|  |
| --- |
| **TRƯỜNG ĐẠI HỌC HOA SEN**  **KHOA KHOA HỌC VÀ CÔNG NGHỆ**    **BÁO CÁO ĐỀ TÀI CÔNG NGHỆ DỊCH VỤ WEB**    **Giảng viên hướng dẫn :** Thầy Võ Hoàng Hải  **Sinh viên thực hiện :** Dương Quốc Việt – MSSV: 09263L  **Lớp :** QL092L  **Thời gian :** 01/09/2011 – 20/12/2011    **Tháng 11/ 20117** |

**LỜI MỞ ĐẦU**

Ngày nay khi xã hội ngày càng phát triển thì nhu cầu chuyển đổi chỗ ở do tính chất của công việc và nhu cầu đi du lịch ngày càng tăng. Do đó việc di chuyển từ địa phương này sang địa phương khác hoặc việc tạm trú, tạm vắng ngày càng gia tăng. Điều này dẫn đến các khó khăn trong công tác quản lý thông tin tạm trú, tạm vắng của một cá nhân cụ thể nào đó.

Từ đó đòi hỏi cần có một chương trình quản lý thông tin tạm trú tạm vắng để giúp cho chính quyền địa phương, cũng như các hộ kinh doanh cá nhân (khách sạn, nhà trọ) quản lý thông tin một cách nhanh chóng, dễ dàng và chính xác hơn.

Từ những yêu cầu nêu trên nhóm đã quyết định chọn đề tài ”**Xây dựng Web service cung cấp thông tin tạm trú tạm vắng**”.

Nhóm đã phân tích và đưa ra một số chức năng cơ bản của một web service cung cấp thông tin tạm trú, tạm vắng như sau:

* Quản lý thông tin tạm trú.
  + Tìm kiếm thông tin tạm trú theo họ tên của người tạm trú.
  + Tìm kiếm thông tin theo số điện thoại của người tạm trú.
  + Tìm kiếm thông tin theo chứng minh nhân dân của người tạm trú.
  + Tìm kiếm thông tin theo địa chỉ email cá nhân.
  + Liệt kê danh sách những người đang tạm trú tại địa chỉ hiện tại.
  + Thống kê giới tính theo địa chỉ hiện tại.
  + Thống kê danh sách nghề nghiệp của những người đang tạm trú tại địa chỉ hiện tại.
  + Tìm kiếm danh sách những người đang ở tại địa chỉ hiện tại theo ngày.
  + Cho phép import dữ liệu từ các file excel, csv, cơ sở dữ liệu giúp cho việc đăng ký thông tin tạm trú một cách nhanh chóng hơn, giúp người quản lý có thể sử dụng lại các danh sách đã có sẵng mà không cần nhập lại dữ liệu.
* Quản lý thông tin tạm vắng.
  + Tìm kiếm thông tin tạm vắng theo tên cá nhân.
  + Tìm kiếm thông tin tạm vắng theo số điện thoại cá nhân.
  + Tìm kiếm thông tin tạm vắng theo địa chỉ email cá nhân.
  + Tìm kiếm thông tin tạm vắng của 1 cá nhân theo chứng minh nhân dân.
  + Thống kê xem tại địa chỉ hiển tại có những cá nhân nào đăng ký tạm vắng.
  + Liệt kệ danh sách tạm vắng của địa chỉ hiện tại theo ngày.
* Quản lý thông tin chung.
  + Thêm thông tin tạm trú, tạm vắng.
  + Cập nhật thông tin tạm trú tạm vắng.
  + Xóa thông tin tạm trú, tạm vắng.
  + Đăng ký thành viên.
  + Login: đăng nhập vào hệ thống.

Danh sách thành viên thực hiện đề tài.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Họ tên | MSSV | Lớp | Email |
| 1 | Dương Quốc Việt | 09263L | QL092L | [viet.dq263l@sinhvien.hoasen.edu.vn](mailto:viet.dq263l@sinhvien.hoasen.edu.vn) |

**MỤC LỤC**

[**LỜI MỞ ĐẦU** 2](#_Toc310503111)

[1. Tìm hiểu lý thuyết 7](#_Toc310503112)

[1.1 Giới thiệu Web service & SOA 7](#_Toc310503113)

[1.1.1 Tổng quan Web service 7](#_Toc310503114)

[1.1.2 Tổng quan SOA 7](#_Toc310503121)

[1.2 Các nguyên tắc thiết kế 8](#_Toc310503123)

[1.2.1 Nhóm nguyên lý thực thi 8](#_Toc310503125)

[1.2.2 Nhóm nguyên lý điều chỉnh 8](#_Toc310503131)

[1.3 Standardized Service contract 8](#_Toc310503135)

[1.4 Service Reusability 9](#_Toc310503136)

[1.5 Service Autonomy 9](#_Toc310503137)

[1.6 Service Statelessness 10](#_Toc310503138)

[1.7 Service Discoverability 11](#_Toc310503139)

[1.8 Service Loose Coupling 12](#_Toc310503140)

[1.9 Service Abstraction 12](#_Toc310503141)

[1.10 Service Composability 13](#_Toc310503142)

[2. Thiết kế chương trình 15](#_Toc310503143)

[2.1 Đối tượng phục vụ 15](#_Toc310503144)

[2.1.1 Hộ gia đình cá nhân 15](#_Toc310503145)

[2.1.2 Chính quyền địa phương 15](#_Toc310503146)

[2.2 Kiến trúc hệ thống 15](#_Toc310503147)

[2.3 Các chức năng chính của chương trình. 16](#_Toc310503148)

[2.3.1 GetInfoByName (tìm kiếm thông tin theo tên) 16](#_Toc310503149)

[2.3.2 GetInfoByPhone (tìm kiếm theo số điện thoại) 16](#_Toc310503151)

[2.3.3 GetInfoByIdNumber (tìm kiếm theo chứng minh nhân dân) 17](#_Toc310503153)

[2.3.4 GetInfoBySex (tìm kiếm theo giới tính & địa chỉ) 18](#_Toc310503155)

[2.3.5 GetListByAddress (tìm kiếm theo địa chỉ) 19](#_Toc310503157)

[2.3.6 GetListOccupationByAddress (thống kê nghề nghiệp theo địa chỉ) 20](#_Toc310503159)

[2.3.7 GetInfoByEmail (tìm kiếm thông tin theo địa chỉ email) 20](#_Toc310503161)

[2.3.8 GetListByDate (tìm kiếm danh sách theo ngày) 21](#_Toc310503163)

[2.3.9 CountSexByAddress (thống kê giới tính tại địa chỉ hiện tại) 22](#_Toc310503164)

[2.3.10 GetData (hiển thị tất cả thông tin tạm trú, tạm vắng) 23](#_Toc310503166)

[2.3.11 ImportData (đăng ký thông tin tạm trú từ danh sách có sẵng) 23](#_Toc310503168)

[2.3.12 Login (đăng nhập vào hệ thống) 24](#_Toc310503169)

[2.3.13 InsertData (thêm thông tin tạm trú/ tạm vắng) 24](#_Toc310503170)

[2.3.14 UpdateData (cập nhật thông tin tạm trú/ tạm vắng) 25](#_Toc310503171)

[2.3.15 DeleteData (xóa thông tin tạm trú/ tạm vắng) 25](#_Toc310503172)

[2.3.16 Đối tượng chứa thông tin tạm trú tạm vắng 26](#_Toc310503173)

[2.3.17 Đối tượng chứa thông tin đăng nhập 27](#_Toc310503174)

[2.4 Mô tả cơ sở dữ liệu 27](#_Toc310503175)

[2.4.1 Bảng tamtrutamvang 27](#_Toc310503176)

[2.4.2 Bảng Users 28](#_Toc310503177)

[3. Hiện thực 29](#_Toc310503178)

[3.1 Các màn hình chính của chương trình. 29](#_Toc310503179)

[3.1.1 Giao diện chính của chương trình. 29](#_Toc310503180)

[3.1.2 Form đăng nhập vào hệ thống. 29](#_Toc310503181)

[3.1.3 Form thêm, sửa, xóa thông tin tạm trú/ tạm vắng. 30](#_Toc310503182)

[3.1.4 Tìm kiếm thông tin theo tên. 31](#_Toc310503183)

[3.1.5 Thống kê giới tính theo địa chỉ hiện tại. 31](#_Toc310503184)

[3.1.6 Thống kê nghề nghiệp 32](#_Toc310503185)

[3.1.7 Tìm kiếm thông tin theo ngày 32](#_Toc310503186)

[3.2 Vài đoạn mã đặc biệt 33](#_Toc310503187)

[3.2.1 Service cung cấp chức năng thêm dữ liệu 33](#_Toc310503188)

[3.2.2 Service cung cấp chức năng tìm kiếm theo ngày 34](#_Toc310503190)

[3.2.3 Cách thức lấy dữ liệu phía client 35](#_Toc310503192)

[3.3 Cách cài đặt và chạy chương trình 37](#_Toc310503194)

[3.3.1 Cài đặt cơ sở dữ liệu 37](#_Toc310503195)

[3.3.2 Cài đặt hosting 38](#_Toc310503196)

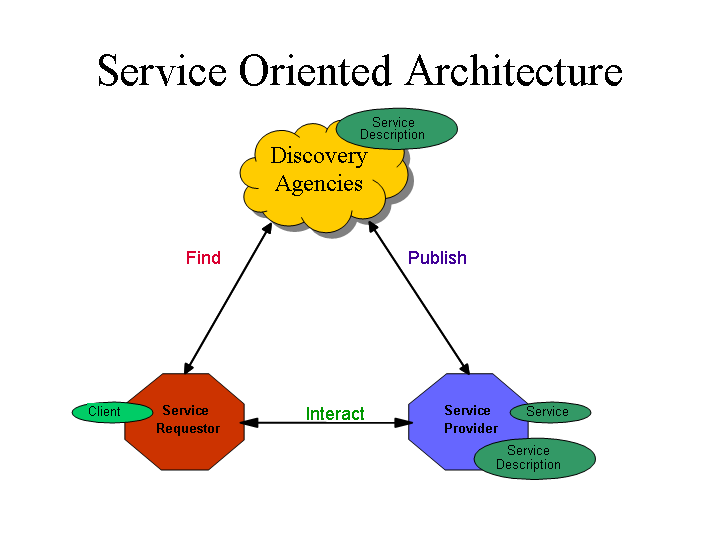
[3.3.3 Chạy chương trình phía Client 38](#_Toc310503197)

[4. Kết luận 39](#_Toc310503198)

[5. Tài liệu tham khảo 40](#_Toc310503199)

1. Tìm hiểu lý thuyết
   1. Giới thiệu Web service & SOA
      1. Tổng quan Web service

Web service phát triển dựa trên khái niệm căn bản của quá trình phân bố các components. Trong đó tất cả các hoạt động sẽ được thực hiện thông qua Internet.



Hình 1.1: Kiến trúc của Web service

Web service gồm 3 thành phần chính:

SOAP: là giao thức truyền dữ liệu được xây dựng dựa trên XML, HTTP, và SMTP.

WSDL: là các lược đồ XML được dùng để mô tả các chức năng, giao diện (interface) của 1 web service.

UDDI: là tổ chức quản lý các services, nơi có sự tham gia của các tổ chức như IBM, Microsoft, ... tham gia cung cấp các dịch vụ.

* + 1. Tổng quan SOA

Kiến trúc hướng dịch vụ (SOA) cho phép các đơn vị logic đơn lẻ tồn tại tự chủ nhưng không cô lập so với các đơn vị khác. Các đơn vị logic này vẫn phải phù hợp với một tập hợp các nguyên tắc cho phép chúng phát triển độc lập, trong khi vẫn duy trì đủ sự tương đồng và các tiêu chuẩn. Trong SOA, các đơn vị này của logic được gọi là dịch vụ. Có thể nghĩ SOA là 1 tập hợp các services.

* 1. Các nguyên tắc thiết kế

Có 8 nguyên tắc thiết kế theo hướng kiến trúc và được chi thành 2 nhóm như sau:

* + 1. Nhóm nguyên lý thực thi
* Standardized Service Contract: dịch vụ đưa ra một thỏa thuận thông tin liên lạc, theo quy định chung bởi một hoặc nhiều mô tả dịch vụ và tài liệu liên quan.
* Service Reusability: các dịch vụ sẽ được chi thành các dịch vụ con với mục đích cho việc tái sử dụng các service.
* Service Autonomy: các dịch vụ sẽ có các quyền tự chủ.
* Service Statelessness: giảm việc lưu trữ các thông tin riêng.
* Service Discoverability: được thiết kế theo hướng mô tả bên ngoài cho phép người dùng tìm kiếm và sử dụng dựa trên các cơ chế khám phá có sẵng.
  + 1. Nhóm nguyên lý điều chỉnh
* Service Loose Coupling: giảm thiểu sự phụ thuộc giữa các service với nhau.
* Service Abstraction: che giấu thông tin logic của service khi hiển thị thông tin ra bên ngoài.
* Service Composability: tập hợp các service có thể phối hợp và ghép lại với nhau.
  1. Standardized Service contract
* Các Service contract thông thường bao gồm mô tả kỹ thuật và phi kỹ thuật. Các tài liệu thường được miêu tả như một dạng giao diện và được biết như API. Do đó các service trong kho dịch vụ phải tuân theo các chuẩn thiết kế contract giống nhau.
* Có 3 loại chuẩn hóa Service contract.
* Chuẩn hóa chức năng (Functional Expression Standardization): Các operation của service khi định nghĩa cần phải áp dụng các qui tắc đặt tên chuẩn, các qui tắc này được áp dụng cho các input message, output message và tên phải phù hợp với ngữ nghĩa của chúng. Điều này nhành giúp diễn giải service contract chính xác hơn, tăng khả năng liên kết nội tại và tính sử dụng lại của service, ngoài ra còn giảm khả năng trùng lắp dữ liệu.
* Chuẩn hóa mô hình dữ liệu (Data Model Standardization): yêu cầu phát triển các mô hình dữ liệu phải được chuẩn hóa để tạo kiến trúc dữ liệu được chuẩn hóa nhằm tăng khả năng sử dụng lại các service trong hệ thống, nhưng điều này không yêu cầu chuẩn hóa toàn bộ data model toàn hệ thống.
* Chuẩn hóa chính sách (Policy Standardization): các policy mô tả giới hạn sử dụng các service, để một service có thể sử dụng lại thì các behavioral requirement cần được trình bày theo phong cách phù hợp với việc sử dụng các policy expression được chuẩn hóa dựa trên các từ vựng chuẩn và phải phù hợp với những đặc điểm của ngôn ngữ.
  1. Service Reusability
* Là khả năng sử dụng lại của service.
* Mục tiêu: giúp tăng khả năng thu hồi vốn đầu tư ban đầu, tăng tính linh hoạt bằng cách thực hiện nhanh chóng các yêu cầu nghiệp vụ trong tương lai của doanh nghiệp.
* Đặc điểm thiết kế:

+ Service được xác định bởi các thuyết luôn luôn đúng của dịch vụ.

+ Service có tính dùng chung cao.

+ Service contract của service có tính dùng chung và có thể mở rộng.

+ Các service logic có thể sử dụng đồng thời.

* 1. Service Autonomy
* Autonomy thể hiện khả năng tự điều chỉnh, được thể hiện qua sự tự do điều khiển đưa ra quyết định cho bản thân mà không cần sự chấp thuận từ phía bên ngoài dịch vụ.
* Để tăng tính tự trị đòi hỏi sự thực thi chương trình phải được cô lập hơn vì vậy sẽ làm tăng mức độ độc lập, kết quả là giúp chương trình tăng độ tin cậy và khả năng dự đoán.
* Mục tiêu:

+ Tăng quyền điều khiển của 1 service trên môi trường thực thi của nó.

+ Tăng độ tin cậy, khả năng dự đoán đặc biệt là được kết hợp và sử dụng lại.

* Có 2 loại service autonomy:

+ Runtime autonomy: mức độ điều khiển của một service trong quá trình xử lý trong thời điểm mà service được gọi và thực thi.

+ Design-time autonomy: mức độ tự do của chủ service để tạo sự thay đổi đối với 1 service trong thời gian tồn tại của nó.

* Đo lường khả năng tự trị của service:

+ Service contract autonomy: phạm vi khả năng của 1 contract không được trùng lắp với những contract khác. Điều này dễ dàng đạt được khi service contract được chuẩn hóa và tách rời với môi trường thực thi.

+ Shared autonomy.

+ Service logic autonomy: là mức độ phổ biết nhất có thể đạt được khi phát triển tùy chỉnh cho một service mới.

+ Pure autonomy.

* 1. Service Statelessness
* Khi sự kết hợp các service ngày càng phức tạp thì số lượng dữ liệu hoạt động cần được quản lý trong suốt thời gian sống của composition. Đề tăng khả năng mở rộng và sự thực thi trong kho service thì các service và kiến trúc cần được thiết kế hỗ trợ sự ủy nhiệm và trì hoãn các việc quản lý trạng thái.
* Mục tiêu: tăng khả năng mở rộng service, hỗ trợ việc thiết kế agnostic service logic và cải tiến khả năng sử dụng lại service.
* Nguyên lý thiết kế: tùy thuộc vào service model và cách tiếp cận trì hoãn được dùng mà có các loại đặc điểm thiết kế khác nhau được thực hiện.
* Các loại state:

+ Active state: qui định của service sau khi nó được gọi và thực thi bởi một chương trình sử dụng.

+ Passive state: một service được load vào bộ nhớ nhưng nó không thực hiện bất kỳ chức năng nào của nó một cách tích cực.

+ Stateful: service được active và sử dụng thêm vùng nhớ tạm để lưu trữ tạm thời và xử lý state data. Stateful gồm 3 loại: session data, context data, business data.

+ Stateless: service được active và không xử dụng thêm vùng nhớ tạm để xử lý.

* 1. Service Discoverability
* Khả năng khám phá, giải thích và trình bày service.
* Điều này sẽ giúp cho xác định service đã tồn tại hay cần phải xây dựng mới. Để làm được điều này đòi hỏi thông tin mô tả service phải chính xác thể hiện đầy đủ khả năng của service, nếu mô tả không đầy đủ thì người dùng sẽ mất cơ hội sử dụng lại service điều này dẫn đến sự trùng lắp với các service đã tồn tại.
* Mục tiêu: các service được đặt ở vị trí như các resource có khả năng khám phá cao, mục đích và khả năng của service được trình bày rõ ràng bởi con người cũng như chương trình phần mềm.
* Đặc điểm thiết kế:

+ Trang bị dữ liệu phù hợp.

+ Trang bị thêm những thông tin mô tả với mục đích sử dụng của chúng và những khả năng với con người.

+ Nếu một service đã được đăng ký thì các bản ghi đăng ký được phổ biến là những meta information.

+ Nếu một service không được đăng ký thì các tài liệu sơ lược về service được thiết kế bổ sung cho Service contract và đó là dạng cơ bản cho các bản ghi đăng ký trong tương lai.

+ Khả năng khám phá và trình bày dịch vụ.

+ Cải thiện sự truyền chất lượng thông tin của dữ liệu mô tả dịch vụ.

* Những loại khám phá:

+ Design-time discovery: tiến trình khám phá được thực hiện một cách thủ công bởi con người.

+ Runtime discovery: service có khả năng đưa ra những truy vấn discovery động và kết quả là một quá trình tự động hóa.

* 1. Service Loose Coupling
* Service loose coupling nhằm giảm sự phụ thuộc vào các service khác mà bản thân service sử dụng, cũng như giảm sự phụ thuộc vào chương trình sử dụng bản thân service.
* Mục tiêu: giảm sự phụ thuộc giữa các service, tăng sự thực thi độc lập.
* Đặc điểm thiết kế: sự tồn tại của một service contract phải tách rời về công nghệ và sự thực thi. Không phụ thuộc vào xử lý bên ngoài, giảm tối thiểu các yêu cầu có sự gắn kết với consumer.
* Các loại service contract coupling:

+ Logic to contact coupling: phạm vi của 1 service được giới hạn bởi contract của nó, đây là dạng liên kết mang tính tích cự.

+ Contract to logic coupling: phạm vi của 1 service phụ thuộc vào logic bên dưới, đây là loại liên kết mang tính tích cực vì các chương trình service consumer tạo ra sụ ràng buộc với service contract.

+ Contract to technology coupling: là sự phụ thuộc vào công nghệ giao tiếp độc quyền. Loại liên kết này không mang tính tích cực vì khi service phụ thuộc vào công nghệ sẽ dẫn đến consumer cũng sẽ phụ thuộc vào công nghệ.

+ Contract to implement coupling: là sự phụ thuộc vào môi trường thực thi, đây cũng là loại liên kết không mang tính tích cực.

+ Contract to functional coupling: là sự phụ thuộc của service đối với external logic.

* 1. Service Abstraction
* Các service contract chỉ chứa các thông tin cần thiết và các thông tin này sẽ giới hạn những thông tin gì sẽ được publish trong các service contract.
* Mục tiêu: giữ cho số lượng và nội dung thông tin súc tích, cân bằng và ngăn ngừa sự truy cập không cho phép.
* Đặc điểm thiết kế: thông tin sẽ được trừu tượng về công nghệ, logic vác các chức năng bên ngoài dịch vụ, thông tin về một service sẽ được điều khiển hoặc ẩn trong một môi trường cụ thể.
* Có 4 loại Meta abstraction:

+ Technology information: mô tả thực thi về kỹ thuật bên dưới. Thông tin chi tiết về kỹ thuật sẽ được ẩn đi để tạo sự tự do về thay đổi kỹ thuật mà không ảnh hưởng đến người sử dụng service.

+ Function information: metadata sẽ mô tả service có những chức năng nào.

+ Programatic logic information: mô tả cách thức dịch vụ thực hiện khả năng của nó. Được áp dụng ở những mức thiết kế chi tiết thấp như thuật toán, xử lý lỗi, ...

+ Quality of service information: metadata mô tả service behavior, các giới hạn và yêu cầu tương tác. Chất lượng của service data là một giới hạn của vùng thông tin liên quan đến độ tin cậy.

* 1. Service Composability
* Khả năng kết hợp các service lại với nhau.
* Mục tiêu: tất cả các mục đích của service reusability có thể áp dụng, và là một dạng sử dụng lại dịch vụ.
* Các khái niệm và thuật ngữ tích hợp:

+ Composition.

+ Composition Instance.

+ Composition member.

+ Composition controller.

+ Composition sub-controller.

+ Composition member capability.

+ Composition controller capability.

+ Designated controller.

+ Collective composability.

+ Service activity.

+ Point-to-point.

+ Primitive composition.

+ Complex composition.

+ Composition initiator.

1. Thiết kế chương trình
   1. Đối tượng phục vụ
      1. Hộ gia đình cá nhân

Là những chủ nhà trọ, chủ khách sạn hay các doanh nghiệp thường xuyên có khách hàng cư trú lại doanh nghiệp của mình và có nhu cầu quản lý thông tin khách hàng của mình.

* + 1. Chính quyền địa phương

Ngoài việc phục vụ cho các hộ gia đình cá nhân, chương trình còn có thể ứng dụng cho chính quyền địa phương dùng để quản lý thông tin tạm trú, tạm vắng của các cá nhân tại địa phương của mình.

* 1. Kiến trúc hệ thống

Hình 2.1: Kiến trúc hệ thống

Hệ thống chia làm 3 phần chính: Client application, WCF Service Host và WCF Service Library.

* Client applicaiton: là ứng dụng phía người sử dụng, có thể là windows application hoặc web applicaion.
* WCF Service Host: chứa các file cấu hình kết nối đến các Service Library.
* WCF Service Library: là hệ thống chính của WS cung cấp thông tin tạm trú tạm vắng, các chức năng của hệ thống sẽ được mô tả tại đây.
  1. Các chức năng chính của chương trình.
     1. GetInfoByName (tìm kiếm thông tin theo tên)

Chương trình cho phép người quản lý tìm kiếm thông tin tạm trú, tạm vắng theo tên của một cá nhân. Khi người dùng cung cấp thông tin là họ tên của một cá nhân cụ thể, chương trình sẽ cho biết được cá nhân đó đã từng đăng ký tạm trú, tạm vắng tại những địa chỉ nào. Ngoài ra hệ thống còn cung cấp địa chỉ hiện tại đang cư trú của cá nhân đó.

**TranferRecord[] GetInfoByName(string Name, string type)**

* Tham số truyền vào: Name, type.
* Kết quả trả về: mãng đối tượng TranferRecord
* [OperationContract(Name="FindInfoByName")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Name | nvarchar(64) | Họ tên cần tìm kiếm  Ví dụ: Dương Quốc Việt |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một mãng đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.1: tìm kiếm thông tin theo tên

* + 1. GetInfoByPhone (tìm kiếm theo số điện thoại)

Hệ thống cho phép người dùng tìm kiếm thông tin tạm trú, tạm vắng của một cá nhân cụ thể theo số điện thoại. Khi người sử dụng chương trình cung cấp thông tin là số điện thoại của một cá nhân, hệ thống sẽ trả về tất cả dữ liệu tạm trú/ tạm vắng của cá nhân liên quan.

**TranferRecord[] GetInfoByPhone(string Phone, string type)**

* Tham số truyền vào: Phone, type
* Kết quả trả về: mãng đối tượng TranferRecord
* [OperationContract(Name="FindInfoByPhone")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Phone | nchar(10) | Số điện thoại cần tìm kiếm  Ví dụ: 0909455034 |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.2: tìm kiếm theo số điện thoại

* + 1. GetInfoByIdNumber (tìm kiếm theo chứng minh nhân dân)

Hệ thống cho phép tìm kiếm thông tin tạm trú, tạm vắng của một cá nhân theo chứng minh nhân dân. Người sử dụng chỉ cần nhập vào số chứng minh nhân dân, hệ thống.

**TranferRecord[] GetInfoByIdNumber(string IDNumber, string type)**

* Tham số truyền vào: IdNumber, type
* Kết quả trả về: danh sách các đối tượng
* [OperationContract(Name="FindInfoByIdNumber")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| IDNumber | nchar(10) | Số chứng minh nhân dân/ passport  Ví dụ: 023859423 |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.3: tìm kiếm theo chứng minh nhân dân

* + 1. GetInfoBySex (tìm kiếm theo giới tính & địa chỉ)

Hệ thống cho phép kê liệt kê danh sách những người đang tạm trú hoặc tạm vắng theo giới tính và địa chỉ hiện tại.

**TranferRecord[]GetInfoBySex (string Address, string Sex, string type)**

* Tham số truyền vào: Address, Sex, type
* Kết quả trả về: dãy các đối tượng.
* [OperationContract(Name="ListPersonBySex")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Address | nvarchar(128) | Địa chỉ cần tìm  Ví dụ: 023859423 |
| Sex | nchar(10) | Giới tính. Ví dụ: Nam/ Nữ |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.4: tìm kiếm theo giới tính & địa chỉ.

* + 1. GetListByAddress (tìm kiếm theo địa chỉ)

Hệ thống cho phép người sử dụng tìm kiếm theo địa chỉ, người dùng nhập địa chỉ cần tìm. Hệ thống sẽ trả về danh sách những người đang tạm trú hoặc tạm vắng tại địa chỉ đã nhập.

**TranferRecord[] GetListByAddress(string Address, string type)**

* Tham số truyền vào: Address, type
* Kết quả trả về: mãng các đối tượng.
* [OperationContract(Name="ListPersonByAddress")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Address | nvarchar(128) | Địa chỉ cần tìm |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.5: tìm kiếm theo địa chỉ

* + 1. GetListOccupationByAddress (thống kê nghề nghiệp theo địa chỉ)

Ngoài các chức năng cơ bản, hệ thống còn cho phép người quản lý thống kê giới tính tại địa chỉ hiện tại, chức năng này giúp người quản lý dễ dàng trong việc thống kê giới tính cho địa phương của mình.

**TranferRecord[] GetListOccupationByAddress(string Address, string type)**

* Tham số truyền vào: Address, type
* Kết quả trả về: mãng các đối tượng.
* [OperationContract(Name="ListOccupationByAddress")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Address | nvarchar(128) | Địa chỉ cần tìm |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.6: thống kê nghề nghiệp theo địa chỉ

* + 1. GetInfoByEmail (tìm kiếm thông tin theo địa chỉ email)

Hệ thống cho phép người dùng tìm kiếm thông tin tạm trú, tạm vắng theo địa chỉ email.

**TranferRecord[] GetInfoByEmail(string Email, string type)**

* Tham số truyền vào: Email, type
* Kết quả trả về: mãng các đối tượng.
* [OperationContract(Name = "FindInfoByEmail")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Email | nvarchar(64) | Địa chỉ email |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.7: tìm kiếm thông tin theo địa chỉ email

* + 1. GetListByDate (tìm kiếm danh sách theo ngày)

Hệ thống cho phép tìm kiếm danh sách theo địa chỉ từ ngày đến ngày. Khi người dùng cung cấp thông tin là địa chỉ cần tìm, khoảng thời gian cần tìm kiếm, hệ thống sẽ cung cấp danh sách tương ứng.

**TranferRecord[] GetListByDate(string Address, DateTime FromDate, DateTime ToDate, string type)**

* Tham số truyền vào: Address, FromDate, ToDate, type.
* Kết quả trả về: danh sách đội tượng.
* [OperationContract(Name="ListPersonByDate")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Address | nvarchar(128) | Địa chỉ cần tìm |
| FromDate | Date | Ngày bắt đầu tạm trú/ tạm vắng |
| ToDate | Date | Ngày kết thúc tạm trú/ tạm vắng |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng chứa các thông tin cá nhân như họ tên, CMND, địa chỉ tạm trú/ tạm vắng, ... |

Table 2.8: tìm kiếm danh sách theo ngày

* + 1. CountSexByAddress (thống kê giới tính tại địa chỉ hiện tại)

Hệ thống cho phép thống kê giới tính theo địa chỉ hiện tại, chức năng giúp cơ quan địa phương thống kê giới tính, ...

**TranferRecord[] CountSexByAddress(string Address, string type)**

* Tham số truyền vào: Address, type.
* Kết quả trả về: mãng đối tượng thống kê giới tính tại địa chỉ cần tìm.
* [OperationContract(Name="CountSexByAddress")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Address | nvarchar(128) | Địa chỉ tạm trú/ tạm vắng |
| Type | nchar(10) | Loại thông tin cần tìm kiếm (tạm trú/ tạm vắng). |
| TranferRecord | Object | Kết quả trả về là một đối tượng thống kê giới tính của địa chỉ cần tìm. |

Table 2.9: thống kê giới tính tại địa chỉ hiện tại

* + 1. GetData (hiển thị tất cả thông tin tạm trú, tạm vắng)

Chức năng liệt kê tất cả danh sách tạm trú, tạm vắng theo từng hộ gia đình hoặc địa phương.

**TranferRecord[] GetData()**

* Tham số truyền vào: không có.
* Kết quả trả về: danh sách các đối tượng chứa thông tin tạm trú, tạm vắng.
* [OperationContract(Name=”LoadData”)]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| TranferRecord | Object | Kết quả trả về là danh sách các đối tượng chứa thông tin tạm trú, tạm vắng. |

Table 2.10: Lấy dữ liệu

* + 1. ImportData (đăng ký thông tin tạm trú từ danh sách có sẵng)

Cho phép import dữ liệu từ các file excel, csv, cơ sở dữ liệu giúp cho việc đăng ký thông tin tạm trú một cách nhanh chóng hơn, giúp người quản lý có thể sử dụng lại các danh sách đã có sẵng mà không cần nhập lại dữ liệu.

**void ImportData(string[] file)**

* Tham số truyền vào: danh sách các file chứa thông tin khách hàng cần đăng ký tạm trú
* Kết quả trả về: không có.
* [OperationContract(Name = "ImportDataFromFile", IsOneWay = true)]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| file | array | Danh sách các file chứa thông tin khách hàng cần đăng ký tạm trú. |

Table 2.11: đăng ký thông tin tạm trú từ danh sách có sẵng

* + 1. Login (đăng nhập vào hệ thống)

Hệ thống cung cấp chức năng đăng nhập, điều này bắt buộc khi người sử dụng chương trình cần thêm mới, cập nhật hoặc xóa dữ liệu.

**int Login(string username, string password)**

* Tham số truyền vào: username, password
* Kết quả trả về: kiểu dữ liệu int. 1 là đăng nhập thành công, 0 là đăng nhập thất bại.
* [OperationContract(Name = "Login")]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| username | nvarchar(10) | Tài khoản đăng nhập |
| Password | nchar(10) | Mật khẩu đăng nhập |
| Result | Int | Kết quả đăng nhập, thành công hoặc thất bại. |

Table 2.12: đăng nhập vào hệ thống

* + 1. InsertData (thêm thông tin tạm trú/ tạm vắng)

Hệ thống cho phép người sử dụng chương trình đăng ký thông tin tạm trú, tạm vắng.

**void InsertData(TranferRecord data)**

* Tham số truyền vào: đối tượng chứa tất cả thông tin tạm trú/ tạm vắng.
* Kết quả trả về: không có.
* [OperationContract(Name = "ImportDataFromFile", IsOneWay = true)]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Data | TranferRecord | Đối tượng chứa tất cả thông tin cá nhân của người muốn đăng ký thông tin tạm trú/ tạm vắng. |

Table 2.13: thêm thông tin tạm trú/ tạm vắng

* + 1. UpdateData (cập nhật thông tin tạm trú/ tạm vắng)

Hệ thống cung cấp chức năng cập nhật thông tin tạm trú/ tạm vắng. Cho phép người quản lý cập nhật các thông tin của đối tượng tham gia tạm trú hoặc tạm vắng.

**void UpdateData(int Id, TranferRecord data)**

* Tham số truyền vào: Id, data
* Kết quả trả về: không có
* [OperationContract(Name = "UpdatInformation", IsOneWay = true)]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Id | Int | Id của cá nhân sẽ cập nhật thông tin |
| Data | TransferRecord | Đối tượng chứa các thông tin như: họ tên, ngày sinh, .... đây là những thông tin cần cập nhật. |

Table 2.14: cập nhật thông tin tạm trú/ tạm vắng

* + 1. DeleteData (xóa thông tin tạm trú/ tạm vắng)

Hệ thống cung cấp chức năng xóa thông tin cá nhân của đối tượng tham gia tạm trú/ tạm vắng.

**void DeleteData(int Id)**

* Tham số truyền vào: Id
* Kết quả trả về: không có
* [OperationContract(Name = "DeleteInformation", IsOneWay = true)]

|  |  |  |
| --- | --- | --- |
| Name | Description | Comment |
| Id | Int | Id của cá nhân sẽ xóa thông tin |

Table 2.15: xóa thông tin tạm trú/ tạm vắng

* + 1. Đối tượng chứa thông tin tạm trú tạm vắng

[DataContract]

public class TranferRecord

{

[DataMember]

public int Total;

[DataMember]

public int Id;

[DataMember]

public string FullName;

[DataMember]

public string PhoneNumber;

[DataMember]

public string Email;

[DataMember]

public DateTime Birthday;

[DataMember]

public string Sex;

[DataMember]

public string OriginalAddress;

[DataMember]

public string IDNumber;

[DataMember]

public string Occupation;

[DataMember]

public string CurrentAddress;

[DataMember]

public DateTime FromDate;

[DataMember]

public DateTime ToDate;

[DataMember]

public string Reason;

[DataMember]

public string Description;

[DataMember]

public string Type;

}

* + 1. Đối tượng chứa thông tin đăng nhập

[DataContract]

public class UserRecord

{

[DataMember]

public string username;

[DataMember]

public string password;

}

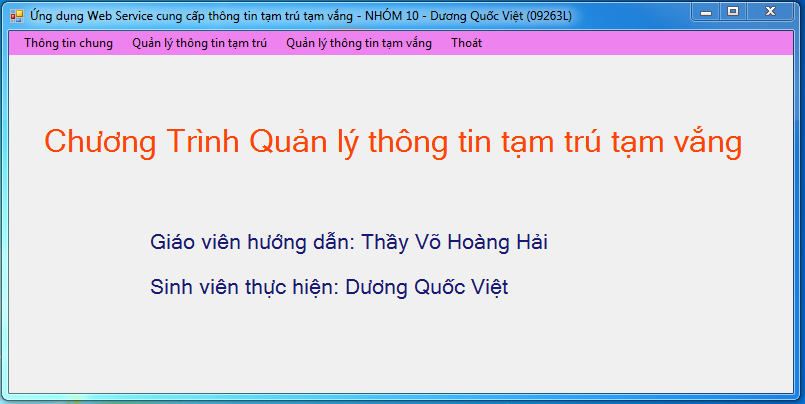
* 1. Mô tả cơ sở dữ liệu
     1. Bảng tamtrutamvang

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Value Type | Null | Description |
| Id | Int | No | Khóa chính |
| FullName | nvarchar(64) | Yes | Họ tên |
| PhoneNumber | nchar(10) | Yes | Số điện thoại |
| Email | nchar(64) | Yes | Địa chỉ email |
| Birthday | date | Yes | Ngày sinh |
| Sex | nchar(10) | Yes | Giới tính |
| OriginalAddress | nvarchar(128) | Yes | Địa chỉ thường trú |
| IDNumber | nchar(10) | Yes | Chứng minh nhân dân/ Passport |
| Occupation | nvarchar(50) | Yes | Nghề nghiệp |
| CurrentAddress | nvarchar(128) | Yes | Địa chỉ hiện tại (tạm trú/ tạm vắng) |
| FromDate | Date | Yes | Ngày bắt đầu tạm trú/ tạm vắng |
| ToDate | Date | Yes | Ngày kết thúc tạm trú/ tạm vắng |
| Reason | Text | Yes | Lý do tạm trú/ tạm vắng |
| Description | Text | Yes | Mô tả chi tiết |
| Type | nchar(10) | Yest | Loại (tạm trú/ tạm vắng). Ví dụ: tạm trú (“TT”), tạm vắng (“TV”) |

* + 1. Bảng Users

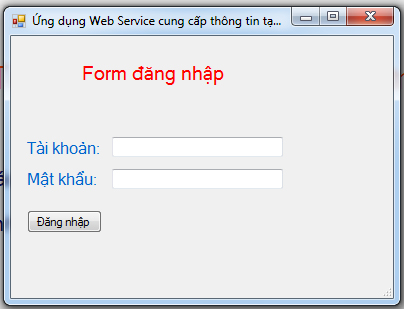
|  |  |  |  |
| --- | --- | --- | --- |
| Name | Value Type | Null | Description |
| Username | nvarchar(10) | No | Tài khoản đăng nhập |
| Password | nchar(10) | No | Mật khẩu đăng nhập |

1. Hiện thực
   1. Các màn hình chính của chương trình.
      1. Giao diện chính của chương trình.



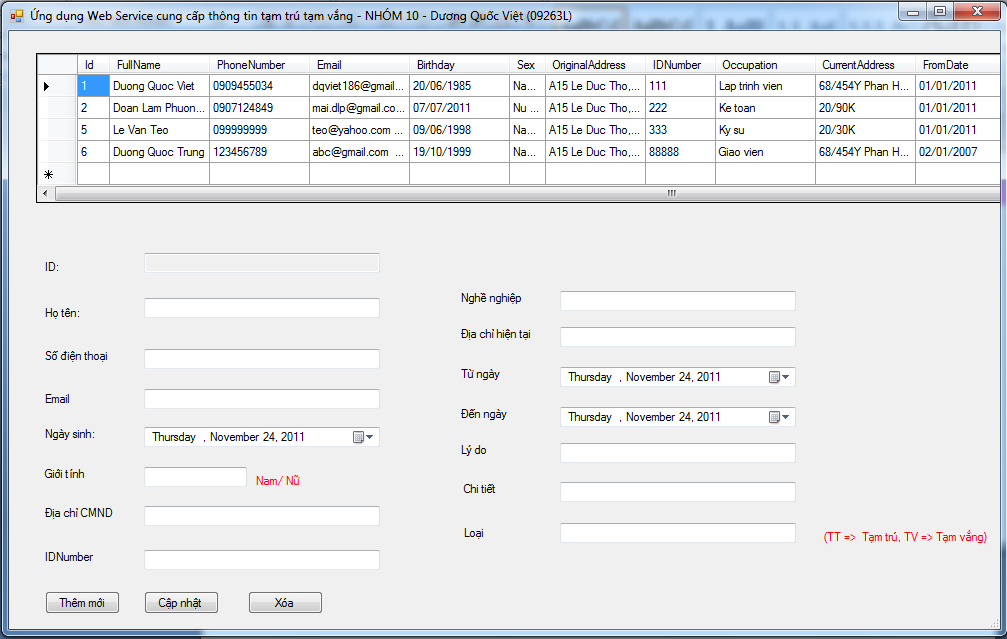
Hình 3.1: Giao diện chính của chương trình

* + 1. Form đăng nhập vào hệ thống.

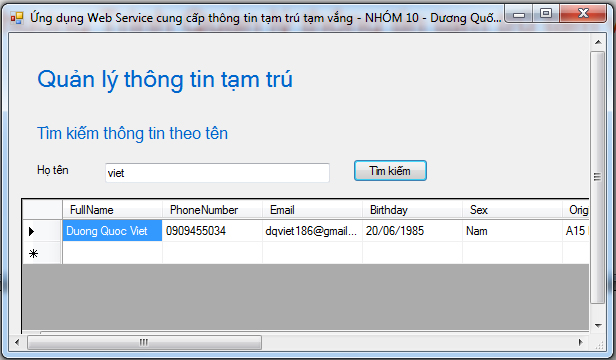


Hình 3.2: Form đăng nhập vào hệ thống

* + 1. Form thêm, sửa, xóa thông tin tạm trú/ tạm vắng.

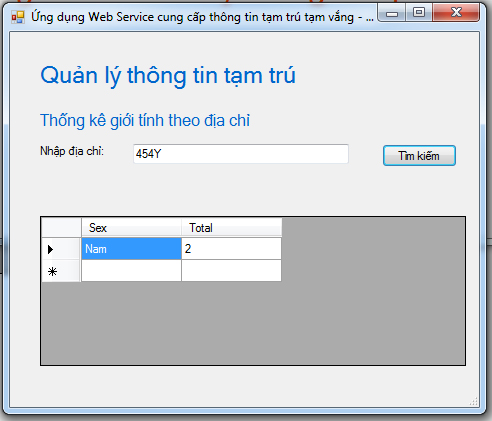
Hình 3.3: Form thêm, sửa, xóa thông tin

* + 1. Tìm kiếm thông tin theo tên.



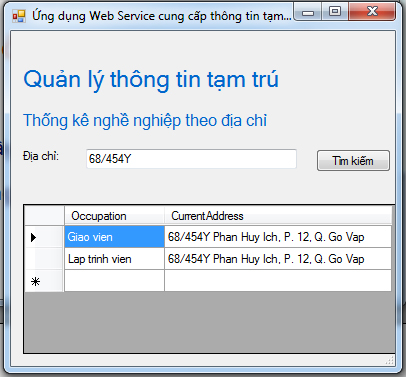
Hình 3.4: Tìm kiếm thông tin theo tên

* + 1. Thống kê giới tính theo địa chỉ hiện tại.



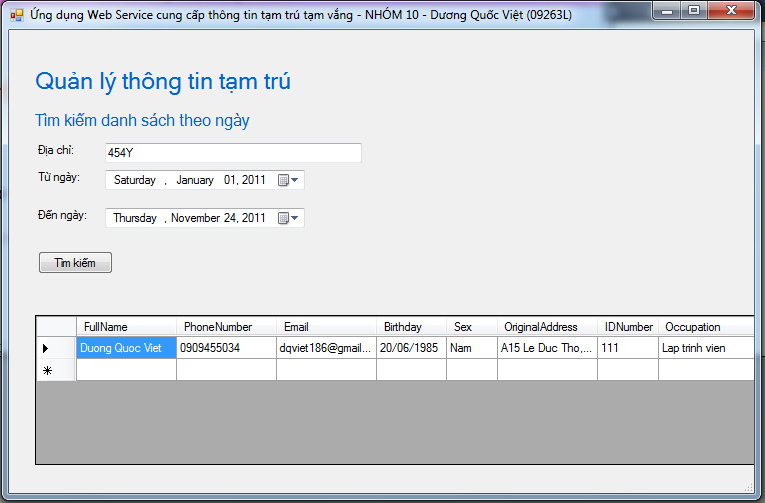
Hình 3.5: Thống kê giới tính theo địa chỉ

* + 1. Thống kê nghề nghiệp



Hình 3.6: Thống kê nghề nghiệp theo địa chỉ

* + 1. Tìm kiếm thông tin theo ngày



Hình 3.7: Tìm kiếm thông tin theo ngày

* 1. Vài đoạn mã đặc biệt
     1. Service cung cấp chức năng thêm dữ liệu

public void InsertData(TranferRecord data)

{

try

{

string connectionString = "Data Source=mavi-PC;Initial Catalog=cnweb;Integrated Security=True";

myConnection = new SqlConnection(connectionString);

// Open connection

myConnection.Open();

string updateCement = "INSERT INTO tamtrutamvang (FullName,PhoneNumber,Email,Birthday,Sex,OriginalAddress,IDNumber,Occupation,CurrentAddress,FromDate,ToDate,Reason,Description,Type) VALUES (@FullName,@PhoneNumber,@Email,@Birthday,@Sex,@OriginalAddress,@IDNumber,@Occupation,@CurrentAddress,@FromDate,@ToDate,@Reason,@Description,@Type)";

SqlCommand cmd = new SqlCommand(updateCement, myConnection);

cmd.Parameters.AddWithValue("@FullName", data.FullName);

cmd.Parameters.AddWithValue("@PhoneNumber", data.PhoneNumber);

cmd.Parameters.AddWithValue("@Email", data.Email);

cmd.Parameters.AddWithValue("@Birthday", data.Birthday.ToString("yyyy-MM-dd"));

cmd.Parameters.AddWithValue("@Sex", data.Sex);

cmd.Parameters.AddWithValue("@OriginalAddress", data.OriginalAddress);

cmd.Parameters.AddWithValue("@IDNumber", data.IDNumber);

cmd.Parameters.AddWithValue("@Occupation", data.Occupation);

cmd.Parameters.AddWithValue("@CurrentAddress", data.CurrentAddress);

cmd.Parameters.AddWithValue("@FromDate", data.FromDate.ToString("yyyy-MM-dd"));

cmd.Parameters.AddWithValue("@ToDate", data.ToDate.ToString("yyyy-MM-dd"));

cmd.Parameters.AddWithValue("@Reason", data.Reason);

cmd.Parameters.AddWithValue("@Description", data.Description);

cmd.Parameters.AddWithValue("@Type", data.Type);

cmd.ExecuteNonQuery();

// close connection

myConnection.Close();

}

catch (PisNotFoundException exp)

{

TrackedFault tf = new TrackedFault(

Guid.NewGuid(),

exp.Message,

DateTime.Now);

throw new FaultException<TrackedFault>(

tf,

new FaultReason("PisNotFoundException"),

FaultCode.CreateReceiverFaultCode(new FaultCode("InsertInfomation")));

}

catch (Exception exp)

{

FaultReasonText reason = new FaultReasonText(exp.Message);

throw new FaultException(new FaultReason(reason), FaultCode.CreateReceiverFaultCode(new FaultCode("InsertInfomation")));

}

}

* + 1. Service cung cấp chức năng tìm kiếm theo ngày

public TranferRecord[] GetListByDate(string Address, DateTime FromDate, DateTime ToDate, string type)

{

try

{

SqlConnection cn = new SqlConnection();

cn.ConnectionString = "Data Source=mavi-PC;Initial Catalog=cnweb;Integrated Security=True";

cn.Open();

string str1 = FromDate.ToString("yyyy-MM-dd");

string str2 = ToDate.ToString("yyyy-MM-dd");

string query = "SELECT \* FROM tamtrutamvang WHERE CurrentAddress like '%" + Address + "%' and FromDate >= '" + str1 + "' or ToDate <= '" + str2 + "' and Type = '" + type + "'";

SqlCommand sql = new SqlCommand(query, cn);

SqlDataAdapter adt = new SqlDataAdapter(sql);

DataSet ds = new DataSet();

adt.Fill(ds);

int rows = ds.Tables[0].Rows.Count;

data = new TranferRecord[rows];

for (int i = 0; i < rows; i++)

{

data[i] = new TranferRecord();

data[i].Id = Convert.ToInt32(ds.Tables[0].Rows[i].ItemArray[0].ToString());

data[i].FullName = ds.Tables[0].Rows[i].ItemArray[1].ToString();

data[i].PhoneNumber = ds.Tables[0].Rows[i].ItemArray[2].ToString();

data[i].Email = ds.Tables[0].Rows[i].ItemArray[3].ToString();

data[i].Birthday = Convert.ToDateTime(ds.Tables[0].Rows[i].ItemArray[4]);

data[i].Sex = ds.Tables[0].Rows[i].ItemArray[5].ToString();

data[i].OriginalAddress = ds.Tables[0].Rows[i].ItemArray[6].ToString();

data[i].IDNumber = ds.Tables[0].Rows[i].ItemArray[7].ToString();

data[i].Occupation = ds.Tables[0].Rows[i].ItemArray[8].ToString();

data[i].CurrentAddress = ds.Tables[0].Rows[i].ItemArray[9].ToString();

data[i].FromDate = Convert.ToDateTime(ds.Tables[0].Rows[i].ItemArray[10]);

data[i].ToDate = Convert.ToDateTime(ds.Tables[0].Rows[i].ItemArray[11]);

data[i].Reason = ds.Tables[0].Rows[i].ItemArray[12].ToString();

data[i].Description = ds.Tables[0].Rows[i].ItemArray[13].ToString();

data[i].Type = ds.Tables[0].Rows[i].ItemArray[14].ToString();

}

cn.Close();

return data;

}

catch (PisNotFoundException exp)

{

TrackedFault tf = new TrackedFault(

Guid.NewGuid(),

exp.Message,

DateTime.Now);

throw new FaultException<TrackedFault>(

tf,

new FaultReason("PisNotFoundException"),

FaultCode.CreateReceiverFaultCode(new FaultCode("GetListByDate")));

}

catch (Exception exp)

{

FaultReasonText reason = new FaultReasonText(exp.Message);

throw new FaultException(new FaultReason(reason), FaultCode.CreateReceiverFaultCode(new FaultCode("GetListByDate")));

}

}

* + 1. Cách thức lấy dữ liệu phía client

// tao dataset

public DataSet CreateData()

{

DataSet ds = new DataSet();

ds.Tables.Add("tamtrutamvang");

//ds.Tables[0].Columns.Add("Id");

ds.Tables[0].Columns.Add("FullName");

ds.Tables[0].Columns.Add("PhoneNumber");

ds.Tables[0].Columns.Add("Email");

ds.Tables[0].Columns.Add("Birthday");

ds.Tables[0].Columns.Add("Sex");

ds.Tables[0].Columns.Add("OriginalAddress");

ds.Tables[0].Columns.Add("IDNumber");

ds.Tables[0].Columns.Add("Occupation");

ds.Tables[0].Columns.Add("CurrentAddress");

ds.Tables[0].Columns.Add("FromDate");

ds.Tables[0].Columns.Add("ToDate");

ds.Tables[0].Columns.Add("Reason");

ds.Tables[0].Columns.Add("Description");

//ds.Tables[0].Columns.Add("Type");

return ds;

}

public DataSet dataTTTV(string Address, DateTime FromDate, DateTime ToDate, string type)

{

DataSet ds = new DataSet();

ServiceReference1.pisClient proxy = new ServiceReference1.pisClient();

TTTVService.TranferRecord[] result = proxy.ListPersonByDate(Address, FromDate, ToDate,type);

ds = CreateData();

DataRow dr;

for (int i = 0; i < result.Length; i++)

{

dr = ds.Tables["tamtrutamvang"].NewRow();

//dr["Id"] = result[i].Id;

dr["FullName"] = result[i].FullName;

dr["PhoneNumber"] = result[i].PhoneNumber;

dr["Email"] = result[i].Email;

dr["Birthday"] = result[i].Birthday.ToString("dd/MM/yyyy");

dr["Sex"] = result[i].Sex;

dr["OriginalAddress"] = result[i].OriginalAddress;

dr["IDNumber"] = result[i].IDNumber;

dr["Occupation"] = result[i].Occupation;

dr["CurrentAddress"] = result[i].CurrentAddress;

dr["FromDate"] = result[i].FromDate.ToString("dd/MM/yyyy");

dr["ToDate"] = result[i].ToDate.ToString("dd/MM/yyyy");

dr["Reason"] = result[i].Reason;

dr["Description"] = result[i].Description;

//dr["Type"] = result[i].Type;

// add row to table

ds.Tables["tamtrutamvang"].Rows.Add(dr);

}

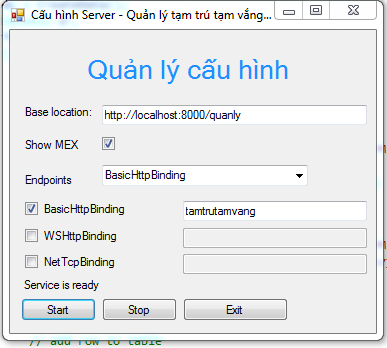
return ds;

}

* 1. Cách cài đặt và chạy chương trình
     1. Cài đặt cơ sở dữ liệu

Đầu tiên ta sẽ chạy chương trình Miscrosoft SQL Server Management Studio, sau đó chọn click phải chuột vào Database -> Attach … -> Chọn cnweb.mdf

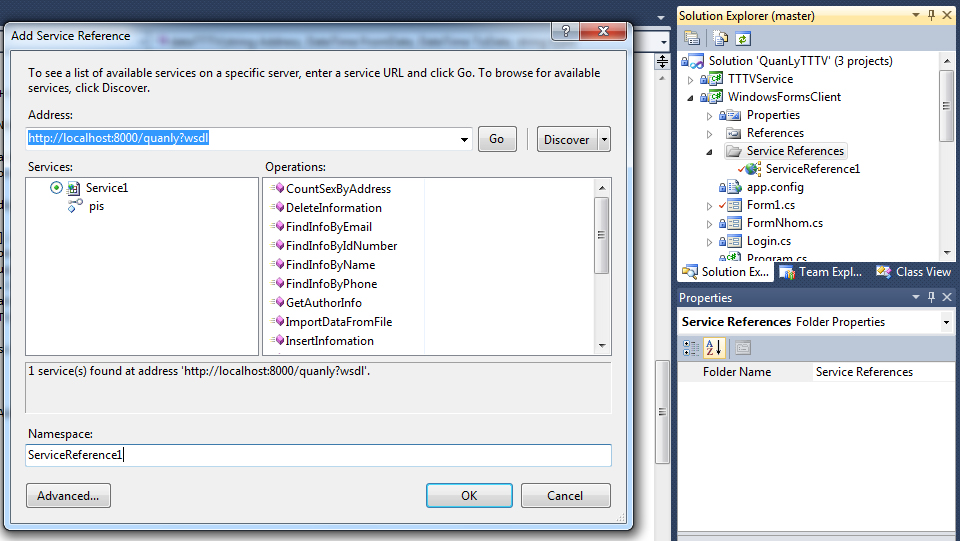
* + 1. Cài đặt hosting



Hình 3.8: cài đặt WCF Hosting

Sau khi điền thông số như trên Nhất Start để start WCF Service Library.

* + 1. Chạy chương trình phía Client



Hình 3.9: cài đặt WCF Service cho Client

Để Client xử dụng được các chức năng hệ thống, client phải dùng Service reference đến WCF Library.

Add service reference như hình trên để cấu hình cho client để client có thể sử dụng các chức năng của hệ thống.

1. Kết luận
   1. ***Ưu điểm***

* Hoàn thành đề tài đúng thời gian đưa ra.
* Giao diện thân thiện đối với người sử dụng.
* Đáp ứng đầy đủ các chức năng cơ bản của việc cung cấp thông tin tạm trú, tạm vắng.
* Có thể áp dụng ở mức độ quản lý cơ quản.
  1. ***Khuyết điểm***
* Giải thuật chưa được tối ưu.
* Chưa xử lý các ngoại lệ.
* Tốc độ xử lý còn chậm.
* Chỉ quản lý ở mức độ đơn giản, chưa quản lý ở mức cao hơn như thống kê theo quận, tỉnh/ thành phố.
* Chưa xử lý việc import dữ liệu từ các file CSV, txt, … để đăng ký thông tin tạm trú, tạm vắng.
* Chưa hiện thực chức năng tìm kiếm theo email.
  1. ***Hướng phát triển***
* Nâng cao tốc độ xử lý.
* Hệ thống cung cấp các báo cáo, thống kê theo thời gian thực. Các bản thống kê được biểu diễn bằng các hình ảnh trực quan.
* Quản trị phân quyền sử dụng một cách chi tiết hơn như: phân quyền phân hệ, chức năng, hành động.
* Kiểm tra thêm các ràng buộc.

1. Tài liệu tham khảo

* <http://msdn.microsoft.com>
* <http://www.telerik.com/help/aspnet-ajax/grid-client-side-binding-to-wcf-service-adonet-dataservice.html>
* <http://www.codeproject.com/KB/combobox/valuemembersample.aspx>
* <http://www.planet-source-code.com>