

ỦY BAN NHÂN DÂN TP. HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC SÀI GÒN

NGUYỄN HOÀNG THANH  
HUỲNH CÔNG KHÁNH

PHẦN MỀM QUẢN LÝ TÀI SẢN  
TRƯỜNG ĐẠI HỌC SÀI GÒN  
PHÂN HỆ QUẢN LÝ TÀI SẢN CỐ ĐỊNH

TÓM TẮT KHÓA LUẬN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN  
TRÌNH ĐỘ ĐÀO TẠO: ĐẠI HỌC

NGƯỜI HƯỚNG DẪN: Th.S CAO THÁI PHƯƠNG THANH

TP. HỒ CHÍ MINH, THÁNG 11 NĂM 2014

## **GIỚI THIỆU ĐỀ TÀI**

### **❖ Lý do chọn đề tài (tính cấp thiết của đề tài)**

Để đáp ứng nhu cầu về quản lý các loại tài sản cố định cho Trường Đại học Sài Gòn, nhà trường mong muốn có được một phần mềm quản lý tập trung các tài sản cố định hiện có ở tất cả các cơ sở nhất là các loại tài sản giá thành cao. Nhờ đó trường sẽ dễ dàng nắm được thông tin về tình hình tài sản của trường ngay khi cần.

Yêu cầu đặt ra cho Khoa Công nghệ thông tin cũng như sinh viên là nghiên cứu, thiết kế và tạo ra được phần mềm đáp ứng đúng nhu cầu sử dụng của trường. Đồng thời, đây cũng là cơ hội để sinh viên có được đề tài và hoàn thành khóa luận tốt nghiệp.

Vì vậy nhóm quyết định chọn đề tài:

“Phần mềm quản lý tài sản Trường ĐH Sài Gòn - Phân hệ quản lý tài sản cố định”

\* Song hành với khóa luận này là một khóa luận của nhóm sinh viên Nguyễn Quốc Dũng, Vương Xương Nhơn với đề tài “Phần mềm quản lý tài sản Trường Đại học Sài Gòn - Phân hệ quản lý thiết bị” (cùng một người hướng dẫn), hai nhóm đã hợp tác để xây dựng một hệ thống lớn là “Phần mềm quản lý tài sản Trường Đại học Sài Gòn”. Đề tài này là một trong hai phân hệ chức năng chính của hệ thống lớn nói trên. Do đó, giữa hai khóa luận có rất nhiều điểm tương đồng, do sử dụng chung các công nghệ và quy trình phát triển.

### **❖ Mục đích nghiên cứu của đề tài**

Mục đích: Tạo ra được ứng dụng quản lý tài sản cố định cho Trường Đại học Sài Gòn.

Mục tiêu:

- + Nghiên cứu các công nghệ hiện đại có liên quan và xoay quanh đề tài, từ đó chọn ra công nghệ phù hợp.
- + Tìm hiểu các mô hình phát triển và xây dựng phần mềm tiên tiến.
- + Thu thập các số liệu thực tế phục vụ cho nghiên cứu.

- + Tin học hóa các nghiệp vụ quản lý tài sản.

#### ❖ **Đối tượng và phạm vi nghiên cứu**

Đối tượng nghiên cứu: chọn “Nghiệp vụ và quy trình quản lý tài sản cố định Trường ĐH Sài Gòn” là đối tượng nghiên cứu.

Phạm vi nghiên cứu:

- + Chỉ nghiên cứu giải pháp cho trường Đại học Sài Gòn.
- + Các số liệu thực tế được lấy từ Phòng Kế hoạch tài chính.
- + Các công nghệ có liên quan sẽ chỉ được nghiên cứu ở mức áp dụng vào thực tiễn, không đi sâu vào các lý thuyết bên trong.
- + Chỉ nghiên cứu các công nghệ có tính ứng dụng cao, và được cập nhật gần đây không quá mười năm.
- + Chỉ tìm hiểu và khảo sát các nhà cung cấp dịch vụ hiện có trong nước như: VDC2, Mất bão,...

#### ❖ **Phương pháp nghiên cứu**

Phương pháp tổng hợp, phân tích (thu thập tài liệu của các tác giả trong, ngoài nước có liên quan đến đề tài một cách có chọn lọc).

Phương pháp so sánh và đưa ra quyết định (tìm ưu nhược điểm của các phương án, chọn phương án hợp lý).

Nghiên cứu và phát triển lý thuyết xoay quanh đề tài.

Phương pháp thử và sai.

Phương pháp chuyên gia (tham khảo ý kiến của các chuyên gia về lĩnh vực, khía cạnh cụ thể).

## **NỘI DUNG**

### **CHƯƠNG 1: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

Trình bày các yêu cầu hệ thống, các bản phân tích, sơ đồ thiết kế cần thiết để chuẩn bị cho bước thực thi. Giai đoạn này rất quan trọng vì sẽ ảnh hưởng đến chất lượng của ứng dụng cũng như khả năng mở rộng về sau.

### **CHƯƠNG 2: THỰC THI**

Trình bày môi trường lập trình, tổ chức và phát triển ứng dụng. Các công nghệ và kỹ thuật lập trình được áp dụng, và kết quả thực thi.

### **CHƯƠNG 3: KIỂM THỬ VÀ TRIỂN KHAI**

Sau bước thực thi thì việc nghiệm thu phần mềm sẽ được diễn ra, và tất nhiên bước kiểm thử là bắt buộc phải có. Phần triển khai thực tế phần mềm cũng sẽ được trình bày ở đây.

### **KẾT LUẬN**

#### **❖ Kết quả đạt được**

Tìm hiểu và áp dụng thành công các kỹ thuật công nghệ mới vào phần mềm.

Phần mềm tạo ra dựa trên các quy trình kỹ thuật đã được học từ nhà trường.

Phần mềm đáp ứng đầy đủ các yêu cầu cơ bản đã đề ra. Đã và đang triển khai chạy thử trong giai đoạn đầu.

Phần mềm chạy được đa nền tảng (Desktop, Mobile).

#### **❖ Hướng phát triển**

Thay đổi và nâng cấp giao diện lẫn chức năng để phù hợp với nhu cầu sử dụng của nhà trường.

Nâng cấp chức năng thống kê. Xuất báo cáo động, thống kê dạng đồ thị trực quan.

Mở rộng phần mềm chạy đa nền tảng (Mac OS, Linux, IOS, Android, Windows Phone,...).

ỦY BAN NHÂN DÂN TP. HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC SÀI GÒN

---

NGUYỄN HOÀNG THANH  
HUỲNH CÔNG KHÁNH

**PHẦN MỀM QUẢN LÝ TÀI SẢN  
TRƯỜNG ĐẠI HỌC SÀI GÒN  
PHÂN HỆ QUẢN LÝ TÀI SẢN CỐ ĐỊNH**

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH: CÔNG NGHỆ THÔNG TIN  
TRÌNH ĐỘ ĐÀO TẠO: ĐẠI HỌC**

NGƯỜI HƯỚNG DẪN: Th.S CAO THÁI PHƯƠNG THANH

**TP. HỒ CHÍ MINH, THÁNG 11 NĂM 2014**

## **LỜI CAM ĐOAN**

Chúng tôi xin cam đoan đây là công trình nghiên cứu của nhóm, các số liệu và kết quả nghiên cứu nêu trong luận văn là trung thực, được các đồng tác giả cho phép sử dụng và chưa từng được sử dụng và chưa từng được công bố trong bất kỳ một công trình nào khác.

Tác giả luận văn

**Nguyễn Hoàng Thanh**

**Huỳnh Công Khánh**

## LỜI CẢM ƠN

Bằng tất cả sự trân trọng, chúng tôi xin bày tỏ lòng biết ơn sâu sắc nhất đến thầy Cao Thái Phương Thanh - người trực tiếp hướng dẫn và chỉ đạo triển khai dự án; xin gửi lời cảm ơn này đến thầy Hoàng Hữu Lượng (Phó Hiệu trưởng Trường Đại học Sài Gòn) và các thầy cô trong phòng Quản trị thiết bị đã tận tình góp ý và tạo mọi điều kiện để nhóm có thể hoàn thành khóa luận đúng tiến độ. Sau bao nhiêu năm tháng dùi mài, để có được thành quả như ngày hôm nay, chúng tôi cũng xin tri ân Ban Giám Hiệu nhà trường cùng toàn thể quý Thầy Cô trong khoa đã tạo điều kiện về cơ sở vật chất và nhiệt tình giảng dạy, chỉ bảo trong suốt quá trình học tập tại trường Đại học Sài Gòn.

Chúng tôi có thể hoàn thành đề tài này là nhờ nhận được hướng nghiên cứu đúng đắn, khoa học; quản lý tiến độ hợp lý và sự đánh giá khách quan từ thầy hướng dẫn, xin cảm ơn thầy đã luôn hướng dẫn tận tình, quan tâm và động viên chúng tôi trong suốt trong quá trình thực hiện đề tài.

Cuối cùng, chúng tôi xin bày tỏ lòng biết ơn đến gia đình, nơi đã cho chúng tôi một định hướng tốt, luôn dõi theo từng bước chân và đã tạo mọi điều kiện về vật chất lẫn tinh thần để giúp chúng tôi thực hiện ước mơ cho tương lai của mình.

Mặc dù nhóm đã cố gắng hoàn tất đề tài bằng hết khả năng của mình, nhưng cũng sẽ không thể tránh khỏi sai sót, nhóm rất mong nhận được sự thông cảm và góp ý của quý Thầy Cô và bạn bè quan tâm. Chúng tôi xin gửi lời chúc sức khỏe và thành đạt tới tất cả quý thầy cô và các bạn.

**Nhóm sinh viên thực hiện đề tài**

## MỤC LỤC

	Trang
Trang phụ bìa .....	i
Lời cam đoan.....	ii
Lời cảm ơn .....	iii
MỤC LỤC.....	1
DANH MỤC CÁC CHỮ VIẾT TẮT .....	4
DANH MỤC CÁC BẢNG.....	4
DANH MỤC CÁC SƠ ĐỒ .....	5
<b>MỞ ĐẦU</b> .....	7
Lý do chọn đề tài (tính cấp thiết của đề tài).....	7
Mục đích nghiên cứu của đề tài .....	7
Đối tượng và phạm vi nghiên cứu.....	8
Phương pháp nghiên cứu.....	8
Kết cấu của đề tài .....	9
<b>NỘI DUNG</b> .....	10
<b>CHƯƠNG 1: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG</b> .....	10
1.1. Tiếp nhận và xử lý yêu cầu .....	10
1.1.1. Yêu cầu chức năng nghiệp vụ .....	10
1.1.2. Yêu cầu chức năng thống kê báo cáo.....	11
1.1.3. Yêu cầu kỹ thuật.....	11
1.2. Lược đồ quan niệm.....	13
1.3. Lược đồ trường hợp sử dụng (Use cases) .....	15
1.4. Lược đồ CSDL mức vật lý .....	16



1.5. Lược đồ lớp (class).....	17
1.5.1. Lược đồ các lớp thực thể.....	17
1.5.2. Các lớp thư viện liên quan .....	20
1.6. Lược đồ tuần tự (sequences) .....	21
1.6.1. Thêm mới tài sản.....	22
1.6.2. Cập nhật thông tin dây .....	23
1.6.3. Chuyển đơn vị .....	24
1.6.4. Chuyển tình trạng tài sản .....	25
1.6.5. Phân rã chức năng chuyển tình trạng tài sản phần 1 .....	26
1.6.6. Phân rã chức năng chuyển tình trạng tài sản phần 2 .....	27
CHƯƠNG 2: THỰC THI.....	28
2.1. Môi trường lập trình và phát triển ứng dụng.....	28
2.2. Mô hình tổ chức ứng dụng .....	28
2.2.1. Tổ chức ứng dụng theo hướng module hóa trong lập trình đa nền tảng (cross-platform).....	28
2.2.2. Mô hình ba lớp trong ứng dụng hướng dữ liệu .....	29
2.2.3. Mô hình MVP (Model-View-Presenter) dành cho ứng dụng Winform Desktop .....	30
2.2.4. Mô hình ASP.NET Webform dành cho ứng dụng Web .....	31
2.3. Các công nghệ và kỹ thuật lập trình được áp dụng.....	32
2.3.1. Công nghệ Entity Framework (EF) trong lập trình dữ liệu hướng đối tượng (OOP).....	32
2.3.2. Công nghệ Sync Framework trong đồng bộ CSDL tập trung.....	42
2.3.3. Công nghệ DevExpress trong lập trình giao diện .....	51

2.3.4. Công nghệ giao diện tùy biến (responsive design) dành cho ứng dụng Web Mobile.....	66
2.4. Kết quả thực thi (các màn hình chức năng) .....	72
<b>CHƯƠNG 3: KIỂM THỬ VÀ TRIỂN KHAI .....</b>	<b>75</b>
3.1. Kiểm thử tự động mức mã nguồn (Unit test) .....	75
3.1.1. Kiểm thử hộp đen (Black box testing) .....	75
3.1.2. Mô hình kiểm thử AAA (Arrange-Act-Assert).....	75
3.2. Kiểm thử chấp nhận (Acceptance test) .....	75
3.3. Các mô hình triển khai .....	83
<b>KẾT LUẬN VÀ KIẾN NGHỊ .....</b>	<b>85</b>
<b>TÀI LIỆU THAM KHẢO.....</b>	<b>86</b>
<b>PHỤ LỤC A .....</b>	<b>89</b>
<b>PHỤ LỤC B .....</b>	<b>92</b>
<b>PHỤ LỤC C .....</b>	<b>96</b>

## DANH MỤC CÁC CHỮ VIẾT TẮT

CSDL	: Cơ sở dữ liệu
EF	: Entity Framework
RWD	: Responsive Web Design

## DANH MỤC CÁC BẢNG

TT	Tên bảng	Trang
1	Bảng 3.1: Kết quả kiểm thử chấp nhận	76
2	Bảng PL1.1: Sơ đồ bảng VITRIS	89
3	Bảng PL1.2: Sơ đồ bảng PHONGS	89
4	Bảng PL1.3: Sơ đồ bảng DONVIS	90
5	Bảng PL1.4: Sơ đồ bảng TAISANS	90
6	Bảng PL1.5: Sơ đồ bảng CHUNGTUS	91
7	Bảng PL2.1: Thiết kế giao diện _CRUDInterface<T>	92
8	Bảng PL2.2: Thiết kế giao diện _EFEEventRegisterInterface	93
9	Bảng PL2.3: Thiết kế lớp ảo _EntityAbstract1<T>	93
10	Bảng PL2.4: Thiết kế lớp ảo _EntityAbstract3<T>	95
11	Bảng PL3.1: Thiết kế lớp cứng ViTri	96
12	Bảng PL3.2: Thiết kế lớp cứng Phong	97
13	Bảng PL3.3: Thiết kế lớp cứng DonVi	98
14	Bảng PL3.4: Thiết kế lớp cứng TaiSan	98
15	Bảng PL3.5: Thiết kế lớp cứng CTTaiSan	99

**DANH MỤC CÁC SƠ ĐỒ**

<b>TT</b>	<b>Tên sơ đồ</b>	<b>Trang</b>
1	Sơ đồ 1.1: Sơ đồ quan niệm quan hệ tài sản - đơn vị	14
2	Sơ đồ 1.2: Sơ đồ quan niệm quan hệ phòng - vị trí	14
3	Sơ đồ 1.3: Sơ đồ quan niệm quan hệ phân quyền	15
4	Sơ đồ 1.4: Sơ đồ trường hợp sử dụng (Usecases)	15
5	Sơ đồ 1.5: Sơ đồ CSDL vật lý quan hệ phòng - vị trí	16
6	Sơ đồ 1.6: Sơ đồ CSDL mức vật lý quan hệ tài sản - log - chứng từ	16
7	Sơ đồ 1.7: Sơ đồ CSDL mức vật lý quan hệ tài sản - đơn vị	17
8	Sơ đồ 1.8: Sơ đồ kế thừa các lớp thực thể	18
9	Sơ đồ 1.9: Sơ đồ lớp quan hệ tài sản – đơn vị	19
10	Sơ đồ 1.10: Sơ đồ lớp quan hệ chứng từ	19
11	Sơ đồ 1.11: Sơ đồ lớp quan hệ phân quyền	19
12	Sơ đồ 1.12: Minh họa cách override một phương thức từ lớp cứng	20
13	Sơ đồ 1.13: Sơ đồ tuần tự cho chức năng "Thêm mới tài sản"	22
14	Sơ đồ 1.14: Sơ đồ tuần tự cho chức năng "Cập nhật thông tin dây"	23
15	Sơ đồ 1.15: Sơ đồ tuần tự cho chức năng "Chuyển đơn vị"	24
16	Sơ đồ 1.16: Sơ đồ tuần tự cho chức năng "Chuyển tình trạng tài sản"	25
17	Sơ đồ 1.17: Sơ đồ tuần tự chức năng "Phân rã chuyển tình trạng tài sản phần 1"	26
18	Sơ đồ 1.18: Sơ đồ tuần tự chức năng "Phân rã chuyển tình trạng tài sản phần 2"	27
19	Sơ đồ 2.1: Mô hình quan niệm tổ chức ứng dụng hướng module hóa	29
20	Sơ đồ 2.2: Mô hình chi tiết tổ chức ứng dụng hướng module hóa	29
21	Sơ đồ 2.3: Mô hình MVP	31
22	Sơ đồ 2.4: Mô hình triển khai TPC trong kế thừa	33

23	Sơ đồ 2.5: Cách hoạt động của kỹ thuật truy vấn lồng	35
24	Sơ đồ 2.6: Tham chiếu ngược trên các quan hệ 1 - n, n - n	36
25	Sơ đồ 2.7: Tương thích ngược trong phiên bản CSDL	37
26	Sơ đồ 2.8: Tương thích xuôi phiên bản CSDL	38
27	Sơ đồ 2.9: Không tương thích phiên bản CSDL (trường hợp 1)	38
28	Sơ đồ 2.10: Không tương thích phiên bản CSDL (trường hợp 2)	38
29	Sơ đồ 2.11: Giao tiếp 2 chiều trong mô hình dữ liệu hướng sự kiện	40
30	Sơ đồ 2.12: Cách hoạt động giữa Singleton và DbContext	42
31	Sơ đồ 2.13: Cách tổ chức lưu trữ của Sync Framework	43
32	Sơ đồ 2.14: Cách hoạt động của Sync Framework	43
33	Sơ đồ 3.1: Mô hình triển khai hệ thống phần mềm	83

## MỞ ĐẦU

### **Lý do chọn đề tài (tính cấp thiết của đề tài)**

- Để đáp ứng nhu cầu về quản lý các loại tài sản cố định cho Trường Đại học Sài Gòn, nhà trường mong muốn có được một phần mềm quản lý tập trung các tài sản cố định hiện có ở tất cả các cơ sở nhất là các loại tài sản giá thành cao. Nhờ đó trường sẽ dễ dàng nắm được thông tin về tình hình tài sản của trường ngay khi cần.

- Yêu cầu đặt ra cho Khoa Công nghệ thông tin cũng như sinh viên là nghiên cứu, thiết kế và tạo ra được phần mềm đáp ứng đúng nhu cầu sử dụng của trường. Đồng thời, đây cũng là cơ hội để sinh viên có được đề tài và hoàn thành khóa luận tốt nghiệp.

- Vì vậy nhóm quyết định chọn đề tài:

“Phần mềm quản lý tài sản Trường ĐH Sài Gòn - Phân hệ quản lý tài sản cố định”

\* Song hành với khóa luận này là một khóa luận của nhóm sinh viên Nguyễn Quốc Dũng, Vương Xương Nhơn với đề tài “Phần mềm quản lý tài sản Trường Đại học Sài Gòn - Phân hệ quản lý thiết bị” (cùng một người hướng dẫn), hai nhóm đã hợp tác để xây dựng một hệ thống lớn là “Phần mềm quản lý tài sản Trường Đại học Sài Gòn”. Đề tài này là một trong hai phân hệ chức năng chính của hệ thống lớn nói trên. Do đó, giữa hai khóa luận có rất nhiều điểm tương đồng, do sử dụng chung các công nghệ và quy trình phát triển.

### **Mục đích nghiên cứu của đề tài**

- Mục đích: Tạo ra được ứng dụng quản lý tài sản cố định cho Trường Đại học Sài Gòn.

- Mục tiêu:

+ Nghiên cứu các công nghệ hiện đại có liên quan và xoay quanh đề tài, từ đó chọn ra công nghệ phù hợp.

- + Tìm hiểu các mô hình phát triển và xây dựng phần mềm tiên tiến.
- + Thu thập các số liệu thực tế phục vụ cho nghiên cứu.
- + Tin học hóa các nghiệp vụ quản lý tài sản.

### **Đối tượng và phạm vi nghiên cứu**

- Đối tượng nghiên cứu: chọn “Nghiệp vụ và quy trình quản lý tài sản cố định Trường ĐH Sài Gòn” là đối tượng nghiên cứu.
- Phạm vi nghiên cứu:
  - + Chỉ nghiên cứu giải pháp cho trường Đại học Sài Gòn.
  - + Các số liệu thực tế được lấy từ Phòng Kế hoạch tài chính.
  - + Các công nghệ có liên quan sẽ chỉ được nghiên cứu ở mức áp dụng vào thực tiễn, không đi sâu vào các lý thuyết bên trong.
  - + Chỉ nghiên cứu các công nghệ có tính ứng dụng cao, và được cập nhật gần đây không quá mười năm.
  - + Chỉ tìm hiểu và khảo sát các nhà cung cấp dịch vụ hiện có trong nước như: VDC2, Mắt bão,...

### **Phương pháp nghiên cứu**

- Phương pháp tổng hợp, phân tích (thu thập tài liệu của các tác giả trong, ngoài nước có liên quan đến đề tài một cách có chọn lọc).
- Phương pháp so sánh và đưa ra quyết định (tìm ưu nhược điểm của các phương án, chọn phương án hợp lý).
- Nghiên cứu và phát triển lý thuyết xoay quanh đề tài.
- Phương pháp thử và sai.
- Phương pháp chuyên gia (tham khảo ý kiến của các chuyên gia về lĩnh vực, khía cạnh cụ thể).

### **Kết cấu của đề tài**

- Đề tài gồm có: phần mở đầu, phần nội dung được trình bày theo từng chương, phần kết luận.

- Phần nội dung gồm có 3 chương:

#### **✓ CHƯƠNG 1: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

+ Trình bày các yêu cầu hệ thống, các bản phân tích, sơ đồ thiết kế cần thiết để chuẩn bị cho bước thực thi. Giai đoạn này rất quan trọng vì sẽ ảnh hưởng đến chất lượng của ứng dụng cũng như khả năng mở rộng về sau.

#### **✓ CHƯƠNG 2: THỰC THI**

+ Trình bày môi trường lập trình, tổ chức và phát triển ứng dụng. Các công nghệ và kỹ thuật lập trình được áp dụng, và kết quả thực thi.

#### **✓ CHƯƠNG 3: KIỂM THỬ VÀ TRIỂN KHAI**

+ Sau bước thực thi thì việc nghiệm thu phần mềm sẽ được diễn ra, và tất nhiên bước kiểm thử là bắt buộc phải có. Phần triển khai thực tế phần mềm cũng sẽ được trình bày ở đây.



## **NỘI DUNG**

### **CHƯƠNG 1: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

#### **1.1. Tiếp nhận và xử lý yêu cầu**

##### **1.1.1. Yêu cầu chức năng nghiệp vụ**

- Trường Đại học Sài Gòn muốn quản lý tài sản của trường theo từng đơn vị, từng vị trí.
- Trường có nhiều đơn vị. Mỗi đơn vị phải có: mã đơn vị, tên đơn vị, mô tả,... Mỗi đơn vị phải thuộc một loại đơn vị và có thể có đơn vị con.
- Mỗi đơn vị được trang bị nhiều tài sản. Mỗi tài sản phải có: tên tài sản, đơn vị tính, số lượng, đơn giá, giá tiền, nguồn gốc, nước sản xuất,... Mỗi tài sản phải thuộc một loại tài sản và có thể có tài sản kèm theo. Tài sản có thể di chuyển qua lại giữa các đơn vị, phòng, vị trí hoặc thay đổi tình trạng của tài sản (đang sử dụng, chờ xử lý, thanh lý,...).
- Trường có nhiều cơ sở, mỗi cơ sở có nhiều dãy, mỗi dãy có nhiều tầng. Mỗi cơ sở, dãy, tầng cần phải có: tên, mô tả,...
- Tài sản có thể thuộc một phòng. Mỗi phòng cần lưu trữ các thông tin sau: tên phòng, loại phòng, vị trí phòng (thuộc cơ sở – dãy – tầng), số chỗ ngồi, mô tả phòng,... Và tài sản có thể thuộc một vị trí (bao gồm cơ sở – dãy – tầng).
- Phòng có thể thuộc tầng hoặc dãy hoặc cơ sở và thuộc một loại phòng nhất định.
- Loại tài sản có thể có loại tài sản con và cần lưu trữ các thông tin sau: mã loại, tên loại, đơn vị tính, loại tài sản hữu hình hoặc vô hình, thuộc loại tài sản, mô tả...
- Mọi sự thay đổi trên tài sản đều phải lưu lại chứng từ. Mỗi chứng từ có thể kèm theo nhiều tập tin chứng từ liên quan.
- Với xu thế sử dụng điện thoại thông minh (smartphone), nhà trường cũng muốn nắm bắt tình hình từ xa thông qua điện thoại nên phần mềm phải bổ sung thêm website

với giao diện web và mobile để nhà trường tiện theo dõi. Tất cả dữ liệu phải đảm bảo tính nhất quán.

- Dữ liệu có thể thống kê, xuất báo cáo theo phòng và theo tài sản. Về thống kê phòng phải có các điều kiện lọc như cơ sở, loại phòng. Về thống kê tài sản phải có các điều kiện lọc như: ngày, loại tài sản, đơn vị quản lý, cơ sở. Có thể kết xuất dữ liệu báo cáo ra các tập tin thông dụng như pdf, excel để thiết kế lại hoặc in ấn trực tiếp.

- Đảm bảo phân quyền chặt chẽ, nhất quán, cho phép phân quyền trên từng đơn vị.

- Mọi thao tác trên cơ sở dữ liệu điều phải lưu lại vào nhật ký hệ thống để quản trị viên dễ dàng theo dõi và nắm bắt tình hình.

#### **1.1.2. Yêu cầu chức năng thống kê báo cáo**

- Thống kê sự tăng giảm tài sản trên toàn trường và từng đơn vị quản lý.

- Thống kê danh sách, số lượng, tổng giá trị tài sản của các phòng theo loại phòng, cơ sở, dãy, tầng.

- Thống kê danh sách, số lượng, giá trị tài sản theo loại tài sản, theo đơn vị quản lý, theo cơ sở, dãy, tầng, phòng.

- Thống kê danh sách, số lượng tài sản tại nơi sử dụng.

\* Các thống kê trên đều phải xuất được báo cáo để in ấn.

#### **1.1.3. Yêu cầu kỹ thuật**

- Các yêu cầu kỹ thuật được xem xét và đưa ra dựa trên các khảo sát về cơ sở hạ tầng ứng dụng hiện tại của các nhà cung cấp dịch vụ liên quan và hạ tầng trang thiết bị hiện có của trường.

##### **1.1.3.1 Yêu cầu về hệ thống thông tin**

- Triển khai được trên hệ thống mạng nội bộ của trường hoặc trên mạng Internet toàn cầu hiện hành.

- Tương thích và vận hành tốt trên các giao thức mạng phổ biến hiện tại như: IP (IPv4, IPv6), FTP (Dùng trong tải lên/xuống các tập tin), HTTP 1.1 (Dùng trong gửi nhận dữ liệu),...
- Có khả năng làm việc khi không có mạng, đồng bộ dữ liệu lên máy chủ tập trung khi có mạng.
- Đảm bảo tính nhất quán về mặt dữ liệu giữa các ứng dụng trên các nền tảng khác nhau, ví dụ: giữa ứng dụng Desktop và ứng dụng Web.
- Yêu cầu chung về bảo mật dữ liệu và an toàn thông tin.
- + Dữ liệu phải có khả năng sao lưu và dễ dàng khôi phục lại khi cần thiết.
- + Dữ liệu chỉ được truy cập bởi những người dùng có thẩm quyền.
- + Các nghiệp vụ liên quan đến bảo mật:
  - Mã hóa mã nguồn, chống dịch ngược.
  - Mã hóa các thông tin nhạy cảm (mật khẩu người dùng, cấu hình cài đặt, ...) theo tiêu chuẩn mã hóa dữ liệu Advanced Encryption Standard (AES) do Viện Tiêu chuẩn và Công nghệ quốc gia Hoa Kỳ (National Institute Standards and Technology – NIST) phát hành ngày 26/11/2001.
  - Các dữ liệu truyền tải giữa máy chủ và máy tính cá nhân người dùng cần đảm bảo độ an toàn bằng cách mã hóa trước khi truyền qua mạng.

#### **1.1.3.2. Yêu cầu phần cứng**

- Vận hành tốt trên các hệ thống máy tính hiện có của trường, cấu hình hệ thống máy tính tối thiểu đề nghị:
  - + CPU: Xung nhịp 1.0 Ghz hoặc cao hơn.
  - + RAM: Dung lượng 512 MB hoặc cao hơn.
  - + Đĩa cứng: Dung lượng trống tối thiểu 5GB.

+ Hiển thị tốt trên các màn hình kích cỡ 15 inch hoặc lớn hơn, độ phân giải 1024x768 hoặc cao hơn.

#### **1.1.3.3. Yêu cầu phần mềm**

- Ứng dụng chạy trên máy tính cá nhân (Windows Desktop Application)

+ Hệ điều hành Windows  $\geq 7$ .

+ Tương thích .NET Framework  $\geq 4$  bản đầy đủ (Full) - vì mỗi phiên bản .NET nhiều bản phân phối, ví dụ: .NET Client Profile.

+ Hệ quản trị CSDL MSSQL Server  $\geq 2008$  (chỉ yêu cầu khi sử dụng ở chế độ không có mạng).

- Ứng dụng Web (Web Desktop Application)

+ Truy cập từ trình duyệt thông qua địa chỉ website.

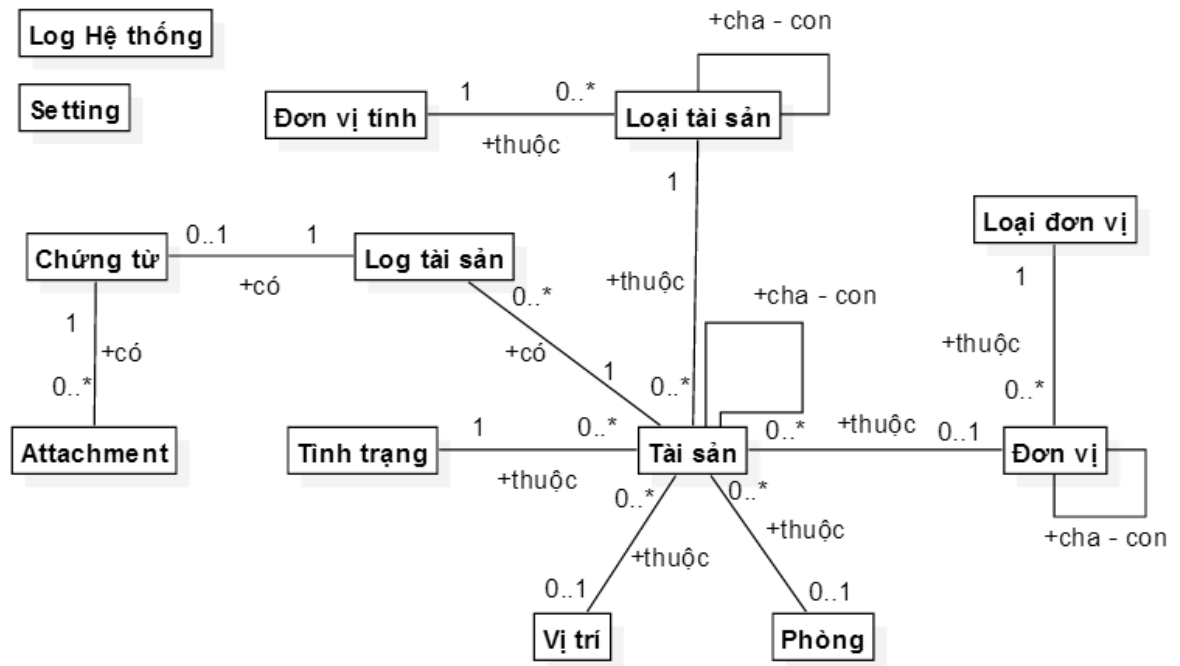
+ Chạy tốt trên trình duyệt Chrome  $\geq 28$ , Firefox  $\geq 29$ , Internet Explorer  $\geq 8$ .

+ Hỗ trợ thiết bị di động.

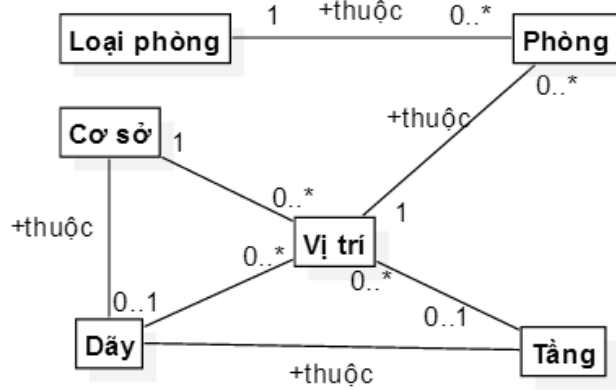
+ Vận hành tốt trên máy chủ IIS ASP.NET  $\geq 4.0$ .

### **1.2. Lược đồ quan niệm**

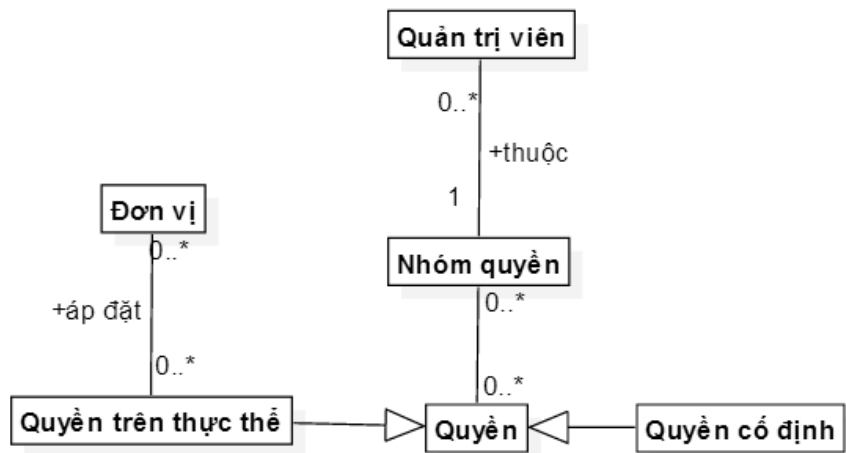
- Do sự dàn trải rộng của sơ đồ nên sẽ trình bày từng sơ đồ con. Các sơ đồ thiết kế sau đây được trình bày theo chuẩn UML 2.0<sup>[18]</sup>.



Sơ đồ 1.1: Sơ đồ quan niệm quan hệ tài sản - đơn vị

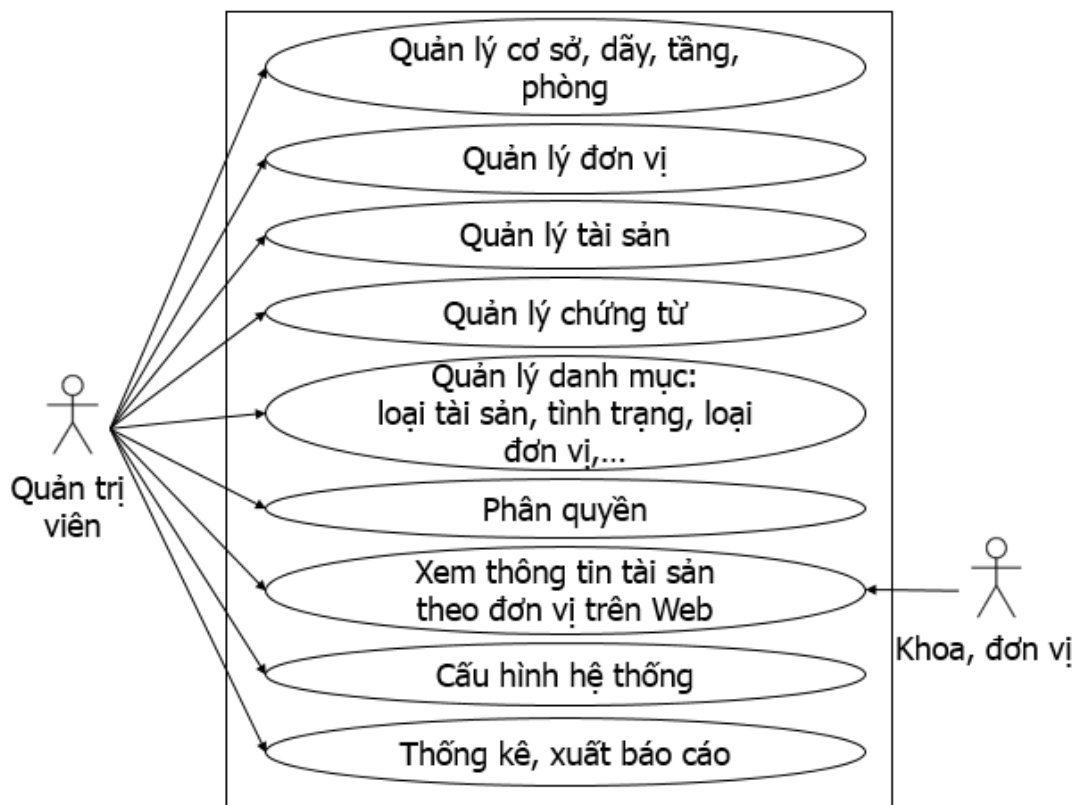


Sơ đồ 1.2: Sơ đồ quan niệm quan hệ phòng - vị trí



Sơ đồ 1.3: Sơ đồ quan niệm quan hệ phân quyền

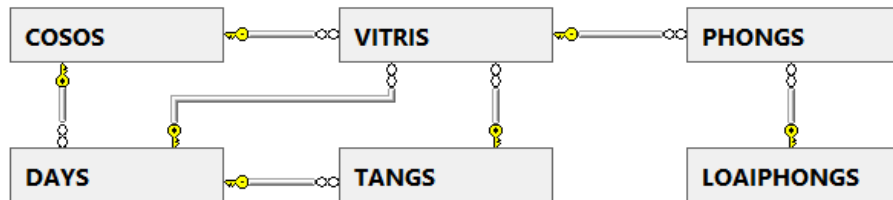
### 1.3. Lược đồ trường hợp sử dụng (Use cases)



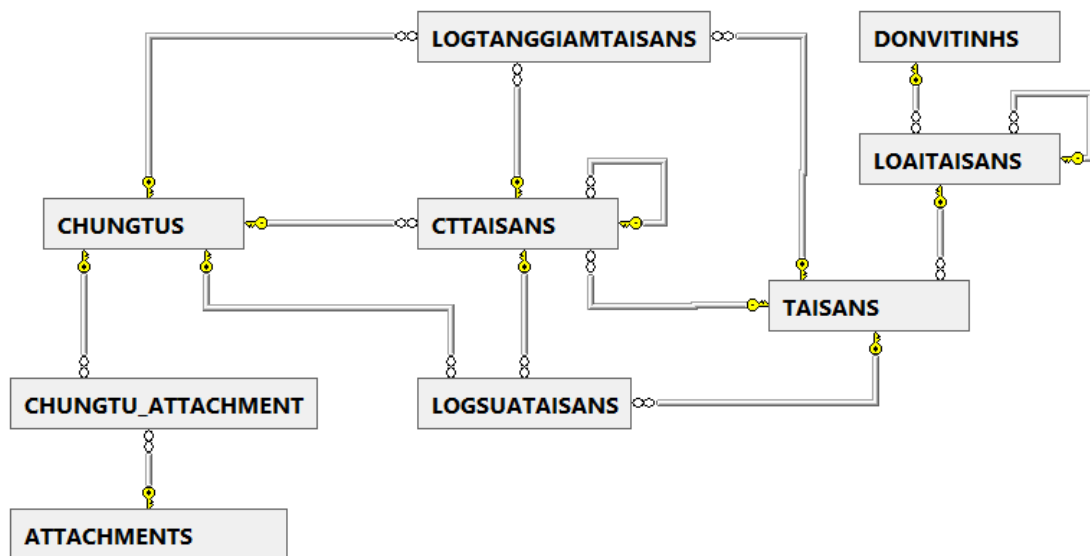
Sơ đồ 1.4: Sơ đồ trường hợp sử dụng (Usecases)

#### 1.4. Lược đồ CSDL mức vật lý

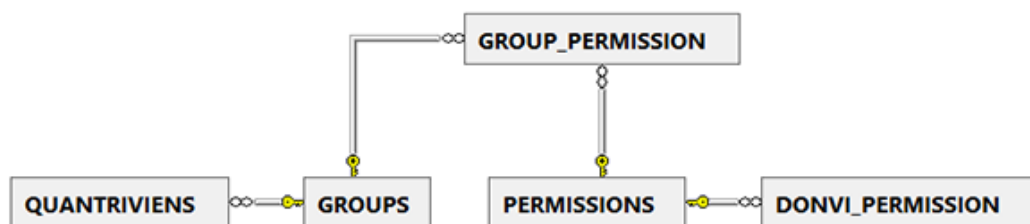
\* Các lược đồ CSDL sau đây được thể hiện trên hệ quản trị MSSQL Server 2008 R2, và được EF tự động tạo khi ánh xạ bản thiết kế lớp xuống.



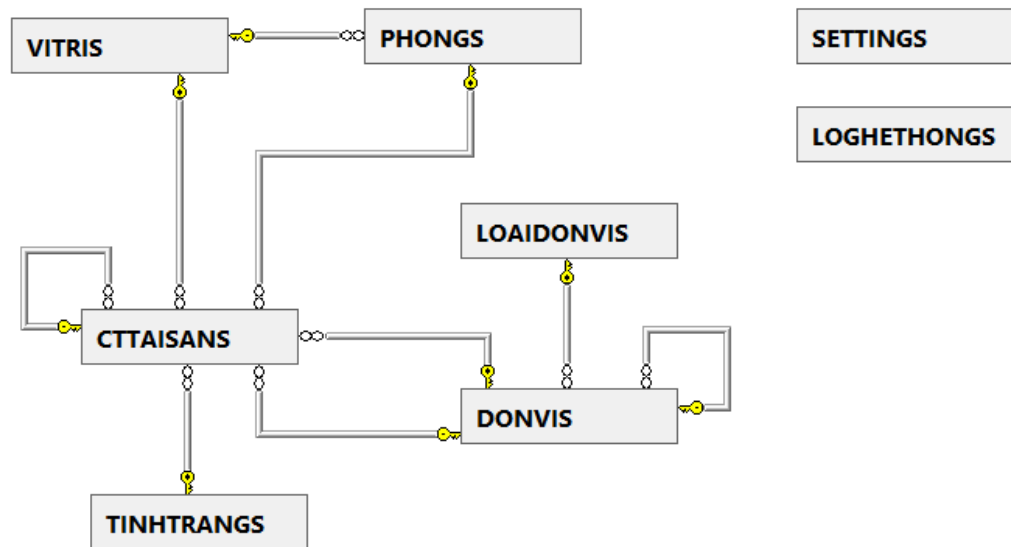
Sơ đồ 1.5: Sơ đồ CSDL vật lý quan hệ phòng - vị trí



Sơ đồ 1.6: Sơ đồ CSDL mức vật lý quan hệ tài sản - log - chứng từ



Hình 1.4.3: Sơ đồ CSDL mức vật lý quan hệ phân quyền



Sơ đồ 1.7: Sơ đồ CSDL mức vật lý quan hệ tài sản - đơn vị

- Chi tiết các bảng trong CSDL vật lý.

(Xem phụ lục A)

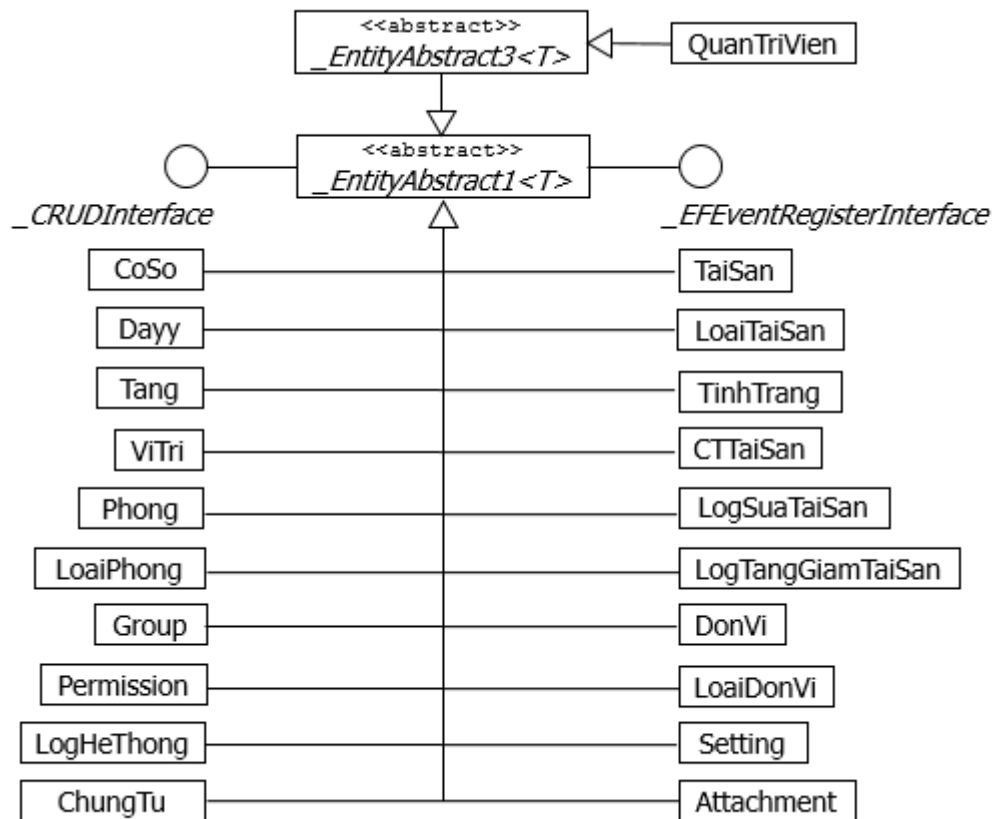
## 1.5. Lược đồ lớp (class)

### 1.5.1. Lược đồ các lớp thực thể

#### 1.5.1.1. Sơ đồ kế thừa giữa các lớp

- Giữa các thực thể sẽ có nhiều thuộc tính giống nhau cần lưu trữ và còn có sự giống nhau trong các chức năng nghiệp vụ, nên việc sử dụng mô hình kế thừa trong thiết kế là điều vô cùng cần thiết. Điều đó khiến cho việc bảo trì các đoạn mã được diễn ra nhanh chóng và tăng tính nhất quán trong toàn cấu trúc.

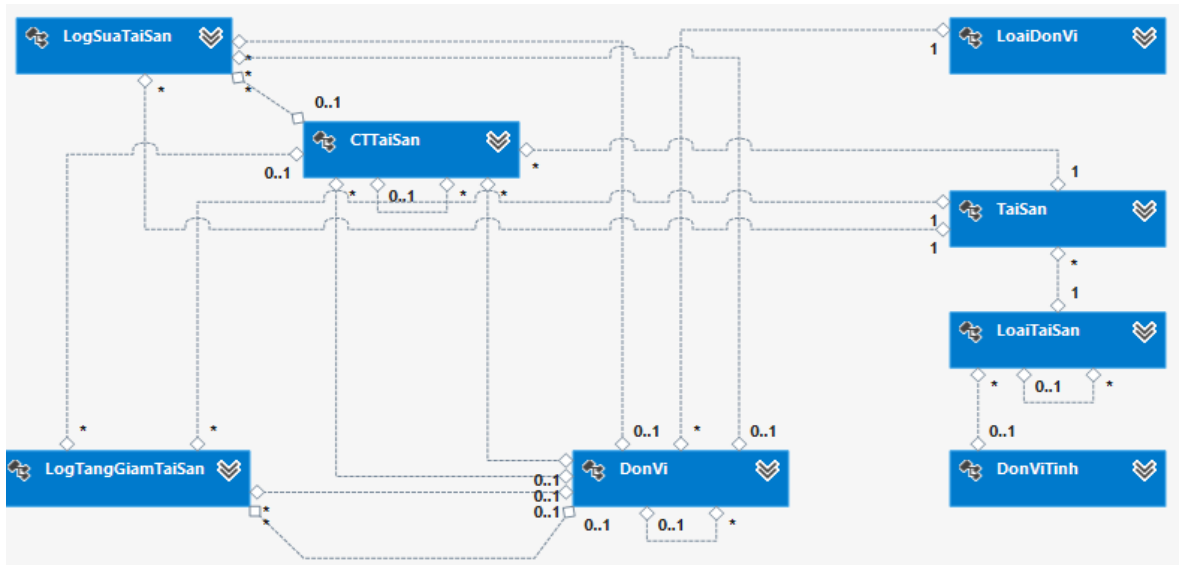




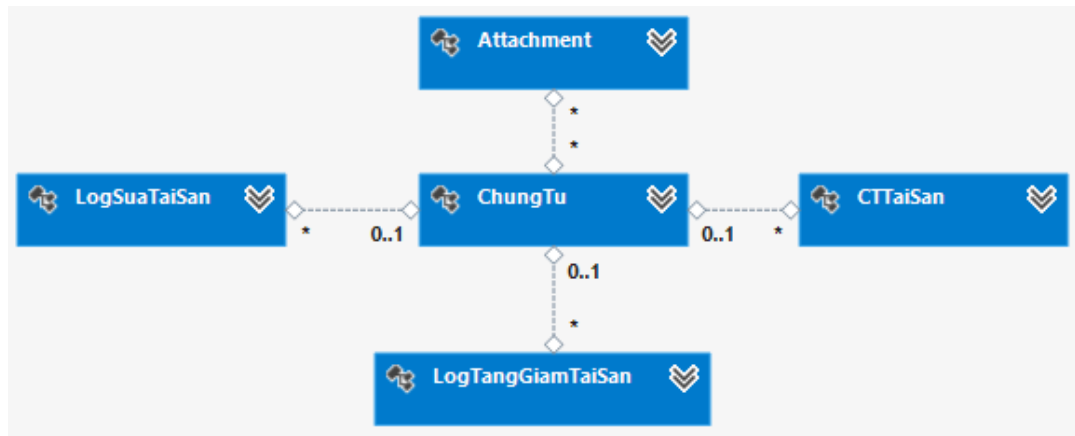
Sơ đồ 1.8: Sơ đồ kế thừa các lớp thực thể

- Một mô hình kế thừa tốt là mô hình mà trong đó các đoạn mã định nghĩa thuộc tính cũng như các đoạn mã lập trình được đặt đúng vị trí, đúng mức (level) cần thiết để tận dụng tối đa mã nguồn đã có, tránh trùng lặp.

### 1.5.1.2. Sơ đồ quan hệ giữa các lớp



Sơ đồ 1.9: Sơ đồ lớp quan hệ tài sản – đơn vị



Sơ đồ 1.10: Sơ đồ lớp quan hệ chứng từ



Sơ đồ 1.11: Sơ đồ lớp quan hệ phân quyền

- Khi các lớp cứng (concreted class) kế thừa từ lớp ảo, chúng sẽ có được các chức năng sẵn có mà không cần phải lập trình lại, nếu vì lý do nào đó mà các lớp cứng muốn bổ sung các đoạn mã phụ, thì sẽ gọi đè (override).

```
protected override void init()
{
    base.init();
    vitris = new List<ViTri>();
    tangs = new List<Tang>();
}
```

*Sơ đồ 1.12: Minh họa cách override một phương thức từ lớp cứng*

- Chi tiết bản thiết kế các giao diện và lớp ảo.

(Xem phụ lục B)

- Chi tiết bản thiết kế các lớp cứng.

(Xem phụ lục C)

### 1.5.2. Các lớp thư viện liên quan

- Ngoài các lớp thực thể chính thì xoay quanh còn có lớp các thư viện hỗ trợ, các thư viện trợ giúp (helper), cấu hình liên quan...

\* Vì lý do số lượng lớp thư viện quá nhiều nên sau đây chỉ liệt kê tên và mô tả.

- Thư viện trợ giúp CSDL (Không gian tên: TSCD.Global):

- + client\_database: Cung cấp các phương thức giao tiếp với CSDL ở máy chủ địa phương.

- + server\_database: Cung cấp các phương thức giao tiếp với CSDL ở máy chủ tập trung.

- + working\_database: Cung cấp các phương thức giao tiếp với CSDL không phụ thuộc máy chủ.

- Thư viện trợ giúp (không gian tên: SHARED.Libraries): DateTimeHelper, Debug, DatabaseHelper, FileHelper, FTPHelper, HTTPHelper, ImageHelper, MobileDetect,

PermissionHelper, ReportHelper, ServerTimeHelper, StringHelper, SkinHelper, GUID.

- Thư viện cấu hình (không gian tên: TSCD.Global):

+ remote\_setting: Các cấu hình áp đặt cho toàn máy trạm.

- ftp\_host: Các cấu hình để tải các tập tin lên máy chủ.
- http\_host: Các cấu hình để tải tập tin từ máy chủ về.

+ local\_setting: Các cấu hình áp đặt lên máy trạm hiện hành.

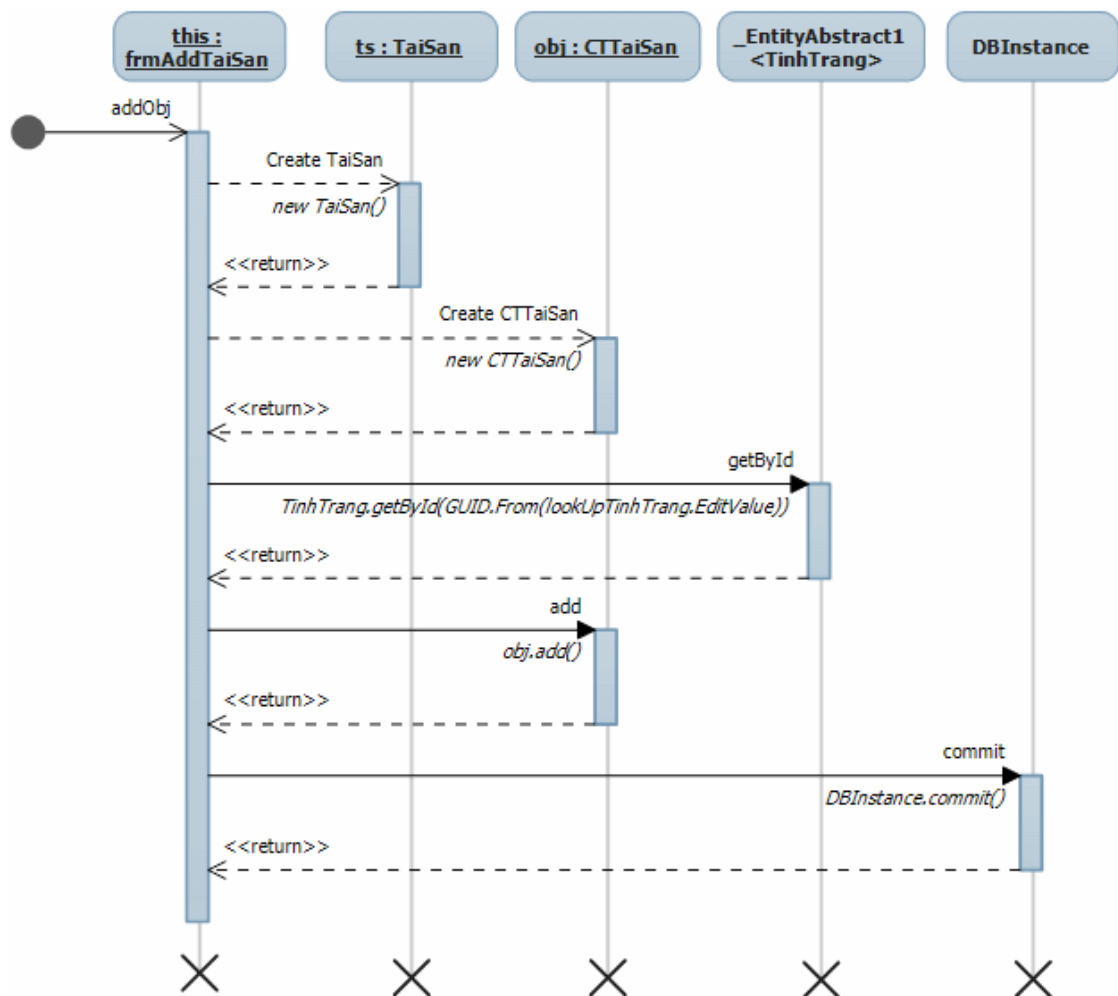
+ sync: Các cấu hình đồng bộ CSDL.

### **1.6. Lược đồ tuần tự (sequences)**

- Do hạn chế về không gian trình bày cũng như không thể đưa hết mọi bản thiết kế vào trong báo cáo này nên nhóm chọn ra năm chức năng chính để trình bày.

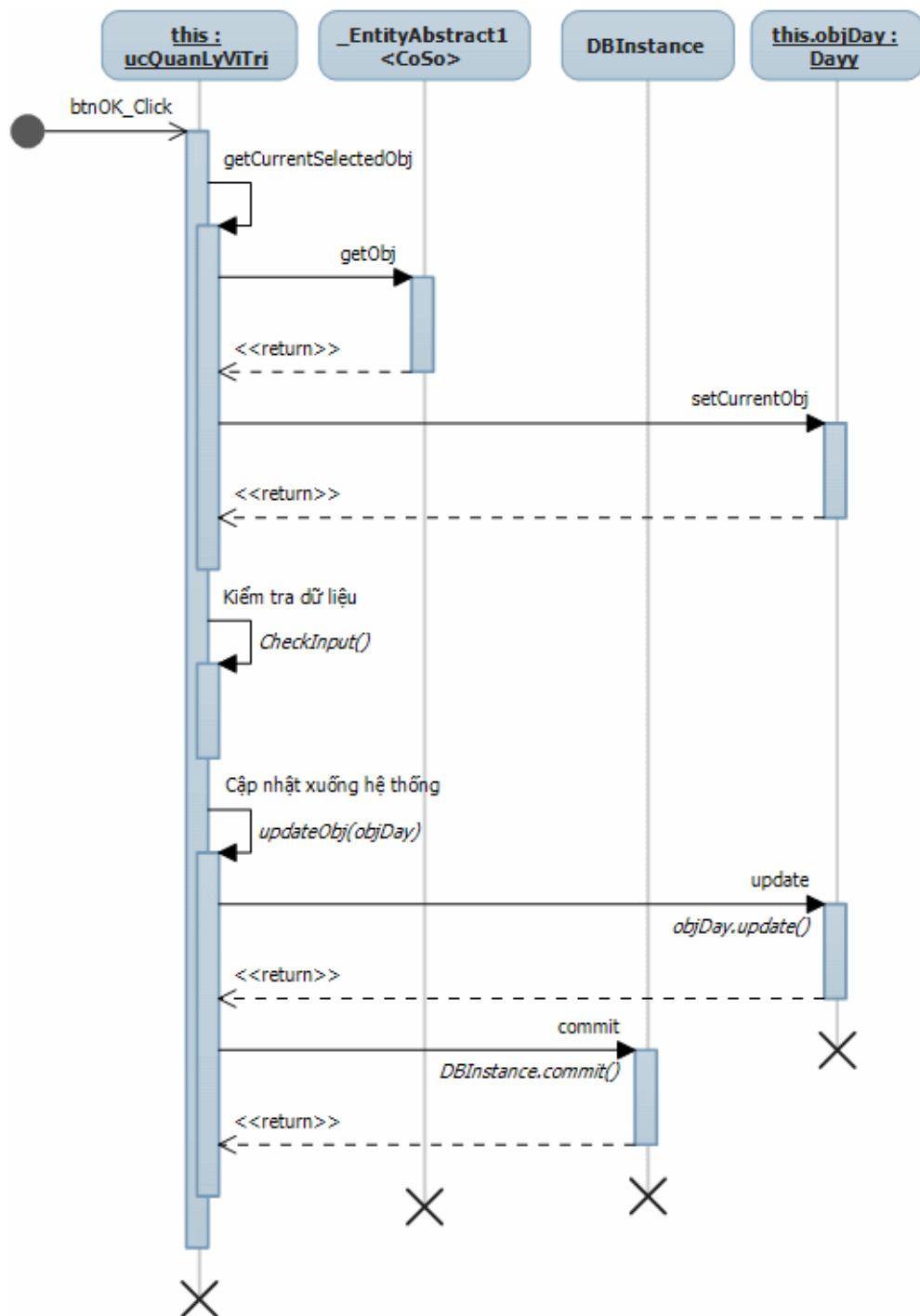
- Các sơ đồ sau được thể hiện tối đa tới mức hai tính từ lời gọi hàm đầu tiên.

### 1.6.1. Thêm mới tài sản



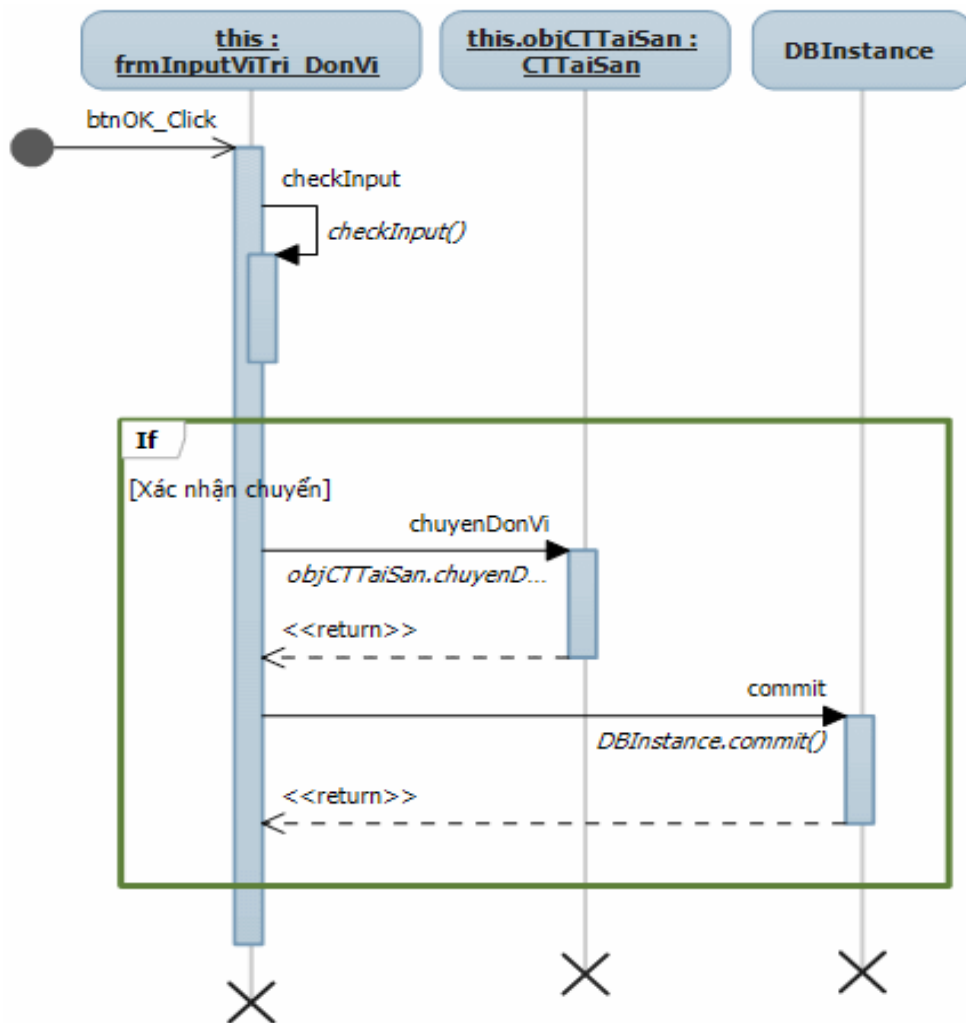
Sơ đồ 1.13: Sơ đồ tuần tự cho chức năng "Thêm mới tài sản"

### 1.6.2. Cập nhật thông tin dây



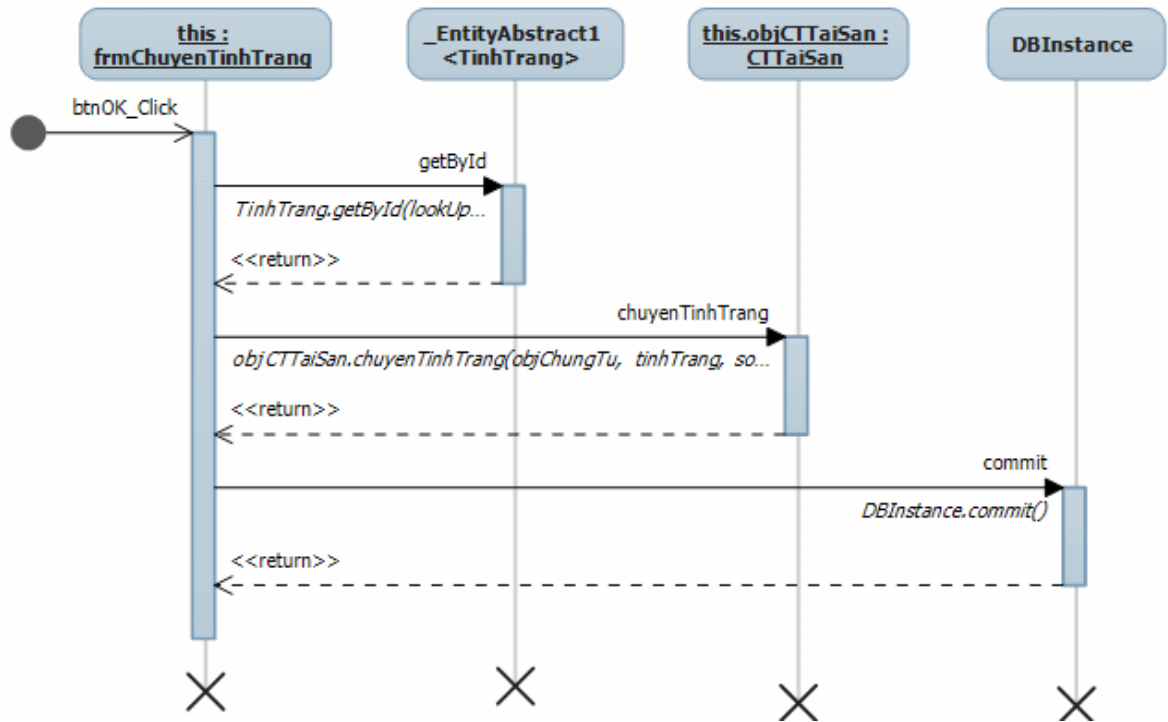
Sơ đồ 1.14: Sơ đồ tuần tự cho chức năng "Cập nhật thông tin dây"

### 1.6.3. Chuyển đơn vị



Sơ đồ 1.15: Sơ đồ tuần tự cho chức năng "Chuyển đơn vị"

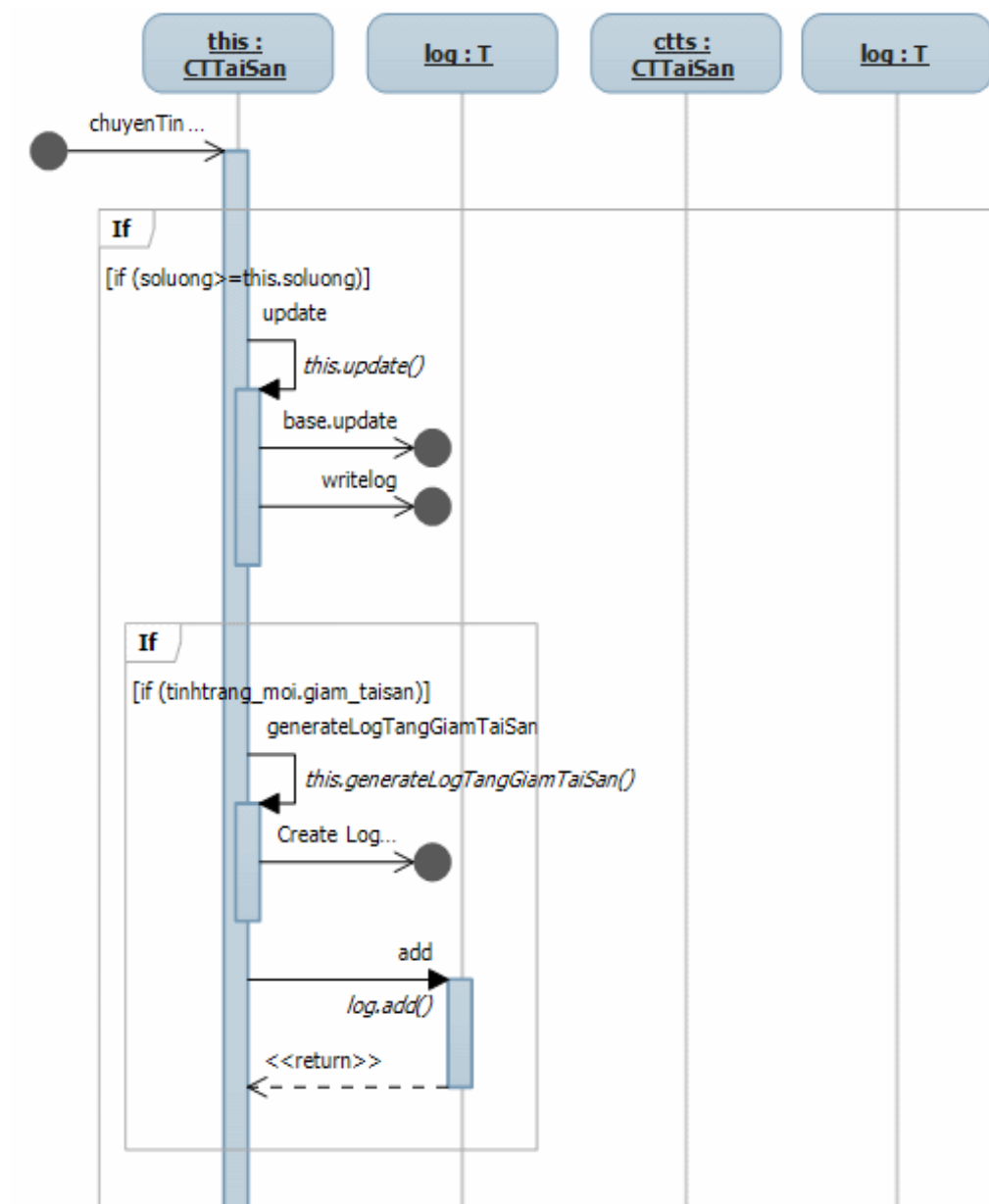
#### 1.6.4. Chuyển tình trạng tài sản



Sơ đồ 1.16: Sơ đồ tuần tự cho chức năng "Chuyển tình trạng tài sản"

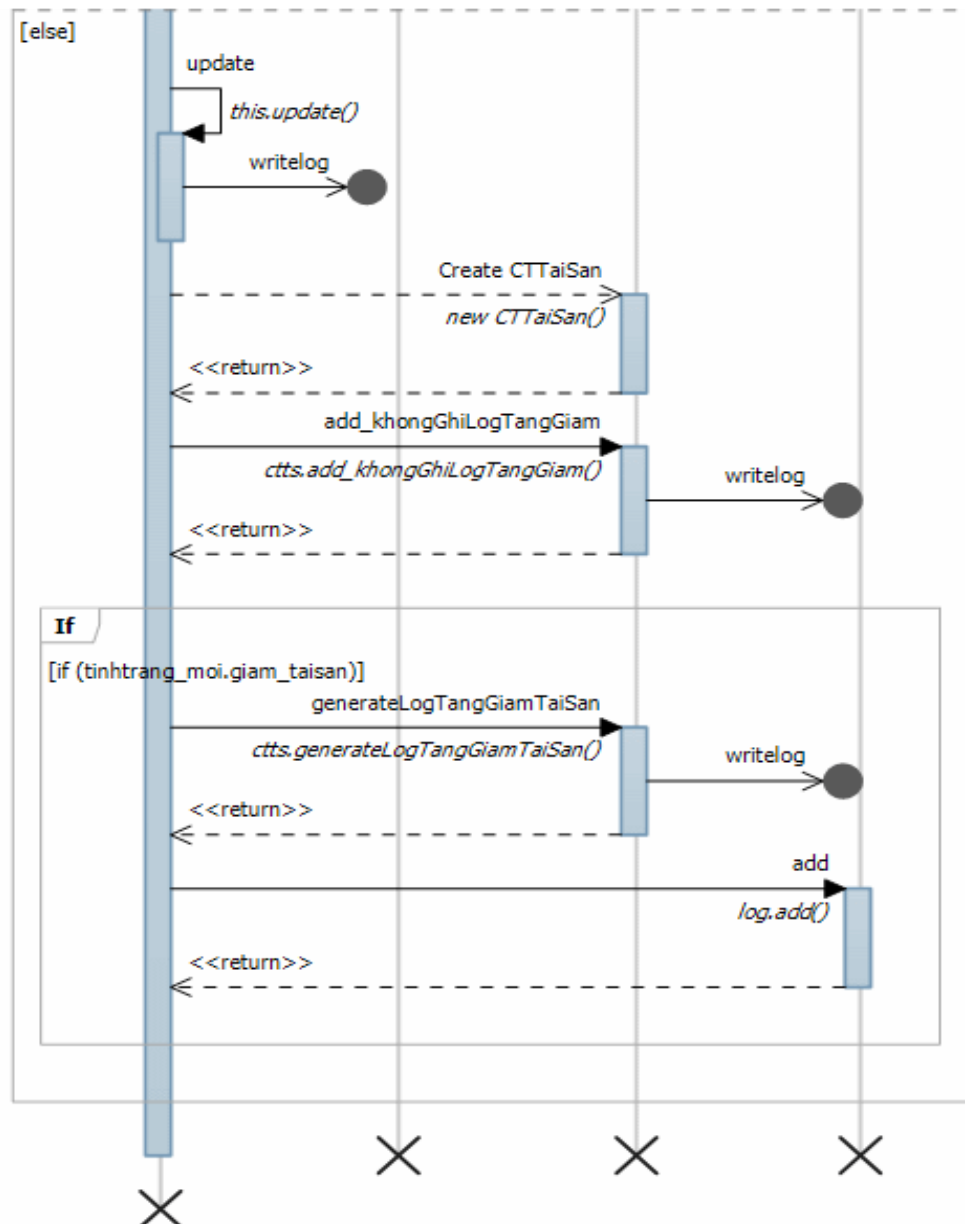


### 1.6.5. Phân rã chức năng chuyển tình trạng tài sản phần 1



Sơ đồ 1.17: Sơ đồ tuần tự chức năng "Phân rã chuyển tình trạng tài sản phần 1"

### 1.6.6. Phân rã chức năng chuyển tình trạng tài sản phần 2



Sơ đồ 1.18: Sơ đồ tuần tự chức năng "Phân rã chuyển tình trạng tài sản phần 2"

## **CHƯƠNG 2: THỰC THI**

### **2.1. Môi trường lập trình và phát triển ứng dụng**

- Hệ điều hành: Windows 7.
- Ngôn ngữ lập trình: C#.
- Nền tảng .NET Framework 4.0.
- Công cụ hỗ trợ soạn thảo và biên dịch: Visual Studio 2012.
- Hệ quản trị CSDL MSSQL Server Express 2008.
- Máy chủ web IIS 8.0 Express.
- Thư viện MS Sync Framework 2.1.
- Thư viện DevExpress 13.2.9.
- Thư viện Bootstrap 3.0.
- Trình quản lý mã nguồn (Source control): Git từ nhà cung cấp Github, Inc.

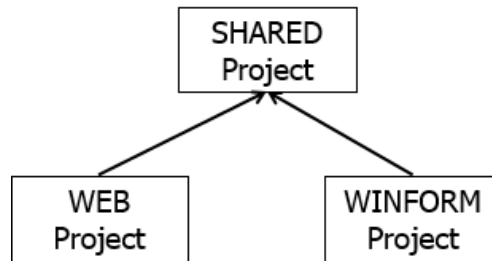
### **2.2. Mô hình tổ chức ứng dụng**

#### **2.2.1. Tổ chức ứng dụng theo hướng module hóa trong lập trình đa nền tảng (cross-platform)**

- Công nghệ .NET có thể được vận hành trên nhiều nền tảng hệ thống, trong đó phổ biến nhất là trên nền Windows Desktop Application, Windows Mobile Application, Windows Web Server IIS và thậm chí trên một số hệ thống hiếm như .NET MonoDevelop trên MAC OS,...
- Do đó, việc tổ chức các thành phần (module) riêng biệt sẽ giúp tận dụng được lại mã nguồn cho các ứng dụng chạy trên các nền tảng khác nhau. Đồng thời tăng tính thống nhất trong chương trình.

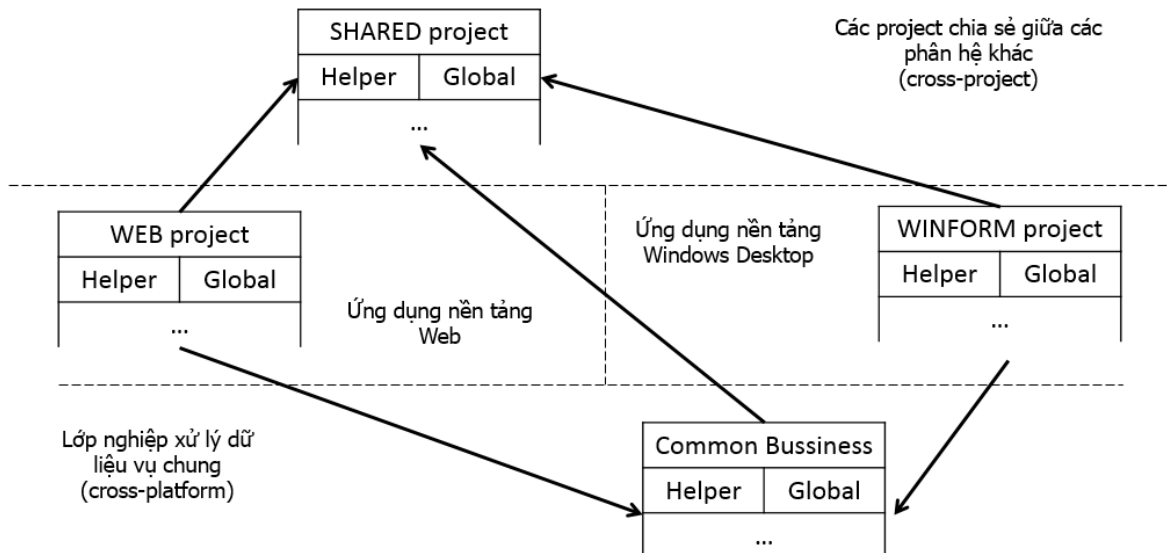
- Mô hình phân tích hướng module

+ Mô hình quan niệm:



Sơ đồ 2.1: Mô hình quan niệm tổ chức ứng dụng hướng module hóa

+ Mô hình chi tiết:



Sơ đồ 2.2: Mô hình chi tiết tổ chức ứng dụng hướng module hóa

### 2.2.2. Mô hình ba lớp trong ứng dụng hướng dữ liệu

- Các ứng dụng mà công việc diễn ra ở nhiều tầng khác nhau mà trong đó chức năng của mỗi tầng hầu như hoàn toàn độc lập thì việc tách các tầng này ra thành các lớp chức năng sẽ khiến cho công việc bảo trì và nâng cấp diễn ra nhanh chóng và nhẹ nhàng hơn. Ứng dụng hướng dữ liệu là một trong những dạng này<sup>[19]</sup>.

- Mô hình 3 lớp phổ biến bao gồm:

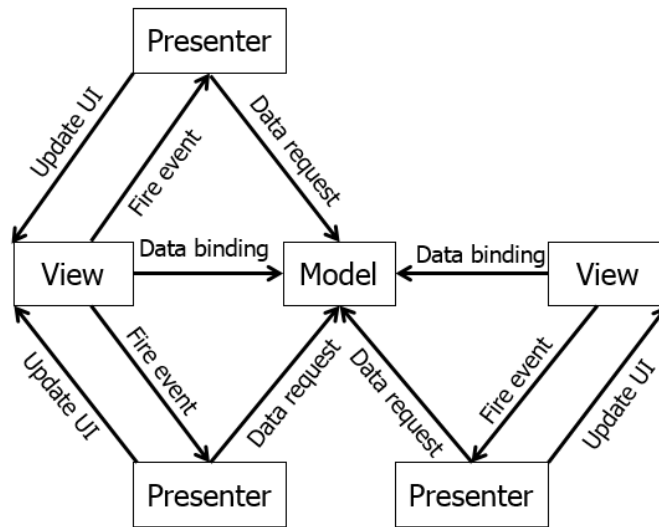
+ Tầng truy xuất dữ liệu (Data Access Layer - DAL).

- + Tầng nghiệp vụ chức năng (Business Logic Layer - BLL).
- + Tầng hiển thị dữ liệu (Presentation Layer - PL).
- Tùy thuộc vào điều kiện sẵn có của hệ thống mà ta sẽ sử dụng những tầng đã có và chỉ thiết kế cho những thành phần chưa có.
- Cụ thể trong đề tài này, khi áp dụng công nghệ EF thì mặc định đã có sẵn tầng truy xuất dữ liệu Data Access Layer, nên ta chỉ việc kết hợp với 2 tầng còn lại của ứng dụng là được.
- Hai tầng trong ứng dụng này bao gồm:
  - + Tầng nghiệp vụ chức năng (Business Logic Layer - BLL): chính là các lớp thực thể và các nghiệp vụ được định nghĩa trên thực thể. Đây là trái tim của hệ thống, mọi thao tác logic về xử lý và điều khiển luồng dữ liệu đều được diễn ra ở đây.
  - + Tầng hiển thị dữ liệu (Presentation Layer - PL): chính là các giao diện ứng với các nền tảng Windows Desktop hay Web,... Hiện ứng dụng có 3 lớp hiển thị ở tầng này (lớp ứng dụng Windows Desktop, lớp ứng dụng Web Desktop và lớp ứng dụng Web Mobile dành cho thiết bị di động).

### **2.2.3. Mô hình MVP (Model-View-Presenter) dành cho ứng dụng Winform Desktop**

- Trong hầu hết trường hợp, các giao diện cần được tái sử dụng nhằm đảm bảo tính thống nhất và bảo trì mã nguồn thuận lợi. Khi đó cần tìm giải pháp tách riêng giao diện ra khỏi các thành phần xử lý. Mô hình MVP là một trong những giải pháp đó.
- Các tính năng MVP:
  - + Tận dụng lại các thiết kế giao diện.
  - + Tận dụng lại các logic lập trình trên giao diện.
  - + Tạo sự thống nhất trong giao diện giữa các dự án (project).

- Mô hình MVP



Sơ đồ 2.3: Mô hình MVP

- Custom User control: Coi như là một module MVP nhỏ được tái sử dụng cho nhiều khu vực.

#### 2.2.4. Mô hình ASP.NET Webform dành cho ứng dụng Web

+ Website Quản lý tài sản cố định sử dụng ASP.NET WebForm.

- Các tính năng khác của WebForm:

+ Sử dụng các thiết kế User Control (các control riêng do lập trình viên tự phát triển) chỉ có ở WebForm.

+ Do đặc điểm Code behind (lập trình tách biệt khỏi giao diện) theo kiểu Event - driven (hướng sự kiện) sẽ thích hợp cho những tác vụ đòi hỏi giao tiếp nhiều giữa máy chủ và máy khách.

## **2.3. Các công nghệ và kỹ thuật lập trình được áp dụng**

### **2.3.1. Công nghệ Entity Framework (EF) trong lập trình dữ liệu hướng đối tượng (OOP)**

#### **2.3.1.1. Tổng quan**

- EF Là một DB ORM (Database Object - Relational Mapping) giúp ánh xạ cơ sở dữ liệu quan hệ lên một khung nhìn các đối tượng và tập hợp các đối tượng<sup>[13]</sup>.
- EF là một mã nguồn mở được chính Microsoft phát triển dựa trên nền tảng .NET.
- EF giúp người lập trình tiết kiệm được rất nhiều thời gian và công sức trong việc truy xuất và làm việc với các CSDL quan hệ, bởi vì bản thân EF đã cung cấp đầy đủ các tính năng và công nghệ tiên tiến giúp độc lập hóa khối CSDL ra khỏi lớp truy xuất.
- Được ứng dụng nhiều trong các nền tảng dạng Domain - Driven Design, cho phép người lập trình ảo hóa CSDL vật lý, giảm sự phụ thuộc vào CSDL vật lý, từ đó lập trình viên có thể làm việc trong suốt với CSDL nền, không quan tâm Hệ quản trị CSDL đích hay phiên bản khác nhau, miễn được EF hỗ trợ.
- Phiên bản mới nhất: 6.1.1.
- Phiên bản sử dụng trong đề tài: 6.1.1.

#### **2.3.1.2. Mô hình triển khai Code first**

- Thiết kế cấu trúc CSDL bằng cách định nghĩa các lớp (class) trước (Code first): là một cách tiếp cận mới trong việc thiết kế cho các ứng dụng hướng CSDL, thay vì thiết kế cấu trúc CSDL vật lý trước thì Code first cung cấp các đặc tả để người lập trình có thể định nghĩa cấu trúc CSDL bằng các lớp và các logic, ràng buộc trên lớp; tận dụng được các tính năng kế thừa trên lớp, sau đó EF sẽ tự động ánh xạ bản thiết kế xuống cấu trúc CSDL.
- Code first trong bản thiết kế CSDL mới hoàn toàn (Code first to new Database): EF hỗ trợ 2 giải pháp làm việc với CSDL.

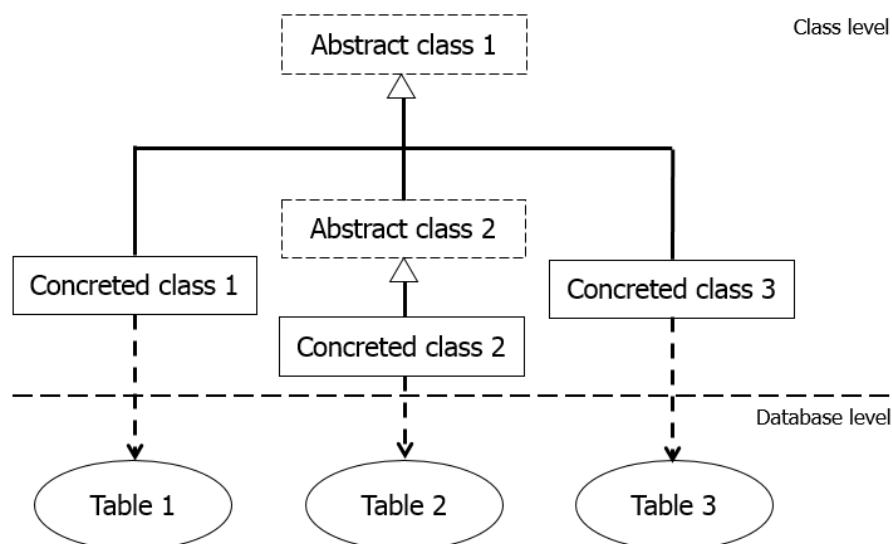
+ Giải pháp 1: Đối với CSDL đã có từ trước, người lập trình nếu muốn áp dụng Code first thì cần phải tìm cách đưa bản thiết kế CSDL vật lý lên mức lớp trong mã nguồn, sau đó chỉ định bản thiết kế lớp này ứng với cấu trúc CSDL vật lý hiện tại. Kể từ thời điểm này, thiết kế sẽ được thực hiện trên mức lớp trước, sau đó sẽ được ánh xạ lại trong cấu trúc CSDL vật lý.

+ Giải pháp 2: Nếu chưa có bản thiết kế CSDL vật lý sẵn, thì lập trình viên chỉ cần tạo bản thiết kế lớp trong mã nguồn như mong muốn, sau đó gọi trình ánh xạ của EF để tạo cấu trúc CSDL vật lý. Đề tài này do được nghiên cứu mới hoàn toàn nên giải pháp 2 được áp dụng.

### 2.3.1.3. Ánh xạ TPC (Table Per Concrete class) trong kế thừa thuộc tính

- Định nghĩa: TPC là cách thiết kế mà trong đó mỗi lớp thực (lớp cứng) sẽ ánh xạ thành một bảng trong CSDL vật lý. Tận dụng được các đoạn mã logic bằng cách thiết kế mô hình kế thừa, đa hình<sup>[20]</sup>.

- Trong TPC, các lớp ảo (abstract) sẽ không được ánh xạ xuống CSDL vật lý, các lớp này chỉ dùng cho mục đích kế thừa. Muốn ánh xạ một thực thể xuống CSDL vật lý thì lớp thực thể đó bắt buộc phải là một lớp cứng.



Sơ đồ 2.4: Mô hình triển khai TPC trong kế thừa



#### 2.3.1.4. Các cách biểu diễn quan hệ 1 - n hoặc n - n trong EF

- Gọi tập hợp  $A = \{ \text{Table } A_0, \text{Table } A_1, \dots, \text{Table } A_m \}$  chứa các thực thể quan hệ nhiều.
- Gọi tập hợp  $B = \{ \text{Table } B_0, \text{Table } B_1, \dots, \text{Table } B_n \}$  chứa các thực thể ở quan hệ 1.
- Một tích Đề - các diễn tả mối quan hệ 1 - n hoặc n - n từ  $B_i$  đến  $A_j$  ( $i < [B], j < [A]$ )  
 $B \times A \{ (b,a) \mid b \in B, a \in A \}.$

- Mỗi quan hệ 1 - n trong CSDL quan hệ có thể được biểu diễn bằng 2 cách sau:

+ Cách 1: Nâng cấp quan hệ 1 - n thành quan hệ n - n và ngầm định không sử dụng chiều tham chiếu ngược lại: khi đó mỗi quan hệ  $(B \times A)_i$  ( $i < [B \times A]$ ) sẽ liên kết với  $(1+1) + 1 = 3$  Table vật lý.

=> Tổng các bảng vật lý tối thiểu cần thiết để biểu diễn quan hệ trên  $(B \times A)$  là:

$$\text{Sum} = [A] + [B] + [B \times A]$$

=> Phù hợp với trường hợp mỗi quan hệ cần có thêm các thuộc tính kèm theo.

+ Cách 2: Truyền thống, đối tượng ở quan hệ nhiều sẽ có n khóa ngoại trỏ đến đối tượng ở quan hệ 1: khi đó mỗi quan hệ  $(B \times A)_i$  ( $i < [B \times A]$ ) sẽ liên kết ứng với  $1 + 1 = 2$  bảng vật lý.

=> Tổng các bảng vật lý tối thiểu cần thiết để biểu diễn quan hệ trên  $(B \times A)$  là:

$$\text{Sum} = [A] + [B]$$

\* EF làm việc được với cả 2 cách biểu diễn trên. Tùy thuộc vào từng trường hợp cụ thể mà lựa chọn phương án thích hợp. Nhóm sử dụng cách 1 trong việc biểu diễn mối quan hệ giữa chứng từ và tập tin đính kèm (attachment).

- Mỗi quan hệ n - n trong CSDL quan hệ có thể được biểu diễn bằng 2 cách:

+ Cách 1: tương tự cách 1 trong biểu diễn quan hệ 1 - n (nhưng vì đã là quan hệ n - n nên không cần nâng cấp).

+ Cách 2: Sử dụng thuộc tính đa trị trong từng trường định nghĩa khóa ngoại, tuy nhiên sẽ vi phạm dạng chuẩn CSDL 1NF (dạng chuẩn thấp nhất) do chứa thuộc tính đa trị, và không được EF hỗ trợ, nên không khả thi khi triển khai.

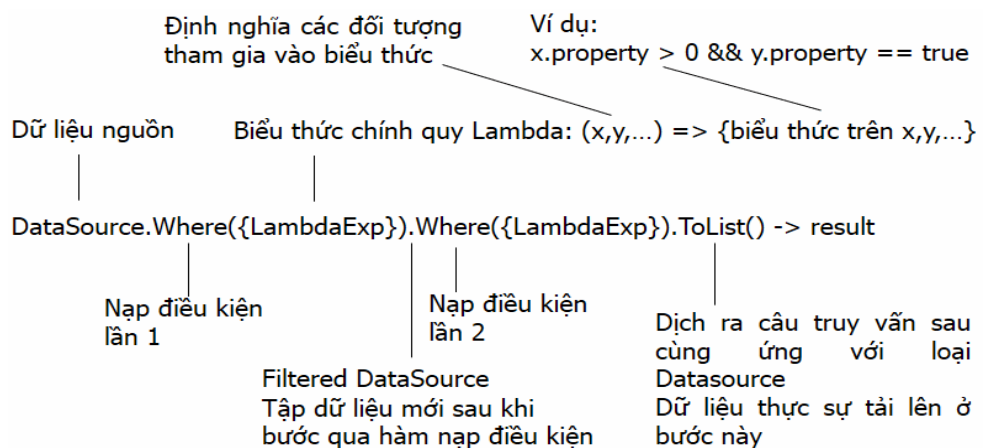
### 2.3.1.5. Công nghệ truy vấn LINQ

- LINQ to Entities: EF sử dụng LINQ để truy vấn trên CSDL. LINQ là một công nghệ truy vấn CSDL không phụ thuộc vào dữ liệu nguồn, tức là có thể dùng LINQ để truy vấn và thực thi các thao tác dữ liệu trên: tập hợp/danh sách các đối tượng, tập tin XML hay các hệ quản trị CSDL khác nhau.

- LINQ IEnumerable và biểu thức chính quy Lambda

+ LINQ IEnumerable: là một giao diện (interface) định nghĩa trong LINQ, cho phép tạo ra các câu truy vấn lồng nhau trên một tập hợp có thực thi lớp giao diện này. Kết quả của một loạt các thực thi sẽ được trả về ngay sau khi có lời gọi để chuyển từ tập hợp không chính quy (non-generic) sang tập hợp chính quy (generic). Đây là tính năng rất hay trong LINQ, mà nhờ đó tiết kiệm được chi phí cũng như thời gian thực thi dữ liệu, vì câu truy vấn cuối cùng chỉ được kết lại và gọi chạy khi hoàn tất một loạt các truy vấn lồng nhau thay vì phải tải dữ liệu lên sau mỗi bước nạp điều kiện.

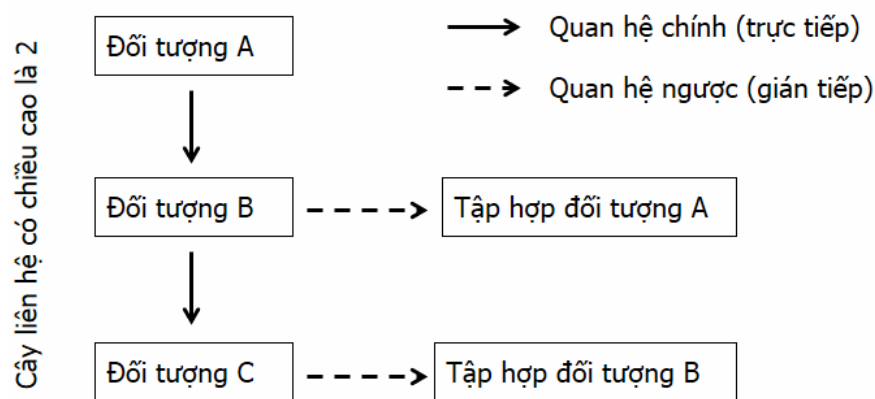
+ Lambda là một dạng biểu thức chính quy được sử dụng trong các truy vấn đến các lớp dữ liệu có đăng ký thực thi lớp IEnumerable.



Sơ đồ 2.5: Cách hoạt động của kỹ thuật truy vấn lồng

### 2.3.1.6. Các tính năng khác được áp dụng trong đề tài

- Trì hoãn tải dữ liệu (Lazy loading): Trì hoãn tải dữ liệu khi chưa cần thiết, dữ liệu chỉ được tải khi có lời gọi tới lần đầu tiên (Giảm thời gian nạp dữ liệu ban đầu). Lazy loading rất hữu ích trong trường hợp các đối tượng có mối quan hệ phức tạp qua lại với nhau và cây liên hệ có chiều cao lớn hơn một. Mặc định EF được cấu hình bật Lazy loading.



Tham chiếu mức 2 từ A đến C có dạng: A->B->C

Sơ đồ 2.6: Tham chiếu ngược trên các quan hệ 1 - n, n - n

- Phiên bản CSDL (Database Version): (Upgrade/Downgrade/Rebase): EF xem mỗi sự thay đổi trong bản thiết kế lớp (class) sẽ tương ứng với một phiên bản CSDL mức vật lý, khi bản thiết kế có sự thay đổi, EF sẽ ghi nhận lại sự thay đổi đó và tạo ra các đoạn mã để điều chỉnh cấu trúc CSDL hiện tại lại cho khớp với bản thiết kế mới. Những đoạn mã này được gọi là một phiên bản của CSDL, và sẽ được định danh bằng tên mã nhằm phục vụ cho quá trình dịch chuyển phiên bản.

- Dịch chuyển phiên bản CSDL (Migration to database Version):

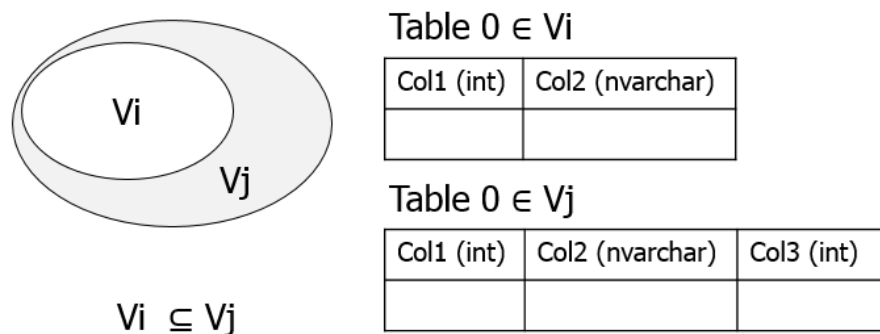
- + Nâng cấp (Upgrade): là hình thức dịch chuyển từ phiên bản thấp  $V_i$  lên phiên bản cao hơn  $V_j$  ( $i < j$ ).
- + Hạ cấp (Downgrade): là hình thức di chuyển từ phiên bản cao  $V_j$  xuống phiên bản thấp hơn  $V_i$  ( $i < j$ ).

\* EF cho phép nhảy cóc giữa các phiên bản, ví dụ có thể nhảy từ phiên bản v3 xuống v1 mà không cần thông qua v2 và ngược lại, miễn là các v1, v2, v3 đã được định danh trước đó. Tính năng này rất hữu ích khi dự án được thực hiện ở quy mô nhóm, khi đó các lập trình viên sẽ tiết kiệm được rất nhiều thời gian cho khâu cập nhật CSDL trong suốt quá trình lập trình và kiểm thử.

- Sự tương thích giữa các phiên bản CSDL: đây cũng là vấn đề được quan tâm tới trong khi thiết kế và vận hành ứng dụng. EF chỉ có thể làm việc trên phiên bản CSDL  $V_i$  nếu  $V_i$  tương thích với phiên bản CSDL hiện tại (ứng với thiết kế lớp (class) hiện tại). Tùy thuộc vào sự thay đổi giữa các phiên bản mà có thể tương thích hoặc không tương thích cụ thể như sau:

- Xét 2 phiên bản CSDL  $V_i$  và  $V_j$  ( $i < j$ ):

+ Tương thích ngược:  $V_j$  được xem là tương thích ngược với  $V_i$  nếu CSDL  $V_j$  bao trùm CSDL  $V_i$ .



Sơ đồ 2.7: Tương thích ngược trong phiên bản CSDL

+ Tương thích xuôi:  $V_i$  được xem là tương thích xuôi với  $V_j$  nếu CSDL  $V_i$  bao trùm CSDL  $V_j$  (điều này rất hiếm khi xảy ra vì đa phần bản thiết kế mới  $V_j$  luôn mở rộng hơn so với bản cũ  $V_i$ ).

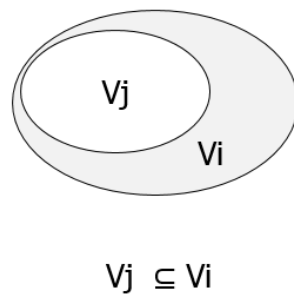


Table 0 ∈ Vi

Col1 (int)	Col2 (nvarchar)	Col3 (int)

Table 0 ∈ Vj

Col1 (int)	Col2 (nvarchar)

Sơ đồ 2.8: Tương thích xuôi phiên bản CSDL

+Không tương thích: Trong các trường hợp còn lại.

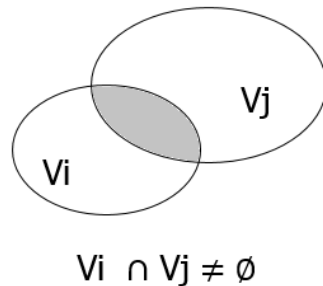


Table 0 ∈ Vi

Col1 (int)	Col2 (nvarchar)

Table 0 ∈ Vj

Col1 (int)	Col21 (int)	Col3 (int)

Sơ đồ 2.9: Không tương thích phiên bản CSDL (trường hợp 1)

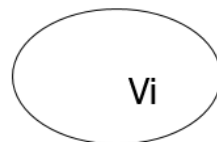


Table 0 ∈ Vi

Col1 (int)	Col2 (nvarchar)

Table 0 ∈ Vj

Col11 (int)	Col21 (int)	Col3 (int)

Sơ đồ 2.10: Không tương thích phiên bản CSDL (trường hợp 2)

\* Các ví dụ minh họa trên chỉ thể hiện ở mức đơn giản nhất, trên thực tế việc xét tính tương thích sẽ phức tạp hơn nhiều do sự kết hợp của nhiều bảng và định nghĩa các thuộc tính trên bảng.

- Kiểm định mô hình (Model checking): Kiểm tra cấu trúc CSDL có bị thay đổi bởi các tác nhân ngoài hệ thống hay không? Đảm bảo hệ thống làm việc ổn định và đúng đắn. Model cheking là cực kỳ quan trọng khi làm việc với EF bởi vì các trường thuộc

tính của dữ liệu vật lý gắn chặt với các lớp tương ứng của ứng dụng khi bộ máy ánh xạ hoạt động, nên chỉ với một thay đổi nhỏ về định nghĩa của CSDL vật lý cũng sẽ khiến EF không hoạt động.

- Trình khởi tạo CSDL tùy biến (Custom Database Initializer)<sup>[11]</sup>: Chỉ định phương thức khởi tạo CSDL, EF cho phép lựa chọn và tùy biến nhiều chỗ trong quá trình tạo tự động cấu trúc CSDL khi ánh xạ bản thiết kế xuống CSDL mức vật lý, các tính năng bao gồm: tự động tạo CSDL nếu chưa có, tự động tạo bảng nếu chưa có, tự động tạo dữ liệu mẫu ban đầu,...)

- + CreateDatabaseIfNotExists: mặc định của EF. Tự động tạo CSDL nếu chưa có.

- + DropCreateDatabaseIfModelChanges: tự động tạo lại CSDL khi cấu trúc bị thay đổi. Tuy nhiên không an toàn vì chỉ một sơ suất nhỏ trong khâu thiết kế cũng sẽ dẫn đến nguy cơ mất CSDL.

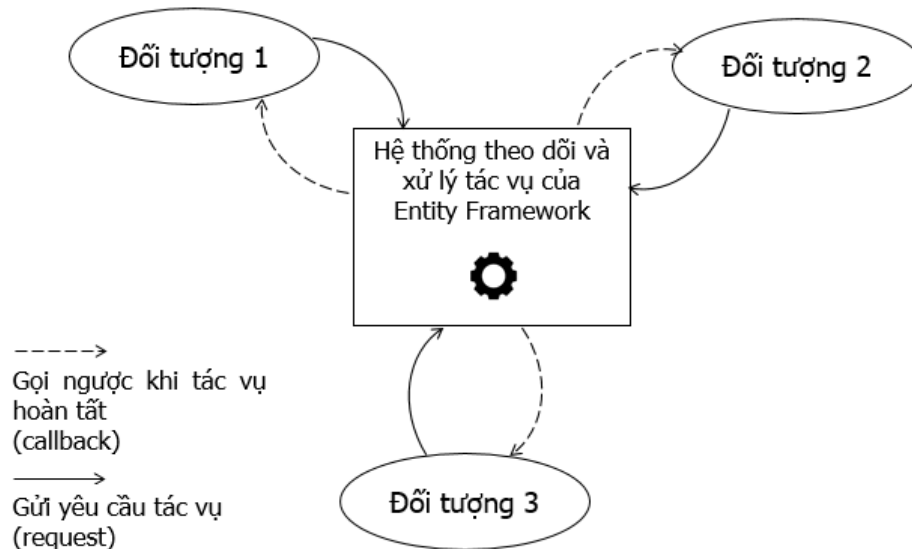
- + DropCreateDatabaseAlways: luôn luôn xóa và tạo mới lại CSDL mỗi khi ứng dụng khởi chạy, phù hợp với các ứng dụng sử dụng CSDL như là một bộ nhớ tạm trong lúc làm việc, và muốn CSDL rỗng cho mỗi phiên mới.

- + Custom DB Initializer: nếu các định nghĩa trên không đáp ứng được yêu cầu thì người lập trình có thể tự định nghĩa một trình khởi tạo dữ liệu riêng cho CSDL.

- Tạo dữ liệu mẫu mặc định (Data Seeding): cung cấp các đặc tả để tạo dữ liệu mặc định ban đầu khi tạo mới một CSDL, ví dụ: Tài khoản quản trị mặc định, các giá trị cài đặt mặc định,...

- Mô hình dữ liệu hướng sự kiện (Event-Driven model - BootStrapper): tương tự như mô hình lập trình giao diện hướng sự kiện được áp dụng trong WinForm, mô hình dữ liệu hướng sự kiện cũng cho phép các đối tượng tham gia đăng ký các sự kiện xảy ra khi hệ thống theo dõi của EF làm việc. Được ứng dụng trong các nghiệp vụ như: sau khi cập nhật thành công thì trường "date\_modified" sẽ có giá trị thời gian hiện tại,

sau khi sửa đổi một đối tượng sẽ ghi nhật ký hệ thống... Các nghiệp vụ này sẽ được thực thi một cách tuần tự và chính xác như mong muốn của lập trình viên.



Sơ đồ 2.11: Giao tiếp 2 chiều trong mô hình dữ liệu hướng sự kiện

- Khai báo proxy cho các thuộc tính (property virtual proxy): EF đòi hỏi thuộc tính khóa ngoại của các lớp phải được khai báo dạng ảo (virtual) để EF được phép tạo proxy ẩn trong các kỹ thuật theo dõi hay Lazy loading khi các thuộc tính này được truy xuất hoặc sửa đổi.
- Trình quản lý giao dịch (Transaction Manager) trong các kỹ thuật quay ngược (rollback): Trong một ứng dụng lớn, mỗi nghiệp vụ có thể có nhiều tác vụ con được thực thi. Hệ quản trị CSDL chỉ xử lý đơn nguyên dữ liệu (hoặc là tất cả thao tác trên dữ liệu đều được thực thi hoặc là không) ở mức rất thấp (mức tác vụ INSERT, DELETE,... trên từng đối tượng). Do đó, nếu muốn đảm bảo tính đơn nguyên ở mức cao hơn (cả một nghiệp vụ hoàn chỉnh) thì việc điều khiển và gọi trình quản lý giao dịch đúng thời điểm sẽ giải quyết được bài toán “đảm bảo toàn vẹn CSDL”. EF cung cấp một cơ chế Transaction rất đơn giản và hiệu quả.
- Trạng thái của đối tượng (Entity State) và tính năng cập nhật chọn lọc: Để có thể theo dõi được các sự thay đổi dữ liệu trên các đối tượng trong lúc thực thi (runtime),

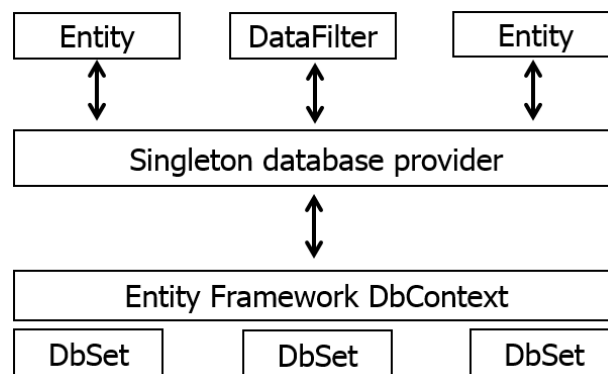
EF đưa ra định nghĩa về trạng thái của các đối tượng, trong đó một đối tượng có thể thuộc một trong các trạng thái sau:

- + Attached: Đối tượng mới khởi tạo và được đưa vào hệ thống theo dõi, tuy nhiên chưa được lưu xuống CSDL.
  - + Detached: Đối tượng đã bị loại khi hệ thống theo dõi.
  - + Added: Đối tượng được đưa vào hàng đợi, chờ thêm vào CSDL.
  - + Modified: Đối tượng được đánh dấu là đã bị thay đổi ít nhất một thuộc tính, được đưa vào hàng đợi chờ cập nhật xuống CSDL.
  - + Unchanged: Đối tượng được đánh dấu là sạch, có thể là mới được khởi tạo hoặc là mới được tải lên từ CSDL.
  - + Deleted: Đối tượng được đánh dấu là bị xóa, được đưa vào hàng đợi chờ xóa khỏi CSDL.
- Trình lọc dữ liệu (DataFilter) trong hiển thị dữ liệu: Một đối tượng chỉ mang các thuộc tính trực tiếp của bản thân nó, trong lập trình giao diện, thông thường khi hiển thị thông tin một đối tượng nào đó, ta thường hiển thị các thuộc tính gián tiếp (thuộc tính của khóa ngoại), do đó trình xử lý giao diện sẽ không làm việc trực tiếp với các đối tượng này mà làm việc thông qua một lớp mặt nạ (mask) gọi là lớp lọc dữ liệu (DataFilter), nhiệm vụ của DataFilter là kết các đối tượng có liên quan lại với nhau sau đó chọn ra các thuộc tính cần hiển thị.
- Ngữ cảnh CSDL (Database Context) và cơ chế hoạt động lớp truy xuất CSDL (Singleton Database Instance Provider):
- + EF xem Database Context là một ngữ cảnh truy xuất đến CSDL, trên đó chứa các định nghĩa về nguồn dữ liệu. Database Context là không gian làm việc của EF. Có thể có nhiều Database Context được định nghĩa trên cùng một CSDL.
  - + Singleton giúp các lớp thực thể nhìn thấy cùng một DbContext trong suốt phiên làm việc, bởi vì EF đòi hỏi các đối tượng sinh ra từ các lớp thực thể phải



thống nhất về DbContext, một đối tượng không thể được theo dõi bởi các DbContext khác nhau.

+ Phiên làm việc được đánh dấu từ lúc DbContext được khởi tạo cho đến khi bị hủy bỏ (Dispose), các đối tượng nằm ngoài phiên làm việc được xem là không hợp lệ và không có ý nghĩa về mặt dữ liệu, muốn làm việc lại trên các đối tượng này nhất thiết phải được tải lại trong một phiên làm việc DbContext khác.



Sơ đồ 2.12: Cách hoạt động giữa Singleton và DbContext

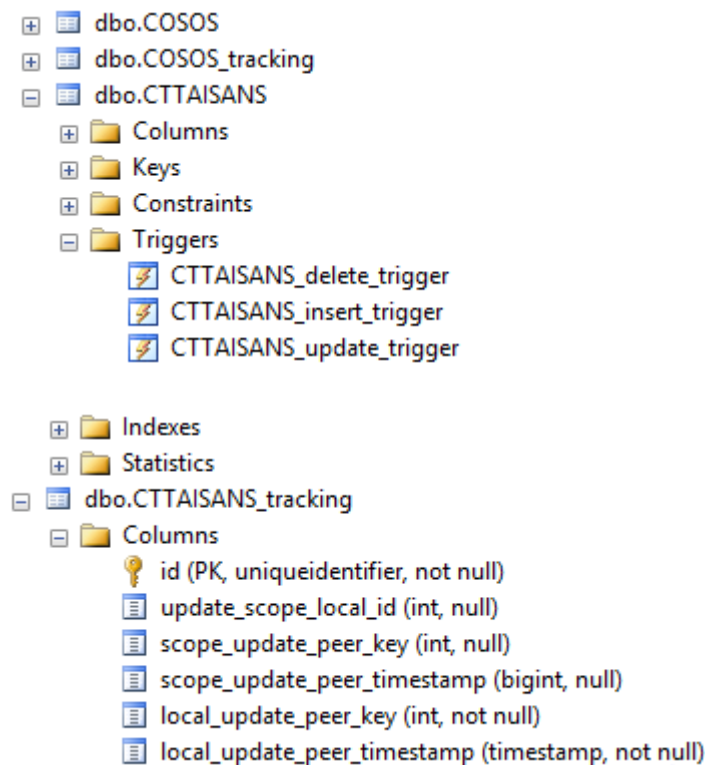
### 2.3.2. Công nghệ Sync Framework trong đồng bộ CSDL tập trung

#### 2.3.2.1. Tổng quan

- Định nghĩa: Sync Framework là công nghệ được Microsoft phát triển với mục đích chính là đồng bộ dữ liệu qua lại giữa các nguồn dữ liệu. Dữ liệu nguồn ở đây có thể là hệ thống tập tin hoặc là một CSDL của một hệ quản trị nào đó, hiện Sync Framework hỗ trợ các dữ liệu nguồn tương thích chuẩn ADO.NET mà trong đó hệ quản trị MSSQL Server hoàn toàn đáp ứng được các yêu cầu trên<sup>[14]</sup>.

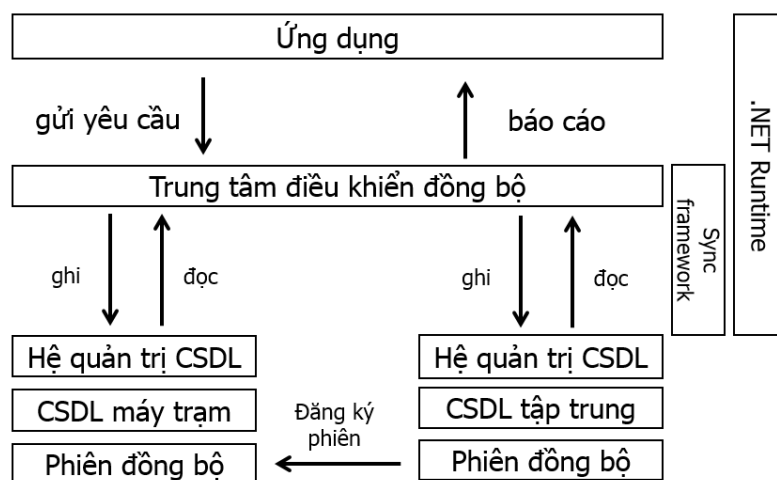
- Đồng bộ dữ liệu giữa các CSDL với mức đơn vị dữ liệu là bảng.

- Sử dụng kỹ thuật trigger trên từng bảng, trigger có nhiệm vụ thu thập và bắt các sự thay đổi về dữ liệu trên CSDL, sau đó lưu trữ lại trong các bảng theo dõi (tracking table, mỗi bảng được chỉ định trong Sync Scope sẽ phát sinh ra một bảng theo dõi tương ứng) mà Sync Framework tạo ra khi một Sync Scope được cài đặt.



Sơ đồ 2.13: Cách tổ chức lưu trữ của Sync Framework

- Nhờ các kỹ thuật lưu trữ và theo dõi dữ liệu như đã được giới thiệu ở trên mà Sync Framework sẽ chỉ đồng bộ những dữ liệu sai khác giữa các dữ liệu nguồn, do đó quá trình xử lý và truyền nhận dữ liệu sẽ tiết kiệm được thời gian và tài nguyên hệ thống, khác với các giải pháp truyền thống là phải tải mới toàn bộ dữ liệu.



Sơ đồ 2.14: Cách hoạt động của Sync Framework

- Phiên bản mới nhất hiện tại: 2.1

- Phiên bản sử dụng trong đề tài: 2.1

\* Trong phạm vi ứng dụng của đề tài này, sẽ chỉ xem xét đến nguồn dữ liệu là hệ quản trị CSDL, cụ thể là MSSQL Server.

#### **2.3.2.2. Đồ thị tiến trình đồng bộ trong kỹ thuật tránh deadlock**

- Đồ thị tiến trình đồng bộ: như đã đề cập ở phần trên, Sync Framework đồng bộ ở mức đơn vị dữ liệu là bảng, nên thứ tự các bảng trong một tiến trình đồng bộ là vô cùng quan trọng vì đặc thù ràng buộc khóa ngoại của CSDL quan hệ. Nếu bảng B có chứa khóa ngoại tham chiếu đến bảng A thì bảng A phải được xử lý trước bảng B.

=> Cần phải xây dựng đồ thị tiến trình cho phiên đồng bộ.

- Đặc điểm của đồ thị tiến trình đồng bộ:

+ Do thứ tự bảng trước sau được xem xét nên đồ thị là đồ thị có hướng.

+ Do trong CSDL quan hệ không được phép chứa có liên hệ vòng nên đồ thị là đơn đồ thị, không chứa chu trình (vì chu trình sẽ gây chết tiến trình đồng bộ (deadlock)) và không bao giờ là một đồ thị liên thông mạnh.

+ Đồ thị có thể liên thông hoặc không liên thông tùy thuộc vào sự giao nhau giữa các tập quan hệ bảng.

+ Đồ thị có thể có nhiều đồ thị con (các thành phần liên thông).

+ Các thành phần liên thông có thể là các cây, trường hợp này cây có thể được tách ra thành nhiều cây con để xử lý song song bằng các tiến trình song song hoặc cũng có thể được xử lý tuần tự bằng một tiến trình duy nhất, tùy thuộc thiết kế đơn luồng hay đa luồng lúc thực thi.

- Xem các bảng là các đỉnh (V - vertexes).

- Xem các liên hệ khóa ngoại từ bảng này đến bảng kia là các cung (E - edges), trong đó một cung có nút con là bảng chứa khóa ngoại và nút cha là bảng tham chiếu đến.

- Ta được đồ thị có hướng  $G = \{V, E\}$ , trong đó:

+ Tập hợp các nút:  $V = \{\text{Table } 0, \text{Table } 1, \dots, \text{Table } n\}$ .

+ Tập hợp các cung:  $E = \{e_0 = (V_i, V_j), e_1 = (V_p, V_q), \dots, e_m = (V_t, V_v)\}$   
 $(i, j, p, q, t, v < n)$ .

- Nguyên tắc xây dựng tiến trình đồng bộ đơn nhất:

+ Chỉ có một tiến trình P duy nhất xử lý cho toàn đồ thị.

+ Các nút trong từng đồ thị sẽ lần lượt được đưa vào danh sách hàng đợi Q sao cho mệnh đề sau luôn đúng:

“Với mọi  $Q_i, Q_j$  thuộc Q nếu cung  $e = (Q_i, Q_j)$  thuộc E thì j phải nhỏ hơn i”

(tức là nếu bảng A có chứa khóa ngoại đến bảng B thì bảng B phải được xử lý đồng bộ trước bảng A).

=> Phương pháp này có thể được thực hiện bằng giải thuật sắp xếp trong đó điều kiện so sánh là “xét cung tạo thành có thuộc đồ thị hay không”.

\* Bên cạnh cách xây dựng tiến trình đơn nhất thì còn một giải pháp khác là tiến trình song song. Trong tiến trình song song, người ta quan tâm tới dạng đặc biệt của đồ thị là cây (đồ thị vô hướng nên không chứa chu trình), khi đó mỗi nhánh đồng cấp sẽ được chia thành các tiến trình con để xử lý đa luồng. Do phương pháp này phức tạp và chỉ phù hợp với ứng dụng quy mô lớn, nên nhóm chỉ dừng lại ở mức tham khảo.

- Không gian đồng bộ (Sync Scope)<sup>[15]</sup>:

+ Định nghĩa: Sync Scope là được hiểu như là định nghĩa về một phiên đồng bộ trên một CSDL cụ thể, chứa các thông tin về tập hợp bảng cần đồng bộ. Một CSDL có thể có nhiều Sync Scope được thiết lập sẵn thông qua việc mở rộng vùng lưu trữ trên CSDL bằng các bảng tạm và các procedure chức năng.

+ Cài đặt một Sync Scope lên CSDL có sẵn: là một loạt các thao tác cần thiết để khởi tạo và định danh một Sync Scope lên trên CSDL đã có sẵn dữ liệu hoặc

CSDL mới hoàn toàn, trong đó việc chỉ định danh sách các bảng cần đồng bộ được xem là quan trọng nhất, khi các phương thức khởi tạo Sync Scope được gọi, Sync Framework sẽ thực hiện nhiệm vụ còn lại.

+ Gỡ bỏ một Sync Scope có sẵn ra khỏi CSDL: thao tác gỡ bỏ Sync Scope sẽ ngược lại với các bước khi cài đặt trước đó. Chỉ cần chỉ định dữ liệu nguồn và tên Sync Scope, sau đó gọi phương thức gỡ bỏ, Sync Framework sẽ thực hiện nhiệm vụ còn lại. Việc gỡ bỏ Sync Scope sẽ đồng nghĩa với việc CSDL sẽ không thể cung cấp phiên đồng bộ cho các trình quản lý đồng bộ, và do đó sẽ không tham gia vào hệ thống.

+ Áp đặt Sync Scope giữa các CSDL (Fetching scope among databases): là sao chép các thông tin về các định nghĩa phiên, các cấu hình liên quan,... từ một Sync Scope trên CSDL A có sẵn sang một Sync Scope mới trên CSDL mới B, khai báo rằng CSDL A và B có thể “bắt tay” được với nhau, lúc này trình quản lý đồng bộ mới có thể nhìn thấy và làm việc được trên cả hai CSDL này, cũng có thể nói đây là bước thiết lập một cầu nối (pipeline) dữ liệu. Việc áp đặt Sync Scope từ một CSDL này lên một CSDL là rất quan trọng. Trong một tiến trình đồng bộ cụ thể thì thao tác này được thiết lập sớm nhất ngay khi có thể. Một CSDL có thể bắt tay với nhiều CSDL khác, đây là tính năng sẽ được ứng dụng trong các mô hình triển khai máy trạm khi đưa vào vận hành.

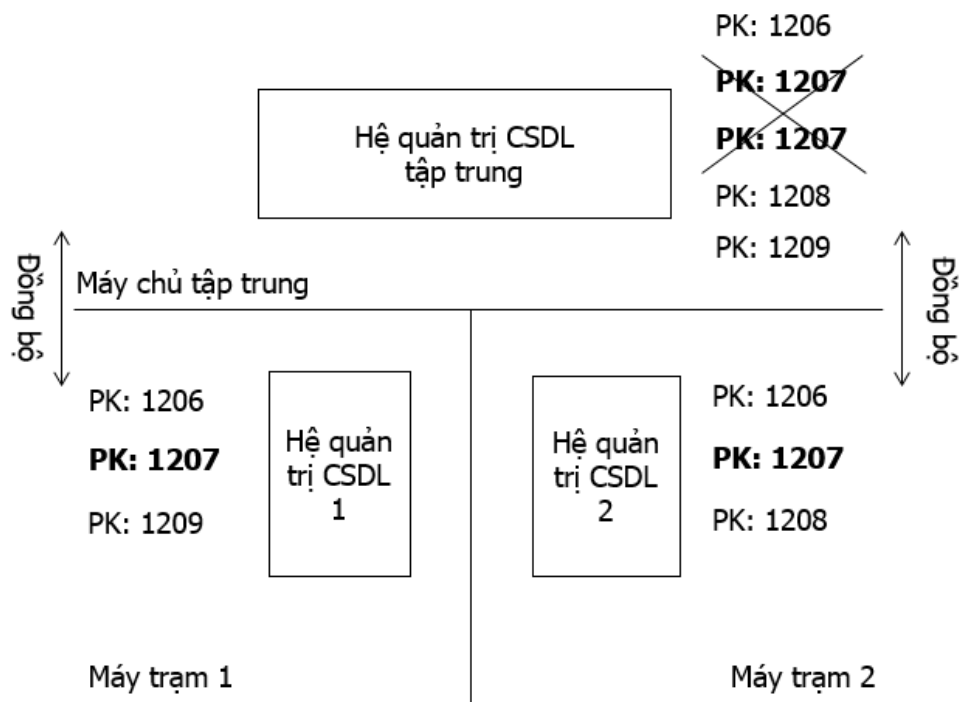
+ Hướng đồng bộ (chỉ lên/chỉ xuống/2 chiều), tính thông nhau giữa các cầu nối (Sync Direction (Up/Down/Bidirectional link)): khi một cầu nối được thiết lập giữa hai CSDL thì việc quy định hướng của luồng đồng bộ được xem xét và quyết định tùy thuộc vào yêu cầu về chức năng và quyền hạn của các máy trạm. Ví dụ: trong hệ thống có những máy trạm chỉ muốn sao chép dữ liệu từ máy chủ tập trung xuống chỉ để để xem và thống kê mà không có các thao tác thay đổi (read only) thì hướng đồng bộ chỉ xuống (Download only) được áp dụng.

\* Việc chọn giải pháp phù hợp sẽ làm tăng tính an toàn dữ liệu vì việc cập nhật sửa đổi đã được kiểm soát ở mức CSDL thấp hơn so với mức ứng dụng (Application), tính bảo mật dữ liệu cũng được tăng lên.

### 2.3.2.3. Đụng độ dữ liệu và giải pháp khóa chính GUID

- Đụng độ dữ liệu: trong hệ đồng bộ thì các đụng độ về mặt dữ liệu là không thể tránh khỏi. Đụng độ đặc trưng cho tính không nhất quán trên cấu trúc CSDL (ví dụ: có nhiều hơn 2 khóa chính trùng nhau trong cùng một bảng).

- Đụng độ vật lý trên khóa chính (primary key conflict)<sup>[14]</sup>: Xét hai CSDL độc lập có cùng cấu trúc bảng và dữ liệu, nếu khóa chính được thiết lập dạng tự động tăng (Auto Increasement) thì khi gọi phương thức chèn mới (INSERT) trên hai CSDL, hệ quản trị CSDL địa phương ở cả hai CSDL trên sẽ có khả năng tạo ra khóa chính trùng nhau rất cao (do tính tuần tự trong cấp phát). Nên khi đồng bộ dữ liệu sẽ bị trùng khóa chính, một trong hai dữ liệu mới đó sẽ không được hệ quản trị tiếp nhận.

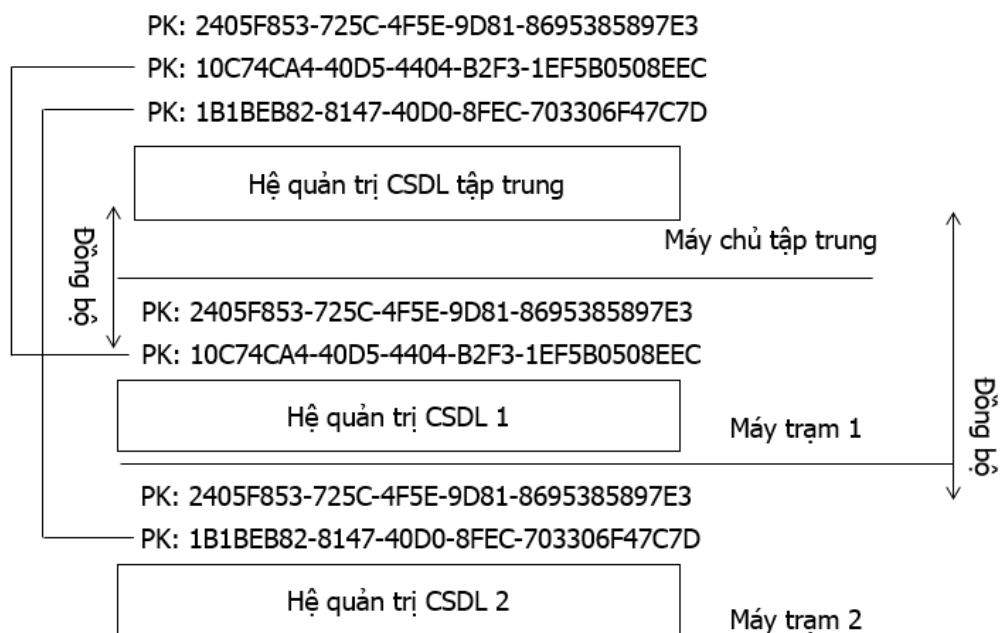


Hình 2.15: Minh họa sự đụng độ khoá chính trong đồng bộ dữ liệu

=> Sử dụng khóa chính kiểu GUID (Globally Unique Identifier)<sup>[14]</sup>: GUID là kiểu giá trị "tự nhiên" (Natural), được tạo ra dựa trên sự kết hợp giữa các giá trị định danh (địa chỉ MAC của card mạng) và ngẫu nhiên (thời gian hiện tại trên hệ quản trị CSDL), được nhiều hệ thống hỗ trợ. Hệ quản trị CSDL sẽ đảm bảo mỗi GUID được cấp phát sẽ là duy nhất trên toàn cầu (mặc dù khả năng trùng là có thể xảy ra trên lý thuyết nhưng thực tế có thể chấp nhận được tùy vào phạm vi sử dụng nội bộ hay liên mạng hay toàn cầu). GUID được chia làm hai loại lớn:

+ GUID ngẫu nhiên: các GUID được tạo ra không theo một trật tự nào cả. Sử dụng trong các trường hợp bảo mật cao (do rất khó để đoán được giá trị cấp phát tiếp theo). Tuy nhiên CSDL lưu trữ GUID dạng này sẽ bị phân mảnh rất nhiều.

+ GUID tuần tự: các GUID được tạo ra theo một trật tự. Giảm sự phân mảnh.



Hình 2.16: Minh họa GUID trong cơ sở dữ liệu

- Độ trễ logic do sự trễ (Delay) dữ liệu: một sự trễ dữ liệu được định nghĩa khi mà sự thay đổi về mặt CSDL ở một máy trạm khác (dù đã đẩy hay chưa đẩy lên máy chủ tập trung) chưa kịp cập nhật cho máy trạm địa phương (local machine) mà máy trạm

địa phương cũng đã tạo nên một sự sửa đổi. Khi mà đưng độ về khóa chính (đưng độ vật lý) đã được giải quyết thì đưng độ logic do sự chậm trễ trong việc cập nhật dữ liệu lên máy chủ tập trung là vấn đề không thể tránh khỏi do đặc thù làm việc khi không có mạng (Offline), và Sync Framework chỉ có thể can thiệp và xử lý tự động khi hạng mục dữ liệu bị thay đổi ở cả 2 nguồn là không giao nhau hoặc hạng mục dữ liệu giao nhau là đơn nhất. Đối với các trường hợp sự thay đổi ở một hạng mục kéo theo sự thay đổi ở các hạng mục liên quan thì Sync Framework không thể can thiệp được.

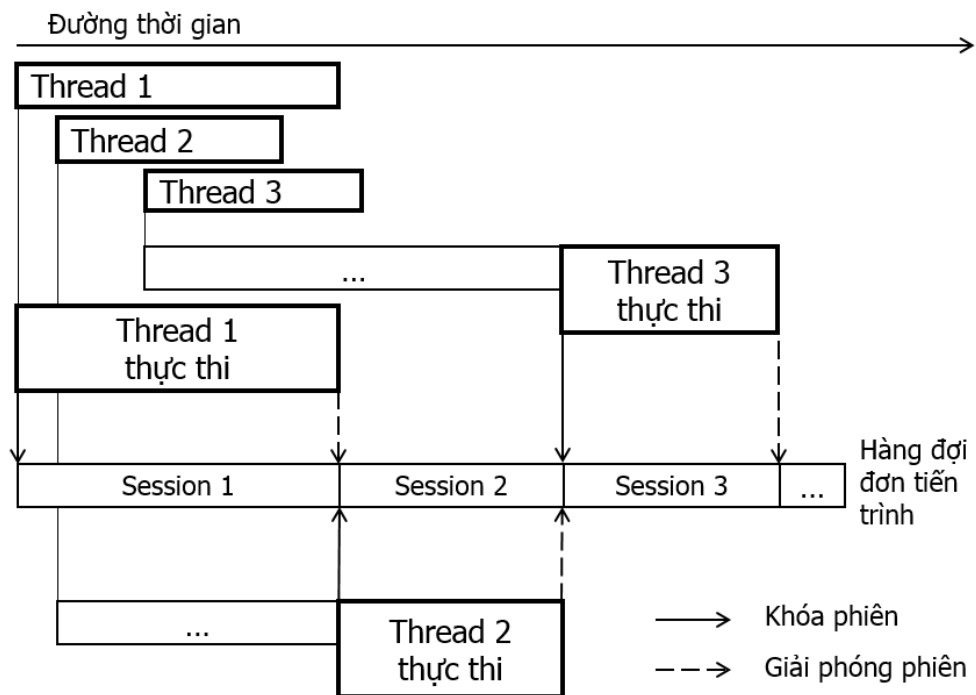
- Đây cũng là cách làm việc chung cho các hệ đồng bộ hiện nay. Việc tránh đưng độ kiểu này phải do lập trình viên tự quy định các chính sách về đồng bộ dữ liệu giữa các máy trạm và máy chủ tập trung.

#### **2.3.2.4. Kỹ thuật khóa Semaphore trong xử lý đa luồng (multi thread) trên hàng đợi (queue)**

- Tác vụ bất đồng bộ (asynchronous action) là các tác vụ được gọi chạy nền bằng các luồng riêng biệt mà không cần biết kết quả trả về (cách hoạt động gần giống giao thức UDP trong truyền tin không xác báo).

- Khóa semaphore giúp ngăn chặn (trong semaphore đơn tiến trình) hoặc giảm thiểu (trong semaphore đa tiến trình) sự đưng độ trong truy cập các tài nguyên chia sẻ từ các tác vụ bất đồng bộ (tạo nên bởi các luồng xử lý song song) bằng cách tạo ra các chính sách về phiên và các giao tác trên hàng đợi.





Hình 2.17: Minh họa semaphore trong xử lý đa luồng trên hàng đợi

- Có thể không cần quản lý các luồng truy xuất dữ liệu song song, vì bản thân hệ quản trị CSDL đã có các chính sách về khóa trên dữ liệu, tuy nhiên lúc này sẽ gây tải lên cả hệ quản trị CSDL địa phương lẫn máy chủ tập trung. Chúng ta nên có giải pháp quản lý ở mức ứng dụng nhằm tránh gây ra các tải không cần thiết lên máy chủ, nhất là trong môi trường liên mạng (tiết kiệm được tài nguyên hệ thống, ví dụ: băng thông đường truyền).

- Cụ thể trong chính sách đồng bộ dữ liệu với máy chủ tập trung: Nhằm đảm bảo dữ liệu có tính sẵn sàng cao và nhất quán, các chính sách sau cần được áp dụng trong kỹ thuật xử lý đồng bộ:

- + Luôn đồng bộ dữ liệu mới nhất từ máy chủ tập trung về trước khi thực hiện các sửa đổi trên CSDL (nhằm hạn chế sự trễ dữ liệu).
- + Sau khi một máy trạm hoàn tất một thao tác sửa đổi trên CSDL thì phải đồng bộ lên máy chủ tập trung ngay lập tức (nhằm đảm bảo hệ thống luôn phản ánh đúng CSDL hiện tại).

### 2.3.3. Công nghệ DevExpress trong lập trình giao diện

#### 2.3.3.1. Tổng quan

- Định nghĩa: DevExpress là một Framework được viết cho nền tảng .NET Framework. Nó cung cấp các control và công nghệ để phục vụ cho quá trình phát triển phần mềm<sup>[5]</sup>.



*Hình 2.18: Logo Devexpress*

- Thành phần của DevExpress gồm có:

- + WinForms Controls: Cung cấp các control cho WinForms.
- + ASP.NET Controls: Cung cấp các control cho WebForms.
- + WPF Controls: Cung cấp các control cho WPF.
- + Silverlight Controls: Cung cấp các control cho Silverlight.
- + XtraCharts: Control cung cấp các loại biểu đồ.
- + XtraReports: Cung cấp các control tạo báo cáo.
- + XPO: Cung cấp môi trường làm việc với database.
- + XAF: Một công nghệ mới giúp việc phát triển phần mềm một cách nhanh chóng.
- + Phần mềm chỉ sử dụng ba controls của Devexpress (WinForms, ASP.NET và XtraReports).



*Hình 2.19: Danh sách Control mà Devexpress hỗ trợ*

- Tính năng: Trải qua hàng loạt phiên bản, DevExpress đã từng bước được nâng cấp, hoàn thiện và thêm mới rất nhiều chức năng. Với phiên bản DevExpress 13.2.9 hiện tại, DevExpress đã cung cấp những công cụ, môi trường tuyệt vời để biến những ý tưởng của lập trình viên thành hiện thực một cách nhanh chóng và dễ dàng.

+ Sử dụng công cụ của Devexpress, bạn không phải tốn nhiều thời gian để thiết kế giao diện hay chức năng, ngoài ra nó giảm được khả năng gây lỗi khi sử dụng công cụ của Devexpress.

- Phiên bản mới nhất: 14.2.

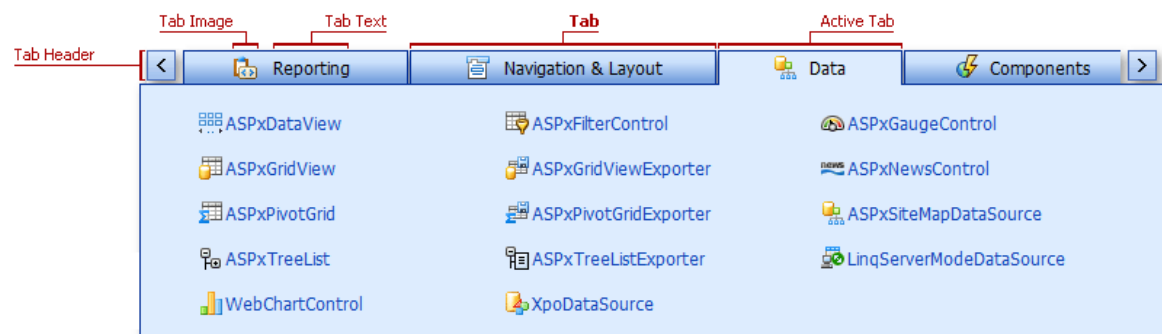
- Phiên bản sử dụng trong đề tài: 13.2.9<sup>[12]</sup>.

### 2.3.3.2. DevExpress dành cho ASP.NET WebForm

- Định nghĩa: Devexpress ASP.NET là một control của Devexpress cho phép các control phục vụ cho Web ASP.NET (bao gồm cả WebForm và MVC (razor và ASP)). Sử dụng control của Devexpress ASP.NET làm cho phần mềm thêm sinh động và chuyên nghiệp hơn.

- Các controls mà website sử dụng từ Devexpress:

✓ ASPxTabControl



Hình 2.20: Minh họa giao diện ASPxTabControl

- Bộ ASPxTabControl bao gồm 2 thành phần nhỏ, giúp ta tạo nên các tab cho trang web của mình. Có thể dùng thành phần APSxTabControl chỉ để thể hiện các Tab hoặc

dùng ASPxPageControl để tạo các tab cùng với nội dung bên trong của từng tab. Cả hai thành phần này đều được hỗ trợ AJAX qua phương thức Callback<sup>[1]</sup>.

- Đặc điểm:

- + Các templates có thể xác định cho từng tab trong cả trạng thái kích hoạt hay không kích hoạt.
- + Có thể thay đổi giao diện, sự thể hiện của từng thành phần một cách trực tiếp qua các thuộc tính hoặc qua CSS.
- + Nhiều định dạng phong phú, dễ dàng chọn và thay đổi.
- + Hỗ trợ hai cách để mở một tab: click chuột hoặc chỉ rê chuột lên trên tab.
- + Để có hỗ trợ AJAX ta set thuộc tính EnableCallbacks thành True, lúc này ASPxPageControl chỉ tải nội dung của tab được mặc định mở chứ không tải hết toàn bộ nội dung của các tab khi hiển thị ở phía người dùng. Khi người dùng nhấn chọn mở các tab, nội dung bên trong sẽ được tải qua sự kiện callback. Một khi nội dung một tab đã được tải lên rồi, người dùng có thể mở hoặc đóng tab này mà không phải gọi sự kiện callback hay postback nữa.

- Tính năng:

- + Dễ dàng chỉnh sửa, thay đổi với trình chỉnh sửa thông minh.
- + Có thể tùy chỉnh khoảng cách giữa các tab.
- + Dễ dàng tùy chỉnh vị trí của các tab.
- + Có thể chèn hình ảnh đại diện cho từng tab.

## ✓ ASPxGridView

Country ▲ City ▲					
	Company Name ▲	Region	Unit Price	Quantity	Total
+ Country: Argentina (\$8,119.10, Count=34)					
- Country: Austria (\$139,496.63, Count=125) (Continued on the next page)					
- City: Graz (\$113,236.68, Count=102) (Continued on the next page)					
	Ernst Handel		21	52	\$1,092.00
	Ernst Handel		18	6	\$108.00
	Ernst Handel		19.5	24	\$468.00
	Ernst Handel		34	60	\$2,040.00
	Ernst Handel		33.25	30	\$997.50
	Ernst Handel		4.5	110	\$495.00
	Ernst Handel		18	45	\$810.00
Count=2155					\$1,354,458.59
Page 1 of 13 (125 items) < [1] 2 3 4 5 6 7 ... 11 12 13 >					

Hình 2.21: Minh họa giao diện ASPxGridView

- ASPxGridView là một Control rất mạnh trong việc hỗ trợ hiển thị dữ liệu dạng lưới, cho phép ta tạo ra các trường hiển thị bằng tay hoặc thông qua CSDL<sup>[1]</sup>.

- Đặc điểm:

- + Tương thích với nhiều trình duyệt.
- + Hỗ trợ AJAX: ta có thể cập nhật nội dung của Control thông qua phương thức callback, không cần thiết phải tải lại toàn bộ trang.
- + Tương tác với người dùng rất đa dạng.
- + Xuất dữ liệu: hỗ trợ xuất dữ liệu ra định dạng PDF, XLS và RTF.
- + Hỗ trợ SEO (Search Engine Optimization): tối ưu hoá công cụ tìm kiếm.
- + Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, SQL Server.
- + Cho phép hiển thị dữ liệu dạng Master - Detail với cấu trúc đa dạng.
- + Có hai chế độ chỉnh sửa: từ Form chỉnh sửa hay chỉnh ngay trên hàng.
- + Chứng thực dòng dữ liệu và chỉ ra lỗi: ASPxGridView cho phép ta xác thực bằng tay các dòng đã chỉnh sửa, và hiển thị thông báo lỗi đối với trường không hợp lệ.

- + Tự động gom nhóm dữ liệu: Cho phép người dùng gom nhóm dữ liệu, không giới hạn số cột.
  - + Tóm tắt dữ liệu đầy đủ: Cho phép hiển thị thông tin thống kê như MIN, MAX, AVG, SUM và COUNT trực tiếp trên lưới.
  - + Cho phép lọc dữ liệu và hiển thị Text: Với mỗi cột ta có thể chỉ định cách dữ liệu được sắp xếp theo giá trị hiển thị của nó. Ngoài ra ta có thể cho phép lọc dữ liệu bất kỳ bằng cách gõ vào giá trị muốn lọc trực tiếp vào ô textbox.
  - + Cho phép lựa chọn nhiều dòng cùng một lúc.
- Tùy biến giao diện hiển thị:
- + Giao diện: Ta có thể tùy chỉnh giao diện của lưới bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.
  - + Hỗ trợ các Template: với mỗi thành phần bên trong ASPxGridView, ta có thể hoàn toàn tùy biến việc hiển thị thông qua các control HTML hay bên phía máy chủ.
  - + Hỗ trợ CSS đầy đủ.

#### ✓ ASPxTreeList

File name	Creation Date
Appearance	2/21/2008 3:41 PM
CSS	2/21/2008 3:41 PM
Data	2/21/2008 3:41 PM
Images	2/21/2008 3:41 PM
Demo	2/21/2008 3:41 PM
IconImages	2/21/2008 3:41 PM
TitleImages	2/21/2008 3:41 PM
CollapsedButton.gif	2/21/2008 3:41 PM

Hình 2.22: Minh họa giao diện ASPxTreeList

- ASPxTreeList là một Control rất mạnh trong việc hỗ trợ hiển thị dữ liệu dạng cây phân hệ, cho phép ta tạo ra các trường hiển thị bằng tay hoặc thông qua Database.

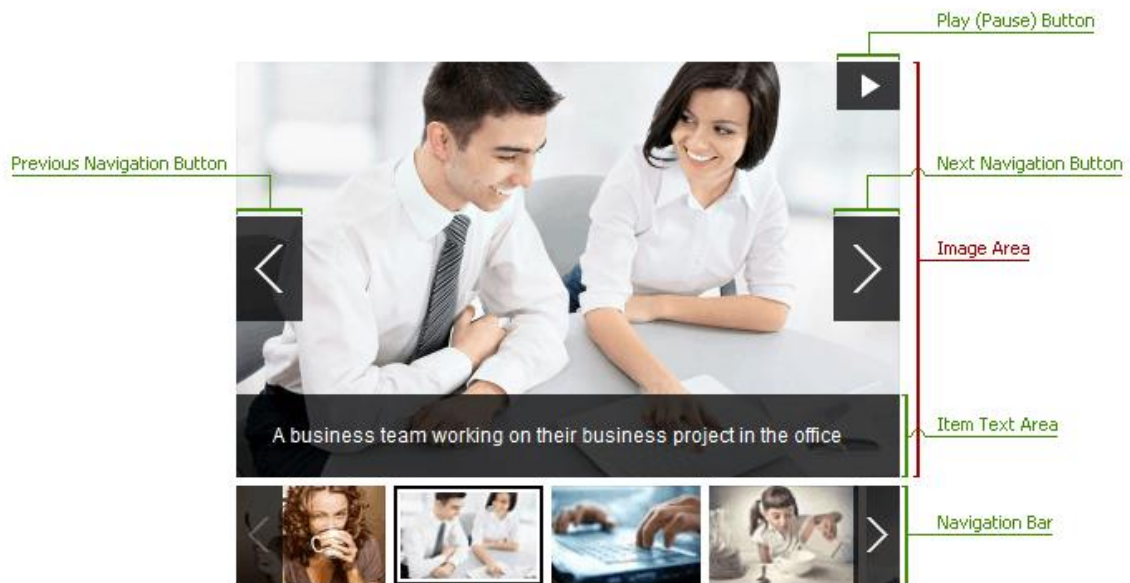
- Đặc điểm:

- + Tương thích với nhiều trình duyệt.
- + Hỗ trợ AJAX: ta có thể tải danh sách child node của Control thông qua phương thức callback, không cần thiết phải tải lại toàn bộ trang.
- + Tương tác với người dùng rất đa dạng.
- + Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, SQL Server.
- + Chứng thực dòng dữ liệu và chỉ ra lỗi: ASPxTreeList cho phép ta xác thực bằng tay các dòng đã chỉnh sửa, và hiển thị thông báo lỗi đối với trường không hợp lệ.

- Tùy biến giao diện hiển thị:

- + Giao diện: Ta có thể tùy chỉnh giao diện của cây bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.
- + Hỗ trợ các Template: với mỗi thành phần bên trong ASPxTreeList, ta có thể hoàn toàn tùy biến việc hiển thị thông qua các control HTML hay bên phía máy chủ.
- + Hỗ trợ CSS đầy đủ.

✓ ASPxImageSlider



Hình 2.23: Minh họa giao diện ASPxImageSlider

- ASPxImageSlider là một Control rất mạnh trong việc hỗ trợ hiển thị hình ảnh, cho phép ta tạo ra các trường hiển thị bằng tay hoặc thông qua Database.

- Đặc điểm:

+ Tương thích với nhiều trình duyệt.

+ Hỗ trợ AJAX: ASPxImageSlider cho phép chứa nhiều hình ảnh cùng lúc, tải dữ liệu dạng động thông qua phương thức callback, không cần thiết phải tải lại toàn bộ trang.

+ Tương tác với người dùng rất đa dạng.

+ Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, SQL Server.

- Tùy biến giao diện hiển thị:

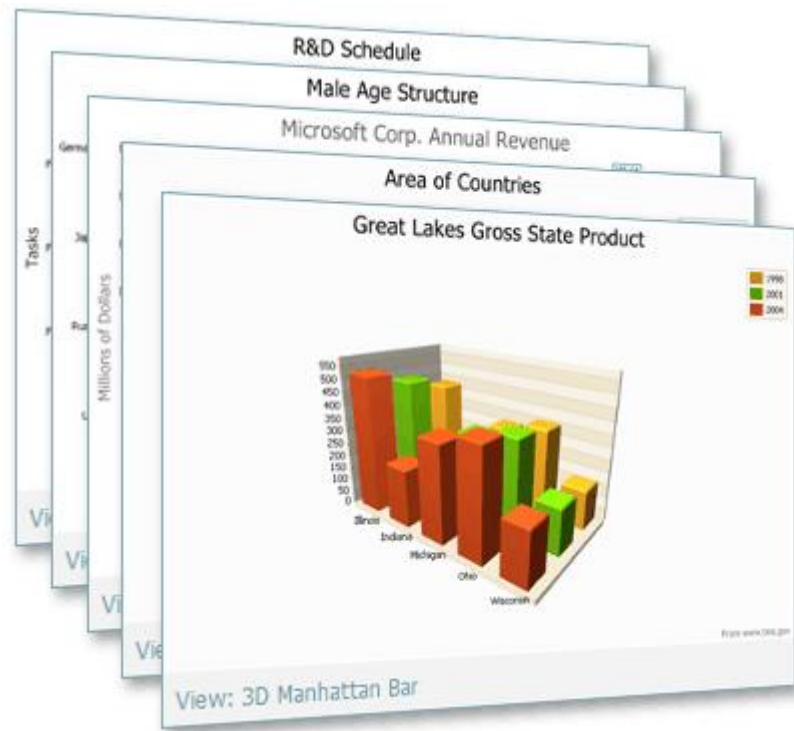
+ Giao diện: Ta có thể tùy chỉnh giao diện của cây bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.



+ Hỗ trợ các Template: với mỗi thành phần bên trong ASPxImageSlider, ta có thể hoàn toàn tùy biến việc hiển thị thông qua các control HTML hay bên phía máy chủ.

+ Hỗ trợ CSS đầy đủ.

✓ ASPxPopupControl



Hình 2.24: Minh họa giao diện ASPxPopupControl

- ASPxPopupControl là một Control rất mạnh trong việc hỗ trợ hiển thị Popup, cho phép ta tạo ra các popup động, tùy chỉnh popup toàn màn hình hay tải dữ liệu động nhờ kết hợp với ASPxLoadingPanel.

- Đặc điểm:

+ Tương thích với nhiều trình duyệt.

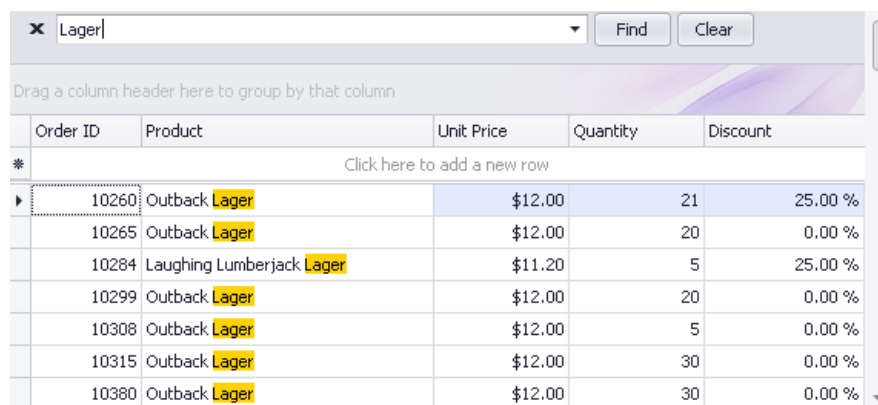
+ Hỗ trợ AJAX: ASPxPopupControl cho phép tải dữ liệu từ trang web cụ thể hoặc trang web local.

+ Tương tác với người dùng rất đa dạng.

- + Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, SQL Server.
- Tùy biến giao diện hiển thị:
  - + Giao diện: Ta có thể tùy chỉnh giao diện của cây bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.
  - + Hỗ trợ các Template: với mỗi thành phần bên trong ASPxPopupControl, ta có thể hoàn toàn tùy biến việc hiển thị thông qua các control HTML hay bên phía máy chủ.
  - + Hỗ trợ CSS đầy đủ.

### 2.3.3.3. Devexpress dành cho .NET WinForms

- Định nghĩa: Devexpress WinForms là một control của Devexpress cho phép các control phục vụ cho WinForms .NET. Sử dụng control của Devexpress WinForms làm cho phần mềm thêm sinh động và chuyên nghiệp hơn.
- Các control mà phần mềm sử dụng từ Devexpress:
  - ✓ GridControl



Order ID	Product	Unit Price	Quantity	Discount
10260	Outback Lager	\$12.00	21	25.00 %
10265	Outback Lager	\$12.00	20	0.00 %
10284	Laughing Lumberjack Lager	\$11.20	5	25.00 %
10299	Outback Lager	\$12.00	20	0.00 %
10308	Outback Lager	\$12.00	5	0.00 %
10315	Outback Lager	\$12.00	30	0.00 %
10380	Outback Lager	\$12.00	30	0.00 %

Hình 2.25: Minh họa giao diện GridControl

- GridControl là một Control rất mạnh trong việc hỗ trợ hiển thị dữ liệu dạng lưới, quản lý một lượng lớn dữ liệu, cho phép ta tạo ra các trường hiển thị bằng tay hoặc thông qua Database<sup>[21]</sup>.

- Đặc điểm:

- + Chạy Thread riêng biệt để xử lý dữ liệu và hiển thị lên màn hình.
  - + Giao diện người dùng đa dạng (Banded Grid View, Advanced Banded Grid View, Card View và Layout View).
  - + Tương tác với người dùng rất đa dạng.
  - + Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, SQL Server.
  - + Cho phép hiển thị dữ liệu dạng Master-Detail với cấu trúc đa dạng.
  - + Tự động gom nhóm dữ liệu: Cho phép người dùng gom nhóm dữ liệu, không giới hạn số cột.
  - + Tóm tắt dữ liệu đầy đủ: cho phép hiển thị thông tin thống kê như MIN, MAX, AVG, SUM và COUNT trực tiếp trên lưới.
  - + Cho phép lọc dữ liệu và hiển thị Text: với mỗi cột ta có thể chỉ định cách dữ liệu được sắp xếp theo giá trị hiển thị của nó. Ngoài ra ta có thể cho phép lọc dữ liệu bất kỳ bằng cách gõ vào giá trị muốn lọc trực tiếp vào ô textbox.
  - + Cho phép lựa chọn nhiều dòng cùng một lúc.
- Tùy biến giao diện hiển thị: ta có thể tùy chỉnh giao diện của lưới bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.

✓ TreeList

Name	Executor	Start date	End date	State
Project: Stanton	Declan Kears	10/1/2011	10/26/2011	
Information Gathering	Declan Kears	10/1/2011	10/10/2011	
Market research	Alexus Maione	10/1/2011	10/5/2011	Completed
Making specification	Daniel Earwood			
Planning	Brown Stjames			
Design	Spinks Hatori			
Design of a web pages	Armando Ordway			
Pages layout	Alexandru Misek			
Testing and Delivery	Manley Difrancesco			
Testing	Leslie Hallquist			
Content	Sonal Kannard			
Development	Lauren Partain	10/23/2011	10/26/2011	
Project: Betaron	Alexus Maione	10/1/2011	10/30/2011	

Hình 2.26: Minh họa giao diện TreeList

- TreeList là một Control rất mạnh trong việc hỗ trợ hiển thị dữ liệu dạng cây, lưới hoặc kết hợp cả hai, cho phép ta tạo ra các trường hiển thị bằng tay hoặc thông qua Database<sup>[22]</sup>.

- Đặc điểm:

- + Hiển thị dạng lồng dữ liệu, không giới hạn số nhánh con.
- + Hỗ trợ chức năng lọc, sắp xếp và tìm kiếm dữ liệu trong cây.
- + Tương tác với người dùng rất đa dạng.
- + Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, MSSQL Server.

- Tùy biến giao diện hiển thị: ta có thể tùy chỉnh giao diện của cây bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.

✓ TreeListLookupEdit

- TreeListLookupEdit là một Control hỗ trợ chọn dữ liệu từ TreeList, như một Combobox nâng cao<sup>[23]</sup>.

Jack Crawford		
Department	Position	Full Name
▼ Developers		
▼ .NET Team		
	Senior Developer	Jim Gordon
	Developer	Alex Marvel
	Developer	Jack Crawford
▼ Web Team		
	Senior Developer	Mark Asher
	Developer	Sandy Broomer

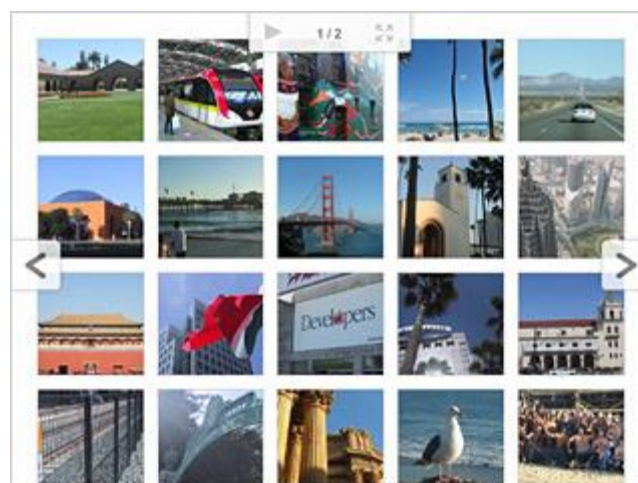
Hình 2.27: Minh họa giao diện *TreeListLookUpEdit*

- Đặc điểm:

- + Hiển thị dạng lồng dữ liệu, không giới hạn số nhánh con.
- + Hỗ trợ chức năng lọc, tìm kiếm dữ liệu trong cây.
- + Tương tác với người dùng rất đa dạng.
- + Hỗ trợ khai thác dữ liệu từ nhiều hệ quản trị cơ sở dữ liệu khác nhau: Microsoft Access, MSSQL Server

- Tùy biến giao diện hiển thị: ta có thể tùy chỉnh giao diện của cây bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.

✓ GalleryControl



Hình 2.28: Minh họa giao diện *GalleryControl*

- GalleryControl là một Control hỗ trợ hiển thị hình ảnh từ dữ liệu hình ảnh dưới dạng binary. Phân loại hình ảnh theo album, loại hình ảnh<sup>[24]</sup>.

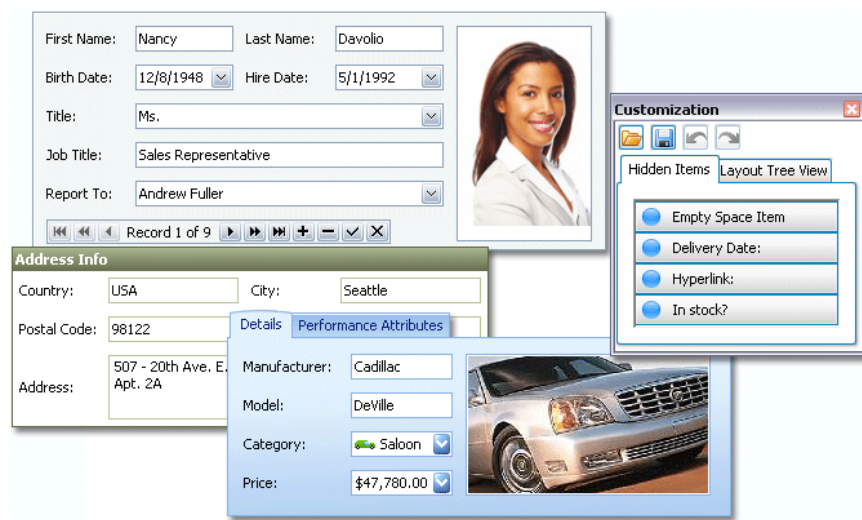
- Đặc điểm:

+ Hiển thị nhiều hình ảnh trong ImageSider, hỗ trợ nút điều hướng hình ảnh.

+ Tương tác với người dùng rất đa dạng.

- Tùy biến giao diện hiển thị: ta có thể tùy chỉnh giao diện của cây bằng cách chọn các định dạng hiển thị khác nhau, chỉ sau vài cái click chuột.

✓ Look & Feel



Hình 2.29: Minh họa giao diện Look & Feel

- Định nghĩa: Là một chức năng của Devexpress cho phép thay đổi giao diện phần mềm với kho giao diện có sẵn hoặc tự thiết kế<sup>[25]</sup>.

- Tính năng: Cho phép thay đổi giao diện một cách trực quan mà không cần phải restart lại phần mềm.

#### 2.3.4.4. Devexpress XtraReport

Contact Name:	Yoshi Nagase	Country:	Japan
Contact Title:	Marketing Manager	Region:	
Phone:	(03) 3555-5011	City:	Tokyo
Fax:		Postal Code:	100
Home Page:			
Address:	9-8 SekimaiMusashino-shi		

Product Name	Product ID	Category	Quantity per Unit	Unit Price	Discontinued
Mishi Kobe Niku	9	Meat/Poultry	18 - 500 g pkgs.	97	<input checked="" type="checkbox"/>

OrderID	Quantity	Discount	Sub Total
Unit price: \$77.6			
10420	20	0.10	\$1,552.0
Total by unit price:			\$1,552.0
Unit price: \$97.0			
10515	16	0.15	\$1,552.0
10507	50	0.25	\$4,850.0

Hình 2.30: Minh họa giao diện XtraReport

- Devexpress XtraReport là một control của Devexpress cho phép xuất báo cáo, in ấn dữ liệu ra nhiều loại tập tin khác nhau, cho phép xuất báo cáo dạng động hay custom. Khẩu lệnh của DevExpress là : “Những gì bạn nhìn thấy là những gì sẽ in”. Sử dụng Devexpress XtraReport làm cho việc xuất báo cáo dễ hơn bao giờ hết<sup>[1]</sup>.

- Đặc điểm: Devexpress XtraReport chứa công cụ để tạo Report từ dữ liệu phức hợp, chế độ xem trước, in và xuất report ra nhiều định dạng khác nhau.

- Tính năng:

+ XtraReport hoạt động được trong cả ứng dụng WinForm và WebForm. Ta có thể chỉ cần tạo một Report và sử dụng ở hai môi trường khác nhau.

+ Tích hợp đầy đủ Visual Studio .Net: Report Designer tích hợp. Hỗ trợ chế độ xem trước dạng Report Viewer (WinForm) hay HTML (WebForm) và dạng in, khi có thay đổi trong thiết kế report, ta không cần biên dịch lại toàn bộ ứng dụng mà vẫn có thể cập nhật được chế độ xem trước kịp thời.

+ XtraReports làm việc với toàn bộ đối tượng dữ liệu được hỗ trợ bởi Visual Studio .NET như : chuẩn .NET Data Objects, Ilist Interface, XML Data Objects.

+ Cho phép lọc dữ liệu dưới với nhiều cấp: Data adapter, Data set, Data views.

- + Cho phép gom nhóm dữ liệu: gom nhóm đa tầng và lồng nhau.
- + Hỗ trợ nhiều control chuẩn như: Label, Line, BarCode, CheckBox, PageInfo, Panel, PictureBox, PageBreak, Table, ZipCode,...
- + Hỗ trợ biểu đồ thông qua control XtraCharts.
- + SubReports: Ta có thể dùng lại các lớp của XtraReport vào ứng dụng qua control Subreport. Chỉ cần thả vào control Subreport, set thuộc tính nguồn Report, ta có hai report từ một nguồn.
- + Hỗ trợ tóm tắt: dễ dàng tạo tóm tắt cho một textbox hay một ô trong bảng. Chỉ cần set hai thuộc tính Summary position (group hay report) và Summary type (Avg, min, max, sum, count...).
- + Hỗ trợ phong phú các định dạng xuất ra: PDF, HTML, MHT, RTF, TXT, CSV và MS Excel. Có thể xuất report ra định dạng hình ảnh như: BMP, EMF, GIF, JPEG, PNG, TIFF, WMF.
- + Importing: có thể nạp lại report cũ của mình từ MS Access, Crystal Reports, Data Dynamics Active Reports vào XtraReport.
- + Tìm kiếm ở chế độ xem trước: giúp cho người dùng có thể tìm những đoạn text mong muốn.
- + Hỗ trợ thừa kế, Bookmark, Watermarks.

#### **2.3.4.5. Tính năng Static Website Loading trong WebForm thông qua AJAX**

- Về AJAX: Ajax được giới thiệu lần đầu tiên vào ngày 18/02/2005 trong một bài báo có tên AJAX : A New Approach to Web Applications của tác giả Jesse James Garrett, công ty AdapativePath. Ngay sau đó thuật ngữ AJAX được phổ biến cực kỳ nhanh chóng trong cộng đồng phát triển Web và cho đến nay nó là một trong những từ khóa được tìm kiếm nhiều nhất trên Internet<sup>[6]</sup>.
- UpdatePanel của Visual Studio hỗ trợ người lập trình web thao tác với ajax một cách dễ dàng thông qua kéo thả, chỉ cần kéo UpdatePanel vào trang aspx, mọi xử lý



viết trong thẻ UpdatePanel này sẽ được Visual Studio tự động chuyển sang xử lý dạng Ajax.

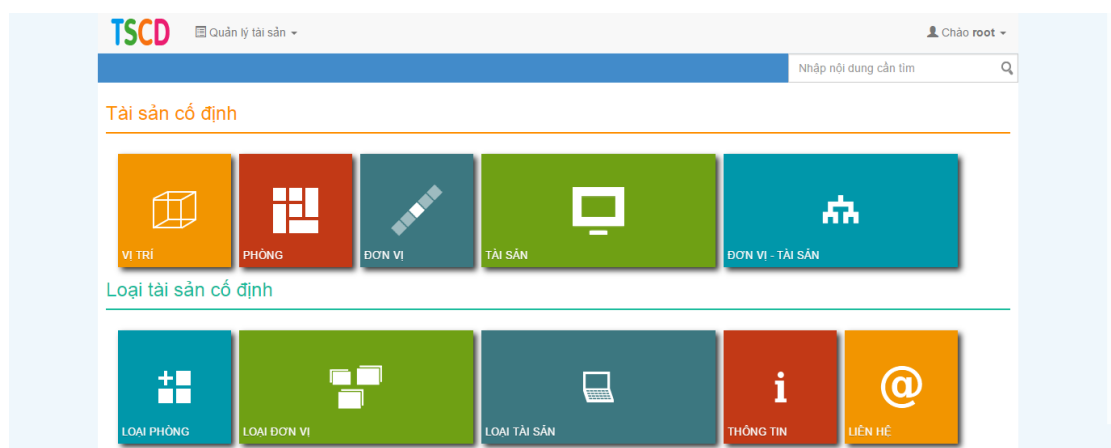
### 2.3.4. Công nghệ giao diện tùy biến (responsive design) dành cho ứng dụng Web Mobile

#### 2.3.4.1. Công nghệ Responsive Web Design (RWD)

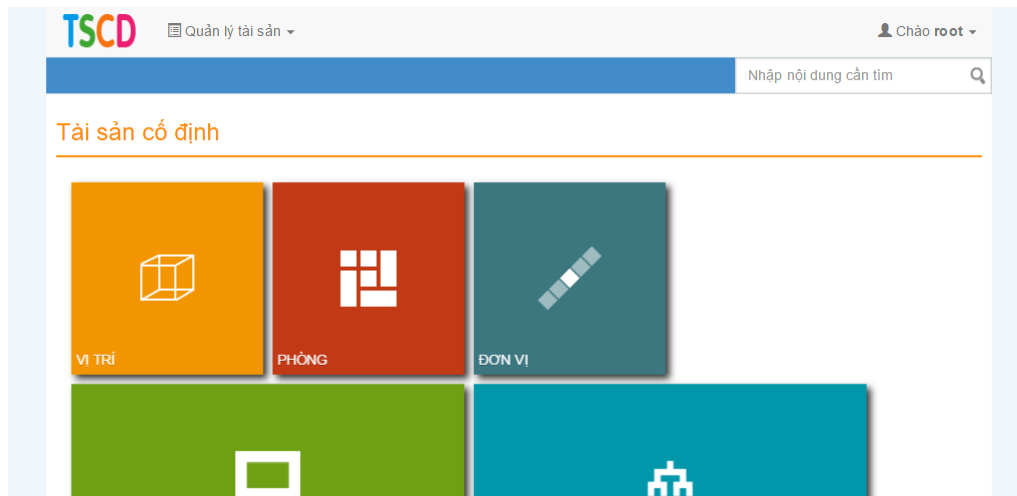
- Định nghĩa: RWD áp dụng nhiều bố cục trang web cho các loại kích cỡ màn hình khác nhau chứ không chỉ thiết kế một giao diện cố định như vẫn thường làm trước đó. Cộng với sự phát triển của các chuẩn HTML5 và CSS3, RWD đã trở thành một thứ quan trọng mà quản trị viên hay chủ sở hữu website cần phải nghĩ tới trong bối cảnh ngày càng nhiều thiết bị di động với đủ các kích cỡ, đủ loại độ phân giải màn hình khác nhau được tung ra thị trường.

- Lợi ích:

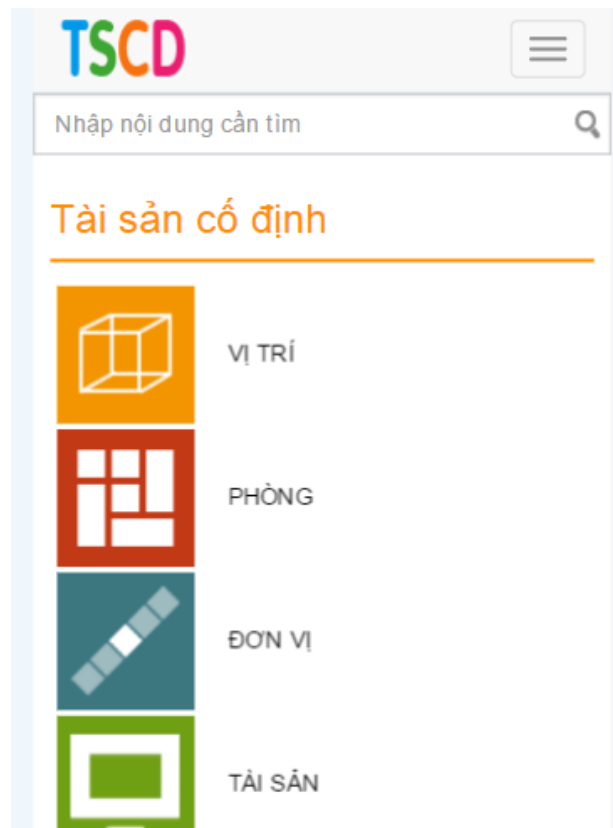
+ RWD dùng để bố cục lại giao diện trang web cho tương thích với nhiều loại kích cỡ màn hình khác nhau.



Hình 2.31: Minh họa giao diện RWD trên Laptop (1366 x 768)



Hình 2.32: Minh họa giao diện RWD trên iPad 3 (1024 x 768)



Hình 2.33: Minh họa giao diện RWD trên iPhone 6 (375 x 667)

+ Trang web có thể hiển thị một cách đầy đủ trên màn hình di động. Tuy nhiên, nếu không áp dụng RWD, trang web khi xem trên thiết bị di động sẽ trở nên nhỏ xíu, bắt buộc phải phóng to và kéo qua lại, lên xuống liên tục để đọc nội

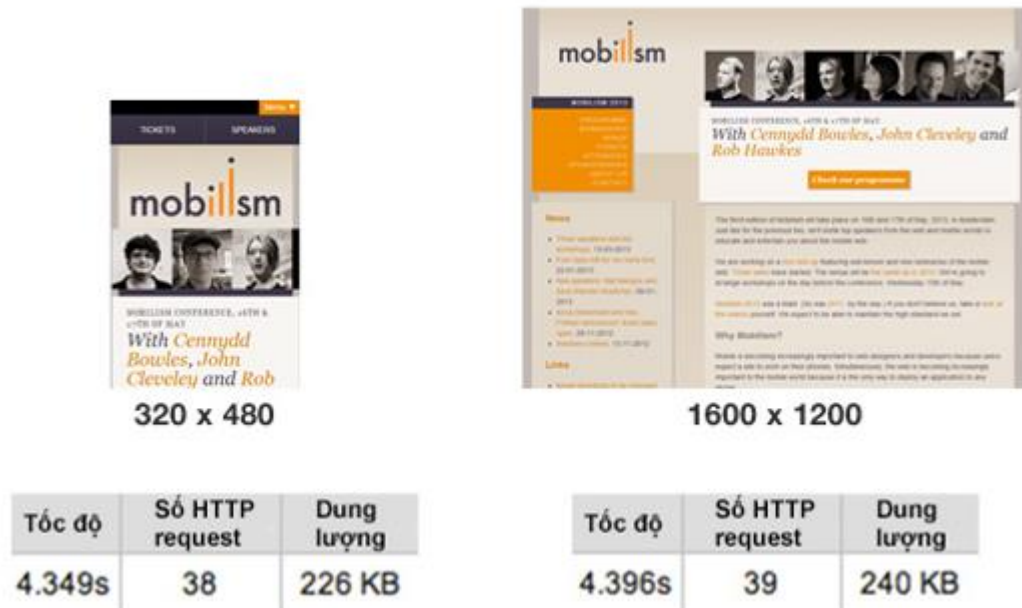
dung. Đây là trải nghiệm tiêu cực và nó khiến bạn nghĩ xấu về website, và điều tất nhiên là bạn chẳng thèm quay lại web đó nữa.

+ Nói tóm lại, RWD là một xu hướng thiết kế hoàn toàn có lợi bởi nó đảm bảo bạn sẽ luôn luôn có những trải nghiệm tốt nhất, đẹp nhất khi xem trang web dù bạn có đang dùng thiết bị nào đi nữa. Nó giúp nhà lập trình web tận dụng tối đa không gian để trình diễn những nội dung cho chúng ta xem theo cách thoải mái và thích thú nhất có thể.

- Hiệu năng:

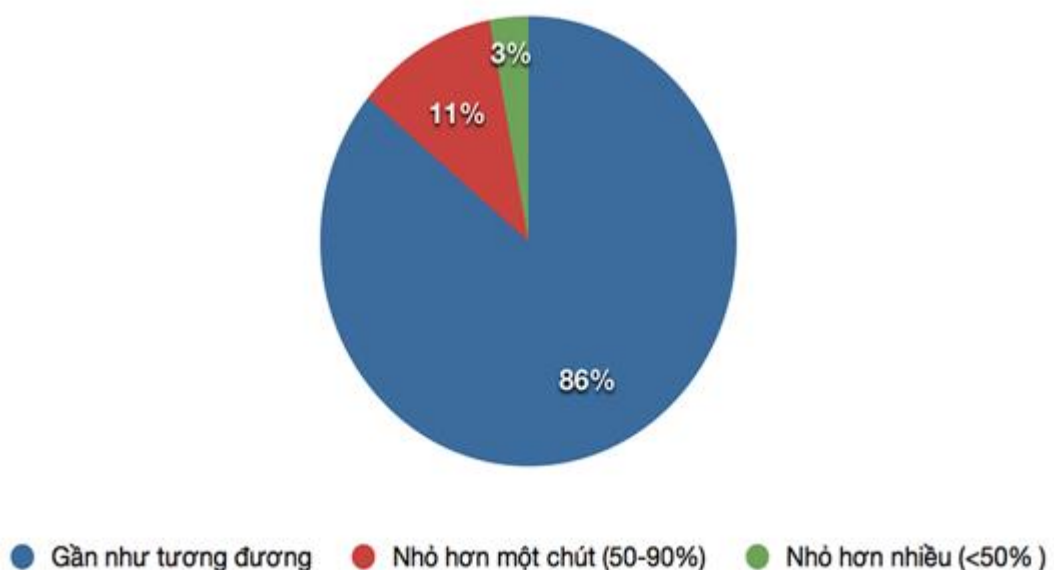
+ Năm 2009, Guy Podjarny, trưởng nhóm kiến trúc sản phẩm của công ty Akamai (một đơn vị chuyên nghiên cứu và tối ưu hóa tốc độ cho các giải pháp điện toán) đã thử nghiệm 347 trang web RWD được trình diễn trên <http://mediaqueri.es> bằng cách dùng Chrome trên nhiều thiết bị khác nhau, cộng với công cụ WebPageTest. Podjarny nhận thấy rằng kích thước tải về cũng như tốc độ tải của các trang web này không khác mấy khi sử dụng trên nhiều thiết bị khác nhau và màn hình với độ phân giải khác nhau. Sau đó Podjarny đã trình bày phát hiện của mình tại một hội nghị chuyên về thiết kế.

+ Vậy thử nghiệm trên có ý nghĩa gì? Nó cho thấy rằng mặc dù trang web đã phản hồi lại với kích thước màn hình bằng cách thu gọn nội dung, ẩn bớt những thành phần không cần thiết, thu nhỏ cỡ ảnh... nhưng điều đó không đồng nghĩa với chuyện web sẽ tải nhanh hơn. Nó cũng không đảm bảo rằng dung lượng tải về sẽ nhỏ hơn, ít chiếm băng thông hơn. Việc tối ưu hóa này hoàn toàn nằm trong tay lập trình viên và nhà thiết kế web.



Hình 2.34: So sánh tốc độ tải trang Web giữa giao diện Mobile và Desktop

- Còn bên dưới là biểu đồ so sánh về kích thước trang của các web RWD trên hai loại độ phân giải do Akamai thực hiện. Đến 86% trang web khi tải giữa hai loại màn hình không cho thấy sự khác biệt rõ rệt về dung lượng trang, tức là chúng ta chỉ tiết kiệm được một khoảng dung lượng không đáng kể khi duyệt web bằng mobile.



Hình 2.35: So sánh giữa kích thước trang Web và độ phân giải màn hình

- Độ phức tạp:

+ Thật ra trước đây các lập trình viên cũng có nghĩ đến một biện pháp khác khi mà RWD chưa phổ biến, đó là xây dựng một phiên bản di động dành cho web (ví dụ: m.tinhte.vn). Cách này cũng tốt, tác dụng gần như tương đương với RWD. Tuy nhiên, nó là một trang tập tin HTML riêng, một file CSS riêng được viết riêng cho thiết bị di động, hình ảnh cũng được thiết lập với kích thước nhỏ hơn.

+ Ngoài ra, một số dịch vụ online cũng có hỗ trợ chuyển RSS thành một trang web riêng biệt. Lập trình viên có thể tận dụng điều này để thiết kế web cho thiết bị di động mà không phải đầu tư quá nhiều công sức. Họ chỉ cần làm cho RSS của mình đầy đủ nhất có thể là xong. Người dùng truy cập từ các smartphone, tablet sẽ không thấy giao diện chính mà chỉ thấy các dòng cập nhật mới nhất, tin tức mới nhất. Như vậy cũng đã đủ đối với một số trang web rồi.

+ RWD thì ngược lại, nó vốn dĩ là phức tạp hơn bởi nhà thiết kế web đang cố gắng nhiều trải nghiệm xem khác nhau chứ không nhắm đến một loại thiết bị cụ thể nào cả. Điều đó có nghĩa là trình duyệt trên máy mobile phải đảm đương một file HTML lớn, một tập tin CSS cũng lớn không kém. Nếu không được tích hợp tốt, RWD có thể làm cho việc duyệt web di động trở nên chậm chạp hơn mặc dù bố cục rất tốt<sup>[2]</sup>.

#### **2.3.4.2. CSS Framework**

- Định nghĩa: là bộ công cụ giúp thiết kế trang web bằng CSS nhanh hơn. Nghĩa là nó được trừu tượng hóa lên một mức cao hơn. Thay vì phải hiểu rõ về các bộ chọn, các thuộc tính và giá trị trong CSS để style cho trang web của mình, thì chỉ cần biết các thành phần có trên trang web như form, navbar, tooltip, dropdown - menu, modal, button,... và thêm nó vào trang html một cách thích hợp. Công việc còn lại là của CSS Framework<sup>[4]</sup>.

- Phiên bản: Hiện tại có rất nhiều CSS Framework được phát triển. Trong đó, có hai loại được phát triển mạnh nhất đến thời điểm này là:

- + Bootstrap (version 2 và version 3)<sup>[7]</sup>.

- + Foundation (version 3, version 4 và version 5)<sup>[8]</sup>.

- Tính năng:

- + Hỗ trợ khả năng Responsive: tức là trang web sẽ tự động co giãn theo kích thước của cửa sổ trình duyệt.

- + Tương thích tốt với thiết bị cỡ nhỏ: với sự phổ biến của smartphone hiện nay, đây là một yếu tố quan trọng. Không cần phải thiết kế một bản riêng cho mobile, với bootstrap bạn chỉ cần thiết kế một lần cho mọi thiết bị.

- + Được tích hợp với thư viện jQuery và tương tác tốt với chuẩn HTML5 và CSS3.

- \* Twitter Bootstrap (gọi tắt là Bootstrap).

- Đặc điểm:

- + Bootstrap là một CSS Framework phổ biến nhất hiện nay do Twitter phát triển.

- + Bootstrap bao gồm các mã CSS và HTML cơ bản cho typography, form, button, table, grid, navigation, và nhiều thành phần khác của website.

- + Bootstrap cung cấp lưới cố định (fixed) rộng 940px và 12 cột. Tất nhiên là cũng có giải pháp cho việc dùng layout dạng động (fluid).

- + Style của các phần tử HTML trong Bootstrap khá đơn giản và thanh lịch. Ví dụ như phần đổ bóng trong input, highlight của bảng biểu, các mã CSS hiển thị cảnh báo, tab, phân trang,...

- Tính năng: Bootstrap giúp chúng ta giảm thiểu thời gian thiết kế HTML và CSS. Bootstrap định nghĩa sẵn các lớp CSS. Công việc của chúng ta chỉ là sử dụng các lớp đó vào mục đích của mình. Bootstrap còn hỗ trợ Responsive Design (giao diện đa thiết bị) rất được ưa chuộng trong thời gian gần đây<sup>[3]</sup>.

- Phiên bản mới nhất: version 2 và version 3.

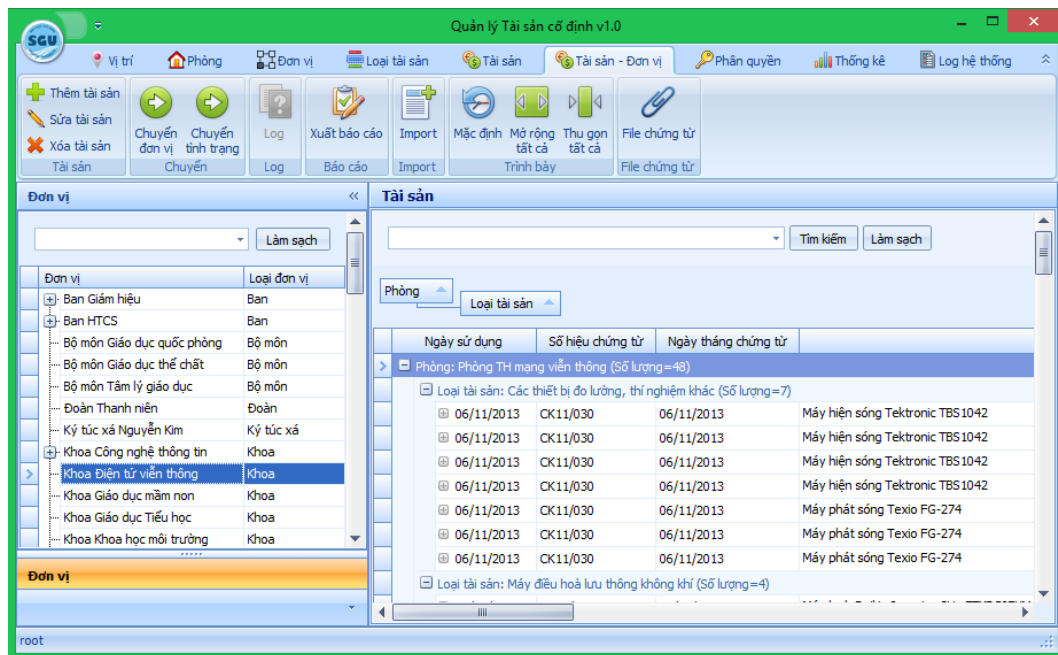
- Phiên bản sử dụng trong ứng dụng: version 3.

## 2.4. Kết quả thực thi (các màn hình chức năng)

\* Do hạn chế về không gian trình bày cũng như không thể đưa hết mọi màn hình chức năng vào trong báo cáo này nên nhóm chọn ra 5 chức năng chính để hiển thị.

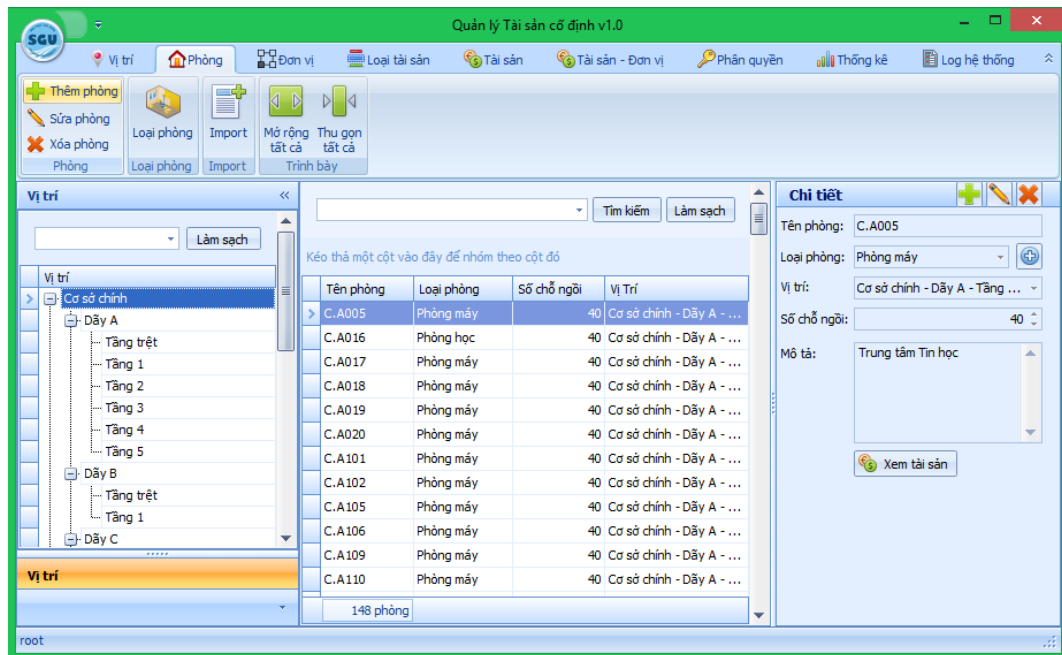
- Ứng dụng Windows Desktop:

+ Chức năng quản lý tài sản theo đơn vị



Hình 2.36: Minh họa màn hình chức năng quản lý tài sản theo đơn vị.

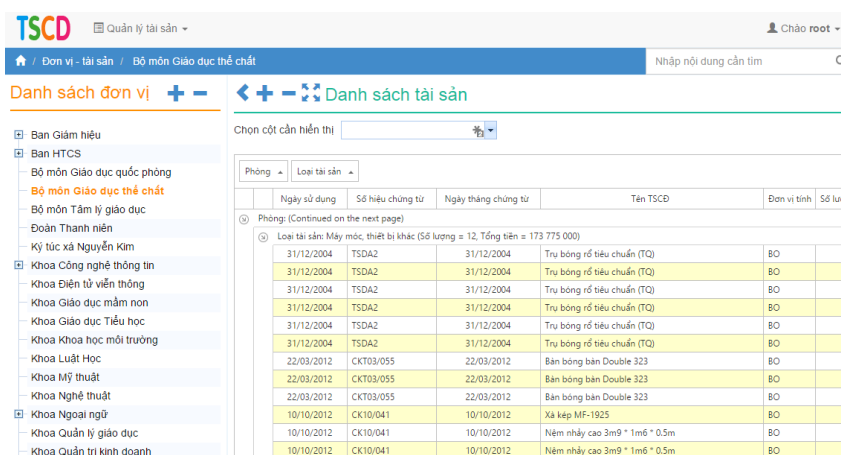
+ Chức năng quản lý phòng



Hình 2.37: Minh họa màn hình chức năng quản lý phòng.

- Ứng dụng Windows Web:


+ Danh sách tài sản theo đơn vị giao diện web



Hình 2.38: Minh họa màn hình danh sách tài sản theo đơn vị giao diện web.



+ Danh sách tài sản theo đơn vị giao diện mobile



Đơn vị - tài sản

Bộ môn Giáo dục thể chất

Danh sách tài sản

#	Tên TSCD	Loại tài sản
1	Bảng trụ bóng rổ	Thiết bị, phương tiện quản lý khác
2	Máy vi tính Pentium 4 HP 5500	Máy vi tính
3	Trụ bóng rổ tiêu chuẩn (TQ)	Máy móc, thiết bị khác

1

2

Thông tin tài sản

Ngày sử dụng:

31/12/2002

Số hiệu:

Ngày tháng:

31/12/2002

Tên tài sản:

Bảng trụ bóng rổ

Đơn vị tính:

CAI

Số lượng:

1

Đơn giá:

7.200.000 VNĐ

Thành tiền:

7.200.000 VNĐ

Nước sản xuất:

Nguồn gốc:

Hệ thống tự động tạo: chuyển từ

Hình 2.39: Minh họa màn hình danh sách tài sản theo đơn vị giao diện mobile.

## CHƯƠNG 3: KIỂM THỬ VÀ TRIỂN KHAI

### 3.1. Kiểm thử tự động mức mã nguồn (Unit test)

#### 3.1.1. Kiểm thử hộp đen (Black box testing)

- Trong Unit test có hai phương pháp kiểm thử cơ bản là kiểm thử hộp trắng (kiểm thử luôn cả dữ liệu vào/ra và cả cách thực hiện cụ thể) và kiểm thử hộp đen (chỉ kiểm thử các bộ dữ liệu vào/ra mà không quan tâm đến cách thực hiện cụ thể).
- Do quy mô của ứng dụng không quá phức tạp nên trong phạm vi đề tài này chỉ chọn giải pháp kiểm thử hộp đen.

#### 3.1.2. Mô hình kiểm thử AAA (Arrange-Act-Assert)

- Định nghĩa: đây là cách tổ chức kiểm thử Unit test phổ biến nhất. Trong đó:
  - + Sắp xếp (Arrange): là lựa chọn các thành phần tham gia kiểm thử. có thể là các biến, các lớp hoặc thậm chí là các dự án ngoài.
  - + Act (Action - hiện thực): là cài đặt các bước, thao tác trong lịch trình kiểm thử nhằm tạo ra các kết quả đầu ra (output) tương ứng với các kết quả đầu vào (input).
  - + Assert (đánh giá): là một so sánh giữa kết quả đầu ra thực tế và kết quả đầu ra mong muốn. Từ đó đưa ra đánh giá cuối cùng.

### 3.2. Kiểm thử chấp nhận (Acceptance test)

- Kiểm thử chấp nhận được thực hiện bởi người dùng hoặc lập trình viên đứng ở góc độ là người dùng.
- Giai đoạn kiểm thử này có thể được thực hiện thủ công hoặc tự động tùy thuộc vào người đứng đầu dự án kiểm thử. Do đặc tính của kiểm thử tự động đòi hỏi phải bỏ ra nhiều thời gian cho phần thiết kế hệ thống hơn là thời gian thực thi kiểm thử, nên nhóm đã quyết định chọn giải pháp kiểm thử thủ công cho giai đoạn này.
- Do lịch trình kiểm thử rất nhiều nên sau đây chỉ liệt kê một vài chức năng đại diện.

Bảng 3.1: Kết quả kiểm thử chấp nhận

Step	Type	Description	Data	Expected result	Actual result	Pass or Fail	Note
<b>Thêm 1 tài sản</b>							
1	Step	Mở chương trình quản lý tài sản cố định					
2	Step	Click vào ribbon tài sản		Ribbon tài sản hiện lên và tài dữ liệu cần thiết			Dữ liệu gồm loại tài sản, vị trí, đơn vị, danh sách tài sản, ...
3	Step	Click vào button "Thêm tài sản"		Form thêm tài sản hiện lên			
4	Step	Điền dữ liệu cho form thêm tài sản	Field "Ngày sử dụng": "01/01/2014" Field "Số hiệu": "ABC53425" Field "Ngày tháng" (chứng từ): "01/01/2014" Field "Mã tài sản": "TS0001" Field "Tên tài sản": "Laptop HP ABCD" Field "Loại tài sản": "Laptop" Field "Số lượng": "5" Field "Đơn giá":				Thêm mới tài sản không có phụ tùng kèm theo

			"13000000" Field "Tình trạng": "Đang sử dụng" Field "Nước sản xuất": "Mỹ" Field "Nguồn gốc": "Mỹ" Field "Ghi chú": "Laptop cho giảng viên"				
5	Step	Click vào button "Ok"		Hiện thông báo thêm tài sản thành công Tài sản mới được thêm vào và hiển thị trên gridcontrol Form thêm tài sản đóng lại Gridcontrol focus vào tài sản vừa được thêm vào			
6	Verify Point	Kiểm tra xem tài sản mới có được thêm vào hay không					
<b>Kết quả</b>						<b>Pass</b>	
<b>Chuyển tài sản từ 1 khoa vào 1 phòng</b>							
1	Step	Mở chương trình quản lý tài sản cố định					
2	Step	Click vào ribbon tài sản		Ribbon tài sản hiện lên và tải dữ liệu cần thiết			Dữ liệu gồm loại tài sản, vị trí, đơn vị, danh sách tài sản,...

3	Step	Chọn 1 tài sản đang ở 1 khoa nào đó					Tài sản được chọn là “Laptop Acer Aspire 3682 NWXCI” đang thuộc khoa công nghệ thông tin, có số lượng là 5.
4	Step	Click vào button "Chuyển đơn vị" trên ribbon.		Form chuyển đơn vị hiện lên  Những thông tin hiện tại của tài sản được hiện lên form chuyển đơn vị.			
5	Step	Điền dữ liệu cho form chuyển đơn vị.	Các field "Ngày sử dụng", "Số hiệu" và "Ngày tháng" được giữ nguyên Field "Số lượng": "1" Field "Phòng": chọn phòng giáo viên khoa ngoại ngữ (tại tầng trệt, dãy A của cơ sở chính). Field “Vị trí” không thay đổi (do chương trình tự set cho) Field "Đơn vị quản lý": chọn khoa ngoại ngữ				Thông tin của tài sản hiển thị đúng trên form chuyển đơn vị.

			Field “Ghi chú”: "Cho khoa ngoại ngữ mượn".				
6	Step	Click vào button “Ok”.		Hiện thông báo chuyển đơn vị cho tài sản thành công.			
7	Verify Point	Sau khi chuyển tài sản vào đơn vị thành công thì chương trình sẽ hỏi có muốn xuất biên bản bàn giao tài sản không.					
8	Step	Click vào button "Cancel"		Form chuyển đơn vị sẽ đóng lại Gridcontrol tài sản sẽ tải lại dữ liệu.			Trường hợp ở đây là chuyển đơn vị thành công
9	Verify Point	Kiểm tra xem gridcontrol tài sản có điều chỉnh lại tài sản "Laptop Laptop Acer Aspire 3682 NWXCI" (ở đây là số lượng giảm xuống còn 4) Gridcontrol phải thêm 1 tài sản mới là "Laptop Acer Aspire 3682 NWXCI", số lượng là 1 và thuộc khoa ngoại ngữ					
<b>Kết quả</b>						<b>Pass</b>	
<b>Chuyển tài sản cho 1 người thuộc khoa nào đó</b>							
1	Step	Mở chương trình quản lý tài sản cố định					
2	Step	Click vào ribbon tài sản		Ribbon tài sản hiện lên và tải dữ liệu cần thiết			Dữ liệu gồm loại tài sản, vị trí, đơn

							vị, danh sách tài sản, ...
3	Step	Chọn 1 tài sản nào đó					Tài sản được chọn là "Laptop HP DV4 - 1202TU Core 2 Duo" có số lượng là 5 và không thuộc bất cứ đơn vị nào
4	Step	Click vào button "Chuyển đơn vị" trên ribbon		Form chuyển đơn vị hiện lên Những thông tin hiện tại của tài sản được hiện lên form chuyển đơn vị			
5	Step	Điền dữ liệu cho form chuyển đơn vị	Các field "Ngày sử dụng", "Số hiệu" và "Ngày tháng" được giữ nguyên Field "Số lượng": "1" Field "Phòng": chọn phòng văn phòng khoa công nghệ thông tin (tại tầng 1, dãy A của cơ sở chính) Field "Vị trí" không thay đổi (do chương trình tự set				Thông tin của tài sản hiển thị đúng trên form chuyển đơn vị

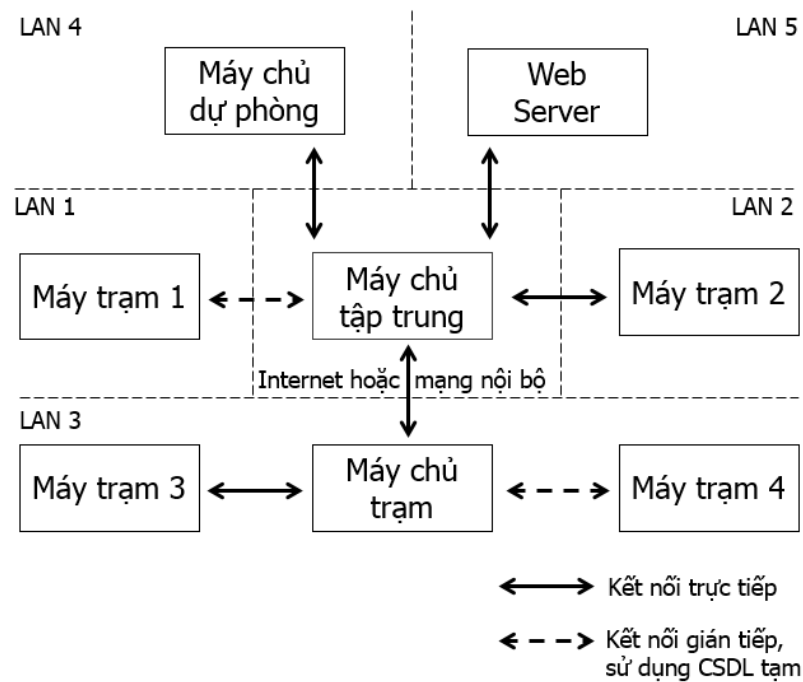




1	Step	Mở webbrowser lên, nhập url là "http://ts.dsesgu.edu.vn/"		Webbrowser hiện trang web của quản lý tài sản cố định			Webbrowser sử dụng ở đây là google chrome
2	Verify Point	Trang login được hiển thị					Vì muốn xem tài sản bắt buộc phải đăng nhập và có quyền xem tài sản
3	Step	Đăng nhập vào trang web	Field "Tài khoản": "root" Field "Mật khẩu": "root"	Đăng nhập vào hệ thống thành công			Tài khoản này là tài khoản đặc biệt, do hệ thống tự tạo và có tất cả quyền
4	Verify Point	Webbrowser được chuyển vào trang quản lý đơn vị		Hiện dữ liệu về các đơn vị			Trang này cho ta xem tài sản của 1 đơn vị, khi chọn 1 đơn vị nào đó
5	Step	Chọn 1 đơn vị (ở đây chọn khoa công nghệ thông tin)		Webbrowser sẽ chuyển sang trang hiện thông tin tài sản của khoa công nghệ thông tin			
6	Verify Point	Tài sản của khoa công nghệ thông tin phải được hiện ra, nếu chưa có hiện thông báo cho người dùng		Những tài sản của khoa công nghệ thông tin được hiện ra			Tài sản được hiển thị như trên winform
Kết quả							Pass

### 3.3. Các mô hình triển khai

- Hệ thống khi được đưa vào sử dụng thực tế sẽ có nhiều cách triển khai, tùy thuộc vào điều kiện hạ tầng mạng và yêu cầu chức năng của từng khu vực. Trong đó cách triển khai sau đây là tổng quát nhất, và được hỗ trợ hoàn toàn trong phần mềm.



Sơ đồ 3.1: Mô hình triển khai hệ thống phần mềm

- Cụ thể quá trình triển khai thực tế:

- + Máy chủ: từ nhà cung cấp dịch vụ Web Host VDC2.
- + Dung lượng lưu trữ trên Host: 1GB.
- + Băng thông hàng tháng: 60GB.
- + Website chạy trên nền IIS ASP.NET 4.0.
- + Các máy trạm người dùng đã cài đặt được cho một máy tính cá nhân ở Phòng Quản trị thiết bị.

- Các lưu ý sau khi triển khai hệ thống:

- + Mặc dù bản thân hệ thống đã có các biện pháp bảo mật và chính sách để vận hành tốt trong điều kiện bình thường, tuy nhiên khi đưa vào vận hành thực tế thì sẽ không thể tránh khỏi các vấn đề phát sinh liên quan như: máy chủ bảo trì hoặc gặp sự cố dữ liệu, quản trị cấp cao (root) quên mật khẩu, cài lại phần mềm trên máy cá nhân, bị tin tặc tấn công,... Khi đó hệ thống có thể sẽ không làm việc ổn định thậm chí ngừng hoạt động.
- + Nên cần phải có ít nhất một quản trị luôn theo dõi giám sát các bất thường xảy ra trên toàn hệ thống, để từ đó đưa ra giải pháp khắc phục nhanh nhất.
- + Cần phải có phương án máy chủ dự phòng khi máy chủ hiện tại không làm việc.
- + Sao lưu dữ liệu thường xuyên tùy thuộc vào tính chất của công việc, tần suất thay đổi CSDL.

## **KẾT LUẬN VÀ KIẾN NGHỊ**

### **Kết quả đạt được**

- Tìm hiểu và áp dụng thành công các kỹ thuật công nghệ mới vào phần mềm.
- Phần mềm tạo ra dựa trên các quy trình kỹ thuật đã được học từ nhà trường.
- Phần mềm đáp ứng đầy đủ các yêu cầu cơ bản đã đề ra. Đã và đang triển khai chạy thử trong giai đoạn đầu.
- Phần mềm chạy được đa nền tảng (Desktop, Mobile).

### **Hướng phát triển**

- Thay đổi và nâng cấp giao diện lẫn chức năng để phù hợp với nhu cầu sử dụng của nhà trường.
- Nâng cấp chức năng thống kê. Xuất báo cáo động, thống kê dạng đồ thị trực quan.
- Mở rộng phần mềm chạy đa nền tảng (Mac OS, Linux, IOS, Android, Windows Phone,...).

## DANH MỤC TÀI LIỆU THAM KHẢO

### Tài liệu tiếng việt

- [1] Nguyễn Mai Linh (2010), *Hướng dẫn sử dụng bộ công cụ DevExpress cho ASP.NET*, Phòng công nghệ thông tin - Đại học Sư phạm TP. Hồ Chí Minh, truy cập ngày 14 tháng 07 năm 2014, <http://123doc.vn/document/1314168-huong-dan-su-dung-bo-cong-cu-devexpress-cho-asp-net.htm>.
- [2] Responsive Web Design là gì và nó giúp ích như thế nào cho việc duyệt web trên thiết bị di động, truy cập ngày 28 tháng 05 năm 2014, <https://www.tinhte.vn/threads/responsive-web-design-la-gi-va-no-giup-ich-nhu-the-nao-cho-viec-duyet-web-tren-thiet-bi-di-dong.2101375>.
- [3] Làm quen với bootstrap grid system , truy cập ngày 08 tháng 06 năm 2014, <http://thachpham.com/web-development/html-css/hoc-bootstrap3-grid-system.html>.
- [4] Giới thiệu và hướng dẫn cách sử dụng Bootstrap, truy cập ngày 19 tháng 06 năm 2014, <http://thuctapcungdoanhnghiep.vn/hoc-lap-trinh-website/gioi-thieu-va-huong-dan-cach-su-dung-bootstrap>.
- [5] Tổng quan về DevExpress, truy cập ngày 08 tháng 06 năm 2014, <http://tympsolution.blogspot.com/2012/03/tong-quan-ve-devexpress.html>.
- [6] Sử dụng Ajax trong website Asp.Net, truy cập ngày 29 tháng 06 năm 2014, <http://thuyvk.com/art/su-dung-ajax-trong-website-aspnet-179>.

### Tài liệu tiếng anh

- [7] Bootstrap . The world's most popular mobile-first and responsive front-end framework, truy cập ngày 01 tháng 06 năm 2014, <http://getbootstrap.com>.
- [8] Foundation. The most advanced responsive front-end framework in the world, truy cập ngày 01 tháng 06 năm 2014, <http://foundation.zurb.com>.
- [9] Responsive CSS Framework Comparison, truy cập ngày 06 tháng 06 năm 2014, <http://responsive.vermilion.com/compare.php>.

- [10] The best web text editor for everyone, truy cập ngày 01 tháng 08 năm 2014, <http://ckeditor.com>.
- [11] Database Initialization Strategies in Code - First, truy cập ngày 13 tháng 07 năm 2014, <http://www.entityframeworktutorial.net/code-first/database-initialization-strategy-in-code-first.aspx>.
- [12] DevExpress Documentation 13.2.9, truy cập ngày 14 tháng 07 năm 2014, <https://documentation.devexpress.com/#HomePage/CustomDocument9453>.
- [13] Entity Framework (EF) Documentation, truy cập ngày 13 tháng 07 năm 2014, <http://msdn.microsoft.com/en-us/data/ee712907.aspx>.
- [14] Microsoft Sync Framework, truy cập ngày 13 tháng 07 năm 2014, [http://msdn.microsoft.com/en-us/library/bb902854\(v=sql.110\).aspx](http://msdn.microsoft.com/en-us/library/bb902854(v=sql.110).aspx).
- [15] Synchronizing SQL Server and SQL Express, truy cập ngày 12 tháng 07 năm 2014, [http://msdn.microsoft.com/en-us/library/ff928700\(v=sql.110\).aspx](http://msdn.microsoft.com/en-us/library/ff928700(v=sql.110).aspx).
- [16] Selecting the Appropriate Sync Framework Components, truy cập ngày 13 tháng 07 năm 2014, [http://msdn.microsoft.com/en-us/library/bb902854\(v=sql.110\).aspx](http://msdn.microsoft.com/en-us/library/bb902854(v=sql.110).aspx).
- [17] Scott Ambler (2002), *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*, Wiley Computer Publishing - John Wiley & Sons, Inc., New York.
- [18] UML 2.0 - Overview, truy cập ngày 10 tháng 09 năm 2014, [http://www.tutorialspoint.com/uml/uml\\_2\\_overview.htm](http://www.tutorialspoint.com/uml/uml_2_overview.htm).
- [19] Vivek Thakur (2008), *ASP.NET 3.5 Application Architecture and Design*, Packt Publishing Ltd., Birmingham - B27 6PA - UK.
- [20] Inheritance with EF Code First: Part 3 - Table per Concrete Type (TPC), truy cập ngày 13 tháng 07 năm 2014, <http://weblogs.asp.net/manavi/inheritance-mapping-strategies-with-entity-framework-code-first-ctp5-part-3-table-per-concrete-type-tpc-and-choosing-strategy-guidelines>.

[21] DevExpress GridControl Documentation, truy cập ngày 18 tháng 06 năm 2014, <http://documentation.devexpress.com/#WindowsForms/CustomDocument3455>.

[22] DevExpress Treelist Documentation, truy cập ngày 14 tháng 06 năm 2014, <http://documentation.devexpress.com/#WindowsForms/CustomDocument2434>.

[23] DevExpress XtraEditorsTreeListLookUpEdit Documentation, truy cập ngày 20 tháng 06 năm 2014, <http://documentation.devexpress.com/#windowsforms/clsDevExpressXtraEditorsTreeListLookUpEdit.topic>.

[24] DevExpress XtraEditorsControlsImageSlider Documentation, truy cập ngày 22 tháng 06 năm 2014, <http://documentation.devexpress.com/#WindowsForms/clsDevExpressXtraEditorsControlsImageSlidertopic>.

[25] DevExpress Look & Feel Documentation, truy cập ngày 22 tháng 06 năm 2014, <http://documentation.devexpress.com/#WindowsForms/CustomDocument1092>.

**PHỤ LỤC A**

- Do số lượng bảng nhiều, không thể trình bày hết, nên chỉ trình bày đại diện một vài bảng.

*Bảng PL1.1: Sơ đồ bảng VITRIS*

Tên cột	Kiểu dữ liệu	Cho phép rỗng
id	uniqueidentifier	<input type="checkbox"/>
coso_id	uniqueidentifier	<input type="checkbox"/>
day_id	uniqueidentifier	<input checked="" type="checkbox"/>
tang_id	uniqueidentifier	<input checked="" type="checkbox"/>
subId	nvarchar(MAX)	<input checked="" type="checkbox"/>
[order]	Bigint	<input checked="" type="checkbox"/>
mota	nvarchar(MAX)	<input checked="" type="checkbox"/>
date_create	Datetime	<input checked="" type="checkbox"/>
date_modified	Datetime	<input checked="" type="checkbox"/>

*Bảng PL1.2: Sơ đồ bảng PHONGS*

Tên cột	Kiểu dữ liệu	Cho phép rỗng
id	uniqueidentifier	<input type="checkbox"/>
ten	nvarchar(255)	<input type="checkbox"/>
loaiphong_id	uniqueidentifier	<input type="checkbox"/>
vitri_id	uniqueidentifier	<input type="checkbox"/>
subId	nvarchar(MAX)	<input checked="" type="checkbox"/>
[order]	Bigint	<input checked="" type="checkbox"/>
mota	nvarchar(MAX)	<input checked="" type="checkbox"/>
date_create	Datetime	<input checked="" type="checkbox"/>
date_modified	Datetime	<input checked="" type="checkbox"/>



*Bảng PL1.3: Sơ đồ bảng DONVIS*

Tên cột	Kiểu dữ liệu	Cho phép rỗng
id	uniqueidentifier	<input type="checkbox"/>
ten	nvarchar(255)	<input type="checkbox"/>
parent_id	uniqueidentifier	<input checked="" type="checkbox"/>
loaidonvi_id	uniqueidentifier	<input type="checkbox"/>
subId	nvarchar(MAX)	<input checked="" type="checkbox"/>
[order]	Bigint	<input checked="" type="checkbox"/>
mota	nvarchar(MAX)	<input checked="" type="checkbox"/>
date_create	Datetime	<input checked="" type="checkbox"/>
date_modified	Datetime	<input checked="" type="checkbox"/>

*Bảng PL1.4: Sơ đồ bảng TAISANS*

Tên cột	Kiểu dữ liệu	Cho phép rỗng
id	uniqueidentifier	<input type="checkbox"/>
ten	nvarchar(MAX)	<input type="checkbox"/>
dongia	Bigint	<input type="checkbox"/>
loaitaisan_id	uniqueidentifier	<input type="checkbox"/>
subId	nvarchar(MAX)	<input checked="" type="checkbox"/>
[order]	Bigint	<input checked="" type="checkbox"/>
mota	nvarchar(MAX)	<input checked="" type="checkbox"/>
date_create	Datetime	<input checked="" type="checkbox"/>
date_modified	Datetime	<input checked="" type="checkbox"/>

*Bảng PL1.5: Sơ đồ bảng CHUNGTUS*

Tên cột	Kiểu dữ liệu	Cho phép rỗng
id	uniqueidentifier	<input type="checkbox"/>
sohieu	nvarchar(MAX)	<input checked="" type="checkbox"/>
ngay	datetime	<input checked="" type="checkbox"/>
subId	nvarchar(MAX)	<input checked="" type="checkbox"/>
[order]	Bigint	<input checked="" type="checkbox"/>
mota	nvarchar(MAX)	<input checked="" type="checkbox"/>
date_create	datetime	<input checked="" type="checkbox"/>
date_modified	datetime	<input checked="" type="checkbox"/>

- Quan hệ n - n giữa thực thể với quyền:

DONVI\_PERMISSION

\* Trong đó, id1 là khóa ngoại đến bảng thực thể tương ứng, id2 là khóa ngoại đến bảng hình ảnh.

- Quan hệ n - n giữa nhóm quyền và quyền:

GROUP\_PERMISSION

## PHỤ LỤC B

### 1. \_CRUDInterface<T>

- Mức truy cập: public.
- Loại: giao diện.
- Mô tả: Các lớp cần các phương thức thông dụng trên một thực thể: Thêm, Xóa, Sửa,... sẽ thực thi lớp giao diện này.

*Bảng PL2.1: Thiết kế giao diện \_CRUDInterface<T>*

Tên phương thức	Kiểu trả về	Mô tả
add()	int	Thêm đối tượng vào CSDL
update()	int	Sửa đối tượng trong CSDL
delete()	int	Xóa đối tượng khỏi CSDL
reload()	T	Tải lại đối tượng
trigger()	void	Ép lazyloading tải dữ liệu ngay
doTrigger()	void	Gọi trigger trên tất cả các thuộc tính là đối tượng đơn và là khóa ngoại
clone()	T	Sao chép thông tin ra đối tượng mới
moveUp()	void	Di chuyển thứ tự đối tượng lên 1 bậc
moveDown()	void	Di chuyển thứ tự đối tượng xuống 1 bậc
niceName()	String	Lấy thông tin cơ bản về đối tượng
prevObj()	T	Trả về đối tượng có thứ tự trước 1 bậc
nextObj()	T	Trả về đối tượng có thứ tự sau 1 bậc

### 2. \_EFEEventRegisterInterface

- Mức truy cập: public.
- Loại: giao diện.

- Mô tả: Các lớp thực thể muốn đăng ký gọi ngược (callback) bởi các sự kiện khi EF làm việc sẽ thực thi lớp giao diện này.

*Bảng PL2.2: Thiết kế giao diện \_EFEventRegisterInterface*

Tên phương thức	Kiểu tham số	Kiểu trả về	Mô tả
onBeforeAdded		void	Được gọi trước khi thêm vào CSDL
onAfterAdded		void	Được gọi sau khi thêm vào CSDL
onBeforeUpdated		void	Được gọi trước khi cập nhật vào CSDL
onAfterUpdated		void	Được gọi sau khi cập nhật vào CSDL
onBeforeDeleted		void	Được gọi trước khi xóa khỏi CSDL
*Không có sự kiện "onAfterDeleted" vì khi đối tượng đã bị loại khỏi hệ giám sát của EF thì sẽ không khả dụng để gọi ngược			

### 3. \_EntityAbstract1<T>

- Mức truy cập: public.
- Loại: lớp ảo.
- Lớp cha: Không có.
- Lớp giao diện thực thi: \_EFEventRegisterInterface, \_CRUDInterface<T>.
- Mô tả: Các lớp thực thể bắt buộc phải có kế thừa trực tiếp hoặc gián tiếp lớp này. Vì đây là các thuộc tính cơ bản cần phải có.
- Thuộc tính và phương thức

*Bảng PL2.3: Thiết kế lớp ảo \_EntityAbstract1<T>*

Mức truy cập	Loại truy cập	Tên	Kiểu dữ liệu	Mô tả
public		id	Guid	Khóa chính

public		subId	string	Mã phụ
public		date_create	DateTime?	Ngày tạo
public		date_modified	DateTime?	Ngày cập nhật gần nhất
public		mota	String	Mô tả
public		order	long?	Dùng để sắp xếp thứ tự
protected	static	db	OurDBContext	Truy xuất đến lớp cung cấp dữ liệu
public	static	USNAME	String	Tên tiếng anh của lớp này
public	static	VNNAME	String	Tên tiếng việt của lớp này
protected		init()	void	Hàm khởi tạo giá trị ban đầu khi đối tượng được tạo mới
protected	static	calculate OrderColumn()	void	Tự động tính toán trường order cho lớp thực thể T
public	static	convert()	List<T>	Chuyển danh sách Id thành danh sách đối tượng
protected		buildLog()	Dictionary <string, string>	Xây dựng thông tin đối tượng khi lưu vết hệ thống

#### 4. \_EntityAbstract3<T>

- Mức truy cập: public.
- Loại: lớp ảo.
- Lớp cha: \_EntityAbstract1<T>.
- Lớp giao diện thực thi: kế thừa từ lớp cha.
- Mô tả: Các lớp cần xác thực đăng nhập sẽ kế thừa lớp này.
- Thuộc tính và phương thức

Bảng PL2.4: Thiết kế lớp ảo *\_EntityAbstract3<T>*

Mức truy cập	Loại truy cập	Tên	Kiểu dữ liệu	Mô tả
public		hoten	String	Họ tên đầy đủ
public		username	String	Tên đăng nhập
public		password	String	Mật khẩu dạng băm
public	static	checkLoginById()	Boolean	Kiểm tra đăng nhập bằng Id
public	static	checkLoginByUserName()	Boolean	Kiểm tra đăng nhập bằng tên đăng nhập
public	static	isUsernameExist()	Boolean	Kiểm tra tên đăng nhập có tồn tại
public	static	getByUserName()	T	Lấy đối tượng bằng tên đăng nhập
public		changePassword()	int	Đổi mật khẩu
public	static	hashPassword()	String	Hàm băm mật khẩu

## PHỤ LỤC C

- Do số lượng bảng nhiều, không thể trình bày hết, nên chỉ trình bày đại diện một vài bảng.

### 1. ViTri

- Mức truy cập: public.
- Loại: lớp cứng.
- Lớp cha: \_EntityAbstract1<ViTri>.
- Lớp giao diện thực thi: kế thừa từ lớp cha.
- Mô tả: vị trí.
- Thuộc tính và phương thức

*Bảng PL3.1: Thiết kế lớp cứng ViTri*

Mức truy cập	Tên	Kiểu dữ liệu	Mô tả
public	coso	CoSo	
public	day	Dayy	
public	tang	Tang	
public	phongs	ICollection<Phong>	
public	request()	ViTri	Lấy đối tượng ViTri theo cơ sở, dãy hoặc tầng

### 2. Phong

- Mức truy cập: public.
- Loại: lớp cứng.
- Lớp cha: \_EntityAbstract1<Phong>.

- Lớp giao diện thực thi: kế thừa từ lớp cha.
- Mô tả: phòng (phòng học, phòng chức năng).
- Thuộc tính và phương thức

*Bảng PL3.2: Thiết kế lớp cứng Phong*

Mức truy cập	Loại truy cập	Tên	Kiểu dữ liệu	Mô tả
public		ten	String	Tên phòng
public		sochongoi	int	Số chỗ ngồi
public		loaiphong	LoaiPhong	Loại phòng
public		vitri	ViTri	Vị trí của phòng
public		cttaisans	ICollection<CTTaiSan>	
public		logsuataisan	ICollection<LogSuaTaiSan>	
public		logtanggiamtaisan	ICollection<LogTangGiamTaiSan>	

### 3. DonVi

- Mức truy cập: public.
- Loại: lớp cứng.
- Lớp cha: \_EntityAbstract1<DonVi>.
- Lớp giao diện thực thi: kế thừa từ lớp cha.
- Mô tả: đơn vị quản lý tài sản.
- Thuộc tính và phương thức



*Bảng PL3.3: Thiết kế lớp cứng DonVi*

Mức truy cập	Loại truy cập	Tên	Kiểu dữ liệu	Mô tả
public		ten	String	Tên cơ sở
public		parent	DonVi	Đơn vị cha
public		loaidonvi	LoaiDonVi	Loại đơn vị
public		permissions	ICollection<Permission>	
		cttaisan_dangsudungs	ICollection<CTTaiSan>	
		cttaisan_dangquanlys	ICollection<CTTaiSan>	
public		childs	ICollection<LoaiDonVi>	

#### 4. TaiSan

- Mức truy cập: public.
- Loại: lớp cứng.
- Lớp cha: \_EntityAbstract1<TaiSan>.
- Lớp giao diện thực thi: kế thừa từ lớp cha.
- Mô tả: tài sản.
- Thuộc tính và phương thức

*Bảng PL3.4: Thiết kế lớp cứng TaiSan*

Mức truy cập	Loại truy cập	Tên	Kiểu dữ liệu	Mô tả
public		Ten	String	Tên thiết bị
public		Dongia	long	Đơn giá
public		nuocsx	String	Nước sản xuất

public		Ten	String	Tên thiết bị
public		loaitaisan	LoaiTaiSan	Thuộc loại tài sản
public		cttaisans	ICollection<CTTaiSan>	
public		logsuataisans	ICollection<LogSuaTaiSan>	
public		logtanggiamtisans	ICollection<LogTangGiamTaiSan>	

### 5. CTTaiSan

- Mức truy cập: public.
- Loại: lớp cứng.
- Lớp cha: \_EntityAbstract1<CTTaiSan>.
- Lớp giao diện thực thi: kế thừa từ lớp cha.
- Mô tả: ghép tài sản với phòng.
- Thuộc tính và phương thức

*Bảng PL3.5: Thiết kế lớp cứng CTTaiSan*

Mức truy cập	Loại truy cập	Tên	Kiểu dữ liệu	Mô tả
public		nguongoc	String	Nguồn gốc
public		ghichu	String	Ghi chú
public		soluong	int	Số lượng
public		phong	Phong	Thuộc phòng
public		thietbi	ThietBi	Thuộc thiết bị
public		tinhtang	TinhTrang	Thuộc tình trạng
public		taisn	TaiSan	Thuộc tài sản

public		parent	CTTaiSan	CTTaiSan cha
public		chungtu	ChungTu	Chứng từ
public		donviquanly	DonVi	Đơn vị quản lý (mang ý nghĩa thống kê)
public		donvisudung	DonVi	Đơn vị sử dụng
public		vitri	ViTri	
public		childs	ICollection<CTTaiSan>	