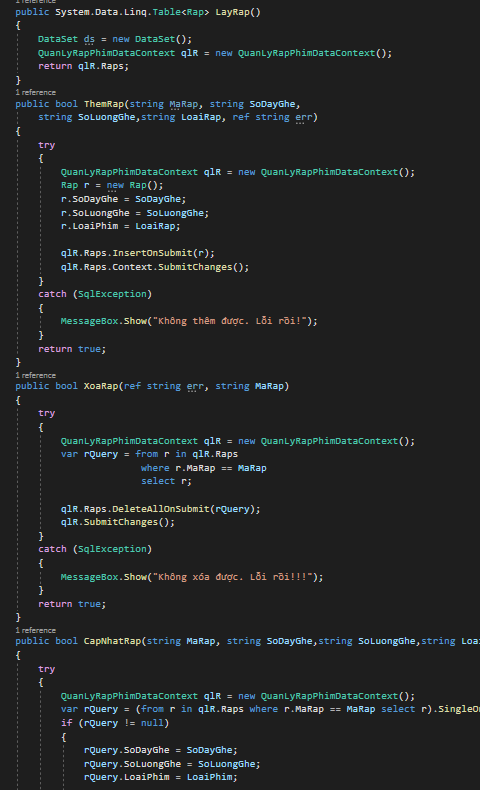
Chứa các hàm được thao tác qua DataContext và Table LinQ bao gồm LayRap()\_Lấy thông tin Rap; ThemRap()\_Thêm thông tin Rap mới vào database; XoaRap()\_Xoá Rap đã có trong database; CapNhatRap()\_Thay đổi thông tin đã có trong database; findRap()\_tìm rạp trong database; findSoDayGhe()\_tìm số dãy ghế trong database; findSoLuongGhe()\_tìm số dãy ghế trong database.



##### .. BLPhimLinQ

#### BLRap

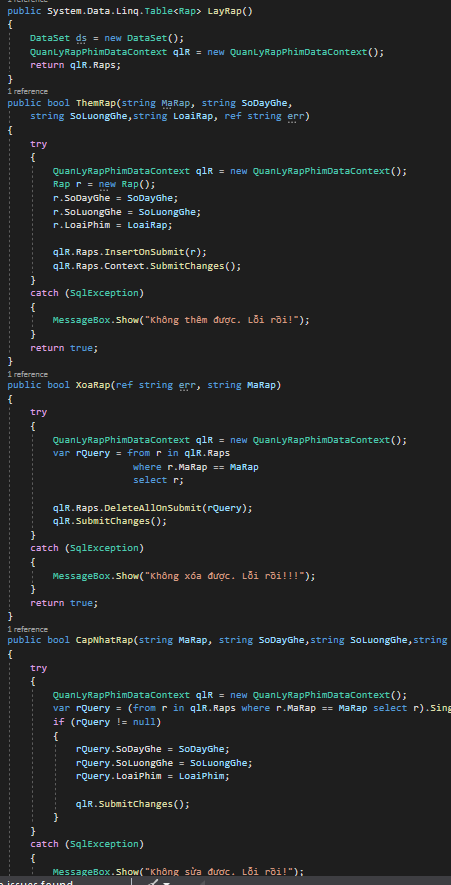
Chứa các hàm được thao tác qua DataContext và Table LinQ bao gồm LayChucVu()\_Lấy thông tin chức vụ; ThemChucVu()\_Thêm thông tin chức vụ mới vào database; XoaChucVu()\_Xoá chức vụ đã có trong database; CapNhatChucVu()\_Thay đổi thông tin đã có trong database.



##### .. BLRapLinQ

#### BLSuatChieu

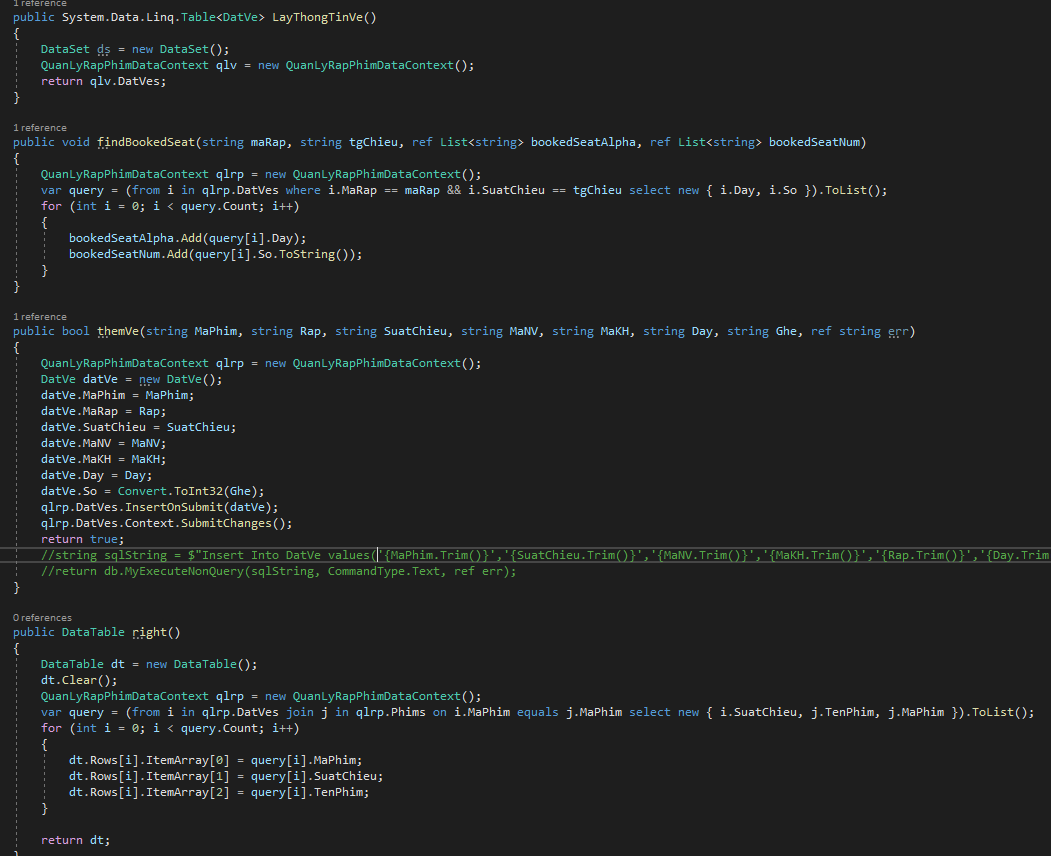
Chứa các hàm được thao tác qua DataContext và Table LinQ bao gồm LaySuatChieu()\_Lấy thông tin của xuất chiếu; ThemSuatChieu()\_Thêm thông tin suất chiếu mới vào database; XoaSuatChieu()\_Xoá suất chiếu đã có trong database; CapNhatSuatChieu()\_Thay đổi thông tin đã có trong database.



##### ..BLSuatChieuLinQ

#### BLDatVe

Class này có chức năng tìm các ghế đã book, ko cho người dùng book nữa(FindBookSeat), chức năng thêm vé cho người dùng lựa chọn vé ngồi(themVe). Và xác nhận thông tin đã book(right)



##### .. BLDatVeLinQ

## Kết luận

### Ưu điểm:

* Đề tài mang tính thiết thực cao.
* Sử dụng được nhiều kiến thức đã học vào đề tài.

### Nhược điểm:

* Hàm renderSeat:
* Khi load khá chậm nếu như rạp quá lớn.
* Mất 1 khoảng thời gian để render khiến người dùng phải chờ đợi.
* Vì sử dụng list<button> để lưu lại nút nên chậm hơn cách kéo thả.
* Database chưa được tối ưu.
* Vì sử dụng linQ nên chỉ ánh xạ 1:1, không thể dùng ánh xạ 1:n hoặc n:n.
* Giao diện chưa sinh động và thân thiện.

### Hướng phát triển:

* Cải thiện tốc độ load ghế.
* Sử dụng entity để có thể ánh xạ 1:n hoặc n:n
* Điều chỉnh database hợp lý hơn.
* Phát triển giao diện đẹp hơn, thân thiện hơn