

LỜI CẢM ƠN

Hoàn thành khóa luận là bước đầu tiên trong quá trình chuẩn bị kiến thức và tích lũy kinh nghiệm cho bản thân trước khi ra trường. Qua đó, chúng em không những được học hỏi tiếp thu thêm được nhiều kiến thức mà còn là cơ hội để chúng em có thể vận dụng những kiến thức trong quá trình học tập vào thực tiễn. Khóa luận thể hiện kết quả của quá trình học tập và là thước đo những kiến thức và kinh nghiệm của chúng em tích lũy được sau 4 năm đại học. Qua đó, chúng em không những được ôn lại kiến thức cũ, học hỏi thêm kiến thức mới mà còn nhận thức được điểm yếu của mình và tìm cách khắc phục.

Chúng em có được nhiều kinh nghiệm như vậy là sự giúp đỡ tận tình của các thầy cô trong Khoa Công nghệ thông tin – trường Đại học Sư phạm Kỹ thuật thành phố Hồ Chí Minh. Chúng em xin chân thành cảm ơn các thầy cô Khoa Công nghệ thông tin đã tạo điều kiện cho chúng em khóa luận. Chúng em vô cùng cảm ơn người đã trực tiếp hướng dẫn chúng em là Thầy Nguyễn Minh Đạo – thầy đã nhiệt tình chỉ dẫn cho chúng em giải quyết các vấn đề khó khăn trong bài làm. Chúng em xin cảm ơn Thầy Lê Văn Vinh – giáo viên phản biện – đã cho chúng em những lời khuyên để hoàn chỉnh báo cáo của mình.

Một lần nữa chúng em xin cảm ơn sự giúp đỡ của thầy cô. Nhóm em xin chúc thầy cô sức khỏe và công tác tốt!

Hồ Chí Minh, ngày 12 tháng 07 năm 2013

Sinh viên

Trần Chí Tâm – Lương Thị Như Quỳnh

Mục lục

PHẦN 1: MỞ ĐẦU.....	1
1.1 Tính cấp thiết của đề tài	1
1.2 Ý nghĩa khoa học và thực tiễn.....	2
1.3 Mục tiêu của đề tài	2
1.4 Đối tượng và phạm vi nghiên cứu.....	2
1.4.1 Đối tượng nghiên cứu.....	2
1.4.2 Phạm vi nghiên cứu.....	3
PHẦN 2: NỘI DUNG.....	4
CHƯƠNG 1: .NET FRAMEWORK 4.5	5
1.1 Tổng quan	5
1.1.1 Giới thiệu.....	5
1.2 Lịch sử phát triển	7
1.2.1 .NET Framework 1.0.....	7
1.2.2 .NET Framework 1.1.....	7
1.2.3 .NET Framework 2.0.....	7
1.2.4 .NET Framework 3.0.....	8
1.2.5 .NET Framework 3.5.....	8
1.2.6 .NET Framework 4.0.....	9
1.2.7 .NET Framework 4.5.....	9
1.3 .NET Framework 4.5	9
1.3.1 .NET cho Windows Store App	9
1.3.2 Portable Class Library.....	9
1.3.3 Các yếu tố mới và cải tiến.....	9
1.3.4 Tools.....	11

1.3.5	Parallel Computing.....	11
1.3.6	Web	12
1.3.7	Networking.....	12
1.3.8	Windows Presentation Foundation (WPF)	12
1.3.9	Windows Communication Foundation (WCF)	13
1.3.10	Windows Workflow Foundation (WF)	14
1.4	Kết luận	16
	CHƯƠNG 2: ENTITY FRAMEWORK 5.0	17
2.1	Tổng quan về Entity Framework	17
2.1.1	Giới thiệu về Entity Framework	17
2.1.2	Các tính năng vượt trội của Entity Framework.....	17
2.1.3	Lợi ích của Entity Framework	18
2.1.4	Các thành phần của Entity Framework	19
2.1.4.1	Conceptual schema definition language (CSDL)	20
2.1.4.2	Store schema definition language (SSDL).....	20
2.1.4.3	Mapping specification language (MSL)	20
2.1.5	Kiến trúc xử lý dữ liệu của Entity Framework	21
2.1.6	Các phiên bản của Entity Framework	22
2.2	Giới thiệu về Entity Framework 5.0.....	24
2.2.1	Giới thiệu.....	24
2.2.2	Các tính năng mới	24
2.2.3	Cài đặt Entity Framework 5.0	25
2.2.3.1	Sử dụng dòng lệnh Package Manager Console.....	25
2.2.3.2	Sử dụng giao diện đồ họa Manage NuGet Package.....	26
2.3	Các hướng tiếp cận Entity Framework.....	27
2.3.1	The Model – first Workfl	27
2.3.1.1	Định nghĩa	27

2.3.1.2	Sử dụng Model –first	27
2.3.2	The database– first workflow	33
2.3.1	Định nghĩa	33
2.3.2	Sử dụng Database – frist	33
2.3.3	The Code – first workflow	35
2.3.3.1	Định nghĩa	35
2.3.3.2	Sử dụng code – first	35
2.3.3.3	Khởi tạo cơ sở dữ liệu - Database Initialization	40
2.3.3.4	Các chiến lược khởi tạo cơ sở dữ liệu trong Code – first	41
2.3.3.5	Cấu hình các lớp miền trong Code – first	43
2.3.3.6	Cấu hình các liên kết ^[4]	47
2.3.3.7	Migration.....	50
2.4	Thao tác với entity sử dụng DbContext^[4]	53
2.4.1	Thêm một thực thể	53
2.4.1.1	Thêm một thực thể đơn	53
2.4.1.2	Thêm thực thể trong mối quan hệ 1 – 1	54
2.4.1.3	Thêm thực thể trong mối quan hệ 1 – nhiều	55
2.4.1.4	Thêm thực thể trong mối quan hệ nhiều – nhiều	56
2.4.2	Cập nhật thực thể	57
2.4.2.1	Cập nhật thực thể đơn	57
2.4.2.2	Cập nhật thực thể trong mối quan hệ 1 – 1	59
2.4.2.3	Cập nhật thực thể trong mối quan hệ một – nhiều	60
2.4.2.4	Cập nhật thực thể trong mối quan hệ nhiều – nhiều	61
2.4.3	Xóa thực thể	62
2.5	Các tính năng mới của Entity Framework 5.0	63
2.5.1	Tạo lớp DbContext trong Entity Framework 5.0	63
2.5.2	Kiểu Enum trong EF 5.0	63
2.5.2.1	Chuyển đổi một thuộc tính thành kiểu Enum	64

2.5.2.2	Thêm Enum mới từ EDM Design.....	65
2.5.2.3	Sử dụng Enum từ một không gian tên khác.....	65
2.5.3	Spatial Data type ^[4]	65
2.5.4	Table-Valued Function.....	66
2.5.5	Batch Import of Stored Procedures.....	69
2.5.6	Multiple diagrams.....	70
2.5.6.1	Tạo một sơ đồ mới và kéo thả các thực thể vào.....	71
2.5.6.1	Di chuyển thực thể từ sơ đồ có sẵn.....	72
2.5.7	Tô màu cho các thực thể trong EDM Designer.....	72
2.6	Kết luận.....	73
CHƯƠNG 3: XÂY DỰNG PHẦN MỀM QUẢN LÝ BÁN HÀNG VẬT LIỆU		
XÂY DỰNG.....		74
3.1	Phân tích hệ thống.....	74
3.1.1	Khảo sát hiện trạng và đặt yêu cầu.....	74
3.1.2	Mô hình giải quyết bài toán.....	75
3.1.3	Usecase tổng quát của phần mềm và website:.....	76
3.1.4	Cơ sở dữ liệu.....	77
3.2	Các chức năng chính.....	91
3.2.1	Thiết lập hệ thống.....	91
3.2.1.1	Thiết lập thông tin cửa hàng.....	91
3.2.1.2	Tạo tài khoản và phân quyền.....	94
3.2.2	Quản lý xuất nhập.....	97
3.2.2.1	Nhập hàng từ nhà cung cấp.....	97
3.2.2.2	Xuất hàng theo hóa đơn.....	100
3.2.2.3	Nhập hàng lỗi từ khách hàng.....	101
3.2.2.4	Xuất hàng lỗi trả nhà cung cấp.....	103
3.2.2.5	Xuất hàng theo hợp đồng.....	105
3.2.3	Quản lý bán hàng.....	107

3.2.3.1	Lập đơn hàng.....	108
3.2.3.2	Lập hóa đơn.....	110
3.2.4	Quản lý hợp đồng.....	112
3.2.4.1	Lập hợp đồng	112
3.2.4.2	Thêm bớt/hàng hợp đồng	117
3.2.5	Quản lý công nợ	120
3.2.5.1	Xem sách công nợ của nhà cung cấp	120
3.2.6	Quản lý thu chi	124
3.2.6.1	Lập phiếu thu tiền.....	124
3.2.6.2	Lập phiếu chi tiền.....	127
3.2.7	Xem thống kê – báo cáo.....	130
3.3	Một số tính năng hỗ trợ	131
3.3.1	Chọn giao diện làm việc khi bắt đầu.....	131
3.3.2	Cho phép xuất các loại giấy tờ theo nhiều loại file.....	131
3.3.3	Hiện thị trạng thái hàng bằng màu sắc	132
3.1	Kết luận	132

DANH MỤC BẢNG BIỂU

DANH MỤC BẢNG BIỂU

Chức Vụ	77
Nhân Viên	77
Quyền	78
Người Dùng	78
Phân Quyền	78
Danh Mục	78
Đơn Vị Tính	79
Nhà Cung Cấp	79
Mặt Hàng	80
Khách Hàng	80
Đơn Hàng	81
Chi Tiết Đơn Hàng	81
Hóa Đơn	81
Chi Tiết Hóa Đơn	82
Phiếu Xuất	82
Chi Tiết Phiếu Xuất	83
Phiếu Nhập	83
Chi Tiết Phiếu Nhập	84
Hợp Đồng	84
Chi Tiết Hợp Đồng	85
Đợt Thanh Toán	85
Đợt Giao	86
Chi Tiết Đợt Giao	86
Phụ Lục	86
Biên Bản Thanh Lý	87
Loại Chi	88
Phiếu Giao	88
Chi Tiết Phiếu Giao	88

DANH MỤC BẢNG BIỂU

Loại Thu	88
Phiếu Thu	89
Phiếu Chi	89
Chốt Tồn.....	90
Danh sách các xử lý giao diện nghiệp vụ thiết lập thông tin cửa hàng.....	93
Danh sách các xử lý giao diện nghiệp vụ tạo tài khoản và phân quyền.....	96
Danh sách các xử lý giao diện nghiệp vụ tạo tài khoản và phân quyền.....	99
Danh sách các xử lý giao diện nghiệp vụ xuất hàng theo hóa đơn	101
Dách các xử lý giao diện nghiệp vụ nhập hàng lỗi từ khách hàng	103
Danh sách các xử lý giao diện nghiệp vụ xuất hàng lỗi trả nhà cung cấp	105
Danh sách các xử lý giao diện nghiệp vụ xuất hàng theo hợp đồng.....	107
Danh sách các xử lý giao diện nghiệp vụ lập đơn hàng.....	109
Danh sách các xử lý giao diện nghiệp vụ lập hợp đồng.....	115
Danh sách các xử lý giao diện thêm/bớt hàng hợp đồng	119
Danh sách các xử lý giao diện nghiệp vụ xem công nợ nhà cung cấp.....	123
Danh sách các xử lý giao diện nghiệp vụ lập phiếu thu tiền hóa đơn mua hàng	126
Danh sách các xử lý giao diện nghiệp vụ lập phiếu chi tiền mua hàng từ nhà cung cấp	129

DANH MỤC HÌNH ẢNH

HÌNH 1.1	KIẾN TRÚC .NET FRAMEWORK.....	6
HÌNH 1.2	MỐI QUAN HỆ GIỮA CLR VÀ THƯ VIỆN ĐẾN CÁC ỨNG DỤNG	7
HÌNH 2.1	TỔNG QUAN VỀ ENTITY FRAMEWORK	17
HÌNH 2.2	ENTITY DATA MODEL – EDM	20
HÌNH 2.3	KIẾN TRÚC ENTITY FRAMEWORK	21
HÌNH 2.4	THIẾT LẬP NỀN .NET FRAMEWORK	26
HÌNH 2.5	GIAO DIỆN ĐỒ HỌA MANAGE NUGET PACKAGE.....	27
HÌNH 2.6	VÙNG LÀM VIỆC ENTITY DATA MODEL DESIGNER	28
HÌNH 2.7	THÊM MỚI 1 ENTITY	28
HÌNH 2.8	BIỂU DIỄN ENTITY TRÊN GIAO DIỆN	29
HÌNH 2.9	GIAO DIỆN THIẾT LẬP QUAN HỆ GIỮA 2 ENTITY	29
HÌNH 2.10	BIỂU DIỄN QUAN HỆ 2 ENTITY TRÊN GIAO DIỆN	30
HÌNH 2.11	HỘP THOẠI CHOOSE DATA SOURCE	30
HÌNH 2.12	HỘP THOẠI CONNECTION PROPERTIES.....	31
HÌNH 2.13	CHỌN CÁC THÀNH PHẦN CƠ SỞ DỮ LIỆU TRONG EDM WIZARD	34
HÌNH 2.14	GIAO DIỆN REFERENCES MANAGER.....	38
HÌNH 2.15	CÁCH CODE – FIRST KHỞI TẠO DỮ LIỆU	40
HÌNH 2.16	PACKAGE MANAGER CONSOLE	51
HÌNH 2.17	THƯ MỤC MIGRATIONS	51
HÌNH 2.18	ADD MIGRATION	51
HÌNH 2.19	FILE ĐIỀU CHỈNH TRONG THƯ MỤC MIGRATIONS	52
HÌNH 2.20	UPDATE MIGRATIONS.....	52
HÌNH 2.21	ENTITY ĐƠN.....	53
HÌNH 2.22	LIÊN KẾT 1-1.....	54
HÌNH 2.23	LIÊN KẾT 1 – NHIỀU	55
HÌNH 2.24	LIÊN KẾT NHIỀU – NHIỀU	56

DANH MỤC HÌNH ẢNH

HÌNH 2.25	CONVERT TO ENUM.....	64
HÌNH 2.26	ADD ENUM TYPE	64
HÌNH 2.27	THÊM CÁC TABLE – VALUED FUNCTION	68
HÌNH 2.28	EDIT TABLE – VALUED FUNCTION	69
HÌNH 2.29	ADD STORED PROCEDURES	70
HÌNH 2.30	ADD NEW DIAGRAM.....	71
HÌNH 2.31	KÉO THẢ ENTITY	72
HÌNH 2.32	MOVE ENTITY.....	72
HÌNH 2.33	TÔ MÀU CHO THỰC THỂ TRONG EDM DESIGNER	73
HÌNH 3.1	MÔ HÌNH GIẢI QUYẾT BÀI TOÁN QUẢN LÝ BÁN HÀNG CỦA CỬA HÀNG VẬT LIỆU XÂY DỰNG.....	75
HÌNH 3.2	USECASE TỔNG QUÁT.....	77
HÌNH 3.3	MÔ HÌNH THỰC THỂ DỮ LIỆU	77
HÌNH 3.4	CHỨC NĂNG THIẾT LẬP HỆ THỐNG	91
HÌNH 3.5	SƠ ĐỒ TUẦN TỰ CHỨC NĂNG THIẾT LẬP THÔNG TIN CỦA HÀNG	92
HÌNH 3.6	GIAO DIỆN NGHIỆP VỤ THIẾT LẬP THÔNG TIN CỦA HÀNG .	93
HÌNH 3.7	GIAO DIỆN WEB HIỂN THỊ THÔNG TIN CỦA HÀNG	94
HÌNH 3.8	SƠ ĐỒ TUẦN TỰ CHỨC NĂNG TẠO TÀI KHOẢN VÀ PHÂN QUYỀN	95
HÌNH 3.9	GIAO DIỆN NGHIỆP VỤ TẠO TÀI KHOẢN VÀ PHÂN QUYỀN .	96
HÌNH 3.10	CHỨC NĂNG QUẢN LÝ XUẤT NHẬP	97
HÌNH 3.11	SƠ ĐỒ TUẦN TỰ CHỨC NĂNG NHẬP HÀNG TỪ NHÀ CUNG CẤP	98
HÌNH 3.12	GIAO DIỆN NGHIỆP VỤ NHẬP HÀNG TỪ NHÀ CUNG CẤP.....	99
HÌNH 3.13	SƠ ĐỒ TUẦN TỰ CHỨC NĂNG XUẤT HÀNG THEO HÓA ĐƠN	101
HÌNH 3.14	GIAO DIỆN NGHIỆP VỤ XUẤT HÀNG THEO HÓA ĐƠN	101

DANH MỤC HÌNH ẢNH

HÌNH 3.15 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG NHẬP HÀNG LỖI TỪ KHÁCH HÀNG	102
HÌNH 3.16 GIAO DIỆN NGHIỆP VỤ NHẬP HÀNG LỖI TỪ KHÁCH HÀNG	103
HÌNH 3.17 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG XUẤT HÀNG LỖI TRẢ NHÀ CUNG CẤP.....	104
HÌNH 3.18 GIAO DIỆN NGHIỆP VỤ XUẤT HÀNG LỖI TRẢ NHÀ CUNG CẤP	105
HÌNH 3.19 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG XUẤT HÀNG THEO HỢP ĐỒNG	106
HÌNH 3.20 GIAO DIỆN NGHIỆP VỤ XUẤT HÀNG THEO HỢP ĐỒNG.....	107
HÌNH 3.21 CHỨC NĂNG QUẢN LÝ BÁN HÀNG.....	108
HÌNH 3.22 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG LẬP ĐƠN HÀNG	108
HÌNH 3.23 GIAO DIỆN NGHIỆP VỤ LẬP ĐƠN HÀNG.....	109
HÌNH 3.24 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG LẬP HÓA ĐƠN	111
HÌNH 3.25 CHỨC NĂNG QUẢN LÝ HỢP ĐỒNG	112
HÌNH 3.26 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG LẬP HỢP ĐỒNG.....	114
HÌNH 3.27 GIAO DIỆN NGHIỆP VỤ LẬP HỢP ĐỒNG	115
HÌNH 3.28 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG THÊM/BỚT HÀNG HỢP ĐỒNG	118
HÌNH 3.29 GIAO DIỆN NGHIỆP VỤ THÊM/BỚT HÀNG HỢP ĐỒNG.....	119
HÌNH 3.30 CHỨC NĂNG QUẢN LÝ CÔNG NỢ.....	120
HÌNH 3.31 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG LẬP DANH CÔNG NỢ CỦA HÀNG NHÀ CUNG CẤP	121
HÌNH 3.32 GIAO DIỆN NGHIỆP VỤ XEM CÔNG NỢ NHÀ CUNG CẤP.....	123
HÌNH 3.33 CHỨC NĂNG QUẢN LÝ THU CHI.....	124
HÌNH 3.34 SƠ ĐỒ TUẦN TỰ CHỨC NĂNG LẬP PHIẾU THU TIỀN HÓA ĐƠN MUA HÀNG.....	125

DANH MỤC HÌNH ẢNH

HÌNH 3.35 GIAO DIỆN NGHIỆP VỤ LẬP PHIẾU THU TIỀN HÓA ĐƠN MUA HÀNG	126
HÌNH 3.36 SƠ ĐỒ TUẦN TỰ CHÚC NĂNG LẬP PHIẾU CHI TIỀN MUA HÀNG TỪ NHÀ CUNG CẤP	128
HÌNH 3.37 GIAO DIỆN NGHIỆP VỤ LẬP PHIẾU CHI TIỀN MUA HÀNG TỪ NHÀ CUNG CẤP	129
HÌNH 3.38 CHÚC NĂNG XEM THỐNG KÊ – BÁO CÁO.....	130
HÌNH 3.39 GIAO DIỆN NGHIỆP VỤ THỐNG KÊ HÀNG HÓA	131
HÌNH 3.40 GIAO DIỆN CHỌN MÀN HÌNH LÀM VIỆC	131
HÌNH 3.41 GIAO DIỆN IN GIẤY TỜ.....	132
HÌNH 3.42 KIỂM TRA TRẠNG THÁI HÌNH	132

DANH MỤC CÁC TỪ VIẾT TẮT

DANH MỤC CÁC TỪ VIẾT TẮT

LINQ	Language-Integrated Query
OOB	Out-of-Band
MEF	Managed Extensibility Framework
ORM	Object/Relational Mapping
POCO	Plain Old CLR Objects
EDM	Entity Data Model
CSDL	Conceptual schema definition language
SSDL	Store schema definition language
MSL	Mapping specification language
EF	Entity Framework
DDL	Data Definition Language
TVF	Table-Valued Function

PHỤ LỤC

GIỚI THIỆU SENCHA TOUCH 2.0	136
-----------------------------------	-----

PHẦN 1: MỞ ĐẦU

1.1 Tính cấp thiết của đề tài

Trong hầu hết mọi lĩnh vực, việc lưu trữ và xử lý dữ liệu là rất quan trọng. Nó cung cấp thông tin cho con người để giải quyết những vấn đề trong đời sống. Cùng với sự phát triển của khoa học – công nghệ, sự phát triển của kinh tế và con người, nguồn dữ liệu trên thế giới ngày càng tăng và đa dạng. Để có thể quản lý các nguồn dữ liệu cần phải có những công cụ hỗ trợ cho việc quản lý được dễ dàng và hiệu quả hơn.

Không đứng ngoài sự phát triển đó, Microsoft cho ra đời công nghệ Entity Framework 3.5 trên nền .Net Framework 3.5. Năm 2012, phiên bản Entity Framework 5.0 trên nền .Net Framework 4.5 ra đời kế thừa được những phiên bản trước đó và phát triển thêm các tính năng hơn, hỗ trợ người lập trình tốt hơn và đáp ứng đa dạng các chương trình hơn. Entity Framework 5.0 trên .Net Framework 4.5 không còn xa lạ trên thế giới nhưng ở Việt Nam các nguồn tài liệu còn hạn chế và chưa có bài bản, chủ yếu là giới thiệu về công nghệ, do đó gây khó khăn cho những người muốn áp dụng vào thực tế.

Hầu hết các ứng dụng đều làm việc với cơ sở dữ liệu, các ứng dụng kết nối dữ liệu qua công nghệ ADO.NET còn gây nhiều khó khăn khi mà lập trình hướng đối tượng ngày càng phổ biến. Entity Framework giúp các lập trình viên có thể tiếp cận với dữ liệu dễ hơn do thao tác với dữ liệu như những đối tượng. Nhờ công nghệ này mà lập trình viên không cần quan tâm đến vấn đề kết nối và làm việc với cơ sở dữ liệu mà tập trung hơn để đáp ứng các tính năng phần mềm theo yêu cầu của thực tiễn.

Với sự yêu thích về các công nghệ mới chạy trên nền tảng tiên tiến, nhóm chúng em đã quyết định chọn đề tài **TÌM HIỂU ENTITY FRAMEWORK 5.0 - .NET FRAMEWORK 4.5 VÀ XÂY DỰNG ỨNG DỤNG MINH HỌA**, một phần phục vụ cho công tác học tập và nghiên cứu, một phần cũng muốn đóng góp một chút công sức vào nền tri thức nước nhà. Chúng em hy vọng những kiến thức chúng em thu thập

được sẽ giúp các bạn sinh viên biết thêm được một công nghệ về kết nối và xử lý dữ liệu và áp dụng nó cho thực tiễn sau này.

1.2 Ý nghĩa khoa học và thực tiễn

Với kết quả nghiên cứu này sẽ đóng góp một phần kiến thức cho nước nhà, đồng thời làm tài liệu Tiếng Việt cho những người quan tâm đến Entity Framework để dàng nghiên cứu hơn và là nền cho sự tìm hiểu các phiên bản sau này.

Nội dung nghiên cứu hầu như tập trung vào nghiên cứu các áp dụng công nghệ vào thực tiễn, nhờ đó các lập trình viên có thể dễ dàng tham khảo để xây dựng không chỉ các ứng dụng quản lý sử dụng Entity Framework mà các chương trình khác có làm việc với cơ sở dữ liệu.

Phần mềm quản lý giao dịch cửa hàng vật liệu xây dựng sử dụng Entity Framework 5.0 trên .Net Framework 4.5 có thể làm một ví dụ mẫu cho các phần mềm khác hoặc có thể đưa vào thực tế sử dụng.

1.3 Mục tiêu của đề tài

Xây dựng tài liệu về cách sử dụng Entity Framework 5.0 trên nền .NET Framework, những điểm mới của Entity Framework 5.0 và .NET Framework 4.5.

Tìm hiểu thêm công nghệ Windows Communication Foundation và công nghệ xây dựng Website mới là Sencha Touch 2.0.

Xây dựng phần mềm quản lý giao dịch cho cửa hàng xây dựng có thể áp dụng vào thực tiễn và cho nhiều cửa hàng xây dựng khác nhau. Phần mềm mở rộng với sự quản lý dữ liệu trên Service, thao tác trên Windows Form và hiển thị sản phẩm trên Website.

1.4 Đối tượng và phạm vi nghiên cứu

1.4.1 Đối tượng nghiên cứu

- ✓ Các tính năng mới của Entity Framework 5.0, .Net Framework 4.5.

- ✓ Bộ thư viện hỗ trợ Entity Framework, Windows Form, WCF, Sencha Touch 2.0.
- ✓ Quy trình phát triển phần mềm.

1.4.2 Phạm vi nghiên cứu

- ✓ Cách sử dụng các tính năng mới của Entity Framework 5.0, .Net Framework 4.5.
- ✓ Các hàm hỗ trợ kết nối, truy vấn dữ liệu trong thư viện System.Data.Entity, System.Linq.
- ✓ Xây dựng Website đơn giản hiển thị sản phẩm dùng Sencha Touch 2.0.
- ✓ Xây dựng WCF Service chứa các phương thức kết nối và xử lý dữ liệu.
- ✓ Xây dựng ứng dụng Windows Form quản lý giao dịch cửa hàng vật liệu xây dựng kết nối với WCF Service.

PHẦN 2: NỘI DUNG

CHƯƠNG 1: .NET FRAMEWORK 4.5

CHƯƠNG 2: ENTITY FRAMEWORK 5.0

**CHƯƠNG 3: PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU
XÂY DỰNG**

CHƯƠNG 1: .NET FRAMEWORK 4.5

1.1 Tổng quan

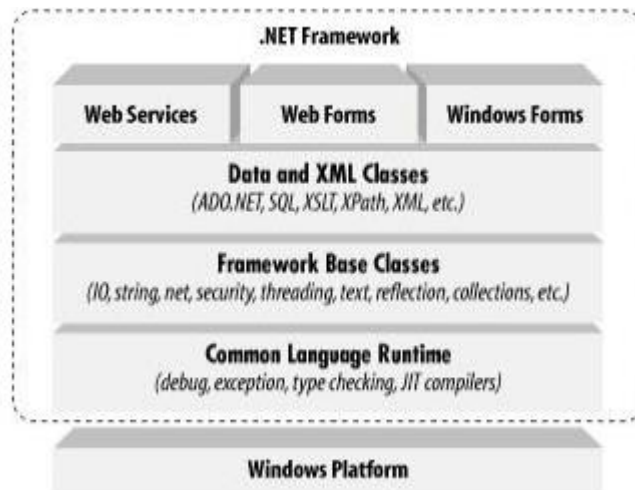
1.1.1 Giới thiệu

.NET Framework là một công nghệ hỗ trợ việc xây dựng và chạy các thể hệ tiếp theo ứng dụng và dịch vụ web XML. **.NET Framework** được thiết kế để hoàn thành các mục tiêu sau đây:

- Cung cấp môi trường lập trình hướng đối tượng phù hợp cho dù mã đối tượng được lưu trữ và thực thi cục bộ, tuy vậy có thể phân phối qua internet hoặc thực thi từ xa.
- Cung cấp môi trường thực thi sao cho giảm thiểu số chương trình được triển khai và xung đột các phiên bản.
- Cung cấp môi trường thực thi mã đầy mạnh sự an toàn của các đoạn mã thực thi, bao gồm các đoạn mã được tạo ra bởi bên thứ ba vô danh hoặc bán tin cậy.
- Cung cấp môi trường thực thi mã có thể loại bỏ được các vấn đề hiệu suất của cách viết và môi trường thông dịch.
- Giúp cho các nhà phát triển có kinh nghiệm nhất quán trên nhiều loại ứng dụng, chẳng hạn như các ứng dụng trên nền tảng Windows và các ứng dụng trên nền tảng web.
- Xây dựng tất cả các giao tiếp dựa trên tiêu chuẩn công nghiệp để chắc chắn rằng mã dựa trên **.NET Framework** có thể tích hợp với bất kỳ mã nào.

.NET Framework bao gồm bộ thực thi ngôn ngữ chung (Common Language Runtime - CLR) và lớp thư viện .NET Framework (FCL). CLR là nền tảng của **.NET Framework**. Bạn có thể nghĩ runtime như là tác nhân quản lý mã tại thời gian thực hiện, cung cấp các ứng dụng cốt lõi như quản lý bộ nhớ, quản lý luồng và remoting, xác nhận mã nguồn an toàn và các hình thức khác của việc chính xác mã nguồn. Trong thực tế, khái niệm về quản lý mã là nguyên tắc cơ bản của runtime. Trong lập trình, mục tiêu runtime được xem như quản lý mã (managed code), trong khi các mã không được thực thi bởi runtime được xem như mã không quản lý. Thư viện là một bộ sưu tập toàn diện, hướng đối tượng mà bạn có thể tái sử dụng để phát triển các

ứng dụng khác nhau, từ các ứng dụng dòng lệnh hoặc giao diện đồ họa (GUI) đến các ứng dụng dựa trên sự sáng tạo cung cấp bởi ASP.NET, chẳng hạn như biểu mẫu web, hoặc dịch vụ web XML.



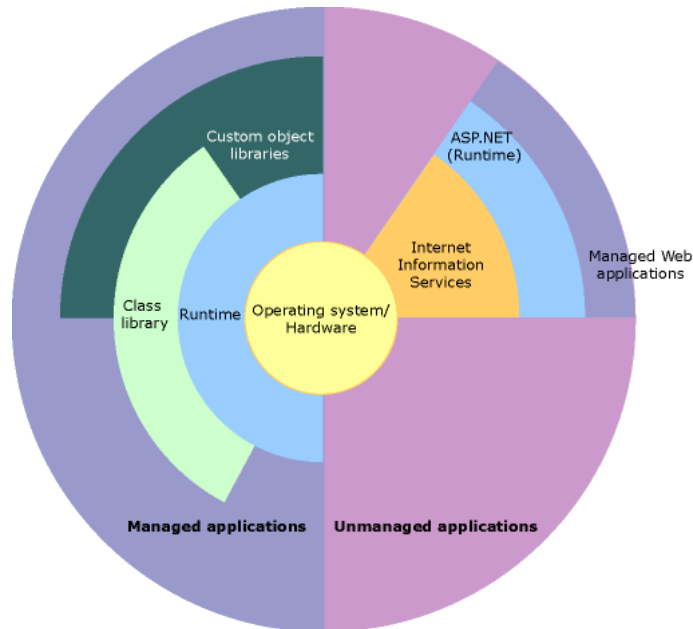
Hình 1.1 Kiến trúc .NET Framework

.NET Framework được lưu trữ bởi các thành phần không quản lý, CLR được tải đến các processes của thành phần này và bắt đầu thực thi managed code, do đó tạo ra môi trường phần mềm có thể khai thác tính năng quản lý và không quản lý. **.NET Framework** không chỉ cung cấp một số host runtime, tuy nhiên vẫn hỗ trợ sự phát triển các host runtime từ bên thứ ba.

Ví dụ như các host ASP.NET runtime đảm bảo sự ổn định, môi trường máy chủ cho các mã quản lý. ASP.NET làm việc trực tiếp theo thời gian thực để phục vụ cho ứng dụng ASP.NET và dịch vụ web XML.

Internet Explorer là ví dụ về một ứng dụng không được quản lý ở thời gian thực (trong hình thức mở rộng MIME). Bằng cách sử dụng Internet Explorer để lưu trữ ở thời gian thực cho phép bạn nhúng các thành phần được quản lý hoặc các Windows Form control trong văn bản HTML. Lưu trữ thời gian thực theo cách này làm cho các mã di động có thể quản lý. Nhưng với những cải tiến đáng kể chỉ cung cấp cho quản lý mã, chẳng hạn như thực thi bán tin cậy và cô lập tập tin lưu trữ.

Minh họa dưới đây cho thấy mối quan hệ giữa CLR và thư viện đến ứng dụng của bạn và hệ thống tổng thể. Đồng thời cho thấy mã quản lý hoạt động trong vòng kiến trúc lớn hơn:



Hình 1.2 Mối quan hệ giữa CLR và thư viện đến các ứng dụng

1.2 Lịch sử phát triển

1.2.1 .NET Framework 1.0

Đây là phiên bản đầu tiên của .NET Framework , nó được phát hành vào năm 2002 cho các hệ điều hành Windows 98, NT 4.0, 2000 và XP. Việc hỗ trợ chính thức từ Microsoft cho phiên bản này kết thúc vào 10/2007, tuy nhiên thời gian hỗ trợ mở rộng được kéo dài đến 14/7/2009.

1.2.2 .NET Framework 1.1

Phiên bản nâng cấp đầu tiên được phát hành vào 4/2003. Sự hỗ trợ của Microsoft kết thúc vào 14/10/2008 và hỗ trợ mở rộng đến 8/10/2013.

1.2.3 .NET Framework 2.0

Kể từ phiên bản này, .NET Framework hỗ trợ đầy đủ nền tảng 64-bit. Ngoài ra, cũng có một số thay đổi trong API; hỗ trợ các kiểu “generic”; bổ sung sự hỗ trợ

cho ASP.NET; .NET Microsoft Framework – một phiên bản .NET Framework có quan hệ với Smart Personal Objects Technology.

1.2.4 .NET Framework 3.0

Đây không phải là một phiên bản mới hoàn toàn, thực tế chỉ là bản nâng cấp của .NET 2.0. Phiên bản 3.0 này còn có tên gọi khác là WinFX, nó bao gồm nhiều sự thay đổi nhằm hỗ trợ việc phát triển và chuyển đổi (porting) các ứng dụng trên Windows Vista. Tuy nhiên, không có sự xuất hiện của .NET Compact Framework 3.0 trong lần phát hành này. Bốn thành phần chính trong phiên bản 3.0:

- ✓ Windows Presentation Foundation (WPF – tên mã là Avalon).
- ✓ Windows Communication Foundation (WCF – tên mã là Indigo).
- ✓ Windows Workflow Foundation (WF).
- ✓ Windows CardSpace (tên mã là InfoCard).

Ngoài ra Silverlight (hay WPF/E) một phiên bản nhánh .NET Framework hỗ trợ các ứng dụng trên nền web, được Microsoft tạo ra để cạnh tranh với Flash.

Có thể minh họa .NET 3.0 bằng một công thức đơn giản:

$$.NET\ 3.0 = .NET\ 2.0 + WPF + WCF + WF + WCS$$

1.2.5 .NET Framework 3.5

Được phát hành vào 11/2007, phiên bản này sử dụng CLR 2.0. Đây có thể được xem là tương đương với phiên bản .NET Framework 2.0 SP1 và .NET Framework 3.0 SP1 cộng lại. .NET Compact Framework 3.5 được ra đời cùng với phiên bản .NET Framework này.

Cũng như phiên bản 3.0, có thể minh họa sự thay đổi của .NET 3.5 bằng công thức:

$$.NET\ 3.5 = .NET\ 3.0 + LINQ + ASP.NET\ 3.5 + REST$$

1.2.6 .NET Framework 4.0

Phiên bản beta đầu tiên của .NET 4 xuất hiện vào 5/2009 và phiên bản RC (Release Candidate) được ra mắt vào 2/2010. Bản chính thức của .NET 4 được công bố và phát hành cùng với Visual Studio 2010 vào 12/4/2010.

1.2.7 .NET Framework 4.5

Những thông tin đầu tiên của .NET Framework 4.5 được Microsoft công bố vào 14/9/2011 tại BUILD Windows Conference, và nó chính thức được ra mắt vào 15/8/2012.

1.3 .NET Framework 4.5

1.3.1 .NET cho Windows Store App

Windows Store apps được thiết kế đặc biệt cho các chuẩn và tận dụng sức mạnh của hệ điều hành Windows. Một tập con của .NET Framework sẵn có cho việc xây dựng các Windows Store app bằng cách sử dụng C# hoặc Visual Basic. Các tập con này gọi là .NET cho các Windows Store app và được thảo luận trong phần tổng quan Windows Dev Center.

1.3.2 Portable Class Library

Project Portable Class Library trong Visual Studio 2012 (hoặc các phiên bản sau này) cho phép bạn viết và xây dựng managed assemblies làm việc đa nền tảng .NET Framework. Dùng một project Portable Class Library, chọn nền tảng (như Windows Phone và .NET cho các Windows Store app) hướng đến. Những types sẵn có và các member trong project của bạn thì tự động giới hạn thành các common type và member trên các nền tảng. Để biết thêm chi tiết, xem Cross-Platform Development với .NET Framework.

1.3.3 Các yếu tố mới và cải tiến

Nhóm .NET Framework đã bắt đầu cung cấp liên tục các tính năng với NuGet. Tính năng released out of band (OOB) cho phép mở rộng nền tảng hỗ trợ và giới thiệu

các chức năng mới. Để biết thêm thông tin, xem .NET Framework và Out-of-Band Releases. Tính năng mới của .NET Framework 4.5:

- ✓ Giảm khả năng khởi động lại hệ thống khi phát hiện và đóng ứng dụng .NET Framework 4 đang hoạt động.
- ✓ Hỗ trợ các mảng có kích thước trên 2 gigabytes (GB) trên nền tảng 64-bit. Tính năng này có thể được kích hoạt trong tập tin cấu hình ứng dụng.
- ✓ Hiệu suất tốt hơn thông qua các background garbage collection cho máy chủ. Khi bạn sử dụng background garbage collection trong .NET Framework 4.5, tính năng này được kích hoạt tự động.
- ✓ Background just-in-time (JIT) compilation, là tùy chọn có sẵn trên bộ xử lý đa lõi để cải thiện hiệu suất ứng dụng.
- ✓ Giới hạn khả năng độ dài của công cụ Regular Expressions sẽ cho phép xử lý các biểu thức chính quy trước khi kết thúc.
- ✓ Khả năng định dạng các culture mặc định cho một miền ứng dụng.
- ✓ Console hỗ trợ mã Unicode (UTF-16).
- ✓ Hỗ trợ phiên bản của cultural string ordering và comparison data.
- ✓ Hiệu suất tốt hơn khi lấy tài nguyên.
- ✓ Cải tiến Zip để file nén có kích thước nhỏ hơn.
- ✓ Có thể tùy chỉnh reflection context để override reflection behavior mặc định thông qua CustomReflectionContext class.
- ✓ Hỗ trợ phiên bản 2008 Internationalized Domain Names in Applications (IDAN) đầy đủ trong lớp System.Globalization.IdnMapping dùng trên Windows 8.
- ✓ So sánh chuỗi delegate để hệ điều hành thực hiện Unicode 6.0, khi .NET Framework dùng trên Windows 8. Khi chạy trên các nền tảng khác, .NET Framework so sánh chuỗi dữ liệu của riêng nó, cho phép thực thi Unicode 5.x.
- ✓ Khả năng tính toán mảng băm cho chuỗi trên một application domain cơ bản.

- ✓ Hỗ trợ các loại tham chiếu giữa Type và TypeInfo classes.

Managed Extensibility Framework (MEF):

Managed Extensibility Framework (MEF) cung cấp các tính năng mới:

- ✓ Hỗ trợ các loại generic.
- ✓ Convention-based programming model cho phép bạn tạo ra các phần trên qui ước đặt tên chứ không phải là thuộc tính.
- ✓ Multiple scopes.
- ✓ Một tập con MEF mà bạn có thể sử dụng khi tạo các Windows Store app. Các tập con này có sẵn trong gói tải về từ thư viện NuGet. Để cài đặt, mở project của bạn trong Visual Studio 2012, chọn Manage NuGet Packages từ Project menu, và search online cho gói Microsoft.Composition.

Asynchronous File Operations:

Trong .NET Framework, tính năng mới không đồng bộ được thêm vào ngôn ngữ C# và Visual Basic. Thêm vào tính năng task-based model cho phép tác vụ thực hiện các hoạt động không đồng bộ. Để sử dụng mô hình này, dùng các phương thức không đồng bộ trong I/O classe.

1.3.4 Tools

Resource File Generator (Resgen.exe) cho phép bạn tạo một tập tin .resw dùng trong các Windows Store app từ một file .resources nhúng trong .NET Framework assembly.

Managed Profile Guided Optimization (Mpggo.exe) cho phép cải thiện thời gian khởi động ứng dụng, sử dụng bộ nhớ (kích thước tập làm việc) và thông lượng bằng cách tối ưu native image assemblies. Công cụ command-line tạo ra hồ sơ dữ liệu cho native image application assemblies.

1.3.5 Parallel Computing

.NET Framework cung cấp một số tính năng mới và cải tiến cho parallel computing. Chúng bao gồm cải thiện hiệu suất, gia tăng control, cải thiện việc hỗ trợ

cho lập trình không đồng bộ, một thư viện dataflow mới và cải tiến các hỗ trợ cho parallel debugging và phân tích hiệu năng.

1.3.6 Web

ASP.NET 4.5 thêm vào mô hình binding cho Web Forms, hỗ trợ WebSocket, xử lý không đồng bộ, cải tiến hiệu suất và một số tính năng khác.

1.3.7 Networking

.NET Framework 4.5 cung cấp một giao diện lập trình mới cho các ứng dụng HTTP. Hỗ trợ cũng bao gồm một giao diện lập trình mới cho việc nhận và tương tác với kết nối WebSocket bằng cách sử dụng HttpListener và các lớp liên quan.

Ngoài ra, .NET Framework 4.5 bao gồm các cải tiến mạng:

- ✓ URI hỗ trợ tương thích RFC.
- ✓ Hỗ trợ phân tích Internationalized Domain Name (IDN).
- ✓ Hỗ trợ Email Address Internationalization (EAI).
- ✓ Hỗ trợ cải tiến IPv6.
- ✓ Hỗ trợ Dual-mode socket.

1.3.8 Windows Presentation Foundation (WPF)

Trong .NET Framework 4.5, Windows Presentation Foundation (WPF) có một số thay đổi và cải tiến:

- ✓ Ribbon control mới, nó cho phép bạn thực thi giao diện người dùng ribbon bằng cách lưu trữ Quick Access Toolbar, Application Menu, và các tab.
- ✓ Interface INotifyDataErrorInfo mới, hỗ trợ xác nhận đồng bộ và không đồng bộ dữ liệu.
- ✓ Tính năng mới cho VirtualizingPanel và Dispatcher classes.
- ✓ Cải tiến hiệu năng khi hiển thị một nhóm dữ liệu lớn bằng cách truy cập bộ các chủ đề non-UI.

- ✓ Data binding cho phép các thuộc tính tĩnh, ràng buộc dữ liệu để tùy chỉnh các loại hiện thực hóa giao diện `IcustomTypeProvider`, và lấy thông tin data binding từ một binding expression.
- ✓ Tái định vị dữ liệu như values change (live shaping).
- ✓ Có khả năng kiểm tra trong khi data context cho một item container bị mất kết nối.
- ✓ Khả năng thiết lập lượng thời gian trôi qua giữa lúc thay đổi thuộc tính và cập nhật dữ liệu nguồn.
- ✓ Cải tiến hỗ trợ hiện thực hóa weak event patterns. Ngoài ra, sự kiện có thể bây giờ chấp nhận phần mở rộng.

1.3.9 Windows Communication Foundation (WCF)

Trong .NET Framework 4.5, các tính năng sau được thêm vào làm nó đơn giản để viết và duy trì các ứng dụng WCF:

- ✓ Đơn giản hóa tập tin cấu hình được tạo ra.
- ✓ Hỗ trợ phát triển contract-first.
- ✓ Hỗ trợ cấu hình ASP.NET tương thích dễ dàng hơn.
- ✓ Thay đổi trong chuyên đổi giá trị thuộc tính mặc định để giảm khả năng bạn phải đặt lại chúng.
- ✓ Cập nhật lớp `XmlDictionaryReaderQuotas` để bạn giảm khả năng cấu hình dung lượng cho thư viện đọc XML.
- ✓ Xác nhận file cấu hình WFC là một phần của quá trình xây dựng, vì thế bạn có thể phát hiện lỗi trước khi chạy ứng dụng.
- ✓ Hỗ trợ trực tuyến không đồng bộ.
- ✓ HTTPS protocol mapping làm dễ dàng hiển thị điểm cuối qua HTTPS với Internet Information Services (IIS).
- ✓ Có khả năng tạo ra siêu dữ liệu trong tài liệu WSDL duy nhất bằng cách đánh dấu? WSDL thành dịch vụ URL.

- ✓ Websockets hỗ trợ giao tiếp hai chiều trên cổng 80 và 443 với hiệu năng tương tự như TCP transport.
- ✓ Hỗ trợ cấu hình dịch vụ trong mã.
- ✓ Chú giải XML Editor.
- ✓ ChannelFactory hỗ trợ bộ nhớ đệm.
- ✓ Nén mã nhị phân.
- ✓ Hỗ trợ UDP transport cho phép nhà phát triển viết dịch vụ tin nhắn sử dụng “fire and forget”. Một khách hàng gửi tin nhắn đến một dịch vụ và không có phản hồi từ dịch vụ.
- ✓ Hỗ trợ nhiều chế độ xác thực trên một single WCF endpoint khi dùng HTTP transport và transport security.
- ✓ Hỗ trợ các dịch vụ WFC sử dụng internationalized domain names (IDNs).

1.3.10 Windows Workflow Foundation (WF)

Các tính năng mới được thêm vào WF trong .NET Framework 4.5:

- ✓ State machine workflows, được giới thiệu như một phần của .NET Framework 4.0.1 (.NET Framework 4 Platform Update 1). Bản cập nhật này bao gồm một số lớp và tính năng mới cho phép nhà phát triển tạo ra state machine workflows. Các lớp và tính năng được cập nhật cho .NET Framework 4.5 bao gồm:
 - Khả năng thiết lập breakpoints trên state.
 - Khả năng sao chép và dán transitions trên workflow designer.
 - Thiết kế hỗ trợ việc chia sẻ trigger transition creation.
 - Tính năng cho phép tạo state machine workflows, bao gồm: StateMachine, State, and Transition.
- ✓ Tính năng nâng cao Workflow Designer:
 - Tăng cường khả năng tìm kiếm trong Visual Studio, bao gồm Quick Find và Find in Files.

- Khả năng tự động tạo ra một Sequence activity khi con activity thứ hai được thêm vào container activity, và bao gồm activities trong Sequence activity.
- Hỗ trợ panning, cho phép các phần có thể nhìn thấy của một quy trình làm việc (workflow) được thay đổi mà không cần sử dụng thanh cuộn (scroll bars).
- Xem phát thảo tài liệu các thành phần của công việc trong chế độ tree-style, và cho phép bạn xem một phần trong xem phát thảo tài liệu.
- Có khả năng chú thích vào activities.
- Có khả năng định nghĩa và hủy các activity delegate bằng cách sử dụng workflow designer.
- Tự động kết nối và tự động chèn activities và transitions trong state machine và sơ đồ workflow.
- ✓ Lưu trữ thông tin trạng thái xem cho workflow trong element đơn của file XML, vì thế bạn dễ dàng xác định vị trí và chỉnh sửa thông tin trạng thái xem.
- ✓ NoPersistScope container activity ngăn activities con mạnh mẽ.
- ✓ Hỗ trợ C# expressions:
 - Projects workflow dùng visual basic sẽ dùng visual basic expressions và project workflow dùng c# sẽ dùng C# expressions.
 - Project C# workflow được tạo ra trong Visual Studio 2012 và có visual basic expressions tương thích với project C# workflow dùng C# expressions.
 - Cải tiến phiên bản:
 - Lớp WorkflowIdentity mới, cung cấp một ánh xạ giữa persisted workflow instance và workflow định nghĩa nó.
 - Thực hiện song song nhiều phiên bản workflow trong cùng một host, bao gồm WorkflowServiceHost.

- Trong Dynamic Update, có khả năng thay đổi định nghĩa của một persisted workflow instance.
- Phát triển dịch vụ Contract-first workflow, hỗ trợ cung cấp tự động tạo ra các activities phù hợp với hợp đồng dịch vụ có sẵn.

1.4 Kết luận

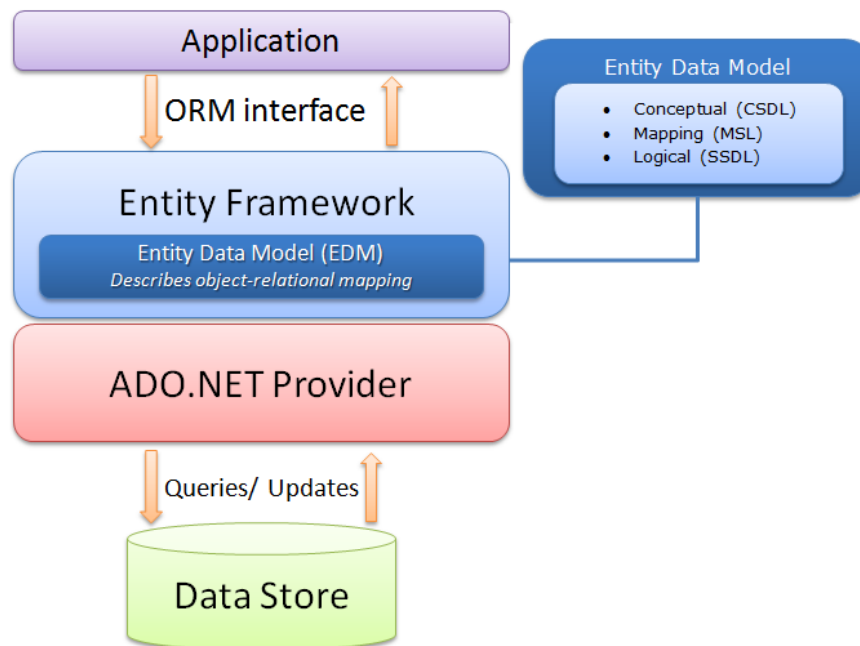
Với những gì đã trình bày, có thể thấy rằng .NET Framework 4.5 bao gồm những cải tiến đặc biệt về ngôn ngữ và nền tảng cho C#, Visual Basic và F# (vì thế có thể dễ dàng viết các đoạn mã asynchronous), sự pha trộn của control flow trong mã synchronous, giao diện cảm ứng, và khả năng mở rộng cho các ứng dụng web. Ngoài ra, .NET Framework 4.5 đã thực hiện những thay đổi liên quan đến ASP.NET, Managed Extensibility Framework, Windows Communication Foundation, Windows Workflow Foundation và Windows Identity Foundation,... và sự bổ sung của các tính năng mới, từ đó mang lại hiệu năng tốt, độ tin cậy và bảo mật cao hơn cho người dùng.

CHƯƠNG 2: ENTITY FRAMEWORK 5.0

2.1 Tổng quan về Entity Framework

2.1.1 Giới thiệu về Entity Framework

The Microsoft® ADO.NET Entity Framework là một bộ ánh xạ đối tượng – quan hệ (*Object/Relational Mapping – ORM*) mà cho phép người lập trình làm việc trên các dữ liệu quan hệ như các đối tượng cụ thể, loại bỏ sự cần thiết của hầu hết các mã plumbing mà người lập trình phải viết. Sử dụng Entity Framework, người lập trình truy vấn bằng **LINQ**, sau đó lấy và thao tác với dữ liệu như các đối tượng được khai báo lúc biên dịch chương trình. Các thành phần ORM của Entity Framework cung cấp các dịch vụ như *change tracking*, *identity resolution*, *lazy loading*, and *query translation* để các lập trình viên tập trung vào việc phát triển ứng dụng kinh doanh logic hơn là nguyên tắc truy cập dữ liệu cơ bản.



Hình 2.1 Tổng quan về Entity Framework

2.1.2 Các tính năng vượt trội của Entity Framework

- Làm việc được với nhiều máy chủ cơ sở dữ liệu (bao gồm *Microsoft SQL Server*, *Oracle* và *DB2*).

- Bao gồm công cụ ánh xạ mạnh mẽ cho phép có thể xử lý trên sơ đồ cơ sở dữ liệu thực tế và làm việc tốt với các thủ tục lưu trữ.
- Cung cấp các công cụ Visual Studio tích hợp trực quan để tạo các mô hình thực thể và tự động tạo mô hình từ một cơ sở dữ liệu có sẵn. Một cơ sở dữ liệu mới có thể được triển khai từ một mô hình, mà có thể toàn quyền hiệu chỉnh.
- Cung cấp Code First để tạo ra mô hình thực thể bằng mã. Code First có thể ánh xạ một cơ sở dữ liệu có sẵn hay tạo một cơ sở dữ liệu từ mô hình.
- Tương thích tốt với các mô hình lập trình ứng dụng .NET bao gồm: *ASP.NET*, *Windows Presentation Foundation* (WPF), *Windows Communication Foundation* (WCF) và *WCF Data Services* (trước kia là *ADO.NET Data Services*).

2.1.3 Lợi ích của Entity Framework

Entity Framework được xây dựng trên mô hình dịch vụ **ADO.NET**, với dịch vụ có sẵn được cập nhật thêm để hỗ trợ cho các tính năng mới của Entity Framework. Bởi vì điều này, các ứng dụng hiện tại được xây dựng trên ADO.NET có thể được chuyển sang Entity Framework dễ dàng với một mô hình lập trình quen thuộc với các lập trình viên ADO.NET.

Sử dụng Entity Framework để viết các ứng dụng theo hướng dữ liệu có các lợi ích:

- Giảm thời gian phát triển; framework cung cấp khả năng truy cập dữ liệu lỗi để người lập trình có thể tập trung vào ứng dụng logic.
- Người lập trình có thể làm việc với sự hỗ trợ của một mô hình đối tượng hướng trung tâm hơn, bao gồm các loại kế thừa (*inheritance*), thành viên phức hợp (*complex members*), và các mối quan hệ. Trong .NET Framework 4, Entity Framework cũng hỗ trợ **Persistence Ignorance** thông qua các thực thể **Plain Old CLR Objects** (POCO).

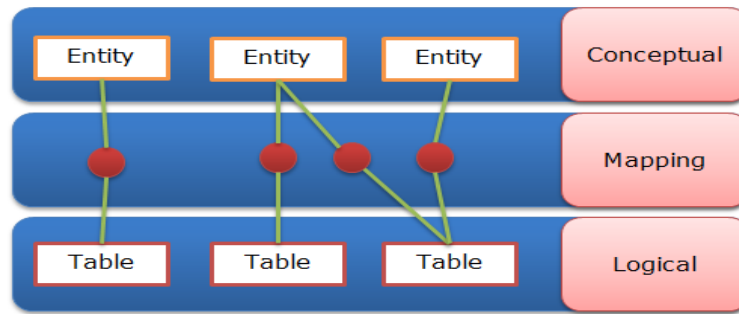
- Các ứng dụng được giải thoát khỏi hard-code phụ thuộc vào công cụ dữ liệu cụ thể hay lược đồ lưu trữ bằng cách hỗ trợ một mô hình ý niệm độc lập với mô hình lưu trữ/vật lí.
- Sự ánh xạ giữa mô hình đối tượng và lược đồ lưu trữ cụ thể có thể thay đổi mà không cần thay đổi code ứng dụng.
- Ngôn ngữ tích hợp truy vấn (**LINQ to Entity**) cung cấp IntelliSense (nhắc lệnh) và thời gian biên dịch xác nhận cú pháp để viết các truy vấn với một mô hình ý niệm.

2.1.4 Các thành phần của Entity Framework

Entity Framework sử dụng **Entity Data Model (EDM)** để mô tả đối tượng ứng dụng cụ thể hoặc mô hình ý niệm mà chương trình phát triển. EDM xây dựng trên mô hình liên kết thực (Entity Relationship model) được biết đến rộng rãi để tăng độ trừu tượng trên lược đồ cơ sở dữ liệu logic. EDM được phát triển với mục tiêu cơ bản trở thành mô hình dữ liệu chung cho bộ công nghệ phát triển và máy chủ của Microsoft. Do đó, EDM được tạo ra sử dụng với Entity Framework cũng có thể được thừa hưởng với *WCF Data Services* (formerly ADO.NET Data Services), *Windows Azure Table Storage*, *SharePoint 2010*, *SQL Server Reporting Services*, and *SQL Server PowerPivot for Excel*, và hơn nữa trong tương lai.

Các thành phần của Entity Framework trong EDM:

- ❖ Entity Framework dựa trên các tập tin **XML** để thực hiện các công việc của mình. Những tập tin đó thực hiện 3 nhiệm vụ: **xác định mô hình ý niệm, xác định mô hình lưu trữ và tạo ra ánh xạ (*mapping*) giữa mô hình với cơ sở dữ liệu vật lí.**



Hình 2.2 Entity Data Model – EDM

2.1.4.1 Conceptual schema definition language (CSDL)

CSDL là một ngôn ngữ dựa trên XML mô tả các thực thể, các mối quan hệ, và các hàm tạo nên một mô hình ý niệm của ứng dụng hướng dữ liệu (data-driven application). Mô hình ý niệm này có thể được sử dụng bởi Entity Framework hay WCF Data Services. Các siêu dữ liệu (metadata) được mô tả với CSDL được sử dụng bởi Entity Framework để ánh xạ các thực thể và mối quan hệ xác định trong mô hình ý niệm với một nguồn dữ liệu.

CSDL được lưu trữ trong tập tin có phần mở rộng là **.csdl** bên trong thư mục chứa project.

2.1.4.2 Store schema definition language (SSDL)

SSDL là một ngôn ngữ dựa trên XML mô tả các mô hình lưu trữ của một ứng dụng Entity Framework.

SSDL được lưu trữ trong tập tin có phần mở rộng là **.ssdl** bên trong thư mục chứa project.

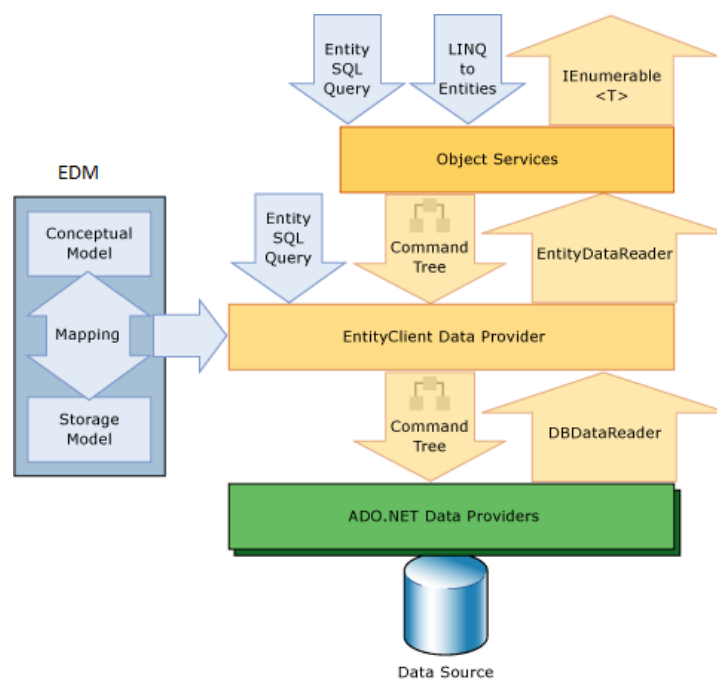
2.1.4.3 Mapping specification language (MSL)

MSL là một ngôn ngữ dựa trên XML mô tả các ánh xạ giữa mô hình ý niệm và mô hình lưu trữ của một ứng dụng Entity Framework.

MSL được lưu trữ trong tập tin có phần mở rộng là **.msl** bên trong thư mục chứa project.

2.1.5 Kiến trúc xử lý dữ liệu của Entity Framework

- **EDM (Entity Data Model):** EDM bao gồm ba phần chính: mô hình ý niệm (Concept model), ánh xạ (Mapping) và mô hình lưu trữ (Storage model).
 - **Concept model:** mô hình ý niệm là các lớp mô hình và các mối quan hệ của nó. Điều này sẽ được độc lập với thiết kế bảng cơ sở dữ liệu của bạn.
 - **Storage model:** mô hình lưu trữ là mô hình thiết kế cơ sở dữ liệu bao gồm bảng, view, thủ tục lưu trữ và các mối quan hệ và các khóa của nó.
 - **Mapping:** bản đồ ánh xạ bao gồm thông tin về sự ánh xạ từ mô hình ý niệm thành mô hình lưu trữ.



Hình 2.3 Kiến trúc Entity Framework

- ❖ **LINQ to Entities:** LINQ to Entities là ngôn ngữ truy vấn được sử dụng để viết các truy vấn đối với các mô hình đối tượng. Nó trả về thực thể được xác định trong mô hình ý niệm.

- ❖ **Entity SQL:** Entity SQL lại là một ngôn ngữ truy vấn giống như LINQ to Entities. Tuy nhiên nó khó khăn hơn một chút so với L2E và cũng phát triển cần phải tìm hiểu nó một cách riêng biệt.
- ❖ **Object Service:** là một entry point xử lý dữ liệu từ cơ sở dữ liệu và trả lại dữ liệu. Object Service chịu trách nhiệm cụ thể hóa quá trình chuyển đổi dữ liệu từ tầng Entity client data provider trả về thành cấu trúc đối tượng thực thể.
- ❖ **Entity Client Data Provider:** trách nhiệm chính của layer này là chuyển đổi các câu truy vấn L2E và Entity Framework thành truy vấn SQL được hiểu bởi cơ sở dữ liệu bên dưới. Nó giao tiếp với ADO.Net data provider để gửi hoặc nhận dữ liệu. EntityClient sử dụng Entity SQL để truy vấn.
- ❖ **ADO.Net Data Provider:** là layer giao tiếp với cơ sở dữ liệu theo tiêu chuẩn ADO.Net.
- ❖ **EntityDataReader** là lớp bên trong EntityClient, được sử dụng để đọc kết quả truy vấn.
- ❖ **DBDataReader** đọc kết quả trả về từ cơ sở dữ liệu.
- ❖ Các lệnh truy vấn từ layer trên sẽ được chuyển thành cấu trúc lệnh dạng cây (**Command Tree**) và chuyển xuống các layer dưới.

2.1.6 Các phiên bản của Entity Framework

Hai phiên bản đầu tiên của Entity Framework (EF 3.5 và EF 4.0) được kèm theo với .NET Framework. Và có số version trùng với số version của nền tảng mà nó được bao gồm. Sau đó EF bắt đầu chuyển thành độc lập và thông qua các tiêu chuẩn của <http://semver.org> về semantic versioning.

Các phiên bản của Entity Framework:

- ❖ **Entity Framework 3.5:** phiên bản đầu tiên của Entity Framework được bao gồm trong NET 3.5 SP1 và Visual Studio 2008 SP1. Phiên bản này hỗ trợ O/RM cơ bản để làm việc với luồng dữ liệu Database First.

- ❖ **Entity Framework 4.0:** Phiên bản này được đưa vào .NET Framework 4.0 và Visual Studio 2010. Các tính năng mới trong phiên bản này bao gồm hỗ trợ POCO, lazy loading, cải thiện khả năng kiểm tra, thể hệ mã tùy biến và tiến trình Model First.
- ❖ **Entity Framework 4.1:** là phiên bản đầu tiên phát hành NuGet. Phiên bản này bao gồm DbContext API và Code First được đơn giản hóa.
- ❖ **Entity Framework 4.1.1:** ngoài các bản vá lỗi, phiên bản này giới thiệu một số thành phần để làm nó dễ dàng hơn cho công cụ lúc thiết kế để làm việc với mô hình Code First.
- ❖ **Entity Framework 4.2:** phiên bản sửa lỗi cho EF 4.1.1.
- ❖ **Entity Framework 4.3:** phiên bản bao gồm tính năng mới Code First Migrations (là quá trình làm cho các ứng dụng hiện có có thể chạy trên các máy khác nhau hay các hệ điều hành khác nhau) cho phép một cơ sở dữ liệu được tạo ra bởi Code First được thay đổi dần khi mô hình Code First của bạn phát triển.
- ❖ **Entity Framework 4.3.1:** Bản vá lỗi bao gồm một số sửa lỗi của phiên bản EF 4.3 và hỗ trợ LocalBb tốt hơn cho người sử dụng EF 4.3 trên Visual Studio 2012.
- ❖ **Entity Framework 5.0:**
 - Phiên bản này có thể được sử dụng trong Visual Studio 2010 và Visual Studio 2012 để viết các ứng dụng .NET 4.0 và .NET 4.5. Với .NET 4.5, phiên bản này giới thiệu một số tính năng mới bao gồm hỗ trợ enum support, table-valued functions, spatial data types and various performance improvements.
 - Nếu bạn tạo ra một mô hình mới bằng cách sử dụng Entity Framework bằng Visual Studio 2012, các gói NuGet EF 5 sẽ được cài đặt cho project của bạn và mã được tạo ra sẽ được sử dụng EF5. Các dự án ASP.NET mới tạo trên Visual Studio 2012 (bao gồm cả dự án MVC) cũng có các gói NuGet EF5 cài theo mặc định.

- Entity Framework Designer trong Visual Studio 2012 cũng giới thiệu hỗ trợ multiple-diagrams cho mỗi mô hình, tô màu các hình dạng trên bề mặt thiết kế và batch import các thủ tục lưu trữ.
- ❖ **Entity Framework 6:** Phiên bản này có thể được sử dụng trong Visual Studio 2013, Visual Studio 2012 và Visual Studio 2010 (runtime only) để viết các ứng dụng nhắm mục tiêu .NET 4.0 và .NET 4.5.
- ❖ **Entity Framework 6.0.1:** Phiên bản sửa lỗi cho EF 6.
- ❖ **Entity Framework 6.0.3:** Phiên bản sửa lỗi cho EF 6.0.1 và 6.
- ❖ **Phiên bản mới nhất: 6.1.**

2.2 Giới thiệu về Entity Framework 5.0

2.2.1 Giới thiệu

Entity Framework 5.0 là một sản phẩm phát hành quan trọng của Microsoft. Nó có thể được sử dụng trong Visual Studio 2010 và Visual Studio 2012 để viết các ứng dụng trên .NET 4.0 và .NET 4.5.

2.2.2 Các tính năng mới

Entity Framework 5.0 bao gồm một số tính năng mới và sửa lỗi cho EF4.3. Hầu hết các tính năng mới chỉ có sẵn trong các ứng dụng .NET 4.5.

Các tính năng mới của EF 5.0:

- Hỗ trợ **Enum** cho phép bạn có các thuộc tính enum trong lớp thực thể của bạn.
 - Các loại dữ liệu không gian có thể được tiếp xúc trong mô hình của bạn bằng cách sử dụng DbGeography và các DbGeometry.
 - Hiệu suất truy vấn được cải tiến
- <http://blogs.msdn.com/b/adonet/archive/2012/02/14/sneak-preview-entity-framework-5-0-performance-improvements.aspx>

- Code-First sẽ phát hiện nếu bạn có sẵn LocalDb hoặc SQL Express để tạo cơ sở dữ liệu mới. Visual Studio 2012 bao gồm LocalDb, trong khi Visual Studio 2010 bao gồm SQL Express.
- Code-First sẽ thêm bảng vào cơ sở dữ liệu hiện tại nếu cơ sở dữ liệu đích không chứa bất kỳ các bảng nào từ mô hình.
- Các tính năng mới của EF Designer trong Visual Studio 2012:
 - **Default DbContext code generation template:** EF Designer sẽ tạo ra một DbContext và các lớp POCO theo mặc định trong Visual Studio 2012.
 - **Multiple-diagrams** cho mỗi mô hình cho phép bạn chia nhỏ mô hình của bạn thành nhiều mô hình nhỏ khi mô hình của bạn quá lớn. Điều này giúp bạn dễ dàng quan sát các mô hình hơn.
 - **Apply color** cho các thực thể trong EF designer.
 - **Table-Valued functions** trong cơ sở dữ liệu có sẵn có thể thêm vào mô hình của bạn. Vì vậy, có thể sử dụng LINQ truy vấn trên kết quả.
 - **Batch import of stored procedures** cho phép nhiều thủ tục lưu trữ được thêm vào mô hình trong quá trình tạo mô hình.

2.2.3 Cài đặt Entity Framework 5.0

2.2.3.1 Sử dụng dòng lệnh Package Manager Console

Visual Studio của bạn được cài đặt **NuGet Package Manager** (tải từ <http://www.nuget.org>). Trong Visual Studio 2012, NuGet Package Manager được cài đặt sẵn.

Mở giao diện **Package Manager Console** và gõ lệnh: **Install-Package EntityFramework –version 5.0.**

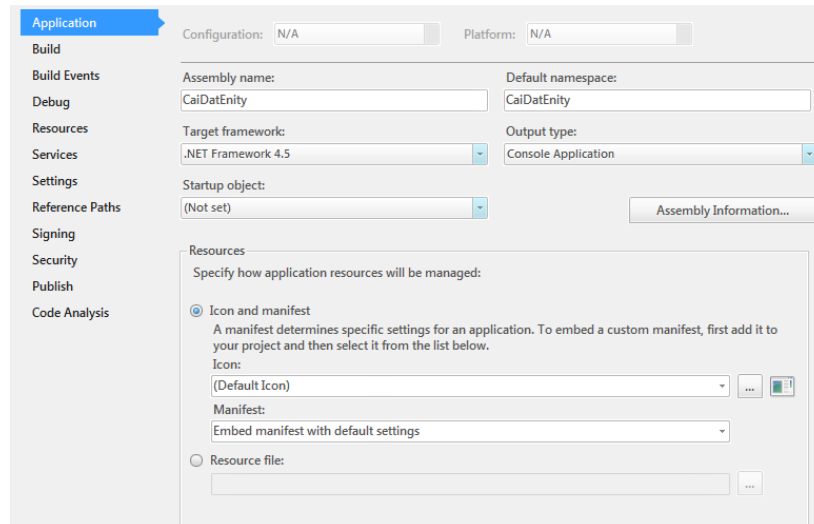
Entity Framework 5.0 sẽ được tải về và tự động cài đặt trên project của bạn.

Để kiểm tra, ta vào thư mục **References** của Project.

ENTITY FRAMEWORK 5.0

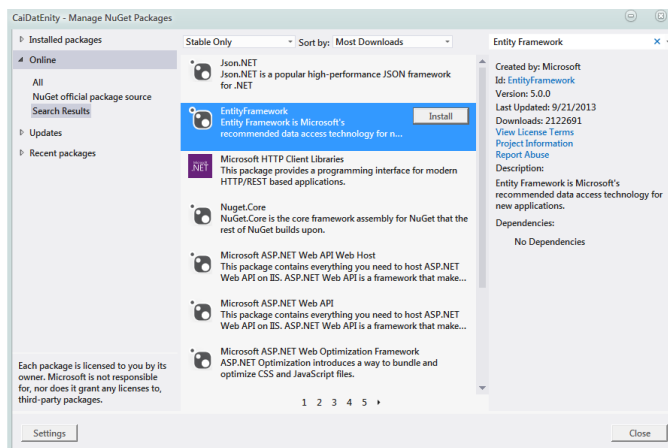
2.2.3.2 Sử dụng giao diện đồ họa Manage NuGet Package

Nhấp trái vào project của bạn chọn Properties để thiết lập nền .NET Framework cho project của bạn:



Hình 2.4 Thiết lập nền .Net Framework

Cài đặt Entity Framework 5.0 bằng các nhấp phải chuột vào project muốn cài đặt và chọn Manage NuGet Package. Chọn Online và tìm kiếm với từ khóa Entity Framework. Nhấn Install để cài đặt:



Hình 2.5 Giao diện đồ họa Manage NuGet Package

2.3 Các hướng tiếp cận Entity Framework

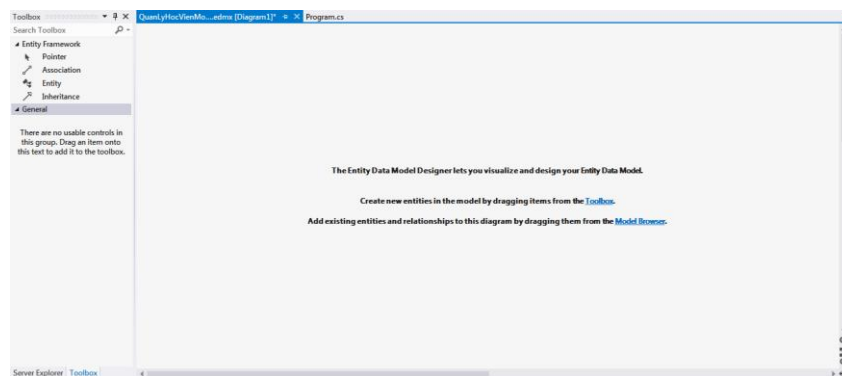
2.3.1 The Model – first Workfl

2.3.1.1 Định nghĩa

- Hướng tiếp cận này được bổ sung từ phiên bản EF 4 trong Visual 2010.
- Bạn có thể tạo một data model rỗng bằng công cụ **Entity Data Model Designer**, khi đó object layer cũng tự động được sinh ra. Sau khi đã hoàn thành việc thiết kế, bạn có thể sử dụng chức năng **Generate Database from Model** để tạo ra các mã DDL (*data definition language*) dựa trên mã SSDL (*Store Schema Definition Language*). Các mã DDL này sẽ được thực thi và lưu thành tập tin .sql.

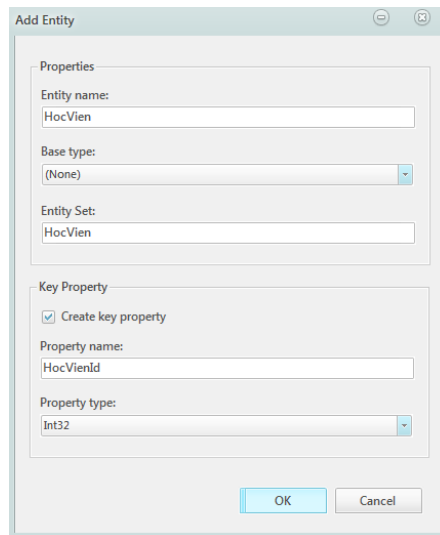
2.3.1.2 Sử dụng Model –first

- Giả sử tạo một Project có tên là **ModelFirst** tạo ra mô hình cơ sở dữ liệu là **QuanLyHocVien**. Quá trình thực hiện được diễn tả qua các bước sau:
 - B1.** Tạo **Project ModelFirst**.
 - B2.** Nhấn chuột phải vào ModelFirst chọn **Add | New Item**.
 - B3.** Chọn **ADO.NET Entity Data Model** và đặt tên cho mô hình muốn tạo ra trong trường Name.
 - B4.** Click **Add. Data Model Wizard** được khởi động.
 - B5.** Chọn **Empty Model** và chọn **Finish**. Vùng thiết kế **Entity Data Model Designer** hiện ra:



Hình 2.6 Vùng làm việc Entity Data Model Designer

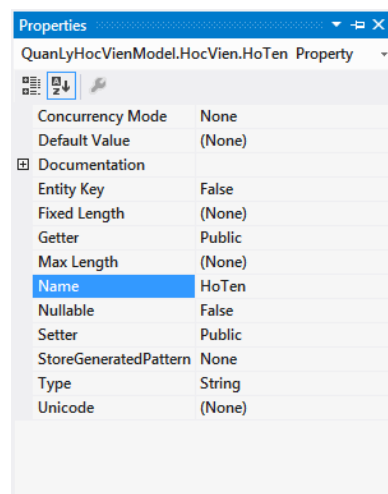
- B6.** Sử dụng **ToolBox** để thiết kế mô hình cơ sở dữ liệu bằng các kéo thả các đối tượng Entity trên ToolBox vào vùng thiết kế hay nhấp phải vào vùng thiết kế chọn **Add New | Entity** để tạo ra các đối tượng cần thiết.

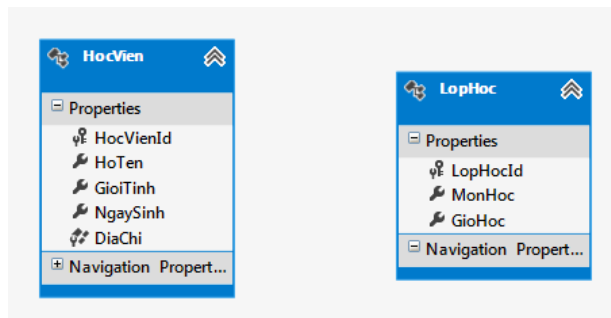


Hình 2.7 Thêm mới 1 entity

Để đặt tên tập hợp các thực thể, ta đánh tên vào trường **Entity Set**. Thiết lập khóa chính tại phần **Key Property**.

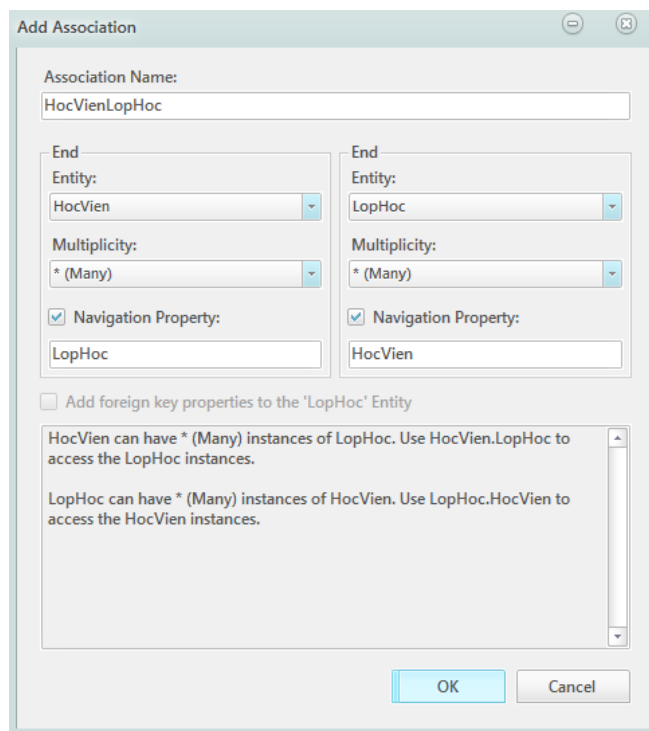
- B7.** Thêm thuộc tính cho các thực thể, ta nhấp phải chuột vào thực thể chọn Add New và chọn các loại thuộc tính muốn thêm. Để chỉnh sửa các thuộc tính ta nhấp chọn vào thuộc tính đó chọn **Properties**.





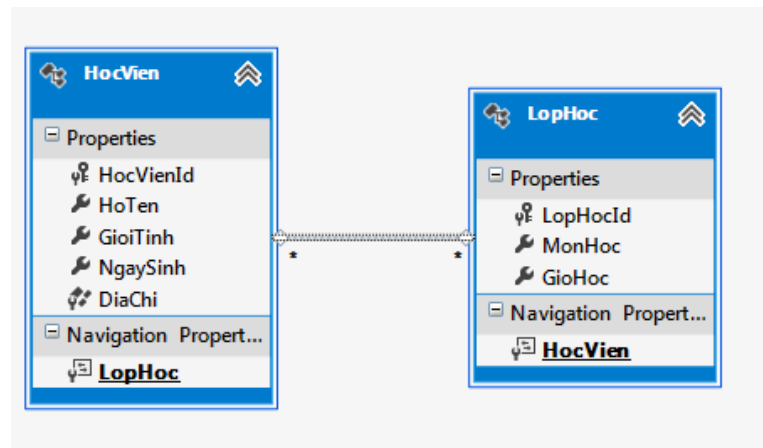
Hình 2.8 Biểu diễn Entity trên giao diện

- B8.** Để tạo mối quan hệ giữa hai thực thể, ta nhấp chuột phải vào thực thể và chọn **Add New | Association**. Hộp thoại **Add Association** hiện ra:



Hình 2.9 Giao diện thiết lập quan hệ giữa 2 Entity

- B9.** Thiết lập các loại liên kết cho các thực thể. Nó sẽ tự sinh ra các kết nối:

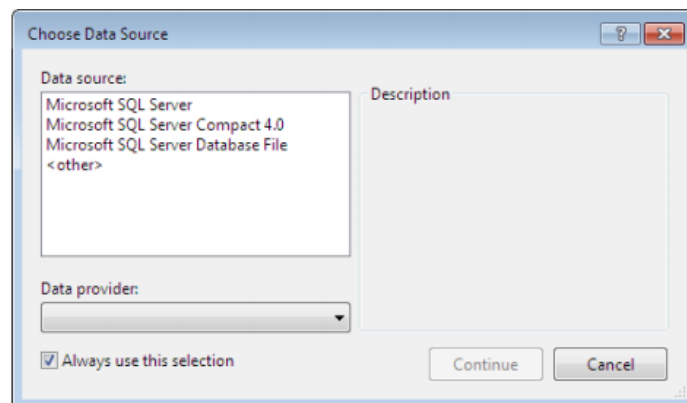


Hình 2.10 Biểu diễn quan hệ 2 Entity trên giao diện

B10. Tạo cơ sở dữ liệu từ mô hình vừa thiết kế. Chọn các thực thể muốn ánh xạ trong vùng thiết kế. Nhấp phải chuột và chọn **Generate Database from Model**.

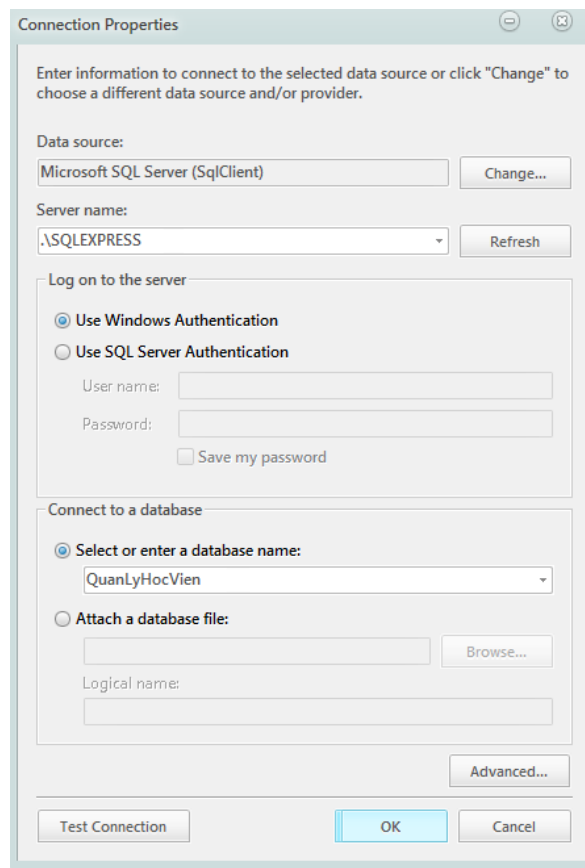
B11. Cửa sổ **Generate Database Wizard** xuất hiện.

B12. Chọn **New Connection**, hộp thoại **Choose Data Source** hiện ra:



Hình 2.11 Hộp thoại Choose Data Source

B13. Chọn **Microsoft SQL Server** và click **OK**, hộp thoại **Connection Properties** hiện ra:



Hình 2.12 Hộp thoại Connection Properties

- B14.** Chọn tên server bạn muốn sử dụng trong **Server name**.
- B15.** Đánh tên cơ sở dữ liệu trong **Select or enter a database name**.
- B16.** Click **OK**. Bạn sẽ thấy một hộp thoại nói với bạn là cơ sở dữ liệu của bạn chưa tồn tại. Visual Studio yêu cầu sự cấp quyền để tạo cơ sở dữ liệu cho bạn.
- B17.** Click **Yes**. Visual Studio tạo cho bạn một cơ sở dữ liệu mới. Đó là một cơ sở dữ liệu trống – nó không chứa bất kỳ một bảng, views, index, hay bất cứ một quan hệ nào đối với 1 cơ sở dữ liệu. Bạn sẽ trở về với hộp thoại **Generate Database Wizard**.
- B18.** Click **Next**. **Generate Database Wizard** tạo ra lệnh ngôn ngữ định nghĩa dữ liệu (Data Definition Language - DDL) để tạo mọi thứ trong mô

hình bạn thiết kế. Bạn có thể thấy được các câu lệnh SQL dùng để làm cho mô hình của bạn thành cơ sở dữ liệu thật và các bảng liên kết.

B19. Click **Finish**. Bạn sẽ thấy được các câu lệnh trong **QuanLyHocVien.EDMX.sql**. Nhưng nó chưa được thực thi. Tất cả những gì **Generate Database Wizard** làm là tạo ra lệnh yêu cầu tạo mô hình cơ sở dữ liệu chức năng của bạn.

B20. Chọn **SQL | Transact-SQL Editor | Execute**. Bạn sẽ thấy một hộp thoại kết nối mà bạn có thể nhập các thông tin cần thiết để kết nối với SQL Server mà bạn đã chọn.

B21. Nhập bất kỳ thông tin cần thiết và nhấn Connect. Visual Studio kết nối tới hệ quản trị cơ sở dữ liệu và thực thi lệnh SQL mà nó tạo ra. Lúc này, cơ sở dữ liệu của bạn thật sự được sử dụng.

❖ Làm việc với các đối tượng Entity vừa tạo:

- Tạo một container lưu trữ các thông tin về cơ sở dữ liệu:

```
QuanLyHocVienModelContainer db = new QuanLyHocVienModelContainer();
```

- Thêm bản ghi vào cơ sở dữ liệu:

```
HocVien h = new HocVien();  
  
h.HocVienId = 51;  
  
h.HoTen = "Tran Chi Tam";  
  
h.GioiTinh = false;  
  
h.NgaySinh = DateTime.Parse("07/04/1992");  
  
h.DiaChi.SoNha = "20";  
  
h.DiaChi.ThanhPho = "Ca Mau";
```

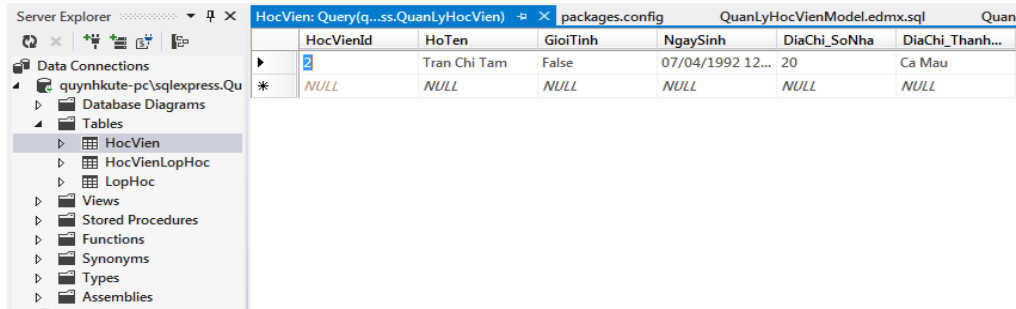
```
LopHoc l = new LopHoc();  
  
l.LopHocId = 02;  
  
l.MonHoc = "He dieu hanh";
```

ENTITY FRAMEWORK 5.0

```
l.GioHoc = "7";
```

- Lưu bản ghi vào cơ sở dữ liệu:

```
db.HocVien.Add(h);  
db.LopHoc.Add(l);  
db.SaveChanges();
```



The screenshot shows the Visual Studio interface. On the left, the 'Server Explorer' pane displays a tree view of a database named 'quynhkute-pc\sqlserver.Qu...'. The 'Tables' folder is expanded, showing 'HocVien', 'HocVienLopHoc', and 'LopHoc'. The 'HocVien' table is selected. On the right, a query window titled 'HocVien: Query(q...ss.QuanLyHocVien)' displays the results of a query. The table has columns: 'HocVienId', 'HoTen', 'GioiTinh', 'NgaySinh', 'DiaChi_SoNha', and 'DiaChi_Thanh...'. The first row shows data for a student named 'Tran Chi Tam' with birth date '07/04/1992 12...'. The second row shows 'NULL' values.

HocVienId	HoTen	GioiTinh	NgaySinh	DiaChi_SoNha	DiaChi_Thanh...
2	Tran Chi Tam	False	07/04/1992 12...	20	Ca Mau
*	NULL	NULL	NULL	NULL	NULL

2.3.2 The database– first workflow

2.3.1 Định nghĩa

- Đây là cách cập nhật hỗ trợ từ phiên bản EF đầu tiên trong Visual Studio 2008.
- Được sử dụng ứng dụng đã có sẵn cơ sở dữ liệu.
- EF sẽ tự động tạo ra mô hình dữ liệu và các lớp đối tượng từ database có sẵn thông qua công cụ **Entity Data Model Wizard**. Có thể thay đổi data model và cập nhật lại vào database.
- Đây là cách tiếp cận phổ biến vì thực hiện đơn giản, nhanh chóng nhưng đòi hỏi cần có cơ sở dữ liệu trước.

2.3.2 Sử dụng Database – frist

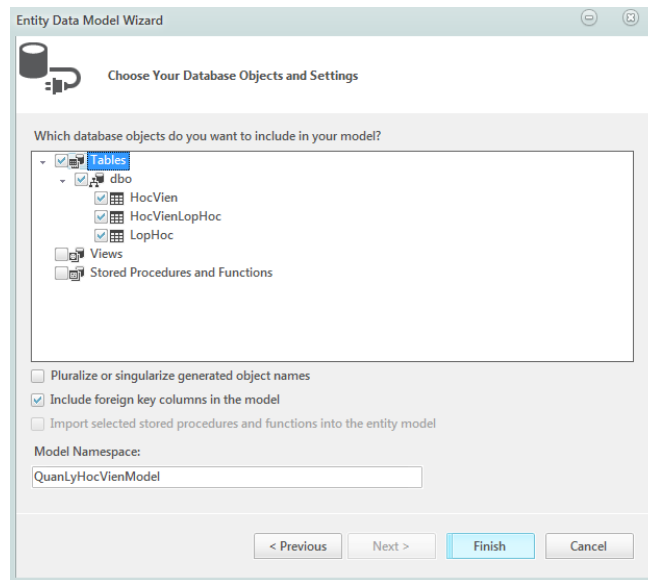
- ❖ Giả sử tạo một Project có tên là **DatabaseFirst** kết nối tới một cơ sở dữ liệu **QuanLyHocVien** mà ta đã tạo từ project **ModelFirst**. Quá trình thực hiện được diễn tả qua các bước sau:

B1. Tạo project và đặt tên project là **DatabaseFirst**.

B2. Nhấp phải vào **DatabaseFirst** trong cửa sổ **Solution Explorer** và chọn **Add | New Item**.

B3. Chọn **ADO.NET Entity Data Model**.

- B4.** Nhập **QuanLyHocVienModel.EDMX** vào trường Name và chọn **Add**. Visual Studio sẽ khởi tạo công cụ **Entity Data Model Wizard**.
- B5.** Chọn mục **Generate From Database** và chọn **Next**. Chọn cơ sở dữ liệu muốn kết nối.
- B6.** Chọn **Next**. Chọn các thành phần của cơ sở dữ liệu muốn sử dụng trong ứng dụng:



Hình 2.13 Chọn các thành phần cơ sở dữ liệu trong EDM Wizard

- B7.** Drill down vào đối tượng **Tables** bằng các click vào các mũi tên cạnh mỗi mục để hiện ra các bảng có trong cơ sở dữ liệu và lấy ra những bảng cần dùng.
- B8.** Chọn **Finish**. Sau đó, ta sẽ thấy mô hình được tạo ra từ cơ sở dữ liệu.
- B9.** Chọn **Buid | Solution** để lưu lại những thay đổi bạn tạo ra và sau đó biên dịch ứng dụng.
- ❖ Làm việc với các đối tượng Entity vừa tạo:
- Tạo đối tượng chứa tập hợp các đối tượng Entity được định nghĩa trong mô hình cơ sở dữ liệu:

```
QuanLyHocVienEntities db = new QuanLyHocVienEntities();
```

ENTITY FRAMEWORK 5.0

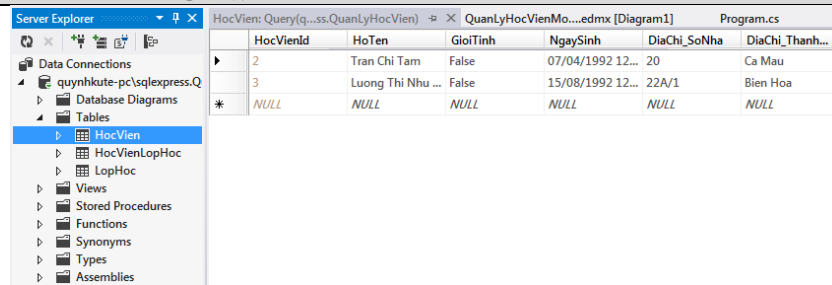
- Tạo các đối tượng mới:

```
HocVien h = new HocVien();
h.HocVienId = 51;
h.HoTen = "Luong Thi Nhu Quynh";
h.GioiTinh = false;
h.NgaySinh = DateTime.Parse("15/08/1992");
h.DiaChi_SoNha = "22A/1";
h.DiaChi_ThanhPho = "Bien Hoa";

LopHoc l = new LopHoc();
l.LopHocId = 01;
l.MonHoc = "Lap trinh";
l.GioHoc = "7";
```

- Thêm bản ghi và lưu lại:

```
db.HocVien.Add(h);
db.LopHoc.Add(l);
db.SaveChanges();
```



HocVienId	HoTen	GioiTinh	NgaySinh	DiaChi_SoNha	DiaChi_Thanh...
2	Tran Chi Tam	False	07/04/1992 12...	20	Ca Mau
3	Luong Thi Nhu ...	False	15/08/1992 12...	22A/1	Bien Hoa
NULL	NULL	NULL	NULL	NULL	NULL

2.3.3 The Code – first workflow

2.3.3.1 Định nghĩa

- Hướng tiếp cận code – first được giới thiệu từ phiên bản EF 4.1.
- Trong hướng tiếp cận này, ta hoàn toàn làm việc với code. Lập trình viên sẽ viết các ra các lớp mô tả các đối tượng thực thể tương ứng với các bảng trong cơ sở dữ liệu. EF sẽ tự động tạo ra cơ sở dữ liệu theo những gì các lớp định nghĩa.
- Để hỗ trợ có việc kiểm tra tính hợp lệ của dữ liệu trong các lớp tự định nghĩa, .NET Framework cung cấp các **DataAnnotation** và **Fluent API**.

2.3.3.2 Sử dụng code – first

- ❖ Giả sử ta tạo một ứng dụng có tên **CodeFirst** và chưa có sẵn cơ sở dữ liệu. Để tạo cơ sở dữ liệu từ code ta thực hiện các công việc sau:

- Tạo project mới

- B1.** Tạo project mới.
- B2.** Đặt tên cho project.
- B3.** Đặt tên **Solution** là **CodeFirstEx**.

- Định nghĩa các lớp khởi tạo

- B1.** Nhấp chuột phải vào mục **CodeFirstEx** trong **Solution Explorer** chọn **Add | New Project**.
- B2.** Chọn **Class Library**.
- B3.** Nhập tên của class (**ClassCodeFirst**) và chọn **OK**. Visual Studio tạo một lớp thư viện mới cho bạn.
- B4.** Xóa mã lớp Class1 cũ.
- B5.** Định nghĩa cho các lớp biểu diễn cho các thực thể trong cơ sở dữ liệu trong **class1.cs**:

```
// Định nghĩa học viên
public class HocVien
{
    public int HocVienId { get; set; }
    public string HoTen { get; set; }
    public bool GioiTinh { get; set; }
    public DateTime NgaySinh { get; set; }
    public DiaChi DiaChi { get; set; }

    public virtual ICollection<LopHoc> LopHocs { get; set; }
}
// Thuộc tính phức hợp
public class DiaChi
{
    public string SoNha { get; set; }
    public string ThanhPho { get; set; }
}
// Định nghĩa lớp học
public class LopHoc
{
```

```
public int LopHocId { get; set; }
public string Mon { get; set; }
public string GioHoc { get; set; }

public virtual ICollection<HocVien> HocViens { get; set; }
}
```

Để định nghĩa liên kết giữa hai thực thể bạn tạo một đường `ICollection` là một đối tượng ảo chứa các thực thể liên kết. Đối tượng ảo này không phải chứa dữ liệu copy của các thực thể liên kết với nó mà chỉ tạo ra đường link tới các thực thể đó.

- Thêm hỗ trợ Entity Framework 5.0 cho lớp thư viện. Sau đó thêm câu lệnh `using` vào đầu **class1.cs**

```
using System.Data.Entity;
```

- Tạo code – first context

Thêm hỗ trợ Entity Framework để **CodeFirstClasses** tự động cung cấp tham chiếu đến các lớp bạn cần. Trong trường hợp này, bạn cần truy cập vào **System.Data.Entity**, nhưng hỗ trợ này đã được thêm vào một cách tự động cho bạn (mở thư mục **References** để kiểm tra). Tuy nhiên, Visual Studio không hiểu cách bạn tạo ra database context như thế nào. Do đó ta phải định nghĩa 1 context cho CSDL.

Thêm vào **class1.cs** các đoạn Code sau:

```
public class QuanLyHocVienDBContext:DbContext
{
    public QuanLyHocVienDBContext():base("QuanLyHocVien2")
    {
    }

    public DbSet<HocVien> HocViens { get; set; }
    public DbSet<LopHoc> LopHocs { get; set; }
}
```

ENTITY FRAMEWORK 5.0

Đoạn mã trên sẽ tạo ra hai kết nối đến cơ sở dữ liệu. Thứ nhất là tạo ra một bảng có tên là *HocViens* lưu trữ các bản ghi về học viên và thứ hai là tạo ra bảng *LopHocs* chứa các bản ghi về lớp học.

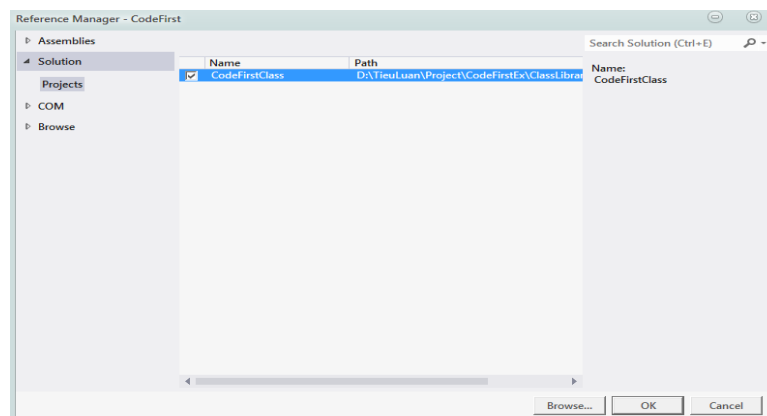
Lớp **QuanLyHocVienContext** kế thừa lớp **DbContext** trong **System.Data.Entity**. **DbContext** tạo ra sự kết nối đến cơ sở dữ liệu để thực hiện các nhiệm vụ **CRUD** (*Creat, Read, Update, Delete*). Tất cả những thay đổi sẽ được thu thập và phân nhóm trong nội dung và sau đó lưu vào cơ sở dữ liệu như một hoạt động hàng loạt (trong các ví dụ trên ta thực hiện thao tác trên dữ liệu nhưng nó vẫn chưa được lưu lại vào cơ sở dữ liệu, chỉ khi gọi đến phương thức **SaveChanges()** thì các thay đổi mới được lưu). Trong phương thức **base()** bạn đặt tên (ở ví dụ này là *QuanLyHocVien2*) cho cơ sở dữ liệu mình muốn tạo ra. Nếu bạn để trống thì tên cơ sở dữ liệu sẽ là *<Tên namespace>.<Tên lớp context>* (trong ví dụ này là *CodeFirstClass.QuanLyHocVienDbContext*).

DbSet<>: Đại diện cho tập hợp các thực thể. Ta có thể hiểu nó tương đương với 1 bảng trong cơ sở dữ liệu quan hệ.

- Tạo và thêm bản ghi vào cơ sở dữ liệu

B1. Thêm hỗ trợ Entity Framework vào ứng dụng của bạn

B2. Nhấp chuột phải vào thư mục **References** trong ứng dụng của bạn để thêm một tham chiếu đến lớp thư viện bạn vừa tạo



Hình 2.14 Giao diện References Manager

ENTITY FRAMEWORK 5.0

B3. Thêm câu lệnh:

```
using CodeFirstClass;
```

B4. Tạo ra các bản ghi mới:

```
HocVien h = new HocVien();  
h.HocVienId = 29;  
h.HoTen = "Nguyen Thi Ngoc Tram";  
h.GioiTinh = false;  
h.NgaySinh = DateTime.Parse("23/11/1992");  
h.DiaChi = new DiaChi {SoNha="30",ThanhPho="Binh Thuan" };  
  
LopHoc l = new LopHoc();  
l.LopHocId = 03;  
l.Mon = "Co so du lieu";  
l.GioHoc = "14";
```

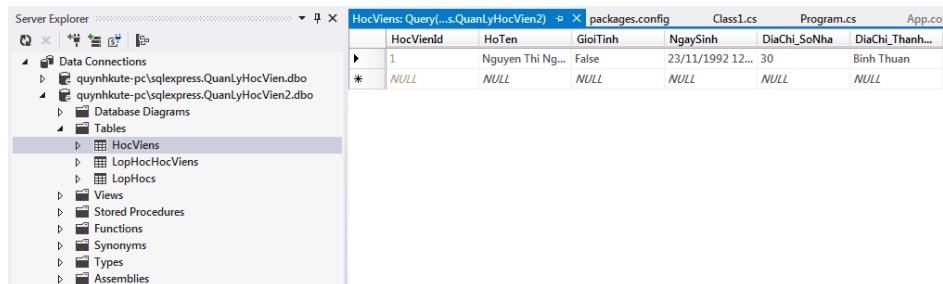
B5. Tạo đối tượng context

```
QuanLyHocVienDbContext db = new QuanLyHocVienDbContext();
```

B6. Thêm bản ghi và lưu lại vào cơ sở dữ liệu

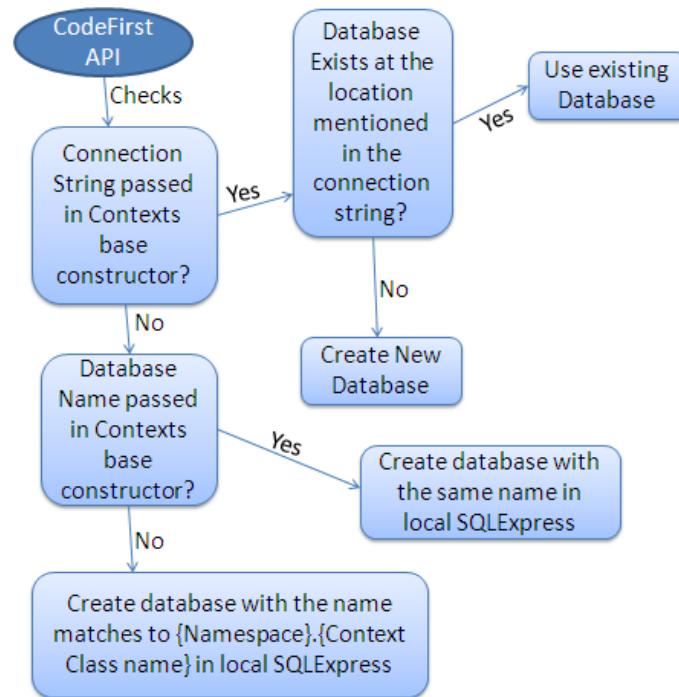
```
db.HocViens.Add(h);  
db.LopHocs.Add(l);  
db.SaveChanges();
```

B7. Chạy chương trình và trong cơ sở dữ liệu sẽ chứa 2 bản ghi vừa tạo.



Cơ sở dữ liệu của bản sẽ được tự động tạo và lưu trữ trong local SQLEXPRESS. Nó sẽ mặc định lấy thuộc tính có tên Id làm khóa chính. Để cấu hình các lớp thực thể, .NET Framework cung cấp cho các **bạn hỗ trợ về DataAnnotation** và **Fluent API** sẽ được giới thiệu vào phần sau.

2.3.3.3 Khởi tạo cơ sở dữ liệu - Database Initialization



Hình 2.15 Cách Code – first khởi tạo dữ liệu

❖ Theo hình trên, bạn có thể truyền các tham số sau đây trong base constructor:

- **Không có tham số:** Nếu không có tham số, nó sẽ tạo ra cơ sở dữ liệu trong local SQLEXPRESS của bạn với tên là <Namespace>.<Tên lớp context>. Trong ví dụ trên là CodeFirstClass.QuanLyHocVienDBContext:

```

public class QuanLyHocVienDBContext : DbContext
{
    public QuanLyHocVienDBContext()
        : base()
    {
    }
}
  
```

- **Tên:** nếu truyền tham số là một “Name” thì cơ sở dữ liệu bạn tạo ra sẽ có tên là tên tham số bạn truyền vào

```

public class QuanLyHocVienDBContext : DbContext
{
  
```

```
public QuanLyHocVienDbContext()
    : base("QuanLyHocVien2")
{
}
}
```

- **Tên chuỗi kết nối – ConnectionStringName:** nếu bạn truyền một chuỗi kết nối trong app.config hay web.config thì cơ sở dữ liệu sẽ được tạo ra theo chuỗi kết nối

```
public class QuanLyHocVienDbContext : DbContext
{
    public QuanLyHocVienDbContext()
        : base("QuanLyHocVienDBConnectionString")
    {
    }
}
```

App.config:

```
<connectionStrings>
  <add name="QuanLyHocVienDBConnectionString"
    providerName="System.Data.SqlClient"
    connectionString="Server=.\SQLEXPRESS;
    Database=QuanLyHocVien2;
    Trusted_Connection=True"/>
</connectionStrings>
```

Với chuỗi kết nối này, cơ sở dữ liệu sẽ được lưu vào server .\SQLEXPRESS trong SQL Server với tên là QuanLyHocVien2. Trong thẻ connectionString phải luôn có `providerName="System.Data.SqlClient"`

2.3.3.4 Các chiến lược khởi tạo cơ sở dữ liệu trong Code – first

Cơ sở dữ liệu sẽ được tạo mới trong lần chạy đầu tiên, nhưng đến lần thứ hai trở đi thì như thế nào? Nó sẽ tạo mới dữ liệu bất cứ khi nào ứng dụng được khởi động? Làm thế nào thay đổi cơ sở dữ liệu khi bạn thay đổi mô hình miền?... Để xử lý các tình huống, bạn phải sử dụng một trong những chiến lược khởi tạo cơ sở dữ liệu.

Có bốn cách cơ bản để xử lý vấn đề này:

- **CreateDatabaseIfNotExists:** nếu cơ sở dữ liệu chưa tồn tại, bộ khởi tạo này sẽ tạo ra cơ sở dữ liệu mới cho bạn. Nếu đã cơ sở dữ liệu đã được tạo trước đó, thì ta sẽ không tạo nữa mà sẽ thao tác trên cơ sở dữ liệu có sẵn. Tuy nhiên khi thay đổi mô hình dữ liệu, phương thức này sẽ ném ra một ngoại lệ.
- **DropCreateDatabaseIfModelChanges:** Khi có sự thay đổi về các entity class, bộ khởi tạo này sẽ xóa cơ sở dữ liệu cũ và tạo lại một cơ sở dữ liệu mới theo sự thay đổi các lớp thực thể đó.
- **DropCreateDatabaseAlways:** Cơ sở dữ liệu cũ sẽ được xóa và tạo cơ sở dữ liệu mới mỗi khi khởi động ứng dụng mà không quan tâm tới mô hình của bạn có bị thay đổi hay không.
- **Custom DB Initializer:** Cho phép bạn khởi tạo tùy theo điều chỉnh riêng của mình nếu các cách trên không đáp ứng yêu cầu của bạn hoặc bạn muốn làm một cái gì đó khác khi nó khởi tạo cơ sở dữ liệu.

Sử dụng các chiến lược khởi tạo:

```
public QuanLyHocVienDBContext (): base("QuanLyHocVienDBConnectionString")
{
    Database.SetInitializer<QuanLyHocVienDBContext>(new
CreateDatabaseIfNotExists<QuanLyHocVienDBContext>());

    //Database.SetInitializer<QuanLyHocVienDBContext>(new
DropCreateDatabaseIfModelChanges<QuanLyHocVienDBContext>());

    //Database.SetInitializer<QuanLyHocVienDBContext>(new
DropCreateDatabaseAlways<QuanLyHocVienDBContext>());

    //Database.SetInitializer<QuanLyHocVienDBContext>(new
QuanLyHocVienDBContextInitializer());
}
```

Các dữ liệu được khởi tạo đầu tiên khi cơ sở dữ liệu của bạn được tạo lại sẽ được định nghĩa trong lớp Seed. Nếu bạn muốn làm thêm những công việc khác khi tạo cơ sở dữ liệu bạn sẽ override lại lớp **Seed**.

```
public class QuanLyHocVienDBContextInitializer :
DropCreateDatabaseAlways<QuanLyHocVienDBContext>
{
}
```

```
protected override void Seed(QuanLyHocVienDBContext context)
{
    //
    //Đoạn code thêm các dữ liệu khi tạo cơ sở dữ liệu
    //
    base.Seed(context);
}
```

Trong ví dụ trên, các dữ liệu sẽ được khởi tạo sau mỗi lần khởi động ứng dụng do được kế thừa từ lớp `DropCreateDatabaseAlways`

Để sử dụng với Custom DB Initializer, lớp khởi tạo của bạn sẽ kế thừa từ `IDatabaseInitializer<Tên context>`. Ví dụ:

```
public class QuanLyHocVienDBContextInitializer :
IDatabaseInitializer<QuanLyHocVienDBContext>
{
    public void InitializeDatabase(QuanLyHocVienDBContext context)
    {
        if (context.Database.Exists())
        {
            if (!context.Database.CompatibleWithModel(true))
            {
                context.Database.Delete();
            }
        }
        context.Database.Create();
    }
}
```

❖ Sử dụng các lớp Seed viết lại:

```
Database.SetInitializer<QuanLyHocVienDBContext>(new
QuanLyHocVienDBContextInitializer());
```

2.3.3.5 Cấu hình các lớp miền trong Code – first

Để có thể tạo cơ sở dữ liệu từ các lớp miền do bạn định nghĩa thì các lớp đó phải tuân theo một số quy ước theo cơ sở dữ liệu để mà EF có thể hiệu được và tạo ra cơ sở dữ liệu cho bạn. Giả sử khi tạo ra một lớp Customer, ta không chỉ định khóa chính nhưng EF sẽ mặc định lấy thuộc tính có 2 ký tự cuối Id làm khóa (CustomerId) nhưng nếu không có thì sẽ báo lỗi không có khóa. Điều cần phải làm ở đây là cấu hình các lớp miền.

Có hai cách để cấu hình:

- Sử dụng các attribute của **DataAnnotation**
- Sử dụng các phương thức của **Fluent API**

❖ DataAnnotation

- ❖ EF Code – first cung cấp các thuộc tính (attributes) để áp dụng vào các lớp miền và các thuộc tính. Bạn phải khai báo không gian tên **System.ComponentModel.DataAnnotations** để sử dụng các attribute.
- ❖ DataAnnotation bao gồm các attribute cơ bản cho việc kiểm chứng phía máy chủ và các thuộc tính liên quan đến cơ sở dữ liệu.
- ❖ Một số thuộc tính kiểm chứng:

Annotation Attribute	Mô tả
Required	Đảm bảo các thuộc tính phải được điền dữ liệu
MinLength	Quy định độ dài tối thiểu của ký tự
MaxLength	Quy định độ dài tối đa của ký tự
StringLength	Xác định độ dài tối thiểu và tối đa của ký tự được cho phép trong một trường dữ liệu

- ❖ Một số thuộc tính liên quan đến lược đồ cơ sở dữ liệu:

Annotation Attribute	Description
Table	Chỉ rõ tên bảng mà các lớp ánh xạ
Column	Chỉ định tên cột và kiểu dữ liệu được ánh xạ bởi các thuộc tính
Key	Cho biết thuộc tính đó là khóa
ComplexType	Đánh dấu các lớp như loại phức hợp
ForeignKey	Đánh dấu khóa ngoại
NotMapped	Cho biết thuộc tính sẽ không được ánh xạ với cơ sở dữ liệu

- ❖ Ví dụ:

```
[Table("DSHocVien")]
public class HocVien
{
    [Key]
    [Column("MaHocVien")]
    public int HocVienId { get; set; }
```

```
[Required]
[StringLength(20)]
public string HoTen { get; set; }

[Required]
public bool GioiTinh { get; set; }

[Required]
public DateTime NgaySinh { get; set; }

public DiaChi DiaChi { get; set; }

[ForeignKey("LopHocId")]
public virtual ICollection<LopHoc> LopHocs { get; set; }
}
// Thuộc tính phức hợp
[ComplexType]
public class DiaChi
{
    public string SoNha { get; set; }
    public string ThanhPho { get; set; }
}
```

❖ Fluent API

- ❖ Phương pháp này cũng sử dụng để cấu hình các lớp miền của bạn bằng cách override lại phương thức **OnModelCreating** trong lớp context.

```
public class QuanLyHocVienDBContext : DbContext
{
    // Specify the name of the database as Rewards.
    public QuanLyHocVienDBContext()
        : base("QuanLyHocVienDBConnectionString")
    {
    }

    // Create a database set for each of the data items.
    public DbSet<HocVien> HocViens { get; set; }
```

```
public DbSet<LopHoc> LopHocs { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    //Configure domain classes using Fluent API here
    base.OnModelCreating(modelBuilder);
}
}
```

❖ Một số cách thức cấu hình thuộc tính ánh xạ sử dụng Fluent API:

- **HasKey():** Thiết lập thuộc tính khóa

```
//Primary Key
modelBuilder.Entity<HocVien>().HasKey(h => h.HocVienId);

//Composite Primary Key
modelBuilder.Entity<HocVien>().HasKey(h => new {h.HocVienId,h.HoTen});
```

- **HasDatabaseGeneratedOption():** tắt kiểu khóa tự động tăng

```
modelBuilder.Entity<HocVien>().Property(h =>
    h.HocVienId).HasDatabaseGeneratedOption(DatabaseGeneratedOption.None)
;
```

- **HasMaxLength():** Thiết lập số ký tự tối đa:

```
modelBuilder.Entity<HocVien>().Property(h=>h.HoTen).HasMaxLength(50);
```

- **IsRequired():** không được để rỗng

```
modelBuilder.Entity<HocVien>().Property(h=>h.HoTen).IsRequired();
```

- **Ignore():** Xác định những thuộc tính, bảng sẽ không được ánh xạ qua cơ sở dữ liệu

```
modelBuilder.Entity<HocVien>().Ignore(h => h.NgaySinh);
modelBuilder.Ignore<HocVien>();
```

- **HasColumnName():** Xác định cột mà thuộc tính đó ánh xạ

```
modelBuilder.Entity<HocVien>().Property(h => h.HocVienId)
```

```
.HasColumnName("MaHocVien");
```

- **IsUnicode():** Cho phép nội dung theo bảng mã Unicode

```
modelBuilder.Entity<HocVien>().Property(h=>h.HoTen).IsUnicode();
```

- **Cấu hình cho thuộc tính đơn của các loại phức hợp:**

```
modelBuilder.ComplexType<DiaChi>().Property(d=>d.ThanhPho).HasMaxLength(20);
```

hoặc:

```
modelBuilder.Entity<HocVien>().Property(h=>h.DiaChi.SoNha).HasMaxLength(20);
```

- **Xác định loại phức hợp:**

```
modelBuilder.ComplexType<DiaChi>();
```

- **ToTable():** Ánh xạ lớp thực thể thành bảng trong cơ sở dữ liệu:

```
modelBuilder.Entity<HocVien>().ToTable("DSHocVien");
```

2.3.3.6 Cấu hình các liên kết^[4]

❖ Liên kết 1 – 1

- Sử dụng DataAnnotation:

```
public class Student
{
    public Student() { }

    public int StudentId { get; set; }
    [Required]
    public string StudentName { get; set; }

    [Required]
    public virtual StudentAddress StudentAddress { get; set; }
}

public class StudentAddress
{
    [Key, ForeignKey("Student")]
    public int StudentId { get; set; }

    public string Address1 { get; set; }
    public string Address2 { get; set; }
    public string City { get; set; }
    public int Zipcode { get; set; }
    public string State { get; set; }
    public string Country { get; set; }
}
```

ENTITY FRAMEWORK 5.0

```
        public virtual Student Student { get; set; }  
    }
```

- Sử dụng Fluent API:

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)  
{  
    modelBuilder.Entity<StudentAddress>()  
        .HasKey(e => e.StudentId);  
    modelBuilder.Entity<StudentAddress>().Property(e => e.StudentId)  
        .HasDatabaseGeneratedOption(DatabaseGeneratedOption.None);  
    modelBuilder.Entity<StudentAddress>()  
        .HasRequired(e => e.Student)  
        .WithRequiredDependent(s => s.StudentAddress);  
    base.OnModelCreating(modelBuilder);  
}
```

❖ Liên kết 1 – nhiều

- Sử dụng DataAnnotation

```
public class Student  
{  
    public Student() { }  
    public int StudentId { get; set; }  
    [Required]  
    public string StudentName { get; set; }  
    public int StdandardId { get; set; }  
    public virtual Standard Standard { get; set; }  
}  
  
public class Standard  
{  
    public Standard()  
    {  
    }  
    public int StandardId { get; set; }  
    public string StandardName { get; set; }  
    public string Description { get; set; }  
    public virtual ICollection<Student> Students { get; set; }  
}
```

- Sử dụng Fluent API

```
public class Student  
{  
    public Student(){ }  
    public int StudentId { get; set; }  
    [Required]  
    public string StudentName { get; set; }  
    //StdId is not following code first conventions name  
    public int StdId { get; set; }  
}
```

```
        public virtual Standard Standard { get; set; }
    }

    public class Standard
    {
        public Standard()
        {
            StudentsList = new List<Student>();
        }
        public int StandardId { get; set; }
        public string StandardName { get; set; }
        public string Description { get; set; }

        public virtual ICollection<Student> StudentsList { get; set; }
    }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        //one-to-many
        modelBuilder.Entity<Student>().HasRequired<Standard>(s => s.Standard)
            .WithMany(s => s.StudentsList).HasForeignKey(s => s.StdId);
    }
}
```

❖ Liên kết nhiều – nhiều

- Sử dụng DataAnnotation

```
public class Student
{
    public Student() { }
    public int StudentId { get; set; }
    [Required]
    public string StudentName { get; set; }
    public int StdandardId { get; set; }

    public virtual ICollection<Course> Courses { get; set; }
}

public class Course
{
    public Course()
    {
    }
    public int CourseId { get; set; }
    public string CourseName { get; set; }

    public virtual ICollection<Student> Students { get; set; }
}
```

- Sử dụng Fluent API

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    modelBuilder.Entity<Student>()
        .HasMany<Course>(s => s.Courses).WithMany(c => c.Students).Map(c =>
    {
        c.MapLeftKey("Student_id");
        c.MapRightKey("Course_id");
    });
}
```

```

        c.ToTable("StudentAndCourse");
    });
    base.OnModelCreating(modelBuilder);
}

```

2.3.3.7 Migration

- ❖ **Migration** là một bộ khởi tạo cũng như *CreateDatabaseIfNotExists*, *DropCreateDatabaseIfModelChanges* và *DropCreateDatabaseAlways*.
- ❖ Được giới thiệu trong EF 4.3.
- ❖ Với các bộ khởi tạo khác, khi ta thay đổi mô hình dữ liệu, các bộ khởi tạo sẽ nhận ra được sự khác biệt giữa mô hình trước và mô hình vừa thay đổi. Để tạo ra cơ sở dữ liệu theo sự thay đổi đó, các bộ khởi tạo sẽ xóa cơ sở dữ liệu cũ (nếu bạn vẫn giữ tên cơ sở cũ cho sự thay đổi) và tạo mới lại. Điều này có nghĩa là bạn sẽ có một cơ sở dữ liệu hoàn toàn mới và rỗng hoặc có dữ liệu do lớp Seed bạn override lại. Nếu bạn vẫn để tên cơ sở dữ liệu cũ khi thay đổi thì những dữ liệu bạn đã thêm hay sửa ở cơ sở dữ liệu cũ sẽ hoàn toàn bị mất. Để khắc phục điều này EF cung cấp cho bạn bộ khởi tạo Migration, nó cho phép cập nhật lại thay đổi của cơ sở dữ liệu mà không cần tạo mới lại nó.

❖ Các bước sử dụng Migration để cập nhật dữ liệu

B1. Enable Migration, gõ lệnh vào Package Manager Console: **Enable-Migrations**

B2. Thay đổi các lớp thực thể

B3. Lưu lại các sự thay đổi thành lớp Migration: **Add-Migration <Tên lớp>**

B4. Cập nhật cơ sở dữ liệu bằng cách gõ lệnh: **Update-database**

❖ Ví dụ

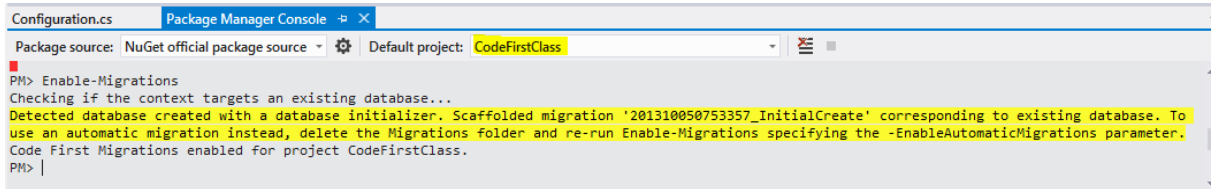
- ❖ Sử dụng lại ví dụ về Code-First. Trong bảng học viên ta đã có một bản ghi:

HocVien: Query(...s.QuanLyHocVien2) -> X		Class1.cs	Program.cs			
	HocVienId	HoTen	GioiTinh	NgaySinh	DiaChi_SoNha	DiaChi_Thanh...
▶	1	Nguyen Thi Ng...	False	23/11/1992 12...	30	Binh Thuan
*	NULL	NULL	NULL	NULL	NULL	NULL

- ❖ Ta sẽ sử dụng Migration để thêm một trường Điện thoại vào bảng:

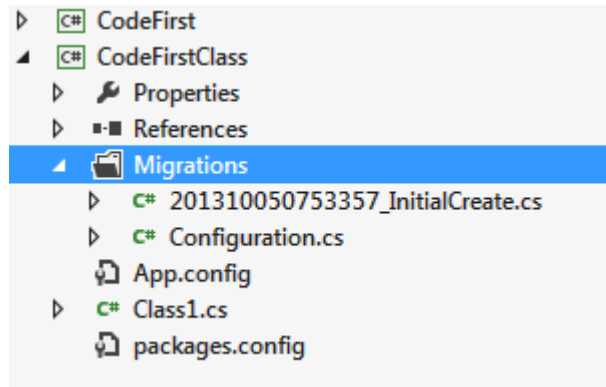
ENTITY FRAMEWORK 5.0

B1. Enable Migration. Ta thực hiện thay đổi trên lớp của thư viện chứa các lớp định nghĩa thực thể. Do đó **Default project** sẽ là thư viện đó.



Hình 2.16 Package Manager Console

Sau khi enable, nó sẽ tự động thêm vào project CodeFirstClass một thư mục **Migration** chứa các lớp Migration trong đó có một lớp **Configuration** có sẵn phương thức **Seed** và một lớp **InitialCreate** định nghĩa mô hình hiện có của ta.

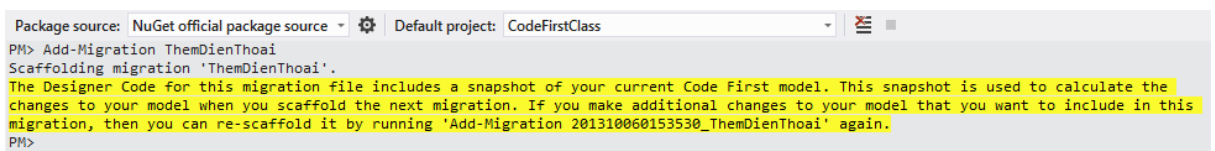


Hình 2.17 Thư mục Migratons

B2. Thêm thuộc tính Điện thoại vào lớp HocVien

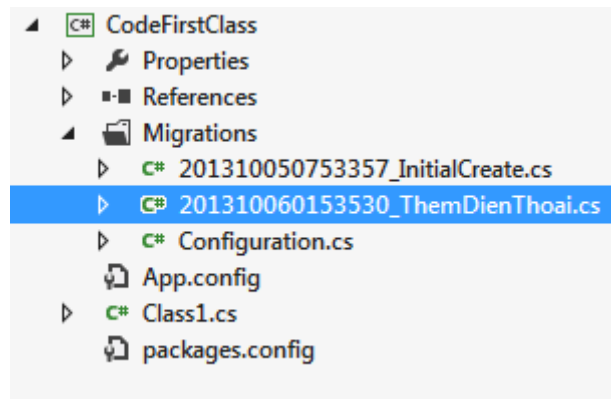
B3. Lưu sự thay đổi vào lớp Migration và đặt tên lớp là ThemDienThoai, gõ lệnh:

Add-Migration ThemDienThoai



Hình 2.18 Add Migration

B4. Lớp ThemDienThoai được tạo ra chứa các thông tin cập nhật mô hình:



Hình 2.19 File điều chỉnh trong thư mục Migrations

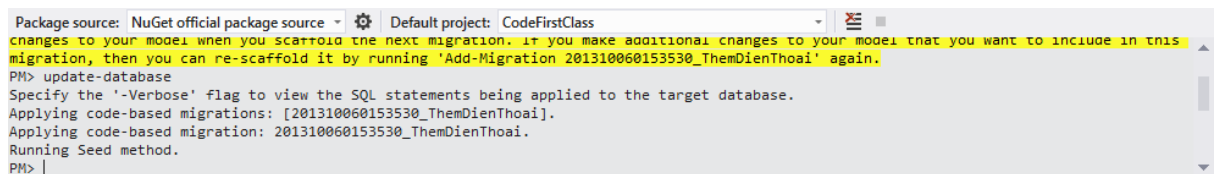
```
public partial class ThemDienThoai : DbMigration
{
    public override void Up()
    {
        AddColumn("dbo.HocViens", "DienThoai", c => c.String());
    }

    public override void Down()
    {
        DropColumn("dbo.HocViens", "DienThoai");
    }
}
```

Phương thức **Up()** chứa thông tin sẽ được cập nhật vào cơ sở dữ liệu. Phương thức **Down()** chứa xử lý các thông tin bị thay đổi.

B5. Cập nhật lại cơ sở dữ liệu: **Update-Database**

Lệnh update này sẽ thực hiện các phương thức Up()-Down() trong lớp Migration lưu lại sau cùng. Ta có thể thực hiện những lớp Migration đã lưu trước đó bằng cách chỉ định lớp Migration để thực hiện update: **Update-Database - TargetMigration:"Tên lớp"**



Hình 2.20 Update Migrations

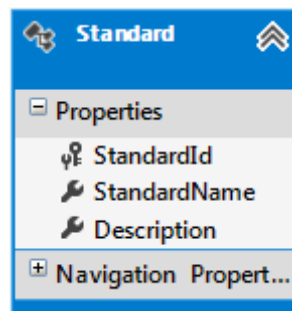
B6. Build Solution. Ta được kết quả:

HocVien: Query(...s.QuanLyHocVien2)		Class1.cs	Program.cs				
	HocVienId	HoTen	GioiTinh	NgaySinh	DiaChi_SoNha	DiaChi_Thanh...	DienThoai
▶	1	Nguyen Thi Ng...	False	23/11/1992 12...	30	Binh Thuan	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2.4 Thao tác với entity sử dụng DbContext^[4]

2.4.1 Thêm một thực thể

2.4.1.1 Thêm một thực thể đơn



Hình 2.21 Entity đơn

- Tạo đối tượng thực thể cần thêm vào cơ sở dữ liệu và gán các giá trị cho các thuộc tính của thực thể:

```
// Tạo đối tượng thực thể Standard

var newStandard = new Standard();

newStandard.StandardName = "Standard 1";
```

- Tạo DbContext và thêm vào tập thực thể chứa thực thể. Sử dụng phương thức SaveChanges để lưu lại các thao tác vừa thực hiện:

```
// Tạo DbContext

using (var dbContext = new SchoolDBEntities())
{
    // Thêm standards vào DBSet dùng phương thức Add

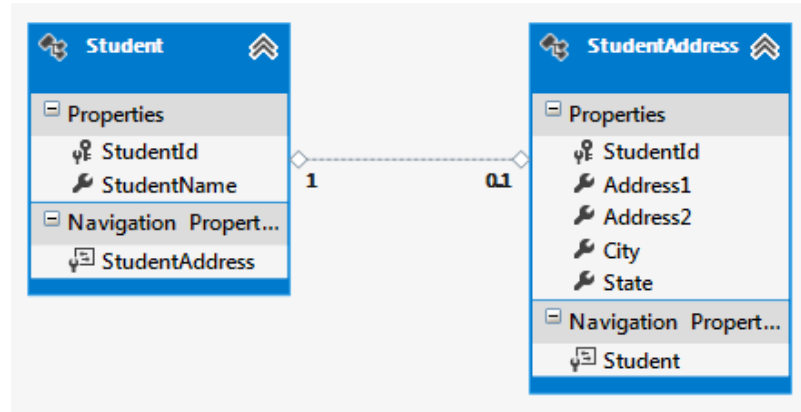
    dbContext.Standards.Add(newStandard);

    // gọi phương thức SaveChange để lưu vào cơ sở dữ liệu

    dbContext.SaveChanges();
}
```

```
}
```

2.4.1.2 Thêm thực thể trong mối quan hệ 1 – 1



Hình 2.22 Liên kết 1-1

- Tạo đối tượng thực thể Student:

```
// Tạo đối tượng thực thể Student
var student = new Student();
student.StudentName = "New Student1";
```

- Tạo thực thể StudentAddress và gán vào thực thể Student:

```
/* Tạo đối tượng thực thể StudentAddress và gán vào StudentAddress
trong student */
student.StudentAddress = new StudentAddress() {
    Address1 = "Student1's Address1",
    Address2 = "Student1's Address2",
    City = "Student1's City",
    State = "Student1's State" };

```

- Tạo DbContext và thêm thực thể Student vào tập các thực thể Students. Sử dụng phương thức Savechanges để lưu lại các thao tác.

```
//Tạo đối tượng DbContext
using (var dbContext = new SchoolDBEntities())
{

```

```
// Thêm student vào DBSet dùng phương thức Add
dbCtx.Students.Add(student);

/* Lưu lại sự thay đổi */
dbCtx.SaveChanges();

}
```

- Ta không cần Add thực thể StudentAddress vì khi tạo thực thể StudentAddress ta đã gán nó cho thuộc tính StudentAddress của thực thể Student.

2.4.1.3 Thêm thực thể trong mối quan hệ 1 – nhiều



Hình 2.23 Liên kết 1 – nhiều

- Tạo mới các thực thể **Standard** và **Teacher**:

```
// Tạo mới một thực thể Standard
var standard = new Standard();
standard.StandardName = "Standard1";

// Tạo mới 3 thực thể Teacher
var teacher1 = new Teacher();
teacher1.TeacherName = "New Teacher1";

var teacher2 = new Teacher();
teacher2.TeacherName = "New Teacher2";

var teacher3 = new Teacher();
teacher3.TeacherName = "New Teacher3";
```

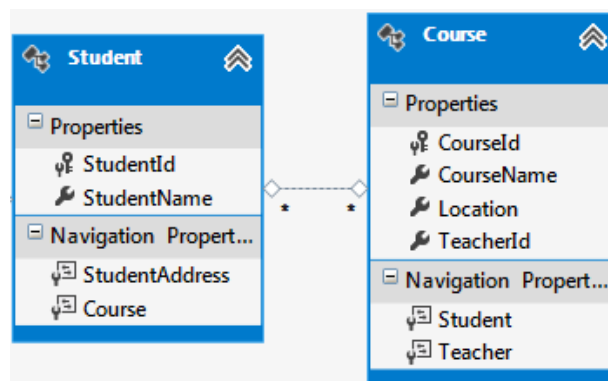
- Thêm các thực thể **Teacher** vào tập các thực thể **Teacher** trong **Standard**:

```
// Thêm các Teacher vào Standard  
  
standard.Teachers.Add(teacher1);  
  
standard.Teachers.Add(teacher2);  
  
standard.Teachers.Add(teacher3);
```

- Tạo DbContext và thêm các thực thể vào:

```
using (var dbCtx = new SchoolDBEntities())  
{  
  
    // Thêm Standart vào DBSet  
  
    dbCtx.Standards.Add(standard);  
  
    // Lưu lại sự thay đổi  
  
    dbCtx.SaveChanges();  
  
}
```

2.4.1.4 Thêm thực thể trong mối quan hệ nhiều – nhiều



Hình 2.24 Liên kết nhiều – nhiều

- Tạo các thực thể Student và Course:

```
// Tạo mới một thực thể Student  
  
var student1 = new Student();  
  
student1.StudentName = "New Student2";  
  
  
// Tạo ra các thực thể Course  
  
var course1 = new Course();
```

```
course1.CourseName = "New Course1";  
course1.Location = "City1";  
var course2 = new Course();  
course2.CourseName = "New Course2";  
course2.Location = "City2";  
var course3 = new Course();  
course3.CourseName = "New Course3";  
course3.Location = "City1";
```

- Thêm các Course vào Student:

```
// Thêm các Course vào thực thể Student  
student1.Courses.Add(course1);  
student1.Courses.Add(course2);  
student1.Courses.Add(course3);
```

- Tạo DbContext và lưu lại:

```
using (var dbContext = new SchoolDBEntities())  
{  
    // Dùng phương thức Add để thêm thực thể DbSet  
    dbContext.Students.Add(student1);  
    // Lưu lại thay đổi dùng SaveChanges  
    dbContext.SaveChanges();  
}
```

2.4.2 Cập nhật thực thể

2.4.2.1 Cập nhật thực thể đơn

- Lấy ra thực thể đã tồn tại.
- Thay đổi các thuộc tính.
- Đánh dấu trạng thái của thực thể là Modified (để chỉ thực thể đó bị thay đổi).
- Gọi phương thức SaveChanges để cập nhật vào cơ sở dữ liệu.

```
Student stud ;

// Lấy Student từ cơ sở dữ liệu
using (var ctx = new SchoolDBEntities())
{
    stud = ctx.Students.Where(s => s.StudentName == "New
Student1").FirstOrDefault<Student>();
}

// Thay đổi tên Student
if (stud != null)
{
    stud.StudentName = "Updated Student1";
}

// Lưu lại sự thay đổi
using (var dbContext = new SchoolDBEntities())
{
    // Đánh dấu thực thể bị thay đổi bằng State
    dbContext.Entry(stud).State =
System.Data.EntityState.Modified;

    dbContext.SaveChanges();
}
```

- Ngoài cách sử dụng trạng thái của thực thể, ta có thể không cần gán các trạng thái cho các thực thể mà gọi luôn phương thức **SaveChanges** để lưu lại những thay đổi.

```
Student stud ;

// Lấy Student từ cơ sở dữ liệu
using (var ctx = new SchoolDBEntities())
{
    stud = ctx.Students.Where(s => s.StudentName == "New
Student1").FirstOrDefault<Student>();
}
```

```
        if (stud != null)
        {
            // Thay đổi tên Student
            stud.StudentName = "Updated Student1";

            // Lưu lại sự thay đổi
            ctx.SaveChanges();
        }
    }
```

2.4.2.2 Cập nhật thực thể trong mối quan hệ 1 – 1

Các cập nhật với mối quan hệ một – một:

- Thêm một thực thể mới vào liên kết của thực thể đã tồn tại nếu thực thể đã tồn tại chưa liên kết với thực thể khác. Ví dụ: thêm một StudentAddress vào Student nếu studentaddress chưa có.
- Thay đổi giá trị các thực thể đã tồn tại. Ví dụ: thay đổi giá trị các thuộc tính của StudentAddress của một Student.
- Xóa thực thể đã được liên kết. ví dụ: Xóa StudentAddress của một Student.

Ví dụ:

```
using (var dbCtx = new SchoolDBEntities())
{
    //Lấy StudentAddress đã tồn tại
    StudentAddress existingStudentAddress =
        dbCtx.StudentAddresses.AsNoTracking()
        .Where(addr => addr.StudentID == stud.StudentID)
        .FirstOrDefault<StudentAddress>();

    //Đánh dấu Student sẽ thay đổi
    dbCtx.Entry(stud).State = System.Data.EntityState.Modified;
```



```
        /*Nếu StudentAddress đã tồn tại (lấy ở trên) khác rỗng thì đánh
dấu xóa */

        if (existingStudentAddress != null)

            dbContext.Entry<StudentAddress>(existingStudentAddress).State

                = System.Data.EntityState.Deleted;

        /* Nếu Student được đánh dấu thay đổi chưa có StudentAddress khác
rỗng thì thêm StudentAddress của Student đó vào cơ sở dữ liệu*/

        if (stud.StudentAddress != null)

            dbContext.StudentAddresses.Add(stud.StudentAddress);

        dbContext.SaveChanges();

    }
```

2.4.2.3 Cập nhật thực thể trong mối quan hệ một – nhiều

Các cập nhật với mối quan hệ một – nhiều:

- Thêm một liên kết mới. Ví dụ: Thêm một Teacher vào tập các thực thể teacher của một Standard.
- Thay đổi giá trị các thuộc tính của thực thể phía nhiều. Ví dụ: Chỉnh sửa Teacher đã tồn tại trong tập các teacher của một Standard.
- Xóa thực một thực thể nhiều ra khỏi tập thực thể. Ví dụ: Xóa một Teacher ra khỏi tập hợp teacher của Standard.

Ví dụ:

```
using (var dbContext = new SchoolDBEntities())
{
    //1- Lấy thực thể Student từ cơ sở dữ liệu
    var existingStudent = dbContext.Students.AsNoTracking().Include(s
=> s.Standard).Include(s => s.Standard.Teachers).Where(s => s.StudentName
== "updated student").FirstOrDefault<Student>();

    var existingTeachers =
existingStudent.Standard.Teachers.ToList<Teacher>();

    var updatedTeachers = teachers.ToList<Teacher>();
```

```
//2- Lấy danh sách teacher được thêm vào
var addedTeachers = updatedTeachers.Except(existingTeachers,
tchr => tchr.TeacherId);

//3- Lấy danh sách teacher sẽ bị xóa
var deletedTeachers =
existingTeachers.Except(updatedTeachers, tchr => tchr.TeacherId);

//4- Lấy danh sách teacher sẽ bị thay đổi
var modifiedTeacher = updatedTeachers.Except(addedTeachers,
tchr => tchr.TeacherId);

//5- Đánh dấu các teacher thêm vào
addedTeachers.ToList<Teacher>().ForEach(tchr =>
dbCtx.Entry(tchr).State = System.Data.EntityState.Added);

//6- Đánh dấu các teacher sẽ bị xóa
deletedTeachers.ToList<Teacher>().ForEach(tchr =>
dbCtx.Entry(tchr).State = System.Data.EntityState.Deleted);

//7- Thay đổi các giá trị của teacher sẽ sửa đổi thành //giá
trị của teacher đã có
foreach(Teacher teacher in modifiedTeacher)
{
    //8- Tìm các teacher đã có theo Id từ cơ sở dữ liệu
    var existingTeacher =
dbCtx.Teachers.Find(teacher.TeacherId);

    if (existingTeacher != null)
    {
        //9- Lấy các giá trị của teacher đã tồn tại
        var teacherEntry = dbCtx.Entry(existingTeacher);
        //10- Ghi đè những thuộc tính đã có của
        //existingTeacher lên teacher
        teacherEntry.CurrentValues.SetValues(teacher);
    }
}
//11- Lưu lại những thay đổi
dbCtx.SaveChanges();
}
```

2.4.2.4 Cập nhật thực thể trong mối quan hệ nhiều – nhiều

Các cập nhật với mối quan hệ nhiều – nhiều:

- Thêm các liên kết mới. Ví dụ: Thêm các Course cho Student, dù đó là Course mới hay đã tồn tại trong cơ sở dữ liệu.
- Có thể loại bỏ một số thực thể trong tập liên kết. Người dùng có thể loại bỏ một số các Course từ bộ sưu tập Course hiện có của Student.

Ví dụ:

```
using (var dbCtx = new SchoolDBEntities())
{
    /* 1- Lấy dữ liệu từ cơ sở dữ liệu */
    var existingStudent = dbCtx.Students.Include("Courses")
        .Where(s => s.StudentName == stud.StudentName)
        .FirstOrDefault<Student>();

    /* 2- Tìm danh sách khóa học sẽ bị xóa bằng cách lấy danh
sách khóa học hiện tại - đi danh sách khóa học đã tồn tại*/
    var deletedCourses =
        existingStudent.Courses.Except(stud.Courses,
            cours => cours.CourseId).ToList<Course>();

    /* 3- Tìm danh sách khóa học được thêm vào bằng cách lấy
danh sách khóa học hiện tại - danh sách khóa học có sẵn */
    var addedCourses =
        stud.Courses.Except(existingStudent.Courses,
            cours => cours.CourseId).ToList<Course>();

    /* 4- Xóa các khóa học ra khỏi bộ tập khóa học của sinh
viên
deletedCourses.ForEach(c =>
    existingStudent.Courses.Remove(c));

    /*5- Thêm khóa học mới
foreach(Course c in addedCourses)
{
    /*6- Thêm những khóa học mới vào tập các khóa học nếu
như chưa có*/
    if (dbCtx.Entry(c).State ==
        System.Data.EntityState.Detached)
        dbCtx.Courses.Attach(c);

    /*7- Thêm vào tập khóa học của sinh viên
existingStudent.Courses.Add(c);
}

    /*8- Lưu thay đổi
dbCtx.SaveChanges();
}
```

2.4.3 Xóa thực thể

- Lấy ra thực thể cần xóa.
- Gọi phương thức Remove.
- Gọi phương thức SaveChanges để lưu lại.

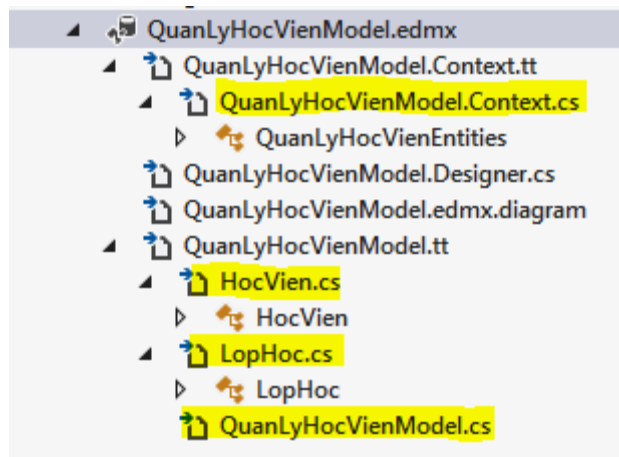
```
using (var dbCtx = new SchoolDBEntities())
{
    //Lấy thực thể cần xóa và dùng phương thức Remove để xóa
    var newtchr = dbCtx.Teachers
        .Where(t => t.TeacherName == "New teacher4")
```

```
.FirstOrDefault<Teacher>();  
dbCtx.Teachers.Remove(newtchr);  
  
//Ngoài ra còn có thể sử dụng trạng thái State = Deleted:  
//dbCtx.Entry(tchr).State = System.Data.EntityState.Deleted;  
  
//Lưu lại sự thay đổi  
dbCtx.SaveChanges();  
}
```

2.5 Các tính năng mới của Entity Framework 5.0

2.5.1 Tạo lớp DbContext trong Entity Framework 5.0

- Khi sử dụng EF 5.0 và Visual 2012 để tạo mô hình từ một cơ sở dữ liệu có sẵn (theo hướng tiếp cận Database-first). Nó sẽ được tự động tạo ra lớp DbContext và các thực thể POCO mà không cần **Add Code Generator Item...**
- Trong ví dụ về hướng Database-first đã nêu trước, khi tạo mô hình từ cơ sở QuanLyHocVien có sẵn, nó đã tự động sinh ra các lớp DbContext và các lớp định nghĩa các thực thể:



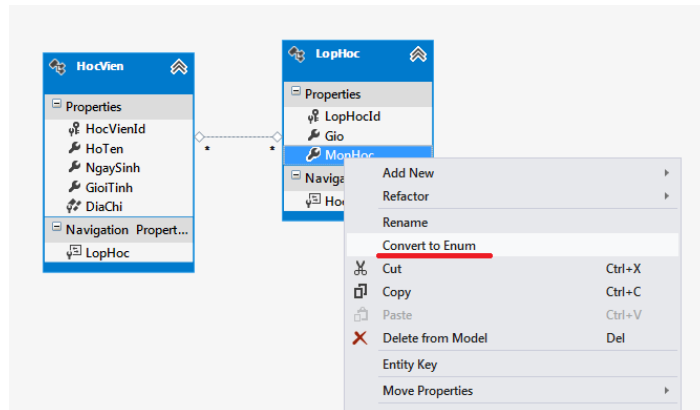
2.5.2 Kiểu Enum trong EF 5.0

- Bạn có thể có kiểu Enum trong project Entity Framework 5.0 trên .NET Framework 4.5.
- Enum chỉ được tạo cho các kiểu dữ liệu: Int16, Int32, Int64, Byte, Sbyte
- Có 3 cách tạo và sử dụng kiểu Enum trong mô hình thực thể dữ liệu bằng 3 cách:
 - Chuyển đổi một thuộc tính có sẵn của thực thể thành Enum từ EDM designer.

- Thêm Enum mới từ EDM designer.
- Sử dụng kiểu Enum từ một không gian tên khác.

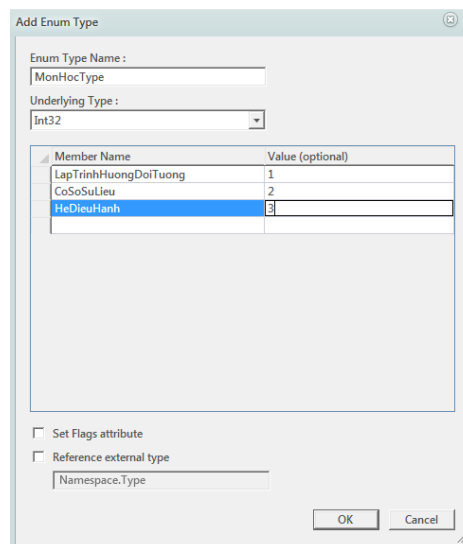
2.5.2.1 Chuyển đổi một thuộc tính thành kiểu Enum

- Nhấp trái vào thuộc tính có thể có kiểu **Enum**, chọn **Convert to Enum**.



Hình 2.25 Convert to Enum

- Hộp thoại **Add Enum Type** xuất hiện, bạn đánh tên **Enum** vào trường **Enum Type Name** và chọn **Underlying Type**.
- Thêm các thành viên trong Enum và giá trị của các thành viên.



Hình 2.26 Add Enum type

- Sau khi chuyển thành Enum, bạn có thể thấy **MonHocType** trong **Enum Type** trong **Model Brower**.
- Trong **Properties** của thuộc tính môn học ta thấy kiểu của nó là **MonHocType**.

2.5.2.2 Thêm Enum mới từ EDM Design

- Click phải vào vùng **EDM Design** chọn **Add New | Enum Type**.
- Nhập các giá trị cho **Enum** như phần trước trong hộp thoại **Add Enum Type**.
- Sau đó **Enum** này sẽ xuất hiện trong thư mục **Enum Types** trong **Model Browser**.
- Bạn có thể gán kiểu **Enum** vừa mới tạo cho các thuộc tính trong Model của bạn bằng các điều chỉnh mục **Type** trong **Properties** của thuộc tính đó là tên của kiểu **Enum** bạn muốn gán.

2.5.2.3 Sử dụng Enum từ một không gian tên khác

- Nếu bạn đã định nghĩa 1 kiểu **Enum** trong code của bạn, bạn có thể thêm kiểu **Enum** đó vào mô hình của bạn.
- Click phải vào vùng **EDM Design** chọn **Add New | Enum Type**.
- Nhập tên kiểu **Enum**.
- Tham chiếu đến kiểu **Enum** bạn tạo ra bằng code bằng cách đánh dấu vào mục **Reference external Type** và điền : *<Tên namespace chứa enum mà bạn định nghĩa>.<Tên kiểu Enum bạn định nghĩa>*
- Sau đó bạn chỉ việc gán kiểu **Enum** này vào thuộc tính bạn muốn.

➤ Sử dụng kiểu Enum

```
LopHoc l = new LopHoc();  
l.LopHocId = 02;  
l.MonHoc = MonHocType.CoSoDuLieu;  
l.Gio = "7";
```

2.5.3 Spatial Data type^[4]

- **MS SQL Server 2008** cung cấp kiểu dữ liệu không gian địa lý đại diện cho dữ liệu trong một hệ tọa độ và hình học vòng quanh trái đất mà dữ liệu đại diện nằm trong một hệ tọa độ **Euclidean** (phẳng).

ENTITY FRAMEWORK 5.0

- Để thao tác với các kiểu dữ liệu không gian, EF 5.0 cung cấp loại dữ liệu: **DbGeography** và **DbGeometry**.
- Để chỉnh các thuộc tính trong mô hình của bạn có kiểu tọa độ, trong Properties của thuộc tính bạn điều chỉnh **Type** là các kiểu tọa độ (**Geography**, **Geometry**).
Hoặc bạn có thể định nghĩa trong Code:

```
public partial class Course
{
    public Course()
    {
        this.Students = new HashSet<Student>();
    }

    public int CourseId { get; set; }
    public string CourseName { get; set; }
    public Nullable<int> TeacherId { get; set; }
    public System.Data.Spatial.DbGeography Location { get; set; }

    public virtual Teacher Teacher { get; set; }
    public virtual ICollection<Student> Students { get; set; }
}
```

- Sử dụng:

```
ctx.Courses.Add(new Course() { CourseName = "New Course",
    Location = DbGeography.FromText("POINT(-122.360 47.656)") });

ctx.SaveChanges();
```

2.5.4 Table-Valued Function

- **Table-Valued Function (TVF)** tương tự như thủ tục lưu trữ với một sự khác biệt quan trọng: kết quả của **TVF** là composable đó có nghĩa là kết quả của **TVF** có thể được sử dụng trong một truy vấn **LINQ**.
- Cách thêm các **Table-Valued Function** vào mô hình thực thể **DatabaseFirst** mà ta đã thực hiện:

- Trong CSDL, ta có **Table-Valued Function LayDSLopHocCuaHocVien** với đối số là **HocVienId**. Hàm này sẽ lấy ra thông tin các lớp học mà học viên có Id muốn tìm tham gia:

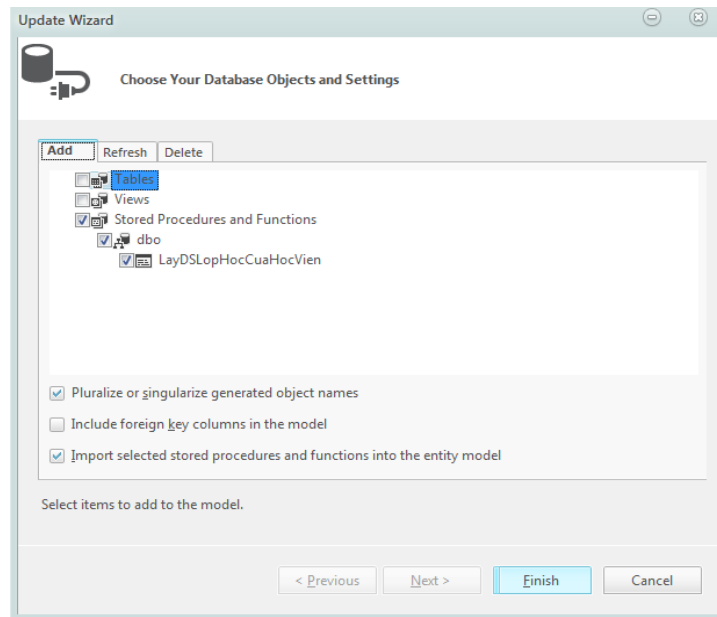
```
USE [QuanLyHocVien]
GO

/***** Object:      UserDefinedFunction      [dbo].[LayDSLopHocCuaHocVien]
*****/

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

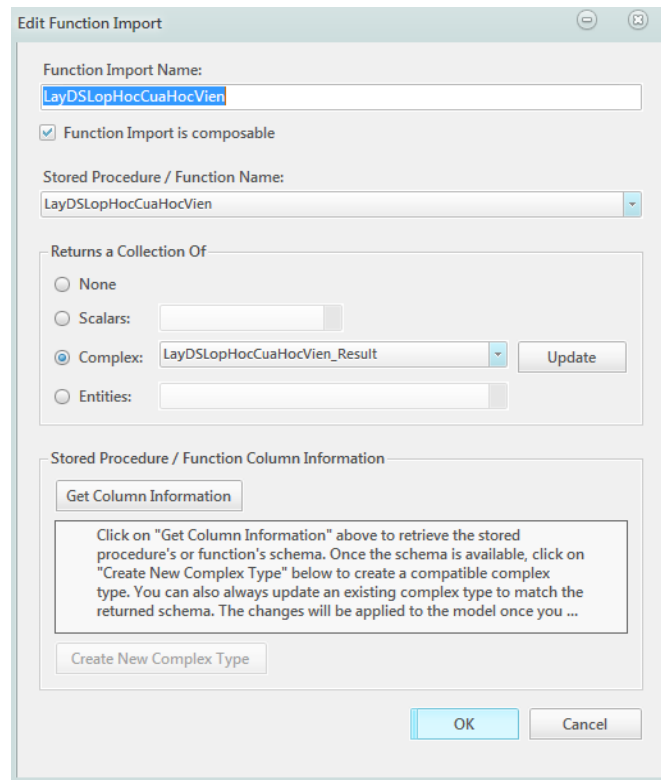
CREATE FUNCTION [dbo].[LayDSLopHocCuaHocVien]
(
    -- Add the parameters for the function here
    @HocVienId int
)
RETURNS TABLE
AS
RETURN
(
    -- Add the SELECT statement with parameter references here
    select c.Mon, c.GioHoc
    from HocVien s left outer join HocVienLopHoc sc on
    sc.HocVien_HocVienId = s.HocVienId join LopHoc c on
    c.LopHocId=sc.LopHoc_LopHocId
    where s.HocVienId = @HocVienId
)
```

- Thêm **TVF** bằng các click phải vào vùng **EDM Design** của mô hình và chọn **Update Model from Database**.
- Trong hộp thoại **Update Wizard** chọn **LayDSLopHocCuaHocVien** tại **Stored Procedures and Functions**.



Hình 2.27 Thêm các Table – Valued Function

- Chọn **Finish**.
- Bạn sẽ thấy **TVF LayDSLopHocCuaHocVien** được hiển thị trong thư mục **Function Imports** và **Stored Procedures/Functions** trong **Model Browser**.
- Trong thư mục **Function Imports**, click phải vào **TVF** vừa thêm và chọn **Edit**.
- Chọn kiểu trả về của **TVF**. Nếu **TVF** của bạn trả về một giá trị, chọn **Scalars** và chọn kiểu giá trị. Nếu là một tập hợp các record, chọn **Complex** và đặt tên đối tượng sẽ chứa các bản record. Nếu trả về thực thể có cùng cấu trúc với thực thể có sẵn trong mô hình, chọn **Entites** và chọn tên Entity cùng cấu trúc. Trong ví dụ này, kiểu trả về là một **Complex** và đối tượng chứa nó là **LayDSLopHocCuaHocVien_Result**.



Hình 2.28 Edit Table – Valued Function

- Sử dụng TVF:

```
using (QuanLyHocVien2Entities db = new QuanLyHocVien2Entities())
{
    var courseList =
db.LayDSLopHocCuaHocVien(1).Where(c=>c.GioHoc=="14").ToList<LayDSLopHocCuaHocVien_Result>();

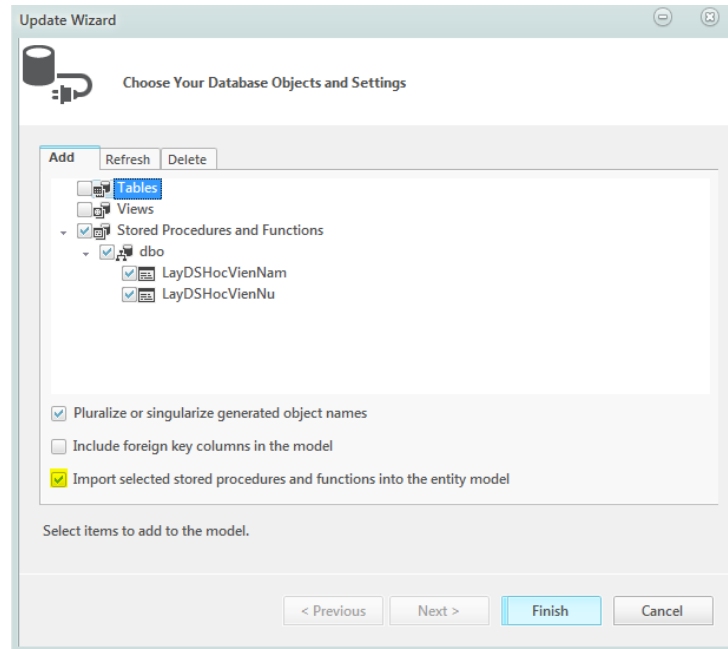
    foreach (LayDSLopHocCuaHocVien_Result cs in courseList)
        Console.WriteLine("Ten Mon: {0}, Gio Hoc: {1}", cs.Mon, cs.GioHoc);
}
```



2.5.5 Batch Import of Stored Procedures

- Trong EF 5.0 cho phép bạn thêm vào mô hình cùng một lúc nhiều thủ tục lưu trữ thông qua **Entity Data Model Wizard** sau khi đánh dấu vào “**Import selected**

stored procedures and functions into the entity model". Ta có thêm update vào mô hình các thủ tục tương tự như làm với TVF.



Hình 2.29 Add Stored Procedures

- Sau khi thêm các thủ tục lưu trữ, bạn cũng **Edit** nó tương tự như TVF.
- Để chắc chắn rằng khi chạy các thủ tục lưu trữ sẽ không đưa ra lỗi bạn cần xác nhận nó bằng cách nhấp phải chuột vào thủ tục lưu trữ trong thư mục Function Imports và chọn '**Validate**'.

2.5.6 Multiple diagrams

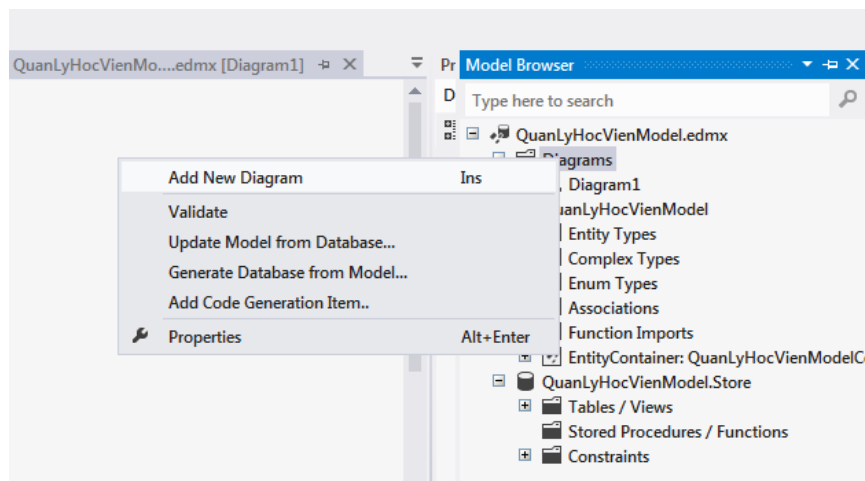
- **Visual Studio 2012** cho phép bạn phân chia mô hình dữ liệu thực thể thành nhiều sơ đồ. Việc này sẽ giúp bạn dễ quan sát các sơ đồ và mối quan hệ của chúng dễ dàng hơn khi mô hình thực thể dữ liệu của bạn trở nên lớn. Mỗi sơ đồ có thể chứa một phần của mô hình dữ liệu thực thể. Giả sử bạn có thể tạo ra sơ đồ HocVien chỉ gồm các thực thể học viên và các thực thể có liên quan tới học viên, còn những thực thể không liên quan bạn không đưa vào. Mục đích của việc tách các thành các sơ đồ này để bạn có thể quan sát sơ đồ của học viên khi cần thiết mà không phải bị khó nhìn bởi sự có mặt của các thực thể và các mối quan hệ không liên quan.

ENTITY FRAMEWORK 5.0

- Việc tách mô hình ra thành nhiều sơ đồ không làm ảnh hưởng tới cơ sở dữ liệu.
- Các cách tạo ra các sơ đồ mới:
 - Tạo một sơ đồ mới và kéo thả các thực thể từ Model Browser vào.
 - Di chuyển một thực thể tại mô hình đã có vào mô hình mới.

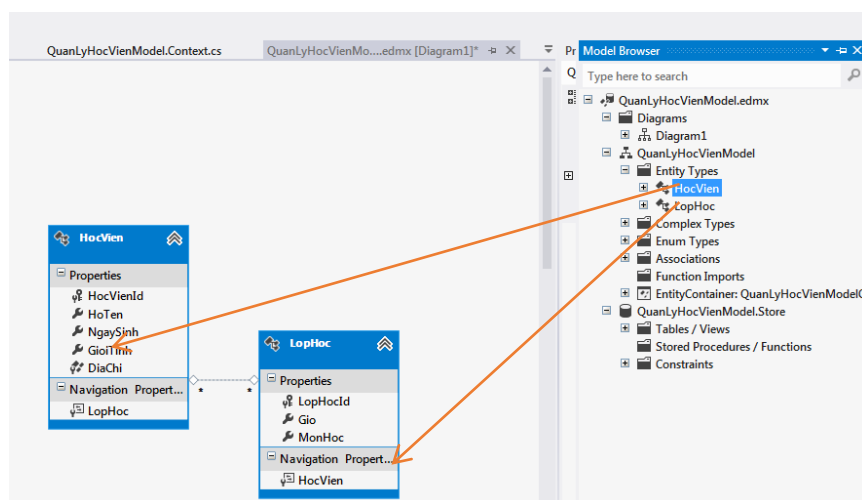
2.5.6.1 Tạo một sơ đồ mới và kéo thả các thực thể vào

- Click phải vào thư mục **Diagrams** trong **Model Browser** và cho **Add New Diagram**.



Hình 2.30 Add New Diagram

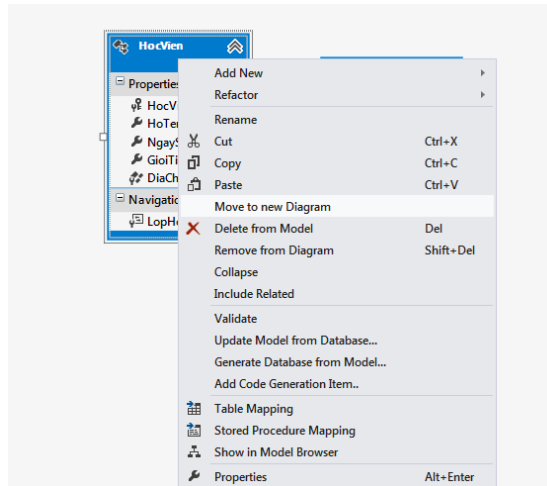
- Bạn có thể đổi tên các sơ đồ vừa tạo.
- Kéo các thực thể từ **Model Browser** và thả vào sơ đồ của bạn.



Hình 2.31 Kéo thả Entity

2.5.6.1 Di chuyển thực thể từ sơ đồ có sẵn

- Click phải vào thực thể và chọn **Move to new Diagram**.

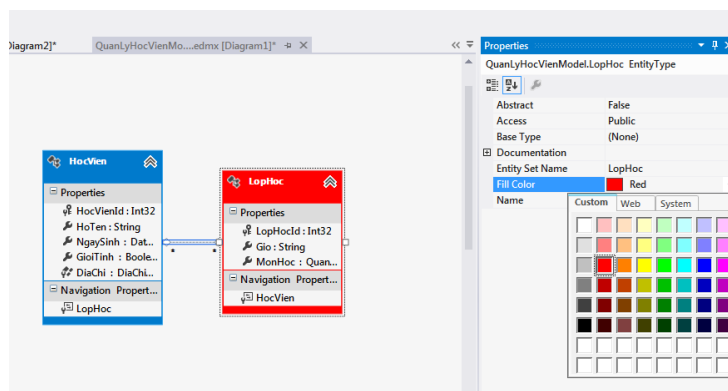


Hình 2.32 Move Entity

- Để đưa những thực thể có quan hệ với một thực thể đang ở trong một sơ đồ nào đó, click chuột phải vào thực thể trong sơ đồ và chọn **Include Related**.
- Khi không muốn thực thể đó nằm trong sơ đồ nữa, click phải vào thực thể và chọn **Remove from Diagram**. Nếu bạn chọn **Delete from Model** thì thực thể đó sẽ không còn trong mô hình dữ liệu thực thể của bạn và bạn không thể sử dụng thực thể đó được nữa.

2.5.7 Tô màu cho các thực thể trong EDM Designer

- Click phải vào thực thể muốn tô màu chọn **Properties**.
- Trong cửa sổ **Properties** chọn màu cho thực thể tại mục **Fill Color**.



Hình 2.33 Tô màu cho thực thể trong EDM Designer

2.6 Kết luận

Với các tính năng mới của EF 5.0, các chương trình thao tác với dữ liệu được phát triển tốt hơn. Kiểu dữ liệu không gian được hỗ trợ vào làm tăng tính đa dạng cho sự áp dụng EF 5.0. Kiểu Enum giúp tăng tính ràng buộc cho dữ liệu, giúp bảo toàn và hạn chế sai dữ liệu. Các tính năng hỗ trợ về EF Designer giúp lập trình viên quản lý các mô hình thực thể dễ dàng và trực quan hơn.

Với các phiên bản trước, việc áp dụng Linq vào truy xuất dữ liệu tuy dễ dàng nhưng tốn nhiều thời gian cho quá trình dịch lệnh. Ở phiên bản này, buffer được thêm vào đã cải tiến hơn hiệu suất truy vấn, tăng tốc cho chương trình.

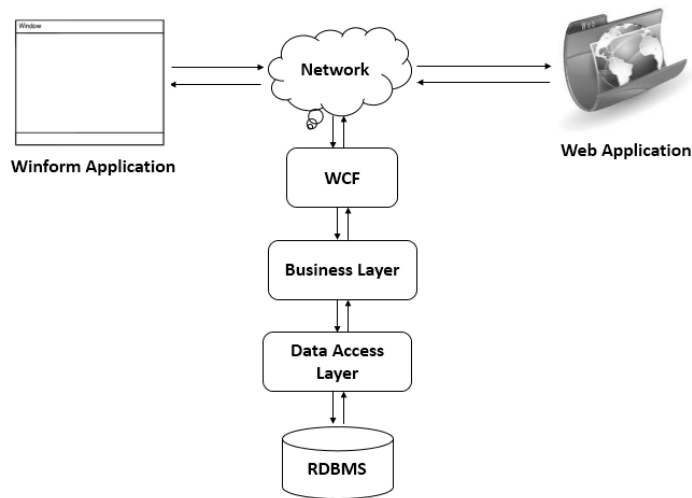
CHƯƠNG 3: XÂY DỰNG PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

3.1 Phân tích hệ thống

3.1.1 Khảo sát hiện trạng và đặt yêu cầu

- Trên thị trường, phần mềm quản lý giao dịch cửa hàng vật liệu xây dựng nói riêng và các phần mềm quản lý nói chung được phát triển đa dạng và đáp ứng được nhu cầu của người sử dụng. Nhiều phần mềm được thương mại hóa và cài đặt được cho nhiều cửa hàng chứ không riêng biệt cho một cửa hàng đặc trưng. Các phần mềm sử dụng các công nghệ kết nối dữ liệu phù hợp để truy cập đến dữ liệu, phục vụ cho lưu trữ và xử lý.
- Hầu hết các ứng dụng quản lý viết trên Windows Form kết nối dữ liệu qua công nghệ ADO.NET.
- Dựa vào phần mềm quản lý giao dịch đã thực hiện tại Tiểu luận chuyên ngành – Tìm hiểu Entity Framework 5.0, .Net 4.5 và xây dựng phần mềm quản lý bán hàng cho cửa hàng vật liệu xây dựng Hồng Tuấn – nhóm phát triển phần mềm mới đảm bảo các chức năng đã nghiên cứu tại phần mềm cũ và thêm các yêu cầu mới:
 - o Mở rộng ứng dụng, phát triển thêm Website hiển thị sản phẩm – đóng vai trò như Catalog mua hàng.
 - o Cho phép làm việc trên nhiều máy.
 - o Sản phẩm áp dụng được cho nhiều cửa hàng khác nhau.
 - o Tăng tính bảo mật.
 - o Thao tác dễ dàng hơn.

3.1.2 Mô hình giải quyết bài toán



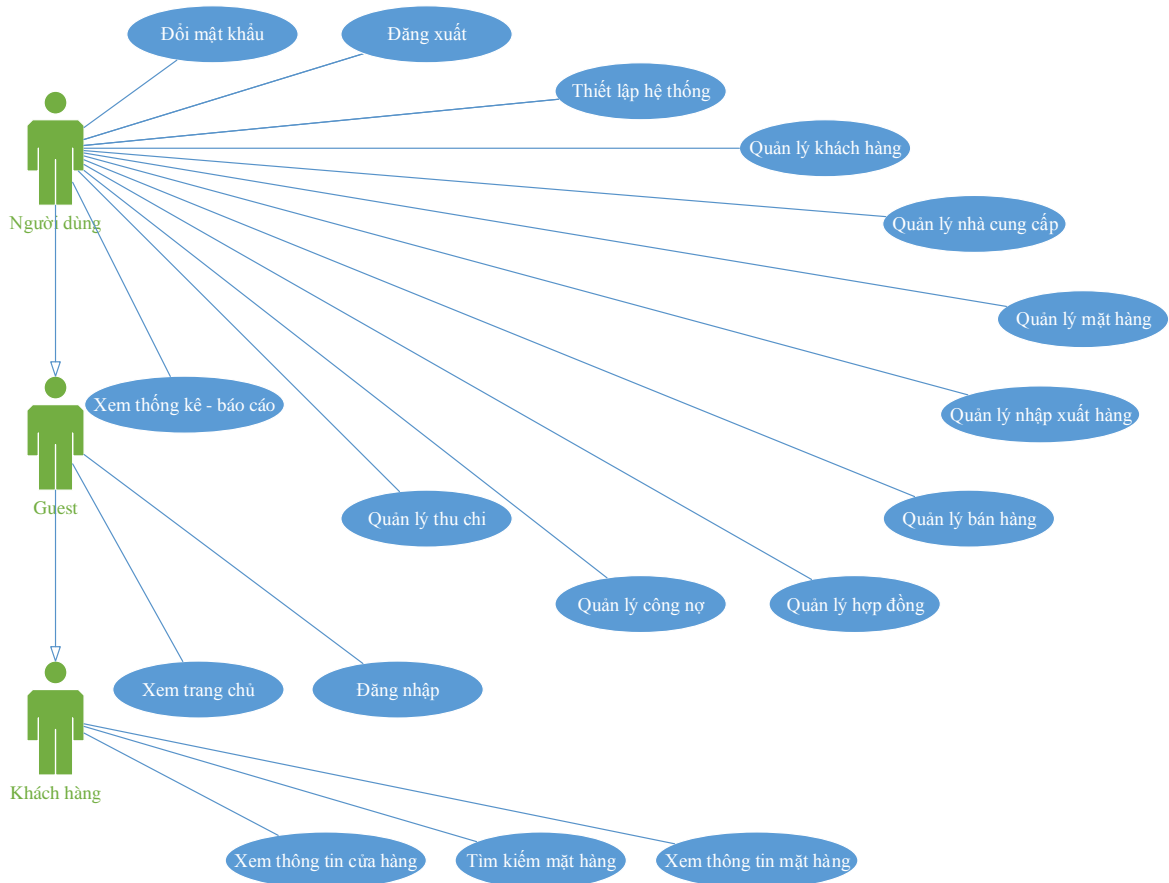
Hình 3.1 Mô hình giải quyết bài toán quản lý bán hàng của cửa hàng vật liệu xây dựng.

Hệ thống này bao gồm:

- Winform, Web application đóng vai trò client cho phép người dùng thực hiện các thao tác.
 - Windows Form Application: Ở đây người dùng được cung cấp ứng dụng có tên là “Quản lý bán hàng” cung cấp đầy đủ các chức năng để cửa hàng có thể thực các giao dịch với khách hàng: Lập hóa đơn bán lẻ, GTGT, hợp đồng... với nhà cung cấp: Nhập hàng,... Đồng thời xác định các khoản thu chi của cửa hàng và xem các thống kê.
 - Web Application: Là một ứng dụng web trên nền tảng Sencha cho phép người dùng xem danh sách các mặt hàng mà cửa hàng cung cấp.
- WCF (Windows Communication Foundation) đảm nhiệm vai trò giao tiếp với các client, gọi các hàm trong Business Layer để thực hiện xử lý, nhận kết quả và trả về cho client.
 - IRESTHRService.cs: Là một giao diện các dịch vụ cung cấp cho client.
 - IRESTHRService.svc: Lớp thực thi giao diện IRESTHRService.cs. Lớp này sẽ gọi các dịch vụ mà Business Layer cung cấp để xử lý các yêu cầu của người dùng.

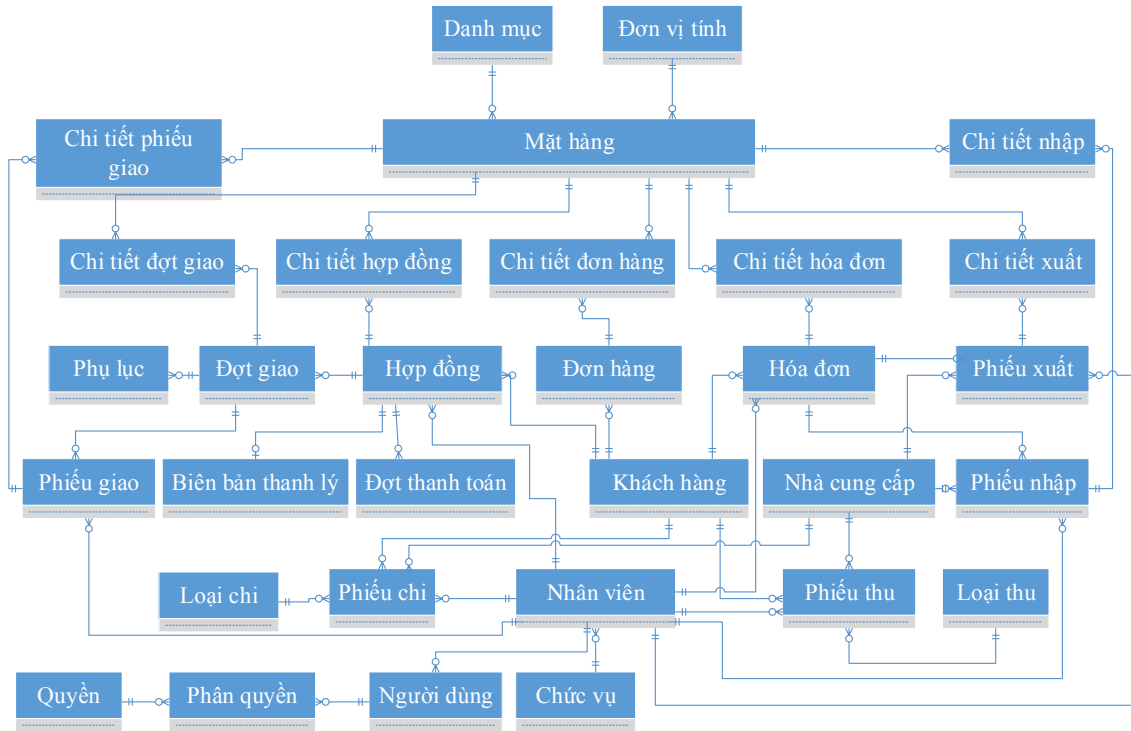
- Business Layer: Thực hiện các nghiệp vụ chính của chương trình, cung cấp các dịch vụ cho lớp WCF.
- Data Access Layer: Đảm nhận các nghiệp vụ liên quan đến lưu trữ và truy xuất dữ liệu.

3.1.3 Usecase tổng quát của phần mềm và website:



Hình 3.2 Usecase tổng quát

3.1.4 Cơ sở dữ liệu



Hình 3.3 Mô hình thực thể dữ liệu

- Cài đặt bảng dữ liệu: Hệ quản trị cơ sở dữ liệu sử dụng: SQL Server 2008

Chức vụ

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã chức vụ</u>	String	Mã chức vụ	StringLength(10)
2	Tên chức vụ	String	Tên chức vụ	

Nhân viên

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã nhân viên</u>	String	Mã nhân viên	
2	Họ tên	String	Họ tên	
3	Giới tính	Boolean	Giới tính	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

4	Ngày sinh	Date Time	Ngày sinh	
5	Địa chỉ	String	Địa chỉ	
6	SĐT	String	SĐT	
7	Ngày vào làm	Date Time	Ngày vào làm	
8	Mã chức vụ	String	Mã chức vụ	
9	Đã xóa	Boolean	Đã xóa	
10	Kiểm tra	Float	Kiểm tra	

Quyền

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Mã quyền	String	Mã quyền	StringLength(10)
2	Tên quyền	String	Tên quyền	

Người dùng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Tên đăng nhập	String	Tên đăng nhập	StringLength(20)
2	Mật khẩu	String	Tên loại	
3	Mã nhân viên	String	Mã nhân viên	StringLength(10)
4	Kiểm tra	Float	Kiểm tra	
5	Đã khóa	Bool	Đã khóa	

Phân quyền

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Tên đăng nhập	String	Tên đăng nhập	StringLength(20)
2	Mã quyền	String	Mã quyền	StringLength(10)

Danh mục

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
-----	------------	--------------	-------	---------

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

1	<u>Mã danh mục</u>	String	Mã danh mục	StringLength(10)
2	Tên danh mục	String	Tên danh mục	
3	Kiểm tra	Float	Kiểm tra	

Đơn vị tính

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	StringLength(10)
2	Tên đơn vị tính	String	Tên đơn vị tính	
3	Kiểm tra	Float	Kiểm tra	

Nhà cung cấp

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã nhà cung cấp</u>	String	Mã nhà cung cấp	
2	Tên nhà cung cấp	String	Tên nhà cung cấp	
3	Địa chỉ	String	Địa chỉ nhà cung cấp	
4	Mã số thuế	String	Mã số thuế	
5	Email	String	Email	
6	Website	String	Website	
7	SĐT	String	Số điện thoại	
8	Số tài khoản	String	Số tài khoản	
9	Quản lý	String	Tên người quản lý nơi cung cấp hàng	
10	SĐT quản lý	String	Số điện thoại người quản lý	
11	Đã xóa	Bool	Đã xóa	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

12	Kiểm tra	Float	Kiểm tra	
----	----------	-------	----------	--

Mặt hàng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã mặt hàng</u>	String	Mã mặt hàng	StringLength(10)
2	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	
3	Tên mặt hàng	String	Tên mặt hàng	
4	Mô tả	String	Mô tả	
5	Số lượng	Int	Số lượng	
6	Đơn giá nhập	Double	Đơn giá nhập	
7	Đơn giá xuất	Double	Đơn giá xuất	
8	Đã xóa	Bool	Đã xóa	
9	Hình ảnh	Byte[]	Hình ảnh	
10	<u>Mã danh mục</u>	String	Mã danh mục	StringLength(10)
11	<u>Mã nhà cung cấp</u>	String	Mã nhà cung cấp	StringLength(10)
12	Kiểm tra	Bool	Kiểm tra	

Khách hàng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã khách hàng</u>	String	Mã khách hàng	StringLength(10)
2	Tên khách hàng	String	Tên khách hàng	
3	Địa chỉ	String	Địa chỉ	
4	SĐT	String	Số điện thoại	
5	Email	String	Email	
6	Mã số thuế	String	Mã số thuế	
7	Số tài khoản	String	Số tài khoản	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

8	Đã xóa	Bool	Đã xóa	
9	Kiểm tra	Float	Kiểm tra	

Đơn hàng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số đơn hàng</u>	String	Số đơn hàng	StringLength(10)
2	Người mua	String	Người mua	
3	Địa chỉ	String	Địa chỉ	
4	Ngày lập	DateTime	Ngày lập	
5	Ghi chú	String	Ghi chú	
6	Mã khách hàng	String	Mã khách hàng	StringLength(10)

Chi tiết đơn hàng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Số lượng	Int	Số lượng	
2	Đơn giá	Double	Đơn giá	
3	Số lượng thiếu	Int	Số lượng thiếu	
4	<u>Mã mặt hàng</u>	String	Mã mặt hàng	
5	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	
6	<u>Số đơn hàng</u>	String	Số đơn hàng	

Hóa đơn

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số hóa đơn</u>	String	Số hóa đơn	
2	Ngày lập	DateTime	Ngày lập	
3	Người mua	String	Người mua	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

4	Địa chỉ	String	Địa chỉ	
5	Tổng tiền	Double	Tổng tiền	
6	<u>Mã khách hàng</u>	String	Mã khách hàng	StringLength(10)
7	<u>Mã nhân viên lập</u>	String	Mã nhân viên lập	StringLength(10)
8	Chiết khấu	Float	Chiết khấu	
9	Tên đơn vị	String	Tên đơn vị	
10	Hình thức thanh toán	String	Hình thức thanh toán	
11	Mã số thuế	String	Mã số thuế	
12	Số tài khoản	String	Số tài khoản	
13	Thành tiền	Double	Thành tiền	
14	Thuế GTGT	Float	Thuế GTGT	

Chi tiết hóa đơn

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Đơn giá xuất	Double	Đơn giá xuất	
2	Đơn giá nhập	Double	Đơn giá nhập	
3	Số lượng	Int	Số lượng	
4	<u>Số hóa đơn</u>	String	Số hóa đơn	
5	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	
6	<u>Mã mặt hàng</u>	String	Mã mặt hàng	

Phiếu xuất

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã phiếu xuất</u>	String	Mã phiếu xuất	StringLength(10)

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

2	Người nhận	String	Người nhận	
3	Ngày xuất	DateTime	Ngày xuất	
4	Thành tiền	Double	Thành tiền	
5	Thuế GTGT	Float	Thuế GTGT	
6	<u>Mã nhân viên lập</u>	String	Mã nhân viên lập	StringLength(10)
7	Chiết khấu	Double	Chiết khấu	
8	<u>Mã phiếu nhập</u>	String	Mã phiếu nhập	StringLength(10)
9	Lý do xuất	String	Lý do xuất	
10	<u>Số hóa đơn</u>	String	Số hóa đơn	StringLength(10)

Chi tiết phiếu xuất

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Số lượng	Int	Số lượng	
2	Đơn giá nhập	Double	Đơn giá nhập	
3	Đơn giá xuất	Double	Đơn giá xuất	
4	<u>Mã mặt hàng</u>	String	Mã mặt hàng	StringLength(10)
5	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	StringLength(10)
6	<u>Mã phiếu xuất</u>	String	Mã phiếu xuất	StringLength(10)

Phiếu nhập

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã phiếu nhập</u>	String	Mã phiếu nhập	StringLength(10)
2	Người giao	String	Người giao	
3	Ngày nhập	DateTime	Ngày nhập	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

4	Thành tiền	Double	Thành tiền	
5	Thuế GTGT	Float	Thuế GTGT	
6	<u>Mã nhân viên lập</u>	String	Mã nhân viên lập	StringLength(10)
7	Chiết khấu	Double	Chiết khấu	
8	Hóa đơn mua hàng	String	Hóa đơn mua hàng	
9	<u>Mã nhà cung cấp</u>	String	Mã nhà cung cấp	StringLength(10)
10	Lý do trả	String	Lý do trả	
11	<u>Số hóa đơn</u>	String	Số hóa đơn	StringLength(10)

Chi tiết phiếu nhập

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Đơn giá xuất	Double	Đơn giá xuất	
2	Đơn giá nhập	Double	Đơn giá nhập	
3	Số lượng	Int	Số lượng	
4	<u>Mã mặt hàng</u>	String	Mã mặt hàng	StringLength(10)
5	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	StringLength(10)
6	<u>Mã phiếu nhập</u>	String	Mã phiếu nhập	StringLength(10)

Hợp đồng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã hợp đồng</u>	String	Mã hợp đồng	StringLength(10)
2	Ngày lập	DateTime	Ngày lập	
3	Ngày kết thúc	DateTime	Ngày kết thúc	
4	Ngày thanh lý	DateTime	Ngày thanh lý	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

5	Đại diện KH	String	Đại diện khách hàng	
6	SĐT đại diện	String	Số điện thoại đại diện	
7	Thành tiền	Double	Thành tiền	
8	Thuế GTGT	Float	Thuế GTGT	
9	Trả trước	Double	Trả trước	
10	Lưu file	String	Lưu file	
11	<u>Mã khách hàng</u>	String	Mã khách hàng	StringLength(10)
12	<u>Mã nhân viên phụ trách</u>	String	Mã nhân viên phụ trách	StringLength(10)

Chi tiết hợp đồng

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Đơn giá xuất	Double	Đơn giá xuất	
2	Đơn giá nhập	Double	Đơn giá nhập	
3	Số lượng	Int	Số lượng	
4	<u>Mã hợp đồng</u>	String	Số hóa đơn	StringLength(10)
5	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	StringLength(10)
6	<u>Mã mặt hàng</u>	String	Mã mặt hàng	StringLength(10)

Đợt thanh toán

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số đợt thanh toán</u>	Int	Số đợt thanh toán	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

2	<u>Mã hợp đồng</u>	String	Mã hợp đồng	StringLength(10)
3	Số tiền thanh toán	Double	Số tiền thanh toán	
4	Ngày thanh toán	DateTime	Ngày thanh toán	

Đợt giao

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số đợt</u>	Int	Số đợt	
2	<u>Mã hợp đồng</u>	String	Mã hợp đồng	StringLength(10)
3	Cách thức giao	String	Các thức giao	
4	Ngày giao	DateTime	Ngày giao	
5	Địa chỉ giao	String	Địa chỉ giao	

Chi tiết đợt giao

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Số lượng	Int	Số lượng	
2	<u>Số đợt</u>	Int	Số đợt	
3	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	
4	<u>Mã hợp đồng</u>	String	Mã hợp đồng	
5	<u>Mã mặt hàng</u>	String	Mã mặt hàng	

Phụ lục

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số đợt</u>	Int	Số đợt	
2	<u>Mã hợp đồng</u>	String	Mã hợp đồng	StringLeng th(10)

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

3	<u>Mã mặt hàng</u>	String	Mã mặt hàng	StringLength(10)
4	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	StringLength(10)
5	<u>Ngày điều chỉnh</u>	DateTime	Ngày điều chỉnh	
6	Loại điều chỉnh	String	Loại điều chỉnh	
7	Số lượng	Int	Số lượng	
8	Ghi chú	String	Ghi chú	
9	Đơn giá nhập	Double	Đơn giá nhập	
10	Đơn giá xuất	Double	Đơn giá xuất	

Biên bản thanh lý

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Ngày thanh lý	DateTime	Ngày thanh lý	
2	Chưa thanh toán	Double	Chưa thanh toán	
3	KHBoiThuong	Double	Khách hàng bồi thường	
4	Cửa hàng hoàn lại	Double	Cửa hàng hoàn lại	
5	Cửa hàng bồi thường	Double	Cửa hàng bồi thường	
6	Tiền thêm bớt hàng	Double	Tiền thêm bớt hàng	
7	Tiền hàng đã giao	Double	Tiền hàng đã giao	
8	<u>Mã hợp đồng</u>	String	Mã hợp đồng	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

Loại chi

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã loại chi</u>	String	Mã loại thu	StringLength(10)
2	Tên loại	String	Tên loại	

Phiếu giao

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số đợt</u>	Int	Số đợt	
2	<u>Mã hợp đồng</u>	String	Mã hợp đồng	StringLength(10)
3	<u>Số phiếu giao</u>	String	Số phiếu giao	StringLength(10)
4	<u>Mã nhân viên lập</u>	String	Mã nhân viên lập	StringLength(10)
5	<u>Mã nhân viên giao</u>	String	Mã nhân viên giao	StringLength(10)
6	Ngày giao	DateTime	Ngày giao	

Chi tiết phiếu giao

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số phiếu giao</u>	String	Số phiếu giao	
2	Số lượng	Int	Số lượng	
3	<u>Mã mặt hàng</u>	String	Mã mặt hàng	StringLength(10)
4	<u>Mã đơn vị tính</u>	String	Mã đơn vị tính	StringLength(10)
5	Đơn giá nhập	Double	Đơn giá nhập	

Loại thu

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
-----	------------	--------------	-------	---------

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

1	<u>Mã loại thu</u>	String	Mã loại thu	StringLength(10)
2	Tên loại	String	Tên loại	

Phiếu thu

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số phiếu thu</u>	String	Số phiếu thu	StringLength(10)
2	Ngày thu	DateTime	Ngày thu	
3	Người nộp	String	Người nộp	
4	Số tiền thu	Double	Số tiền thu	
5	Lý do thu	String	Lý do thu	
6	Số chứng từ	String	Số chứng từ	
7	<u>Mã loại thu</u>	String	Mã loại thu	StringLength(10)
8	<u>Mã nhân viên lập</u>	String	Mã nhân viên lập	StringLength(10)
STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Mã loại chi</u>	String	Mã loại chi	StringLength(10)
2	Tên loại	String	Tên loại	

Phiếu chi

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	<u>Số phiếu chi</u>	String	Số phiếu chi	StringLength(10)
2	Ngày chi	DateTime	Ngày chi	
3	Người nhận	String	Người nhận	
4	Số tiền chi	Double	Số tiền chi	
5	Lý do chi	String	Lý do chi	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

6	Số chứng từ	String	Số chứng từ	
7	<u>Mã loại chi</u>	String	Mã loại chi	StringLength(10)
8	<u>Mã nhân viên lập</u>	String	Mã nhân viên lập	StringLength(10)

Chốt tồn

STT	Thuộc tính	Kiểu dữ liệu	Mô tả	Ghi chú
1	Ngày chốt	DateTime	Ngày chốt	
2	Mã mặt hàng	String	Mã mặt hàng	StringLength(10)
3	Mã đơn vị tính	String	Mã đơn vị tính	StringLength(10)
4	Tồn trước	Int	Tồn trước	
5	Tồn lỗi trước	Int	Tồn lỗi trước	
6	Nhập	Int	Nhập	
7	Xuất	Int	Xuất	
8	Hủy	Int	Hủy	
9	Đã sửa	Int	Đã sửa	
10	Thanh lý	Int	Thanh lý	
11	Khách hàng trả	Int	Khách hàng trả	
12	Hàng lỗi	Int	Hàng lỗi	
13	Trả nhà cung cấp	Int	Trả nhà cung cấp	
14	Tổng hàng tốt	Int	Tổng hàng tốt	

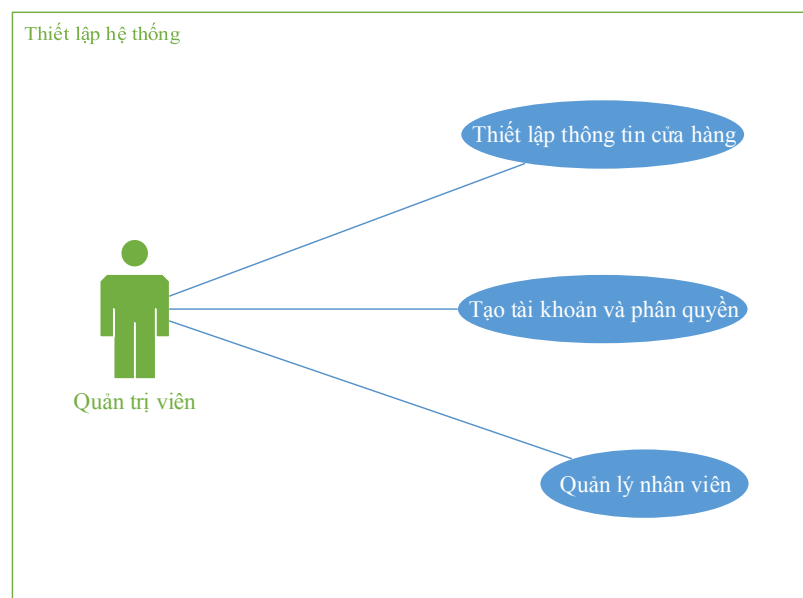
15	Tổng hàng lỗi	Int	Tổng hàng lỗi	
16	Tổng	Int	Tổng	

3.1.4 Công cụ

- Lập trình WinForm: Visual Studio 2012
- Giao diện: DevExpress 13.1.4
- Hệ quản trị cơ sở dữ liệu: SQL Server 2008
- Lập trình Web: JetBrains WebStorm 5.0

3.2 Các chức năng chính

3.2.1 Thiết lập hệ thống

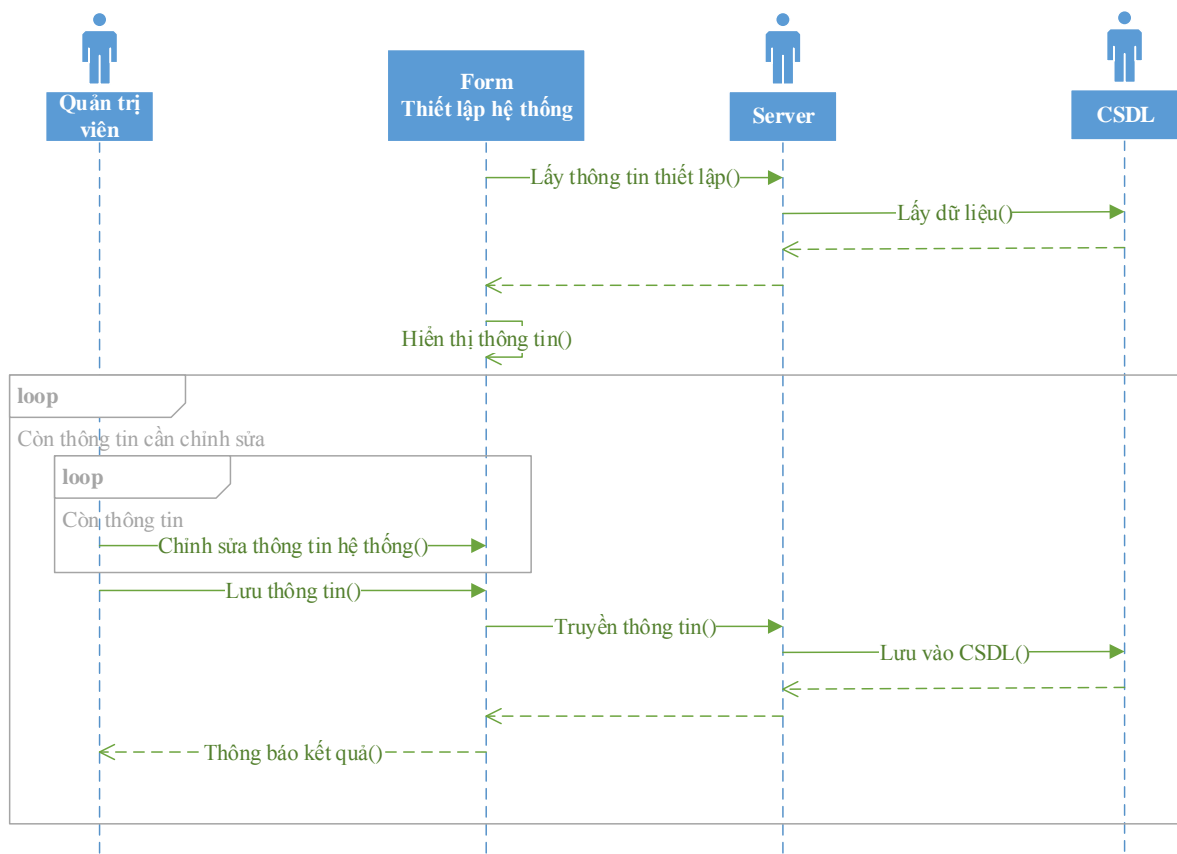


Hình 3.4 Chức năng Thiết lập hệ thống

3.2.1.1 Thiết lập thông tin cửa hàng

- Nội dung: Chính sửa và thay đổi các thông tin cửa hàng như: Tên cửa hàng, địa chỉ, email,... được dùng để hiển thị lên WebSite.
- Các bảng dữ liệu sử dụng: Không có, sử dụng file GioiThieuCuaHang.txt, ThôngTinCuaHang.txt, ThôngTinLienHe.txt.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.5 Sơ đồ tuần tự chức năng Thiết lập thông tin cửa hàng

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

The image displays two screenshots of the 'Thiết lập' (Setup) screen in the software. The top screenshot shows the 'Thông tin cửa hàng' (Store Information) tab selected, with a 'Lưu' (Save) button and a count of 2. The bottom screenshot shows the 'Thông tin liên hệ' (Contact Information) tab selected, with a 'Lưu' (Save) button and a count of 3. Both screenshots show a sidebar with 'Thông tin cửa hàng', 'Tài khoản đăng nhập', and 'Thông tin nhân viên'. The main area contains input fields for store name, address, phone, fax, email, and manager.

Hình 3.6 Giao diện nghiệp vụ Thiết lập thông tin cửa hàng

- Danh sách các xử lý giao diện nghiệp vụ Thiết lập thông tin cửa hàng

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Thay đổi thông tin cửa hàng	Khi người dùng chọn “Lưu”.	
2	Thay đổi giới thiệu cửa hàng	Khi người dùng chọn “Lưu”.	
3	Thay đổi thông tin liên lạc cửa hàng	Khi người dùng chọn “Lưu”.	

- Hiện thị thông tin cửa hàng trên Web:

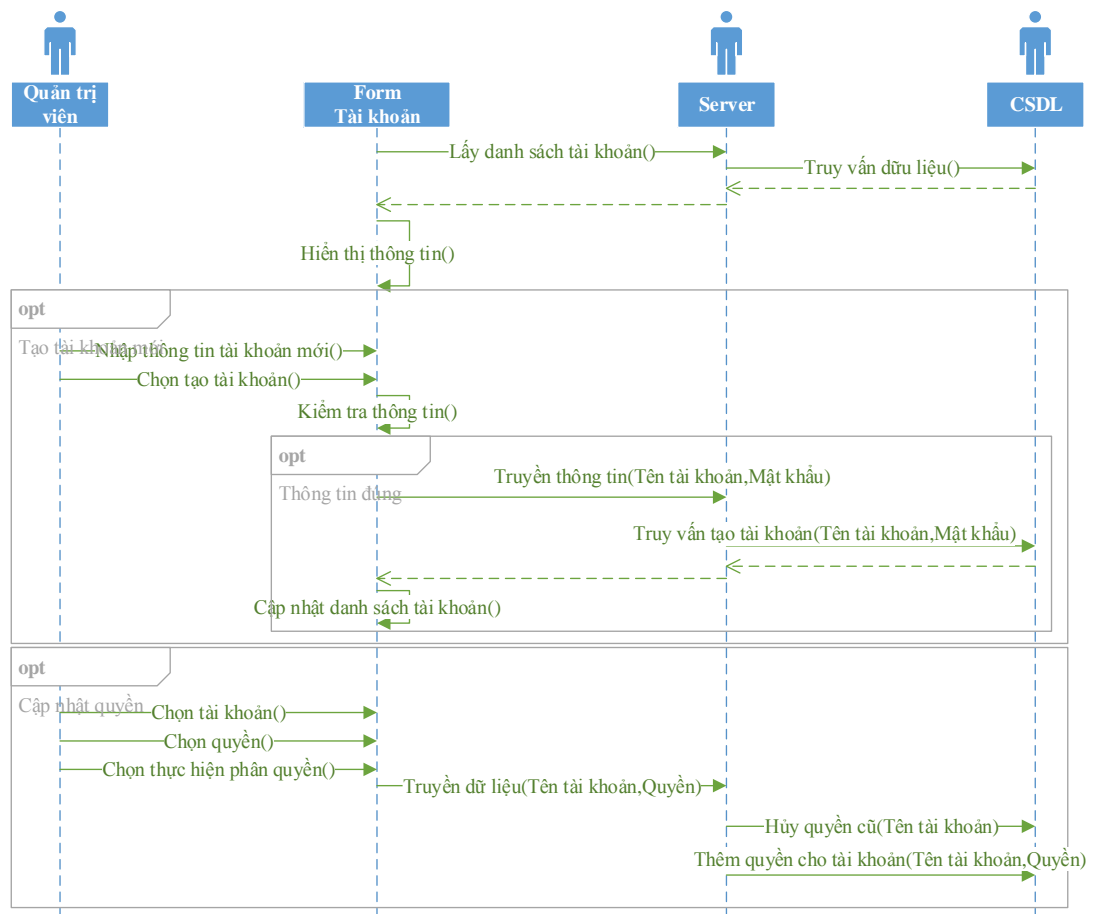
☰ Trang chủ	☰ Liên hệ
 <p>THÔNG TIN:</p> <p>Tên cửa hàng: CÔNG TY TNHH THÉP NAM THÀNH VINH Địa chỉ: 42 Cống Lở - P.15- Q.Tân Bình - Tp.Hồ Chí Minh Số điện thoại: 0909789888 Fax: 08 3526 8694 Email: spkt10110098@gmail.com</p>	<p>Tên cửa hàng: CÔNG TY TNHH THÉP NAM THÀNH VINH Địa chỉ: 42 Cống Lở - P.15- Q.Tân Bình - Tp.Hồ Chí Minh Số điện thoại: 0909789888 Fax: 08 3526 8694 Email: spkt10110098@gmail.com Quản lý: Trần Chí Tâm</p>

Hình 3.7 Giao diện web hiển thị thông tin cửa hàng

3.2.1.2 Tạo tài khoản và phân quyền

- Nội dung: Tạo tài khoản đăng nhập và chọn các quyền cho tài khoản.
- Điểm nổi bật: Phân quyền bên dưới Hệ quản trị Cơ sở dữ liệu SQL Server, từ đó tăng tính bảo mật cho hệ thống.
- Các bảng dữ liệu sử dụng: Người Dùng, Phân Quyền, Nhân Viên, Quyền.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.8 Sơ đồ tuần tự chức năng Tạo tài khoản và phân quyền

- Giao diện chức năng:

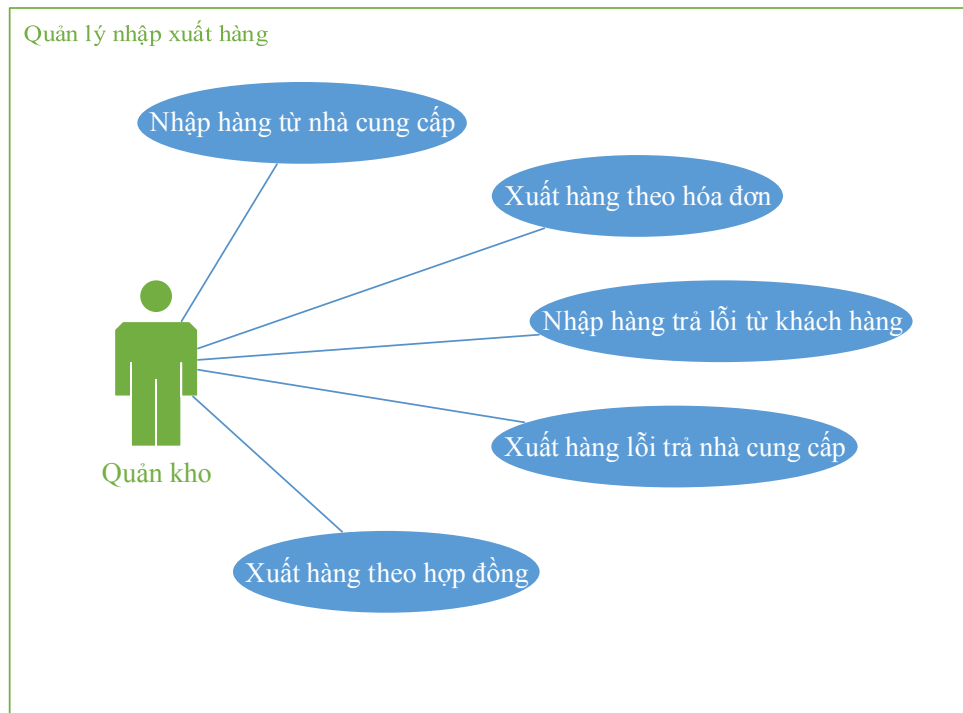
The screenshot shows the 'THIẾT LẬP' (Setup) window of the 'PHẦN MỀM QUẢN LÝ BÁN HÀNG' (Sales Management Software). The interface includes a top menu bar with options like 'TRANG CHỦ', 'THIẾT LẬP', 'MẬT HÀNG', 'NHÀ CUNG CẤP', 'KHÁCH HÀNG', 'NHẬP - XUẤT', 'BÁN HÀNG', 'HỢP ĐỒNG', 'CÔNG NỢ', 'THU - CHI', and 'THỐNG KÊ'. The main area is divided into a left sidebar with 'Thông tin cửa hàng' (Store Information) and 'Thông tin nhân viên' (Employee Information), and a central form for 'Tài khoản đăng nhập' (Login Account). The form contains fields for 'Tên Đăng Nhập' (Login Name), 'Mật Khẩu' (Password), and 'Lập Lại Mật Khẩu' (Reset Password), along with a 'Nhân Viên' (Employee) dropdown. A 'Tìm kiếm' (Search) bar is at the top right. A list of 'Có quyền thao tác' (Operations Allowed) is on the right, including 'Thiết Lập', 'Mật Hàng', 'Nhà Cung Cấp', 'Khách Hàng', 'Nhập Xuất', 'Bán Hàng', 'Hợp Đồng', 'Công Nợ', 'Thu Chi', and 'Thống Kê'. At the bottom, there are buttons for 'Thêm' (Add), 'Lưu' (Save), and 'Sửa' (Edit), and a 'Lưu' (Save) button on the right. The status bar at the bottom shows the time '2:08:28 PM' and the date 'Ngày 01 tháng 06 năm 2014'.

Hình 3.9 Giao diện nghiệp vụ Tạo tài khoản và phân quyền

- *Danh sách các xử lý Giao diện nghiệp vụ Tạo tài khoản và phân quyền*

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Tìm kiếm	Khi người dùng thay đổi nội dung trong khung tìm kiếm.	
2	Hiển thị danh sách nhân viên	Khi giao diện quản lý người dùng được hiển thị	
3	Thêm người dùng	Khi người dùng chọn “Thêm người dùng”.	Khi chọn tính năng này các thông tin sẽ được xóa trắng để thêm thông tin vào.
4	Lưu người dùng	Khi người dùng chọn “Lưu”.	
5	Sửa	Khi người dùng chọn “Sửa”.	
6	Lưu phân quyền	Khi người dùng click vào “In danh sách hàng thiếu”.	

3.2.2 Quản lý xuất nhập

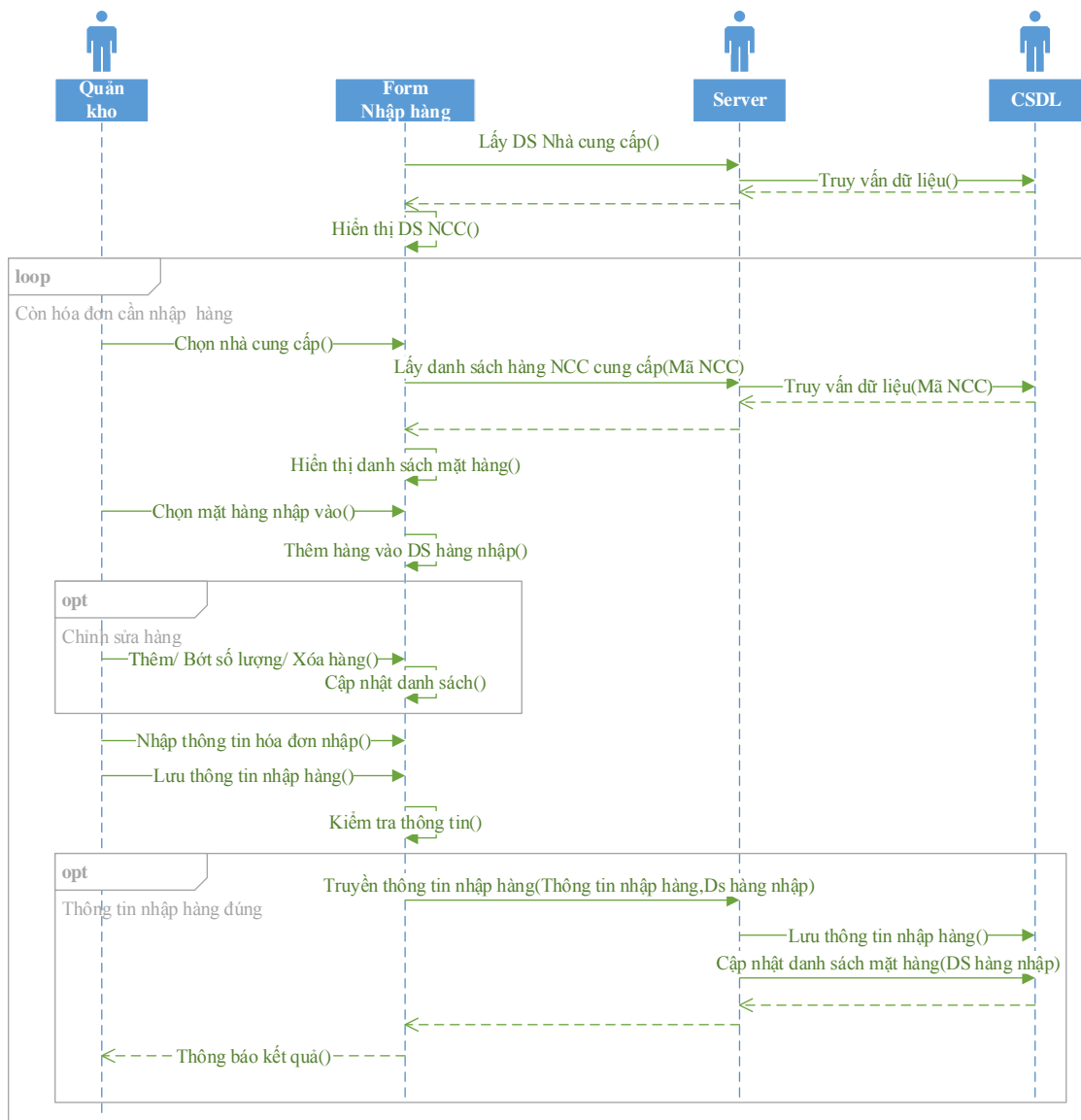


Hình 3.10 Chức năng Quản lý xuất nhập

3.2.2.1 Nhập hàng từ nhà cung cấp

- Nội dung: Lập phiếu nhập kho cho các hóa đơn nhập hàng.
- Các bảng dữ liệu sử dụng: Phiếu nhập, Chi tiết phiếu nhập, Nhà cung cấp, Mặt hàng.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.11 Sơ đồ tuần tự chức năng Nhập hàng từ nhà cung cấp

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

PHẦN MỀM QUẢN LÝ BÁN HÀNG

TRANG CHỦ THIẾT LẬP MẬT HÀNG NHÀ CUNG CẤP KHÁCH HÀNG **NHẬP - XUẤT** BÁN HÀNG HỢP ĐỒNG CÔNG NỢ THU - CHI THỐNG KÊ

Nhập hàng **Xuất hàng**

Từ ngày: Chọn ngày bắt đầu Đến ngày: Chọn ngày kết thúc Xem 1

Tìm kiếm: Nhập từ khóa tìm kiếm Thêm 2

Mã phiếu nhập	Nhà cung cấp	Ngày nhập	Chiết khấu	Thuế GTGT	Tổng tiền hàng	Người giao	Nhân viên lập
PNH0000001	Hà Tiên	5/15/2014	0.00 %	10.00 %	9,000,000 VNĐ	Quỳnh	
PNH0000002	Thanh thanh	5/15/2014	0.00 %	10.00 %	6,000,000 VNĐ		5

Danh sách nhập hàng x

9:43:13 PM Ngày 01 tháng 06 năm 2014

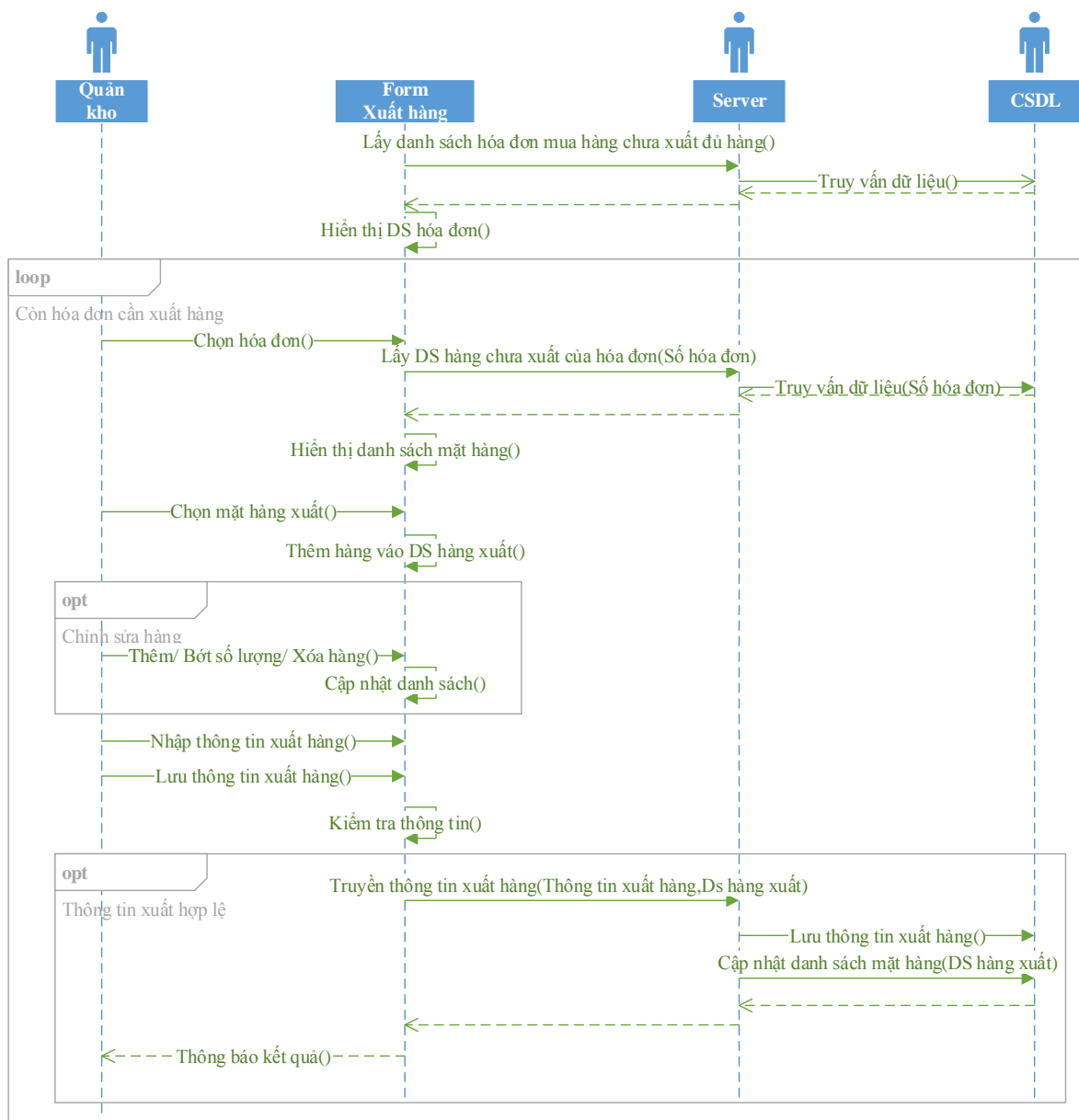
Hình 3.12 Giao diện nghiệp vụ Nhập hàng từ nhà cung cấp

- Danh sách các xử lý Giao diện nghiệp vụ Tạo tài khoản và phân quyền

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Xem danh sách nhập hàng	Sau khi người dùng chọn “Xem”.	
2	Thêm nhập hàng	Sau khi người dùng chọn “Thêm”.	
3	Tìm kiếm	Khi thay đổi thông tin trong khung tìm kiếm.	
4	Danh sách nhập hàng	Khi quản lý nhập hàng từ nhà cung cấp hiển thị.	
5	Chi tiết hàng nhập	Khi người dùng đúp chuột vào phiếu nhập.	

3.2.2.2 Xuất hàng theo hóa đơn

- Nội dung: Lập phiếu xuất hàng trong kho theo hóa đơn mua hàng của khách hàng.
- Các bảng dữ liệu sử dụng: Phiếu xuất, Chi tiết phiếu xuất, Mặt hàng, Khách hàng, Hóa đơn, Chi tiết hóa đơn.
- Sơ đồ tuần tự:



Hình 3.13 Sơ đồ tuần tự Chức năng Xuất hàng theo hóa đơn

- Giao diện chức năng:

Hình 3.14 Giao diện nghiệp vụ Xuất hàng theo hóa đơn

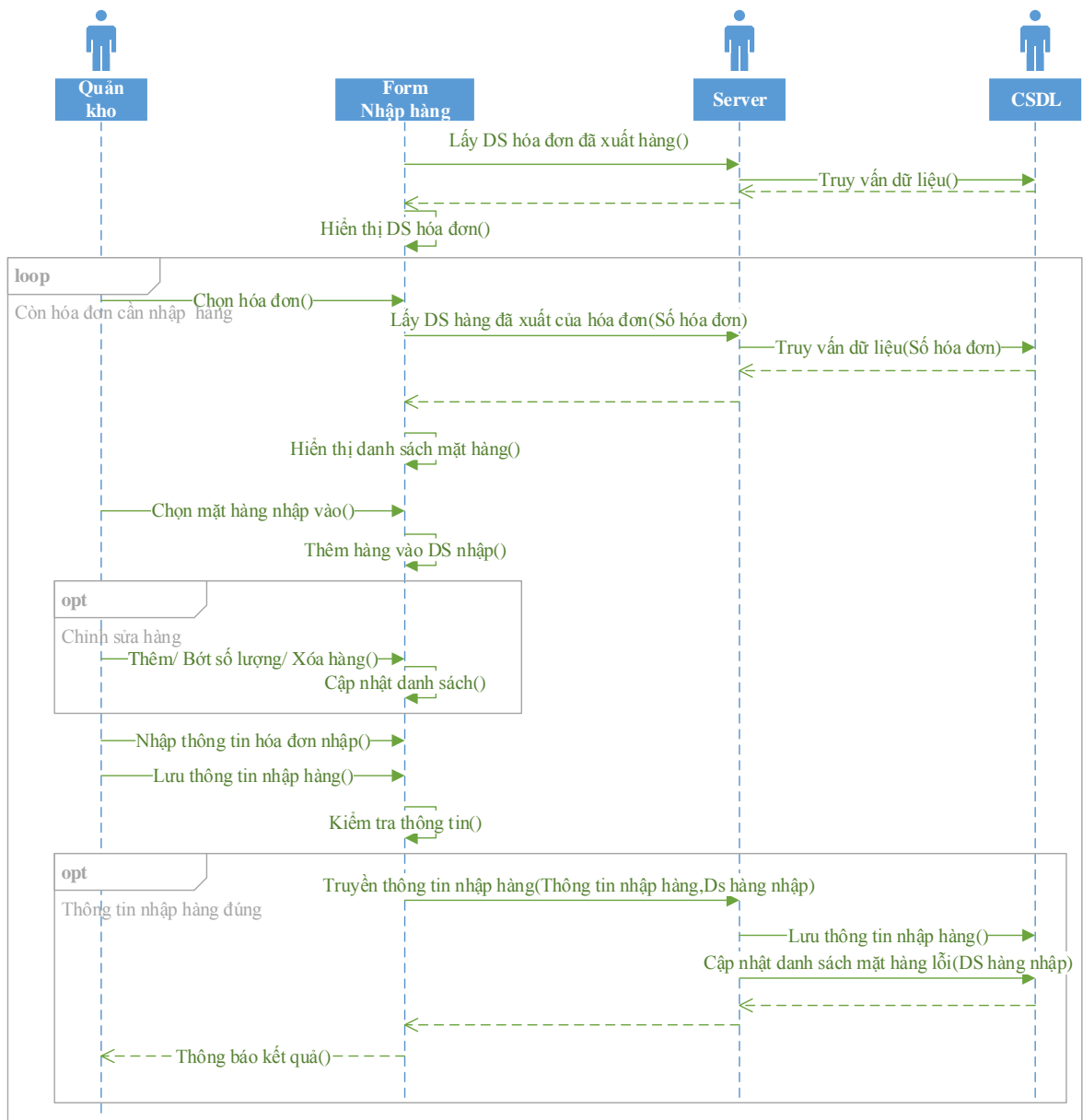
- *Danh sách các xử lý Giao diện nghiệp vụ Xuất hàng theo hóa đơn*

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Hiển thị danh sách hóa đơn	Khi thêm xuất hàng được hiển thị.	
2	Hiển thị mặt hàng theo hóa đơn	Sau khi người dùng chọn hóa đơn.	
3	Danh sách mặt hàng	Sau khi người dùng chọn mặt hàng từ danh sách mặt hàng của hóa đơn.	
4	Lưu phiếu xuất	Sau khi người dùng chọn “Lưu”.	

3.2.2.3 Nhập hàng lỗi từ khách hàng

- Nội dung: Lập phiếu nhập lại hàng bị lỗi đã xuất cho khách hàng.
- Các bảng dữ liệu sử dụng: Phiếu nhập, Chi tiết phiếu nhập, Khách hàng, Mặt hàng lỗi, Phiếu xuất, Chi tiết phiếu xuất, Hóa đơn, Chi tiết hóa đơn.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.15 Sơ đồ tuần tự chức năng Nhập hàng lỗi từ khách hàng

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

Hình 3.16 Giao diện nghiệp vụ Nhập hàng lỗi từ khách hàng

- Danh sách các xử lý Giao diện nghiệp vụ Nhập hàng lỗi từ khách hàng

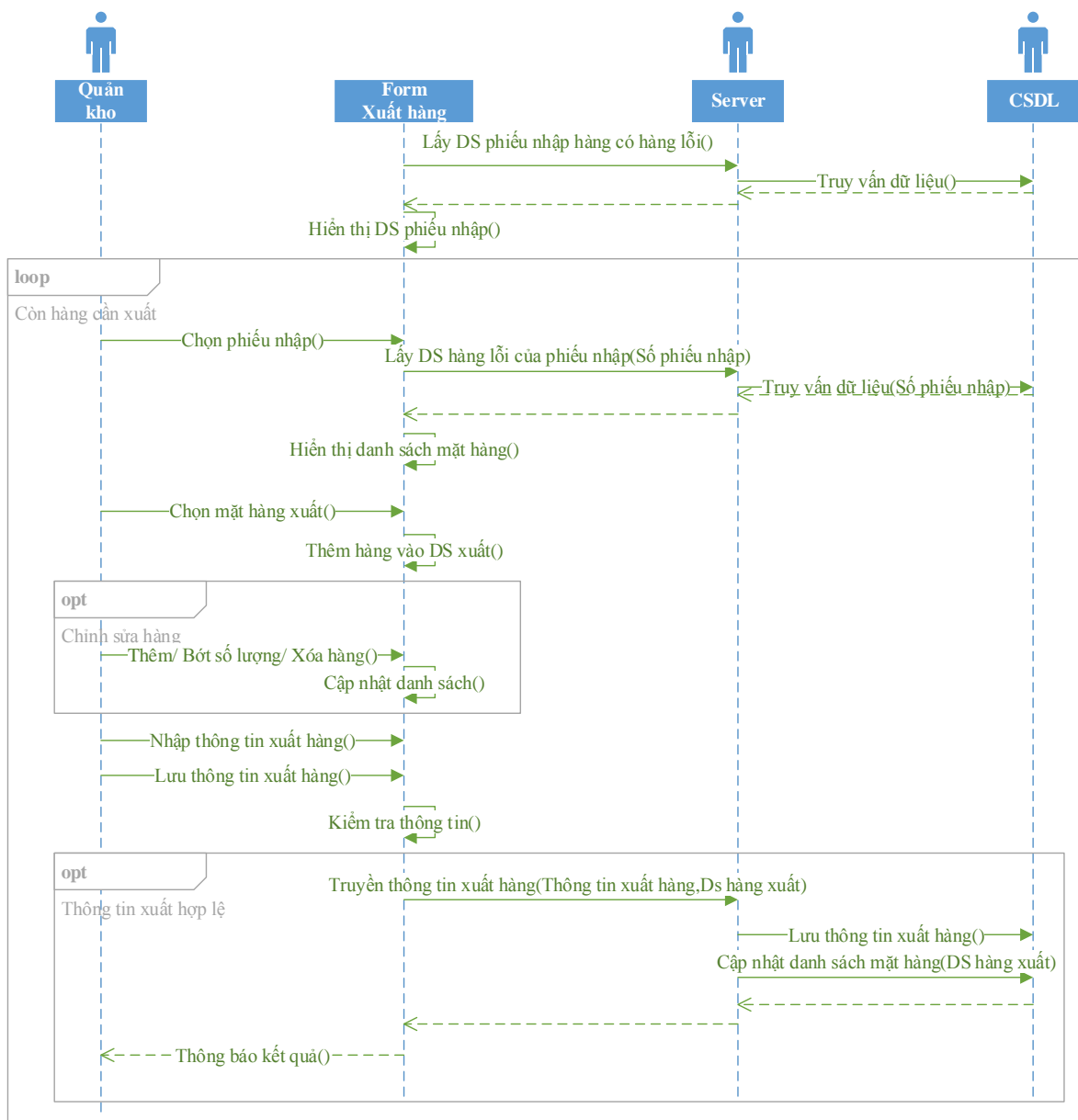
STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Hiển thị danh sách hóa đơn	Khi thêm nhập hàng lỗi được hiển thị.	
2	Hiển thị danh sách hàng nhập	Sau khi người dùng chọn hóa đơn.	
3	Hiển thị danh sách hàng đã chọn	Sau khi người dùng chọn hàng từ danh sách hàng nhập.	
7	Xuất phiếu trả hàng của khách hàng	Sau khi người dùng chọn “Xuất phiếu”.	
8	Lưu phiếu nhập	Sau khi người dùng chọn “Lưu”.	

3.2.2.4 Xuất hàng lỗi trả nhà cung cấp

- Nội dung: Lập phiếu xuất hàng bị lỗi đã xuất cho nhà cung cấp.
- Các bảng dữ liệu sử dụng: Phiếu xuất, Chi tiết phiếu xuất, Nhà cung cấp, Mặt hàng lỗi, Phiếu nhập, Chi tiết phiếu nhập.

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

- Sơ đồ tuần tự:



Hình 3.17 Sơ đồ tuần tự chức năng Xuất hàng lỗi trả nhà cung cấp

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

Hình 3.18 Giao diện nghiệp vụ Xuất hàng lỗi trả nhà cung cấp

- Danh sách các xử lý Giao diện nghiệp vụ Xuất hàng lỗi trả nhà cung cấp

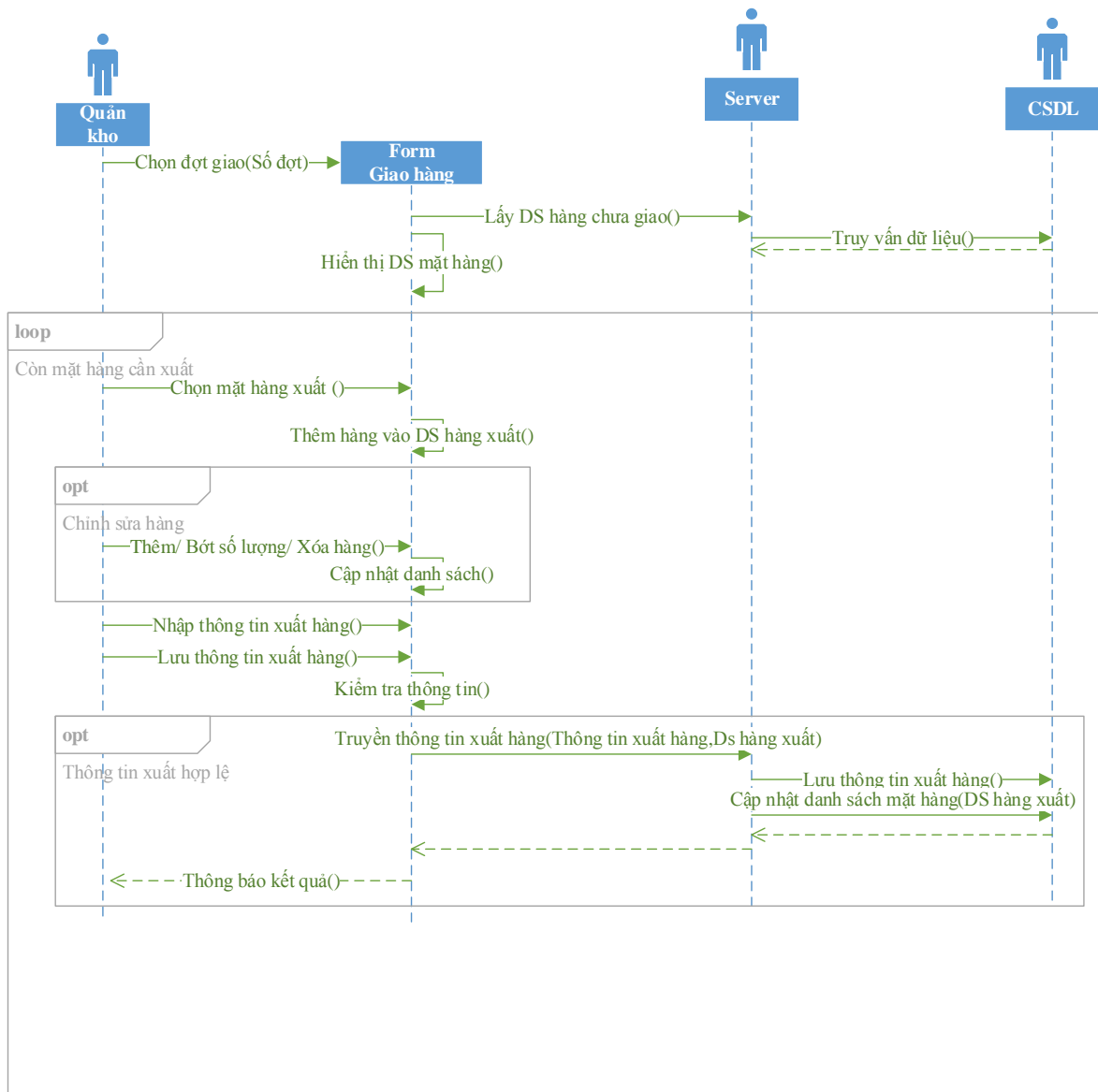
STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Danh sách nhà cung cấp	Khi xuất trả hàng nhà cung cấp hiển thị.	
2	Danh sách phiếu nhập	Sau khi người dùng chọn nhà cung cấp.	
3	Danh sách mặt hàng nhập	Sau khi người dùng chọn phiếu nhập.	
4	Danh sách mặt hàng xuất trả	Sau khi người dùng chọn mặt hàng từ danh sách mặt hàng nhập.	
5	Lưu phiếu xuất trả	Sau khi người dùng chọn “Lưu phiếu xuất trả”.	
6	Xuất phiếu	Sau khi người dùng chọn “Xuất phiếu”.	

3.2.2.5 Xuất hàng theo hợp đồng

- Nội dung: Lập phiếu giao hàng theo đợt giao đã thỏa thuận bên trong hợp đồng với khách hàng.

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

- Các bảng dữ liệu sử dụng: Phiếu giao, Chi tiết phiếu giao, Đợt giao, Chi tiết đợt giao. Mật hàng.
- Sơ đồ tuần tự:



Hình 3.19 Sơ đồ tuần tự chức năng Xuất hàng theo hợp đồng

- Giao diện chức năng:

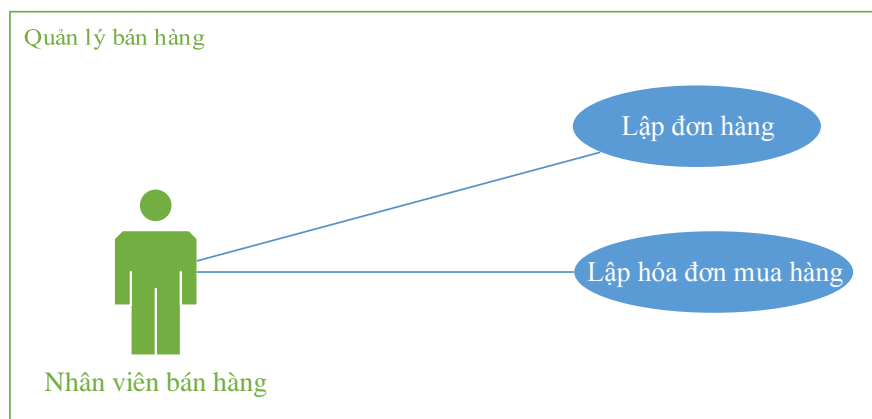
PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

Hình 3.20 Giao diện nghiệp vụ Xuất hàng theo hợp đồng

- Danh sách các xử lý Giao diện nghiệp vụ Xuất hàng theo hợp đồng

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Hiện thị danh sách mặt hàng	Khi thêm phiếu giao được hiển thị.	
2	Hiện thị danh sách nhân viên	Khi lập phiếu giao được hiển thị	
3	Danh sách mặt hàng của phiếu giao	Khi người dùng thêm hoặc hủy mặt hàng.	
4	Lưu phiếu giao	Khi người dùng chọn “Lưu”.	

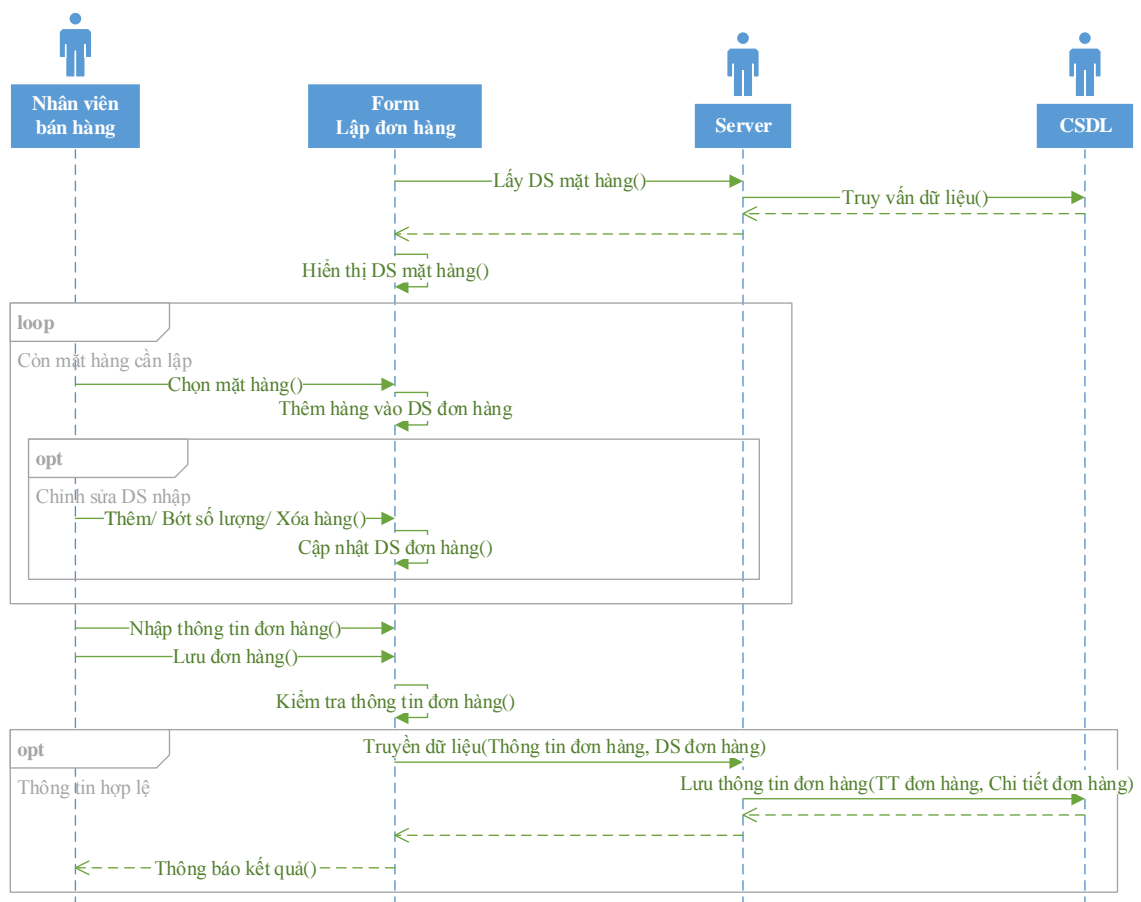
3.2.3 Quản lý bán hàng



Hình 3.21 Chức năng Quản lý bán hàng

3.2.3.1 Lập đơn hàng

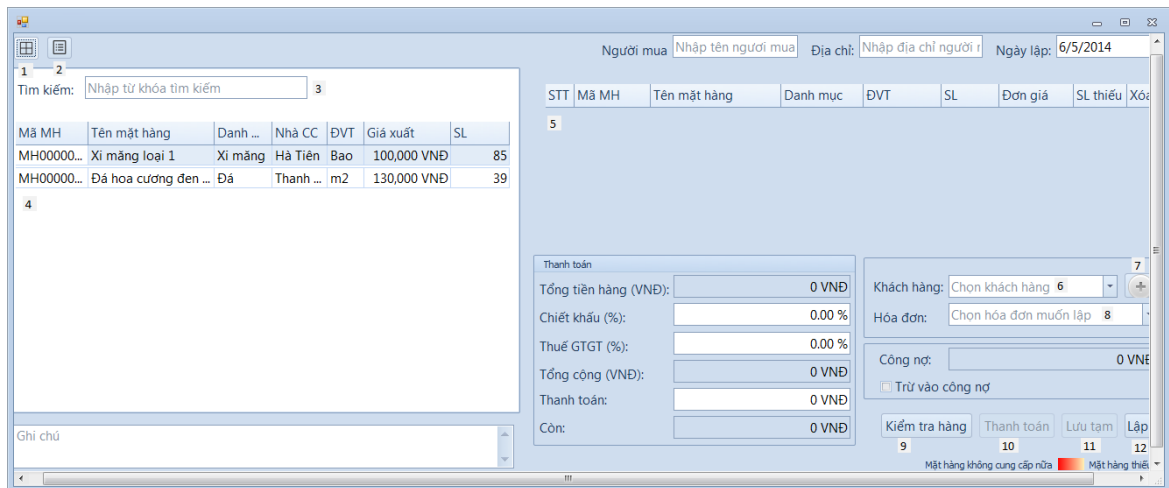
- Nội dung: Lưu lại thông tin khách mua hàng và danh sách mặt hàng mua, dùng để lưu tạm (chưa lập hóa đơn) khi tạm hết hàng hay khách hàng đặt hàng trước.
- Các bảng dữ liệu sử dụng: Đơn hàng, Chi tiết đơn hàng, Khách hàng, Mặt hàng.
- Sơ đồ tuần tự:



Hình 3.22 Sơ đồ tuần tự chức năng Lập đơn hàng

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.23 Giao diện nghiệp vụ Lập đơn hàng

- Danh sách các xử lý Giao diện nghiệp vụ Lập đơn hàng

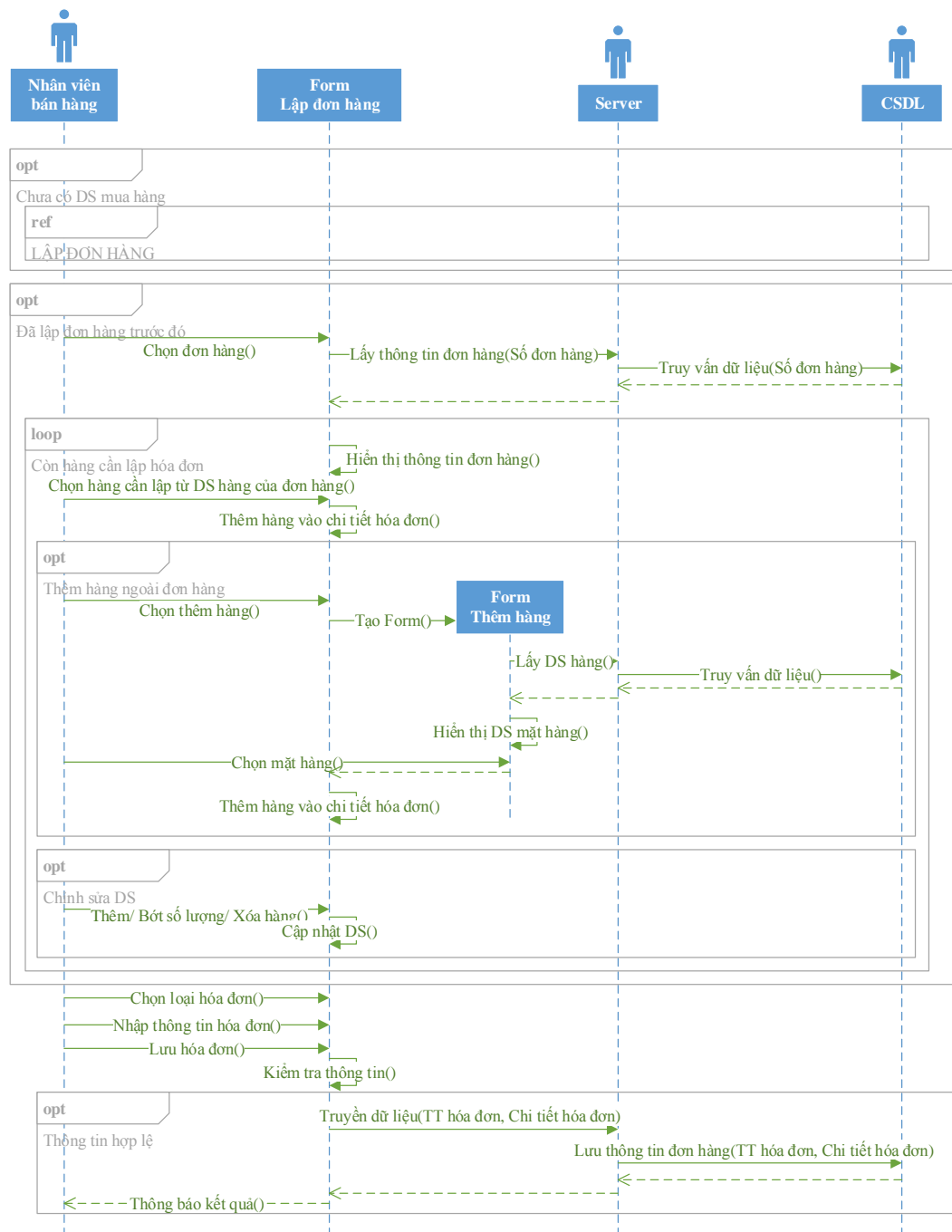
STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Hiển thị chi tiết danh sách sản phẩm	Khi người dùng chọn 	
2	Hiển thị danh sách sản phẩm	Khi người dùng chọn  hoặc lúc thêm đơn hàng hiển thị.	
3	Tìm kiếm sản phẩm	Khi thay đổi thông tin trong khung tìm kiếm.	
4	Hiển thị danh sách sản phẩm	Khi thêm đơn hàng hiển thị hoặc khi người dùng tìm kiếm.	
5	Hiển thị danh sách sản phẩm được chọn	Khi người dùng chọn sản phẩm từ danh sách.	
6	Hiển thị danh sách khách hàng	Khi thêm đơn hàng hiển thị.	
7	Thêm khách hàng	Khi người dùng chọn 	

8	Hiển thị danh sách hóa đơn	Khi thêm đơn hàng hiển thị.	
9	Kiểm tra hàng	Khi người dùng chọn “Kiểm tra hàng”.	
10	Thanh toán	Khi người dùng chọn “Thanh toán”.	
11	Lưu tạm đơn hàng	Khi người dùng chọn “Lưu tạm”.	
12	Lập mới	Khi người dùng chọn “Lập mới”.	

3.2.3.2 Lập hóa đơn

- Nội dung: Lập hóa đơn mua hàng cho khách hàng.
- Các bảng dữ liệu sử dụng: Hóa đơn, Chi tiết hóa đơn, Mặt hàng, Khách hàng.
- Sơ đồ tuần tự:

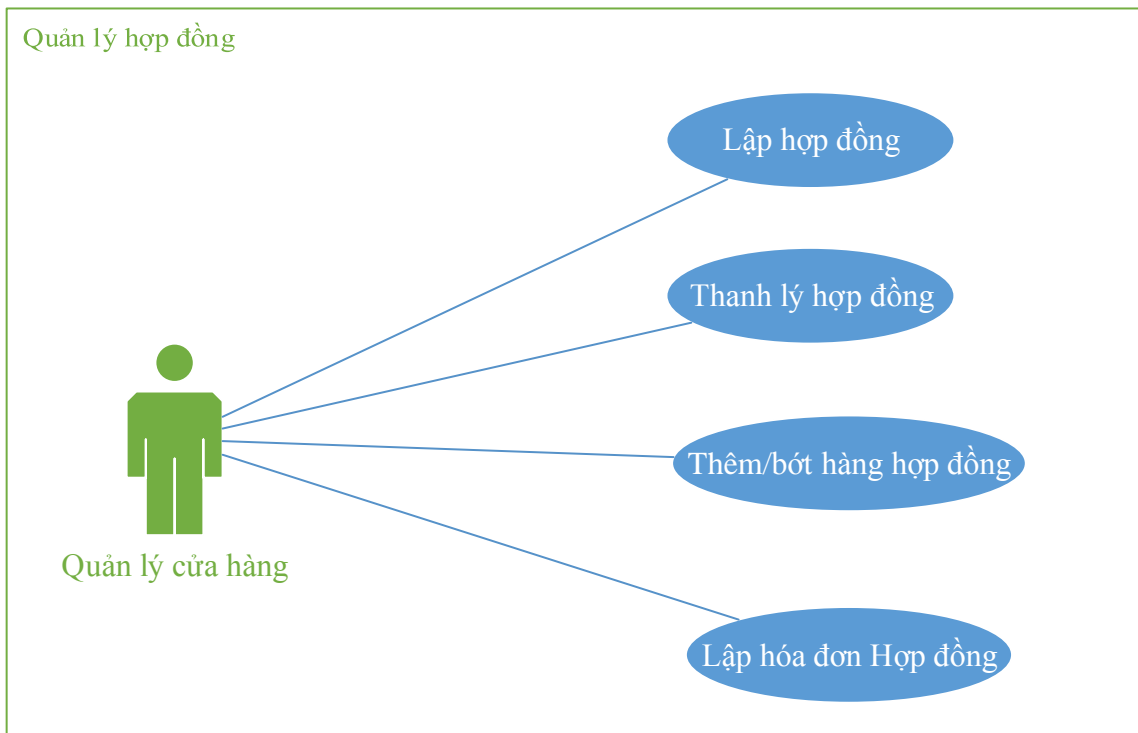
PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.24 Sơ đồ tuần tự chức năng Lập hóa đơn

- Giao diện chức năng: Giống với chức năng lập đơn hàng.

3.2.4 Quản lý hợp đồng

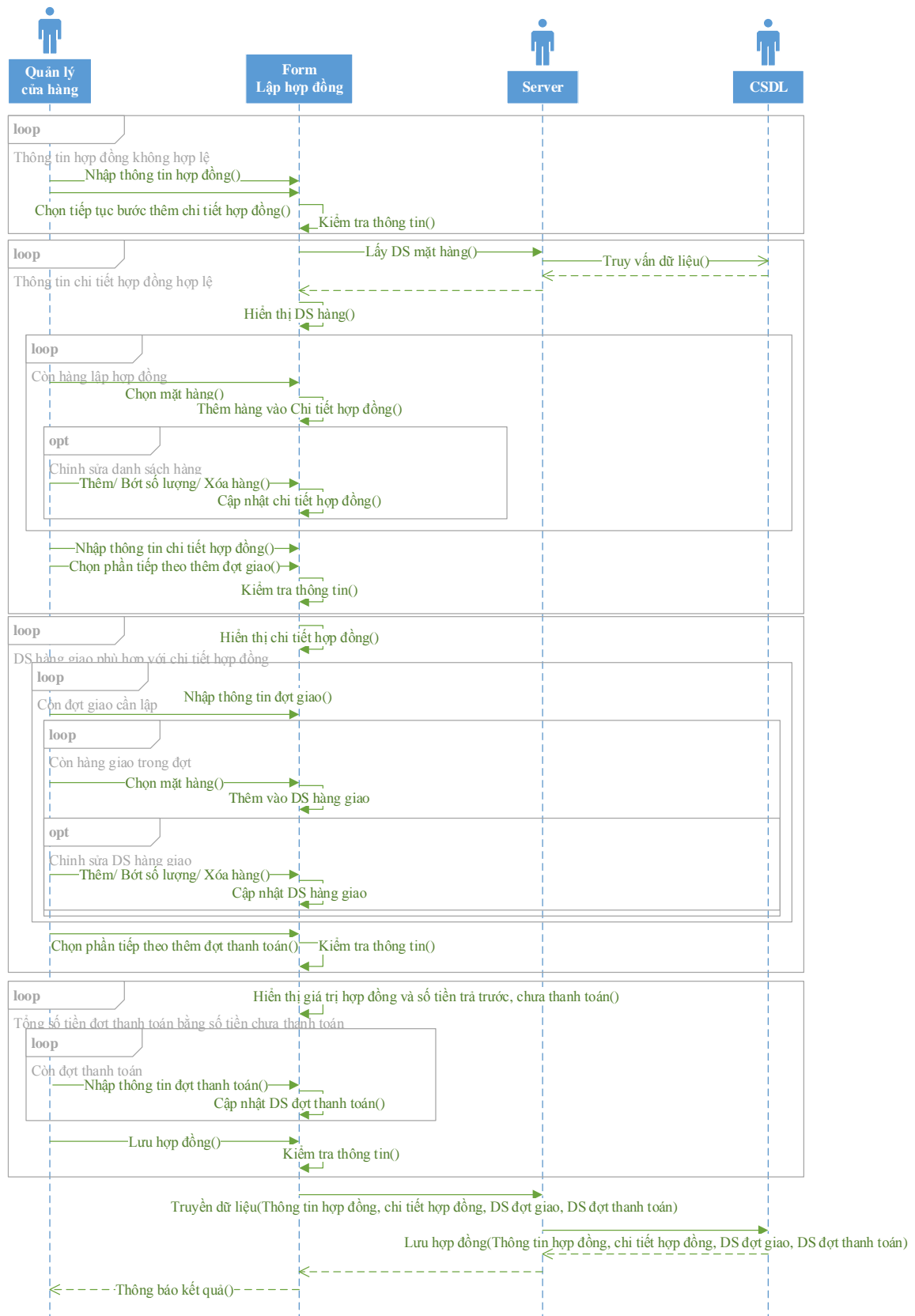


Hình 3.25 Chức năng Quản lý hợp đồng

3.2.4.1 Lập hợp đồng

- Nội dung: Lưu trữ thông tin kí kết hợp đồng gồm thông tin khách hàng, thông tin hợp đồng, chi tiết hợp đồng, thông tin các đợt giao và các đợt thanh toán hợp đồng.
- Các bảng dữ liệu sử dụng: Hợp đồng, Chi tiết hợp đồng, Đợt giao, Chi tiết đợt giao, Đợt thanh toán, Mặt hàng, Khách hàng.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.26 Sơ đồ tuần tự chức năng Lập hợp đồng

- Giao diện chức năng:

THÊM HỢP ĐỒNG

Ngày lập: 6/8/2014

Khách hàng: Chon khách hàng 1

Đại diện khách hàng:

SĐT người đại diện:

Nhân viên phụ trách: Chon đại diện cửa hàng

Ngày kết thúc:

Ngày thanh lý:

Quay lại Kế tiếp Kết thúc

THÊM HỢP ĐỒNG

5 6

7

8

STT	Mã MH	Tên mặt hàng	Danh mục	ĐVT	SL	Đơn giá	Xóa
9							

Thanh toán

Tổng tiền hàng (VNĐ): 0 VNĐ

Chiết khấu (%): 0.00 %

Thuế GTGT (%): 0.00 %

Tổng cộng (VNĐ): 0 VNĐ

10 Quay lại Kế tiếp 10 Kết thúc

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

THÊM HỢP ĐỒNG

Mã MH	Tên mặt hàng	ĐVT	Giá xuất	Số lượng
MH000000...	Xi măng loại 1	Bao	100,000 VNĐ	1

12

Số đợt	Ngày giao	Địa chỉ giao	Cách thức giao	Xóa
THÊM ĐỢT				
1	6/11/2014			

13

STT	Mã MH	Tên mặt hàng	Danh mục	Đơn Vị Tính	SoLuong	Đơn giá	Xóa
1	MH000000001	Xi măng loại 1	Xi măng	Bao	0	100,000 VNĐ	

14

15 Quay lại Kế tiếp 16 Kết thúc

THÊM HỢP ĐỒNG

Thanh toán

Giá trị hợp đồng: 100,000 VNĐ

Thanh toán trước: 50,000 VNĐ

Còn lại: 50,000 VNĐ


Số đợt	Ngày thanh toán	Số tiền thanh toán	Xóa
THÊM ĐỢT			
1	6/11/2014	50,000 VNĐ	

17



18 Quay lại Kế tiếp 19 Kết thúc

Hình 3.27 Giao diện nghiệp vụ Lập hợp đồng

- Danh sách các xử lý Giao diện nghiệp vụ Lập hợp đồng

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Hiển thị danh sách khách hàng	Khi thêm thông tin hợp đồng hiển thị.	
2	Thêm khách hàng	Khi người dùng chọn 	

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỦA HÀNG VẬT LIỆU XÂY DỰNG

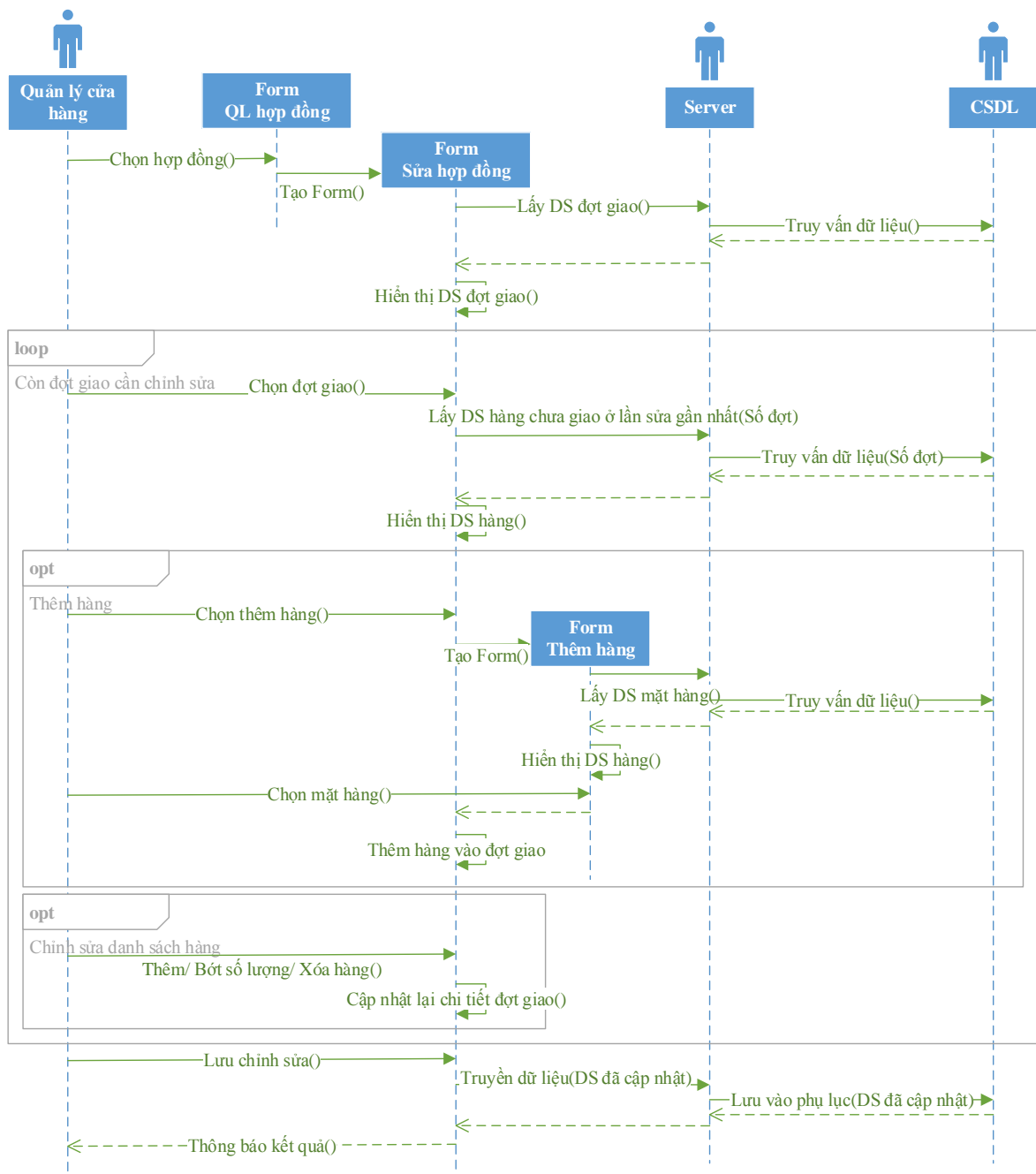
3	Hiển thị nhân viên phụ trách	Khi thêm thông tin hợp đồng hiển thị.	
4	Hiển thị chi tiết hợp đồng cần thêm	Khi người dùng chọn “Kế tiếp”.	
5	Hiển thị chi tiết danh sách mặt hàng	Khi người dùng chọn 	
6	Hiển thị danh sách mặt hàng	Khi người dùng chọn 	
7	Tìm kiếm mặt hàng	Khi thay đổi thông tin trong khung tìm kiếm.	
8	Hiển thị danh sách mặt hàng	Khi chi tiết hợp đồng được hiển thị hoặc khi tìm kiếm.	
9	Hiển thị mặt hàng được chọn	Khi người dùng chọn mặt hàng.	
10	Quay lại thông tin hợp đồng	Khi người dùng chọn “Quay lại”.	
11	Thêm thông tin được giao	Khi người dùng chọn “Kế tiếp”.	
12	Hiển thị danh sách mặt hàng của hợp đồng	Khi thêm thông tin được giao hiển thị.	
13	Danh sách đợt giao	Khi người dùng thêm đợt giao.	

14	Chi tiết đợt giao	Khi người dùng chọn đợt giao.	
15	Quay lại chi tiết hợp đồng	Khi người dùng chọn “Quay lại”.	
16	Thêm thông tin thanh toán	Khi người dùng chọn “Kế tiếp”.	
17	Hiện thị danh sách đợt thanh toán	Khi người dùng thêm đợt thanh toán.	
18	Quay lại danh sách đợt giao	Khi người dùng chọn “Quay lại”.	
19	Hoàn thành thêm hợp đồng	Khi người dùng chọn “Kết thúc”.	

3.2.4.2 Thêm bớt/hàng hợp đồng

- Nội dung: Cho phép khách hàng thêm hay bớt hàng trong hợp đồng theo đợt giao. Danh sách hàng được chỉnh sửa lưu tại bảng Phụ lục.
- Các bảng dữ liệu sử dụng: Đợt giao, Chi tiết đợt giao, Phụ lục, Mặt hàng
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.28 Sơ đồ tuần tự chức năng Thêm/bớt hàng hợp đồng

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

STT	Mã MH	Tên mặt hàng	ĐVT	SL	Đơn giá	Tăng
1	MH00000001	Xi măng Holcim đa dụng	Bao	20	79,000 V...	0
2	MH00000002	Xi măng Fico PCB40	Bao	20	74,000 V...	0
3	MH00000003	Vicem Hà Tiên	Bao	20	75,000 V...	0
4	MH00000009	Gạch ống 8x18 tuyne Phan Thiết (ben)	Viên	100	850 VNĐ	0
5	MH00000011	Gạch ống 8x18 tuyne TQC Bình Thuận (ben)	Viên	100	750 VNĐ	0
6	MH00000014	Gạch thẻ 8x18 tuyne Trung Nguyên (ben)	Viên	100	1,100 VNĐ	0

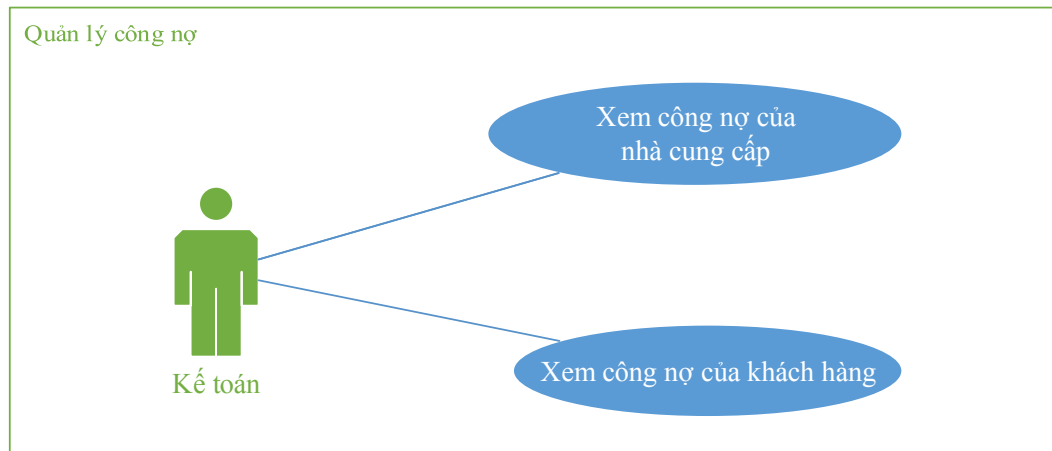
STT	Mã MH	Tên mặt hàng	ĐVT	SL	Ghi chú
	MH00000001	Xi măng Holcim đa dụng	Bao	20	
	MH00000002	Xi măng Fico PCB40	Bao	20	

Hình 3.29 Giao diện nghiệp vụ Thêm/bớt hàng hợp đồng

- Danh sách các xử lý giao diện Thêm/bớt hàng hợp đồng

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Hiển thị danh sách mặt hàng trong hợp đồng	Khi sửa hợp đồng hiển thị.	
2	Hiển thị danh sách đợt giao	Khi sửa hợp đồng hiển thị.	
3	Hiển thị danh sách mặt hàng có trong đợt giao	Khi người dùng chọn đợt giao.	
4	Thêm hàng	Khi người dùng chọn “Thêm hàng”.	

3.2.5 Quản lý công nợ

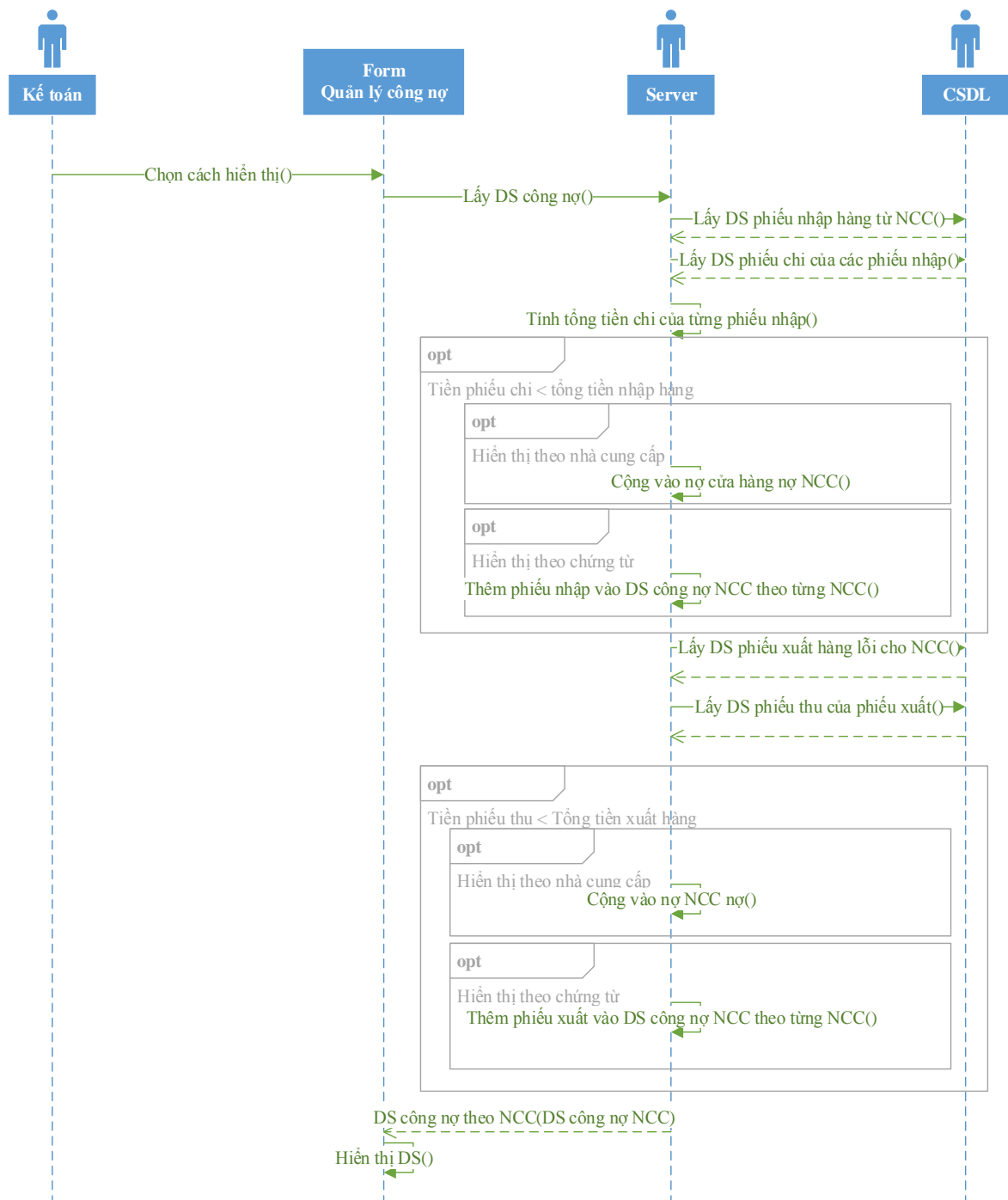


Hình 3.30 Chức năng Quản lý công nợ

3.2.5.1 Xem sách công nợ của nhà cung cấp

- Nội dung: Lấy danh sách nhà cung cấp mà cửa hàng nợ tiền hàng và số nợ ở mỗi nhà cung cấp.
- Các bảng dữ liệu sử dụng: Phiếu xuất, Phiếu nhập, Phiếu thu, Phiếu chi, Nhà cung cấp.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.31 Sơ đồ tuần tự chức năng Lập danh công nợ của hàng nhà cung cấp

- Giao diện chức năng:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG

PHẦN MỀM QUẢN LÝ BÁN HÀNG

TRANG CHỦ THIẾT LẬP MẶT HÀNG NHÀ CUNG CẤP KHÁCH HÀNG NHẬP - XUẤT BÁN HÀNG HỢP ĐỒNG **CÔNG NỢ** THU - CHI THỐNG KÊ

Khách hàng Nhà cung cấp

☒ Phân bổ theo nhà cung cấp 1 ☐ Phân bổ theo các chứng từ 2

Tim kiếm: 3 Cập nhật danh sách 4

Mã nhà cung cấp	Tên nhà cung cấp	Tổng tiền NCC nợ	Cửa hàng còn nợ
NCC0000002	Thanh thanh	0 VNĐ	6,600,500 VNĐ

Danh sách phiếu nhập hàng của hàng chưa thanh toán đủ

Mã phiếu nhập	Ngày nhập	Tổng giá trị	Đã thanh toán	Chưa thanh toán
PNH0000002	5/15/2014	6,600,500 VNĐ	0 VNĐ	6,600,500 VNĐ

5

Công nợ theo nhà cung cấp x

11:46:53 AM Ngày 08 tháng 06 năm 2014

PHẦN MỀM QUẢN LÝ BÁN HÀNG

TRANG CHỦ THIẾT LẬP MẶT HÀNG NHÀ CUNG CẤP KHÁCH HÀNG NHẬP - XUẤT BÁN HÀNG HỢP ĐỒNG **CÔNG NỢ** THU - CHI THỐNG KÊ

Khách hàng Nhà cung cấp

☐ Phân bổ theo nhà cung cấp ☒ Phân bổ theo các chứng từ

Phiếu nhập hàng Phiếu xuất hàng trả lỗi

Tim kiếm: Nhập từ khóa tìm kiếm 6 Cập nhật danh sách 7

Mã phiếu nhập	Ngày nhập	Mã NCC	Tên NCC	Tổng giá trị	Đã thanh toán	Chưa thanh toán
PNH0000002	5/15/2014	NCC0000002	Thanh thanh	6,600,500 VNĐ	0 VNĐ	6,600,500 VNĐ

8

Công nợ theo nhà cung cấp x

11:48:20 AM Ngày 08 tháng 06 năm 2014

PHẦN MỀM QUẢN LÝ BÁN HÀNG

TRANG CHỦ THIẾT LẬP MẶT HÀNG NHÀ CUNG CẤP KHÁCH HÀNG NHẬP - XUẤT BÁN HÀNG HỢP ĐỒNG **CÔNG NỢ** THU - CHI THỐNG KÊ

Khách hàng Nhà cung cấp

☐ Phân bổ theo nhà cung cấp ☒ Phân bổ theo các chứng từ

Phiếu nhập hàng Phiếu xuất hàng trả lỗi

Tim kiếm: Nhập từ khóa tìm kiếm 9 Cập nhật danh sách 10

Mã phiếu xuất	Ngày xuất	Mã phiếu nhập ...	Mã NCC	Tên NCC	Tổng giá trị hàng	Đã hoàn trả	Chưa hoàn trả
---------------	-----------	-------------------	--------	---------	-------------------	-------------	---------------

11

Công nợ theo nhà cung cấp x

11:49:31 AM Ngày 08 tháng 06 năm 2014

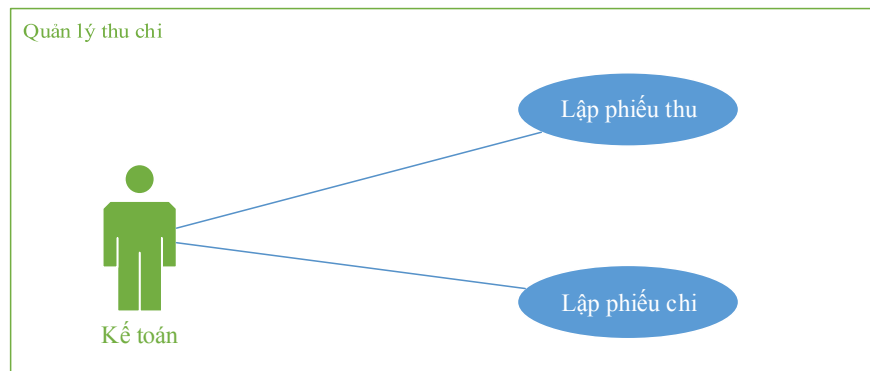
Hình 3.32 Giao diện nghiệp vụ Xem công nợ nhà cung cấp

- *Danh sách các xử lý Giao diện nghiệp vụ Xem công nợ nhà cung cấp*

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Xem công nợ phân bổ theo nhà cung cấp	Khi người dùng chọn “Phân bổ theo nhà cung cấp”.	
2	Xem công nợ phân bổ theo chứng từ	Khi người dùng chọn “Phân bổ theo chứng từ”.	
3	Tìm kiếm	Khi thay đổi thông tin trong khung tìm kiếm.	
4	Cập nhật danh sách	Khi người dùng chọn “Cập nhật danh sách”.	
5	Hiển thị danh sách công nợ theo nhà cung cấp	Khi người dùng chọn “Phân bổ theo nhà cung cấp”.	
6	Tìm kiếm	Khi thay đổi thông tin trong khung tìm kiếm.	
7	Cập nhật danh sách	Khi người dùng chọn “Cập nhật danh sách”.	
8	Hiển thị danh sách công nợ theo phiếu nhập	Khi người dùng chọn tab “Phiếu nhập”.	
9	Tìm kiếm	Khi thay đổi thông tin trong khung tìm kiếm.	
10	Cập nhật danh sách	Khi người dùng chọn “Cập nhật danh sách”.	

11	Hiển thị danh sách công nợ theo phiếu xuất hàng trả lỗi	Khi người dùng chọn tab “Phiếu xuất hàng trả lỗi”.	
----	---	--	--

3.2.6 Quản lý thu chi

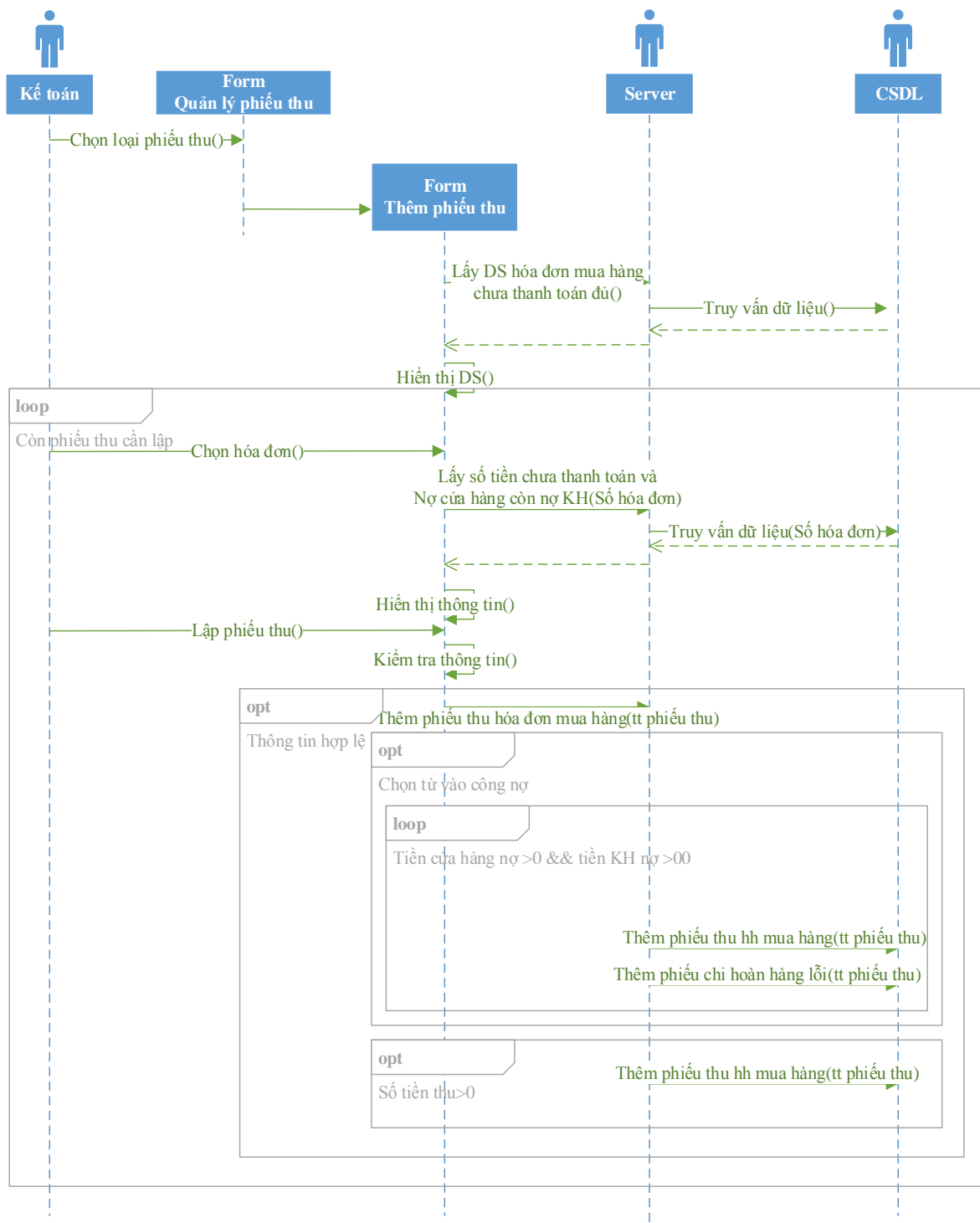


Hình 3.33 Chức năng Quản lý thu chi

3.2.6.1 Lập phiếu thu tiền

- Nội dung: Lập phiếu thu tiền cho cửa hàng. Các loại phiếu thu: Thu tiền hóa đơn mua hàng của khách hàng, thu tiền hoàn hàng lỗi của cửa hàng, thu thanh toán hợp đồng.
- Các bảng dữ liệu sử dụng: Phiếu thu, Hóa đơn, Phiếu xuất, Phiếu nhập, Biên bản thanh lý, Khách hàng, Nhà cung cấp.
- Sơ đồ tuần tự:

PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.34 Sơ đồ tuần tự chức năng Lập phiếu thu tiền hóa đơn mua hàng

- Giao diện chức năng:

Thêm phiếu thu: 1

Người nộp

Phiếu thu hóa đơn mua hàng

Phiếu thu hoàn trả hàng lỗi

Phiếu thu thanh toán hợp đồng

Hóa đơn: [Chọn hóa đơn] 2

Người nộp:

Tiền phải trả: 0 VNĐ

Cửa hàng còn nợ: 0 VNĐ

☐ Trừ vào nợ cửa hàng

Số tiền nộp: 0 VNĐ

Còn: 0 VNĐ

Lý do thu: Nhập lý do thu

Ghi chú: Nhập thông tin ghi chú

Lưu 3

Hình 3.35 Giao diện nghiệp vụ Lập phiếu thu tiền hóa đơn mua hàng

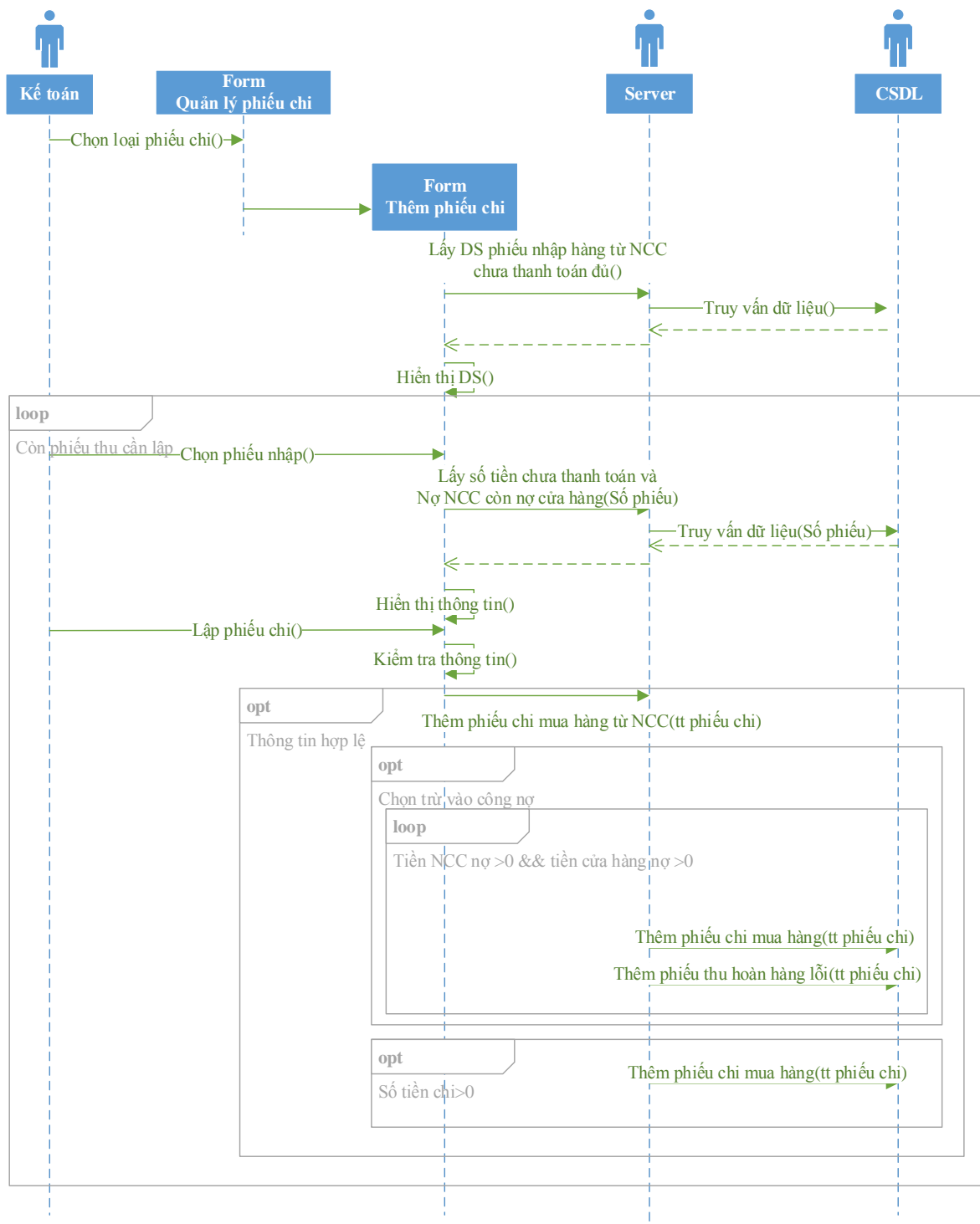
Danh sách các xử lý Giao diện nghiệp vụ Lập phiếu thu tiền hóa đơn mua hàng

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Chọn loại phiếu thu	Khi người dùng chọn loại phiếu thu.	
2	Hiển thị danh sách hóa đơn	Khi thêm phiếu thu được hiển thị.	
3	Lưu phiếu thu	Khi người dùng chọn “Lưu”.	

3.2.6.2 Lập phiếu chi tiền


- Nội dung: Lập phiếu chi tiền cho cửa hàng. Các loại phiếu thu: Chi tiền mua hàng từ nhà cung cấp, chi tiền hoàn hàng lỗi cho khách hàng, chi tiền bồi thường hợp đồng.
- Các bảng dữ liệu sử dụng: Phiếu chi, Hóa đơn, Phiếu xuất, Phiếu nhập, Biên bản thanh lý, Khách hàng, Nhà cung cấp.
- Sơ đồ tuần tự:

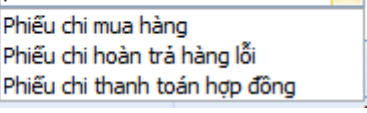
PHẦN MỀM QUẢN LÝ GIAO DỊCH CỬA HÀNG VẬT LIỆU XÂY DỰNG



Hình 3.36 Sơ đồ tuần tự chức năng Lập phiếu chi tiền mua hàng từ nhà cung cấp

- Giao diện chức năng:

Thêm phiếu thu: 


 Phiếu chi mua hàng
 Phiếu chi hoàn trả hàng lỗi
 Phiếu chi thanh toán hợp đồng

Người nhận:

Tiền phải trả:

Nhà cung cấp nợ:


☐ Trừ vào nợ của nhà cung cấp

Số tiền chi:

Còn:

Lý do chi:

Ghi chú:

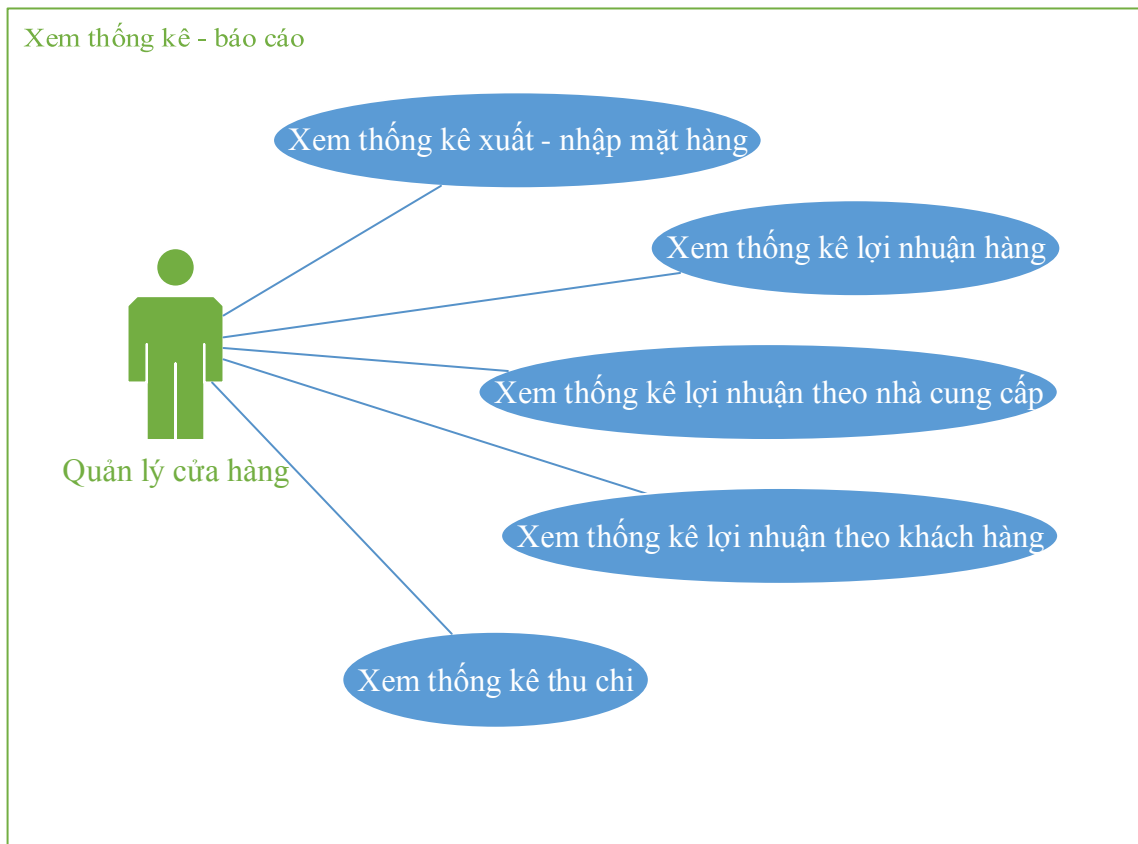


Hình 3.37 Giao diện nghiệp vụ Lập phiếu chi tiền mua hàng từ nhà cung cấp

- Danh sách các xử lý Giao diện nghiệp vụ Lập phiếu chi tiền mua hàng từ nhà cung cấp

STT	Tên xử lý	Điều kiện gọi thực hiện	Ghi chú
1	Chọn loại phiếu chi	Khi người dùng chọn loại phiếu chi.	
2	Hiển thị danh sách phiếu nhập	Khi thêm phiếu chi được hiển thị.	
3	Lưu phiếu chi	Khi người dùng chọn “Lưu”.	

3.2.7 Xem thống kê – báo cáo



Hình 3.38 Chức năng Xem thống kê – báo cáo

THỐNG KÊ

Thống kê mặt hàng Thống kê thu chi Thống kê lợi nhuận hàng Thống kê lợi nhuận theo khách hàng Thống kê lợi nhuận theo NCC

File View Background

Tìm kiếm

Từ ngày: 6/1/2014

Đến ngày: 7/5/2014

Xem thống kê

THÔNG KÊ HÀNG

Từ ngày 6/1/2014 đến ngày 7/5/2014

ST T	Mã MH	Tên mặt hàng	ĐVT	Nhập	Xuất	Tồn trước	Thêm/ bớt (*)	Hủy	Tổng MH	Hàng lỗi	Khách hàng trả	Trả NCC	Tồn lỗi trước	Tổng MH lỗi
Danh mục: Xi măng														
1	MH00000003	Vicem Hà Tiên	Bao	80	10	0	0	0	70	10	10	0	0	10
2	MH00000002	Xi măng Fico PCB40	Bao	50	10	0	0	0	40	10	10	0	0	10
3	MH00000001	Xi măng Holcim đa dụng	Bao	130	10	0	0	0	100	20	10	10	0	20
Tổng				260	30	0	0	0	210	40	30	10	0	40

Page 1 of 5

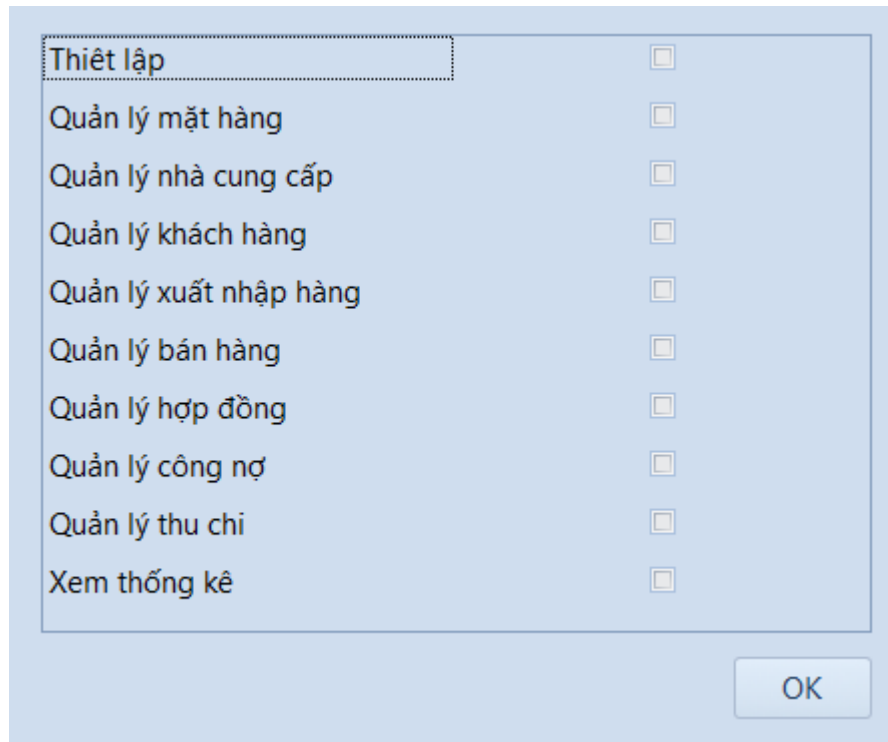
Thống kê hàng hóa

Hình 3.39 Giao diện nghiệp vụ thống kê hàng hóa

3.3 Một số tính năng hỗ trợ

3.3.1 Chọn giao diện làm việc khi bắt đầu

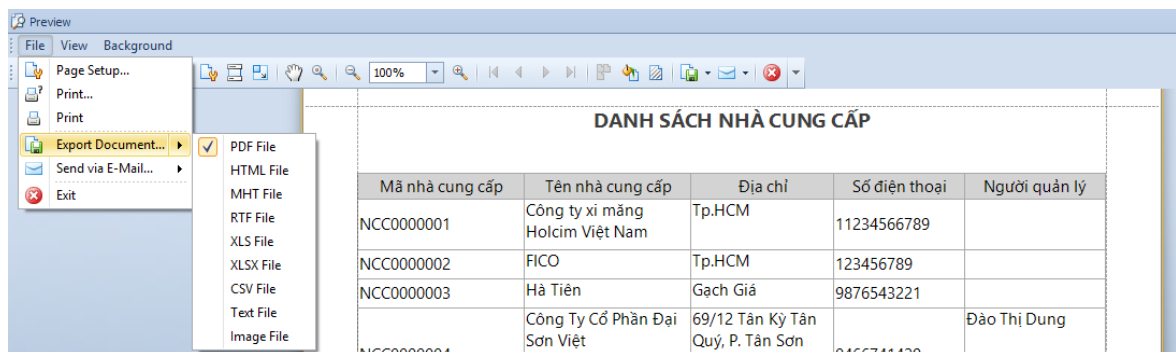
- Người dùng có thể bỏ bớt giao diện khi không cần dùng đến.



Hình 3.40 Giao diện chọn màn hình làm việc

3.3.2 Cho phép xuất các loại giấy tờ theo nhiều loại file

- Một tính năng hỗ trợ từ DevExpress cho phép xuất ra nhiều dạng file: pdf, xlsx, html,...



Hình 3.41 Giao diện In giấy tờ

3.3.3 Hiện thị trạng thái hàng bằng màu sắc

- Cho phép người dùng kiểm tra xem hàng có còn trong kho hay không khi lập đơn hàng.

STT	Mã MH	Tên mặt hàng	Danh mục	ĐVT	SL	Đơn giá	SL thiếu	Xóa
1	MH000000...	Cát đúc (Lagi/ Bắc Bình)	Cát	m3	10	310,000 V...	0	<input type="checkbox"/>
2	MH000000...	Sắt p 18 Miền Nam (V)	Sắt	cây	10	340,500 V...	10	<input type="checkbox"/>
3	MH000000...	Gạch ống 9x19 tuynel Phan Thiết (ben)	Gạch	Viên	100	1,000 VND	0	<input type="checkbox"/>
4	MH000000...	Gạch thẻ 8x18 hoffman Trường Thành (ben)	Gạch	Viên	300	990 VND	300	<input type="checkbox"/>

Mặt hàng không cung cấp nữa ■ Mặt hàng thiếu ■

Hình 3.42 Kiểm tra trạng thái hình

3.1 Kết luận

Phần mềm đã đưa được công nghệ mới vào việc xây dựng chương trình, giúp cho chương trình tận dụng được các máy tính có cấu hình cao trên thị trường hiện nay, giao tiếp với cơ sở dữ liệu nhanh chóng, chính xác, đem lại giao diện thân thiện cho người dùng.

Phần mềm cung cấp các chức năng cần thiết cho giao dịch của một cửa hàng vật liệu xây dựng. Đồng thời hoàn thành được các mục tiêu về mở rộng chương trình đã đề ra. Ngoài ra, phần mềm còn cung cấp thêm các chức năng hỗ trợ thêm cho người dùng.

PHẦN 3: TỔNG KẾT

Ưu điểm:

- Tìm hiểu và nắm được những kiến thức cơ bản của công nghệ .NET Framework 4.5 và Entity Framework 5.0.
- Có khả năng vận dụng được các công nghệ đã tìm hiểu vào thực tế.
- Chương trình có giao diện dễ nhìn, dễ thao tác cho người dùng.
- Các tính năng quản lý được thể hiện rõ ràng và thực hiện tốt.
- Chức năng thống kê được phần mềm thể hiện trực quan.
- Bảo mật tài khoản.
- Sao lưu, phục hồi dữ liệu dễ dàng khi sau khi mất, hư hỏng dữ liệu.
- Khắc phục được các khuyết điểm của phần mềm đã thực hiện ở Tiểu luận và mở rộng hơn:
 - + Mở rộng cho nhiều đơn vị sử dụng.
 - + Đồng bộ giao diện.
 - + Thông tin sản phẩm đầy đủ hơn.
 - + Cho phép làm việc với Service.
 - + Có giao diện Web hỗ trợ xem sản phẩm.
- Áp dụng công nghệ mới: Sencha Touch 2.0.
- Kết hợp nhiều công nghệ với nhau: Windows Communication Foundation, Windows Form, Website.

Hạn chế:

- Do thời gian và khả năng có hạn nên không thể đi sâu để tìm hiểu công nghệ, kiến thức đã tìm hiểu được trình bày một cách ngắn gọn, xúc tích mà người đọc có thể có một cái nhìn tổng quan về hai công nghệ này chứ chưa trình bày đầy đủ tuyệt đối.
- Website chưa cung cấp đặt hàng online.

TỔNG KẾT

- Phần mềm chưa sử dụng thật trong thực tế nên thiếu phản hồi khách quan.

Hướng phát triển:

- Hoàn chỉnh Website cho phép đặt hàng online.
- Đưa sản phẩm ra thị trường để nhận phản hồi và khắc phục những sai sót.
Từ đó thương mại hóa sản phẩm.
- Tiếp tục nghiên cứu các phiên bản mới hơn của công nghệ và vận dụng vào để cải tiến phần mềm.

Tài liệu tham khảo

- [1] John Paul Mueller, *Microsoft ADO.NET Entity Framework Step by Step*, Microsoft Press, Microsoft, 2013
- [2] Developer Express Inc, DevExpress Documentation, Developer Express Inc, USA, 2012
- [3] <http://msdn.microsoft.com/library/vstudio/ms1718688>
- [4] <http://www.entityframeworktutorial.net/>
- [5] <http://code.msdn.microsoft.com/101-LINQ-Samples-3fb9811b>
- [6] Nguyễn Xuân Trúc- Trần Tuấn Khanh, Tiểu luận chuyên ngành “*Tìm hiểu .NET 4.5 và Entity framework 5.0 với C# 5.0 xây dựng ứng dụng quản lý*”, khoa Công nghệ thông tin – ĐH Sư phạm Kỹ thuật thành phố Hồ Chí Minh, 2013
- [6] Trần Chí Tâm – Lương Thị Như Quỳnh, Tiểu luận chuyên ngành “*Tìm hiểu .NET 4.5 và Entity framework 5.0 với C# 5.0 xây dựng ứng dụng quản lý*”, khoa Công nghệ thông tin – ĐH Sư phạm Kỹ thuật thành phố Hồ Chí Minh, 2013
- [7] <http://docs.sencha.com/>

Phụ lục

Giới thiệu sencha touch 2.0

1. Sencha touch 2.0 là gì:

Sencha touch 2.0 là HTML 5 Mobile Application Framework, được xây dựng để cho phép các phát triển xây dựng một ứng dụng nhanh chóng, ấn tượng trên các môi trường iOS, Android, BlackBerry, Kindle Fire và sẽ tiếp tục hỗ trợ nhiều hơn nữa.

Ở phiên bản 2.0 Sencha Touch hỗ trợ MVC tốt hơn so với phiên bản trước đó, bổ sung thêm các API để có thể truy cập các thiết bị phần cứng tốt hơn. Cung cấp free package để có thể thêm ứng dụng vào Apple App Store và Android Play.

2. Tại sao chọn Sencha Touch?

Bạn đang có một ý tưởng phát triển ứng dụng cho mobile và bạn muốn phát triển cho cả 2 nền tảng là iOS và Android, lúc này bạn sẽ phải học Objective C cho iOS và Java cho Android. Việc học 2 ngôn ngữ trên 2 nền tảng khác nhau quả thật rất khó khăn và tốn thời gian của bạn và điều quan trọng là rất có thể bạn đã có kinh nghiệm trong HTML, CSS, JS và muốn tận dụng để phát triển các ứng dụng cho mobile. **Sencha Touch** sẽ là một lựa chọn tốt lúc này, bởi vì hiện nay nó đã cung cấp 1 bộ giao diện, các API điều khiển khá tốt và bạn cũng yên tâm rằng nó có thể hỗ trợ hoặc được bên thứ 3 hỗ trợ biên dịch nó thành một native app.

3. Thiết bị hỗ trợ:

Apple iOS 3+, Android 2.1+ and BlackBerry 6+.