**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC HOA SEN**

KHOA KHOA HỌC VÀ CÔNG NGHỆ



**Tên đề tài: Xây dựng ứng dụng Web Service**

**cung cấp thông tin điểm học tập**

**Giảng viên: Võ Hoàng Hải**

**Sinh viên:**

**1.** Nguyễn Thanh Phong  **080873**

**2.** Đặng Minh Thành **080890**

**3.** Nguyễn Đình Tín **080922**

**Lớp: QL081**

**Khoá: 2008 - 2012**

**Tháng 12 Năm 2011**

**Lời nói đầu**

Ngày nay, Công nghệ thông tin phát triển mạnh mẽ đồng hành với nhu cầu trao đổi thông tin và sử dụng các dịch vụ ngày càng tăng, có nhiều phương thức giúp người dùng có thể sử dụng nhiều dịch vụ, chia sẻ dữ liệu như: rmi, dcom hoặc corba service nhưng còn mang nhiều hạn chế.

Công nghệ Web Services ra đời là một cuộc cách mạng hóa cách thức hoạt động của các dịch vụ B2B và B2C. Web Service đã mở ra một hướng mới cho việc phát triển các ứng dụng trên Internet.Web services kết hợp sử dụng nhiều công nghệ khác nhau cho phép hai ứng dụng cùng ngôn ngữ, độc lập hệ điều hành trao đổi được với nhau thông qua môi trường mạng Internet. Tuy nhiên những công nghệ sử dụng ở đây không nhất thiết phải là những công nghệ mới. Đây là điểm khác biệt của web services so với các công nghệ khác, đó chính là khả năng kết hợp các công nghệ đã có như là XML, SOAP, WSDL, UDDI để tạo ra các service, đặc điểm này làm nổi bật vai trò của Web Services.

Với môn Công nghệ và dịch vụ web chúng tôi học trong học kỳ 11.1A tại trường đại học Hoa Sen, đã giúp cho chúng tôi có cái nhìn cặn cẽ hơn về công nghệ và dịch vụ web. Qua đó chúng tôi đã thực hiện đề tài ”Xây dựng ứng dụng Web Service cung cấp thông tin điểm học tập”, để vận dụng những kiến thức đã học vào việc ứng dụng trong thực tế.

**Mục lục**

[I. TÌM HIỂU LÝ THUYẾT 6](#_Toc311657625)

[1. Service Oriented Architecture 6](#_Toc311657626)

[1.1. Service 6](#_Toc311657627)

[1.2. Các đặc điểm chính của service 6](#_Toc311657628)

[1.3. Service Oriented Architecture 7](#_Toc311657629)

[1.3.1. Các actor trong SOA 7](#_Toc311657630)

[1.3.2. Ích lợi khi sử dụng SOA 8](#_Toc311657631)

[1.3.3. Thông điệp (message) trong SOA 8](#_Toc311657632)

[2. Web Service 9](#_Toc311657633)

[2.1. Cấu trúc Web service 9](#_Toc311657634)

[2.2. Simple Object Access Protocol – SOAP 11](#_Toc311657635)

[2.2.1. Các đặc điểm của SOAP 11](#_Toc311657636)

[2.2.2. SOAP Message Structures 11](#_Toc311657637)

[2.2.3. Web Service Description Language – WSDL 13](#_Toc311657638)

[II. PHÂN TÍCH THIẾT KẾ CHƯƠNG TRÌNH 15](#_Toc311657639)

[1. Kiến trúc hệ thống 15](#_Toc311657640)

[1.1. Nhóm chức năng hỗ trợ người dùng 15](#_Toc311657641)

[1.1.1. CheckLogin 15](#_Toc311657642)

[1.1.2. ChangePassword 15](#_Toc311657643)

[1.1.3. Search\_Subject 16](#_Toc311657644)

[1.1.4. Send\_Feedback 17](#_Toc311657645)

[1.2. Nhóm contracts quản lý thông tin cá nhân 17](#_Toc311657646)

[1.2.1. Get\_Info\_Student 17](#_Toc311657647)

[1.2.2. Update\_Info\_Student () 18](#_Toc311657648)

[1.2.3. Get\_Info\_Lec 18](#_Toc311657649)

[1.2.4. Update\_Info\_Lec 18](#_Toc311657650)

[1.3. Nhóm contracts cung cấp điểm sinh viên 19](#_Toc311657651)

[1.3.1. Get\_Mark (Overloading, Name = “Mark\_Subject\_Single”) 19](#_Toc311657652)

[1.3.2. Get\_Mark (Overloading, Name = “Mark\_Subject\_All”) 20](#_Toc311657653)

[1.3.3. Info\_Semester (Overloading, Name = “Info\_Stu\_Sem\_Only”) 20](#_Toc311657654)

[1.3.4. Info\_Semester (Overloading, Name = “Info\_Stu\_Sem\_All”) 20](#_Toc311657655)

[1.3.5. Get\_Avg\_Semester 21](#_Toc311657656)

[1.3.6. Get\_Avg\_Cumulative 21](#_Toc311657657)

[1.4. Nhóm contracts giành cho giảng viên 22](#_Toc311657658)

[1.4.1. Lec\_Sub\_Single 22](#_Toc311657659)

[1.4.2. Lec\_Sub\_All 22](#_Toc311657660)

[1.4.3. List\_Class 22](#_Toc311657661)

[1.4.4. UploadResource 23](#_Toc311657662)

[1.4.5. DownloadResource 24](#_Toc311657663)

[1.5. Nhóm contracts cung cấp thông tin chung 24](#_Toc311657664)

[1.5.1. List\_Semester 24](#_Toc311657665)

[1.5.2. List\_Subject 24](#_Toc311657666)

[1.6. Nhóm contracts cung cấp dữ liệu thống kê 25](#_Toc311657667)

[1.6.1. Info\_Statistic\_Stu\_General 25](#_Toc311657668)

[1.6.2. Get\_Min\_Mark 25](#_Toc311657669)

[1.6.3. Get\_Max\_Mark 26](#_Toc311657670)

[1.6.4. Get\_Avg\_Mark 26](#_Toc311657671)

[1.6.5. Class\_Stu\_Total. 27](#_Toc311657672)

[1.6.6. Statistic\_Mark\_Equal 27](#_Toc311657673)

[1.6.7. Statistic\_Mark\_Distance 28](#_Toc311657674)

[1.6.8. Statistic\_Mark\_Less 29](#_Toc311657675)

[1.6.9. Statistic\_Mark\_Greater 29](#_Toc311657676)

[2. Datacontract 30](#_Toc311657677)

[3. Cơ sở dữ liệu 41](#_Toc311657678)

[4. Hiện thực chương trình 42](#_Toc311657679)

[4.1. Service host 42](#_Toc311657680)

[4.2. Client 42](#_Toc311657681)

[4.2.1. Đăng nhập, phân quyền 42](#_Toc311657682)

[4.2.2. Thay đổi mật khẩu 43](#_Toc311657683)

[4.2.3. Quản lý profile 44](#_Toc311657684)

[4.2.4. Xem kết quả 44](#_Toc311657685)

[4.2.5. Thống kê tiến trình học tập 45](#_Toc311657686)

[4.2.6. Tìm kiếm 46](#_Toc311657687)

[4.2.7. Gửi phản hồi 46](#_Toc311657688)

[4.2.8. Thông tin lớp môn học 47](#_Toc311657689)

[4.2.9. Tải lên – Tải về tài liệu 49](#_Toc311657690)

[III. KẾT LUẬN 50](#_Toc311657691)

[1. Ưu điểm 50](#_Toc311657692)

[2. Khuyết điểm 50](#_Toc311657693)

[3. Hướng phát triển 50](#_Toc311657694)

[IV. Cài đặt và sử dụng 51](#_Toc311657695)

[1. Cài đặt 51](#_Toc311657696)

[2. Sử dụng 51](#_Toc311657697)

**Phân tích**

1. TÌM HIỂU LÝ THUYẾT

# Service Oriented Architecture

## Service

Về mặt định nghĩa, service là một hệ thống có khả năng nhận một hay nhiều yêu cầu xử lý và sau đó đáp ứng lại bằng cách trả về một hay nhiều kết quả. Quá trình nhận yêu cầu và trả kết quả về được thực hiện thông qua các interface đã được định nghĩa trước đó. Thông thường việc giao tiếp này được thực hiện trên các interface đã được chuẩn hóa và sử dụng rộng rãi.

Một ví dụ đơn giản của service chính là hoạt động của một nhà hàng. Khi khách hàng vào nhà hàng và gọi thức ăn, khách hàng đang tiến hành gởi yêu cầu cho service “phục vụ khách hàng” của nhà hàng. Nhân viên nhà hàng nhậc được yêu cầu của khách, nếu món ăn khách hàng yêu cầu nhà hàng không có hoặc đã hết, nhân viên nhà hàng sẽ từ chối hoặc đề nghị khách hàng gọi món khác. Nếu nhà hàng có thể đáp ứng được yêu cầu của khách, món ăn sẽ được chế biến và mang ra cho khách hàng thưởng thức sau một khoảng thời gian chờ. Ở đây, yêu cầu chính là món ăn mà khách hàng muốn thưởng thức, còn kết quả trả về của service phục vụ nhà hàng chính là từ chối (nếu nhà hàng không đáp ứng được yêu cầu của khách) hay là món ăn mà khách hàng cần.

Một hệ thống được thiết kế theo kiểu hướng service (service oriented) là một hệ thống trong đó các chức năng của hệ thống được xây dựng dựa trên các service có độ kết dính thấp. Các service trong hệ thống giao tiếp với nhau thông qua việc gởi nhận các thông điệp (message).

## Các đặc điểm chính của service

* Có ranh giới rõ ràng (Boundaries Are Explicit)

Mỗi service được xây dựng dựa trên các interface chuẩn hóa đã được sử dụng rộng rãi.  
Chi tiết hiện thực của mỗi service sẽ không được thể hiện ra bên ngoài. Mỗi service chỉ công bố một số các interface của nó cho user có thể dùng để gởi các yêu cầu và nhận kết quả.

* Tính tự trị (Autonomous)

Về mặt lý thuyết, mỗi service có tính độc lập cao, có thể được build và đưa vào sử dụng mà không phụ thuộc vào các service khác.

* Share the Schema and Contract, Not the Class

Về mặt trao đổi dữ liệu, các service không truyền các class và type. Thay vào đó, các class và type sẽ được đặc tả hình thức (data được đặc tả trong schema, behavior được đặc tả thành các contract )

* Service Compatibility Is Based on Policy

Sự tương thích giữa các service được căn cứ vào các policy.   
Tương thích về mặt cấu trúc dựa trên các đặc tả hình thức bao gồm contract (dựa trên Web Service Description Language (WSDL) hoặc Business Process Execution Language for Web Services (BPEL4WS)) và schema (XSD)

Sự tương thích dựa trên policy cung cấp khả năng phân tích cũng như đảm bảo sự tương thích giữa các service.

## Service Oriented Architecture

Service Oriented Architecture (SOA) cung cấp cơ chế cho phép các hệ thống hoạt động trên các platform khác nhau có thể giao tiếp với nhau.

Một hệ thống được xây dựng theo mô hình SOA bao gồm các service thỏa mãn các tính chất của service ở mục 1.2. Mỗi service trong hệ thống có thể được sửa đổi một cách độc lập với các service khác nhằm mục đích đáp ứng một yêu cầu mới từ thực tế.

### Các actor trong SOA

Hình 1.3.1.1 mô tả các actor tham gia trong một hệ thống xây dựng theo SOA.

* Service Provider: Cung cấp stateless service phục vụ cho một nhu cầu nào đó. User (service consumer) không cần quan tâm đến vị trí thực sự mà service họ cần sử dụng đang hoạt động.
* Serive Consumer: User sử dụng service được cung cấp bởi Service Provider
* Service Registry: Nơi lưu trữ thông tin về các service của các Service Provider khác nhau, Service Consumer dựa trên những thông tin này để tìm kiếm và lựa chọn Service Provider phù hợp.

Hình 1.3.1.1: SOA actors

Service Provider sẽ đăng kí thông tin về service mà mình có thể cung cấp (các chức năng có thể cung cấp, khả năng của hệ thống (resource, performance), giá cả dịch vụ, ...) vào Service Registry. Service Consumer khi có nhu cầu về một service nào đó sẽ tìm kiếm thông tin trên Service Registry. Ngoài chức năng hỗ trợ tìm kiếm, Service Registry còn có thể xếp hạng các Service Provider dựa trên các tiêu chí về chất lượng dịch vụ, bầu chọn từ các khách hàng đã sử dụng service, ... Những thông tin này sẽ hỗ trợ thêm cho quá trình tìm kiếm của Service Consumer. Khi đã xác định được Service Provider mong muốn, Service Consumer thiết lập kênh giao tiếp trực tiếp với Service Provider nhằm sử dụng service hoặc tiến hành thương lượng thêm (về mặt giá cả, resource sử dụng, ...)

### Ích lợi khi sử dụng SOA

Sử dụng mô hình SOA trong việc thiết kế hệ thống mang lại lợi ích về mặt kinhtế cũng như kỹ thuật.

* Lợi ích kinh tế
  + Doanh nghiệp có điều kiện tập trung thời gian để tìm kiếm các giải pháp cho các bài toán liên quan đến kinh tế.
  + Thúc đẩy sự phát triển của hệ thống hiện có cũng như cung cấp khả năng mở rộng hệ thống trong tương lai.
* Lợi ích kỹ thuật
  + Hệ thống xây dựng theo mô hình SOA đảm bảo các service trong hệ thống có tính độc lập cao (độ kết dính thấp) (autonomous và loose coupling).
  + Ở góc nhìn người sử dụng, vị trí các service có tính trong suốt (transparency), việc di dời các service đến một máy tính khác không ảnh hưởng khả năng phục vụ yêu cầu khách hàng.
  + Hoạt động của các service có tính động, hành vi của các service tùy thời đểm, tùy yêu cầu cần xử lý mà có sự khác nhau (late binding).

### Thông điệp (message) trong SOA

So với kiểu thiết kế Component-Based, điểm khác biệt chính của SOA là cung cấp khả năng giao tiếp giữa các thành phần trong hệ thống (service) sử dụng thông điệp (message) dựa trên các chuẩn giao tiếp đã được chuẩn hóa (HTTP, FTP, SMTP, ...). Chính nhờ đặc điểm này, hệ thống SOA trở nên độc lập với platform (platform independent). Các service hoạt động trên nền các platform khác nhau vẫn có thể giao tiếp với nhau nhờ vào các interface giao tiếp đã được chuẩn hóa để cộng tác xử lý một tác vụ nào đó.

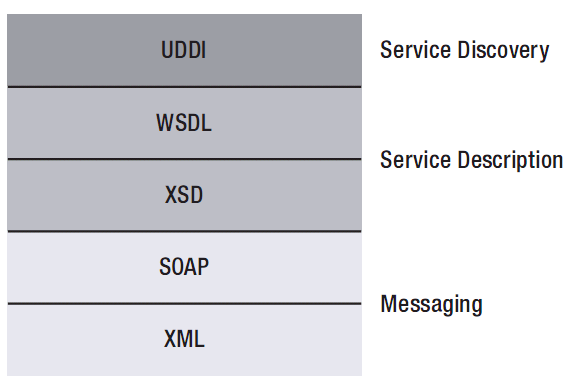
Sử dụng thông điệp (message) để giao tiếp có các lợi thế sau:

* Cross-platform: thông điệp (message) trở thành ngôn ngữ chung của các platform và các ngôn ngữ lập trình khác nhau. Điều này đảm bảo các service trên các platform khác nhau hoạt động với cấu trúc dữ liệu đặc thù của platform đó.
* Asynchronous communications: hoạt động gởi nhận thông điệp được thực hiện theo cơ chế Fire-and-Forget. Sender và Receiver không cần phải chờ thông điệp trả lời sau khi đã gởi đi một thông điệp. Điều này giúp cho Sender và Receiver tiếp tục xử lý công việc sau khi gởi thông điệp mà không cần dừng thực thi để chờ thông điệp trả lời.
* Reliable communication: các thông điệp từ Sender có thể được gởi đến một service trung gian có nhiệm vụ lưu trữ (store) các thông điệp. Service trung gian sẽ gởi (forward) thông điệp cho Receiver khi Receiver có thể xử lý yêu cầu tiếp theo. Cơ chế Store-and-Forward này đảm bảo các thông điệp sẽ không bị thất lạc trong trường hợp Receiver bị quá tải và không thể nhận thêm yêu cầu mới.
* Thread management: Việc trao đổi thông điệp theo cơ chế bất đồng bộ giúp ứng dụng không cần ngừng thực thi để chờ một tác vụ kết thúc mà có thể tạo ra các thread xử lý các công việc khác nhau.
* Remote communication: Các thông điệp lưu trữ thông tin về các đối tượng dữ liệu dưới dạng đặc tả hình thức thay thế việc phải serialization and deserialization các đối tượng dữ liệu truyền qua mạng khi ứng dụng thực hiện remote call một ứng dụng khác.
* End-to-end security: Thông điệp có thể lưu trữ thông tin về security context của kênh giao tiếp. Điều này cung cấp khả năng điều khiển liên quan đến security như authentication and authorization.

# Web Service

Web service là một hiện thực của service sử dụng web protocol để giao tiếp. Web service cung cấp hệ thống phần mềm hỗ trợ việc giao tiếp giữa các máy tính trên network.

## Cấu trúc Web service

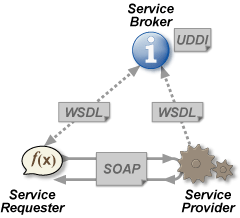


Hình 2.1.1: Web service protocol stack

Hình 2.1.1 mô tả các layer hình thành nên Web service. Hình 2.1.2 mô tả các Web service actor tương ứng với các layer này.

Tương tự với SOA, có 3 actor chính tham gia vào Web service.

* Service Provider: Dùng Web Services Description Language (WSDL) để mô tả dịch vụ mà mình có thể cung cấp cho Service Broker (tương tự với Service Registry trong SOA).
* Service Broker: Lưu trữ thông tin về các service được cung cấp bởi các Service Provider. Cung cấp chức năng tìm kiếm hỗ trợ Service Requester (Service Consumer trong SOA) trong việc xác định Service Provider phù hợp. Thành phần chính của Service Broker là Universal Discovery, Description, and Integration (UDDI) repositories.
* Service Requester: Dùng WSDL để đặc tả nhu cầu sử dụng (loại service, thời gian sử dụng, resource cần thiết, mức giá ...) và gởi cho Service Broker. Bằng việc sử dụng UDDI và chức năng tìm kiếm của Service Broker, Service Requester có thể tìm thấy Service Provider thích hợp. Ngay sau đó, giữa Service Requester và Service Provider thiết lập kênh giao tiếp sử dụng SOAP để thương lượng giá cả và các yếu tố khác trong việc sử dụng service.



*Hình 2.1.2: Web service actors*

## Simple Object Access Protocol – SOAP

SOAP là một protocol giao tiếp dùng trong Web service được xây dựng dựa trên XML. SOAP được sử dụng để đặc tả và trao đổi thông tin về các cấu trúc dữ liệu cũng như các kiểu dữ liệu giữa các thành phần trong hệ thống.

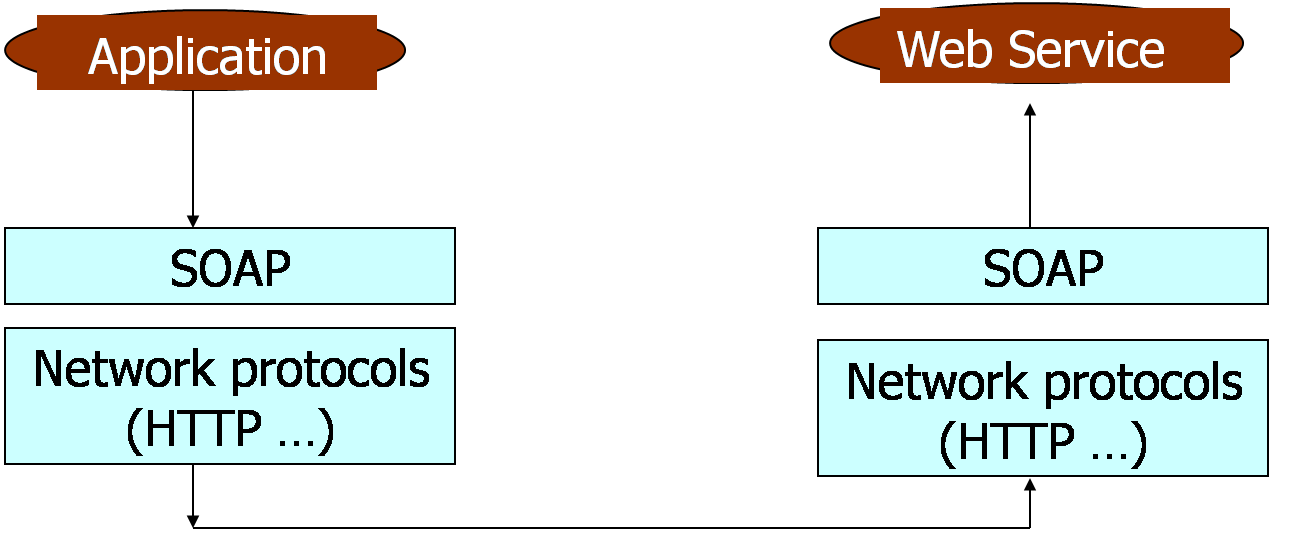
Sử dụng SOAP, ứng dụng có thể yêu cầu thực thi method trên máy tính ở xa mà không cần quan tâm đến chi tiết về platform cũng như các phần mềm trên máy tính đó.

### Các đặc điểm của SOAP

* Khả năng mở rộng (Extensible): Cung cấp khả năng mở rộng phục vụ cho nhu cầu đặc thù của ứng dụng và nhà cung cấp. Các chức năng về bảo mật, tăng độ tin cậy có thể đưa vào phần mở rộng của SOAP. Các nhà cung cấp dịch vụ khác nhau, tùy vào đặc điểm hệ thống của mình có thể định nghĩa thêm các chức năng mở rộng nhằm tăng thêm lợi thế cạnh tranh cũng như cung cấp thêm tiện ích cho người sử dụng.
* Có thể hoạt động trên các network protocol đã được chuẩn hóa (HTTP, SMTP, FTP, TCP, ...)
* Độc lập với platform, ngôn ngữ lập trình hay programming model được sử dụng.

### SOAP Message Structures

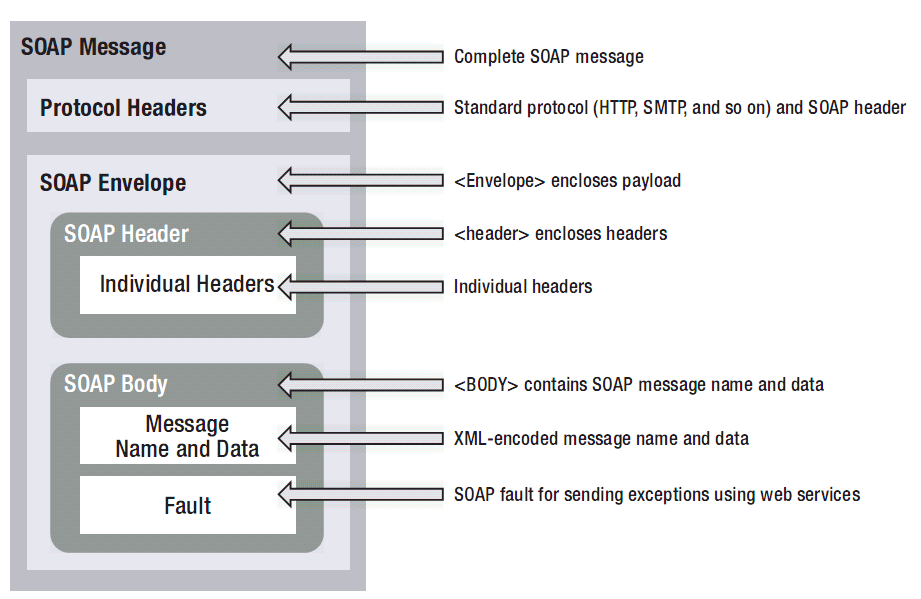
Hình 2.2.2.1 mô tả giao tiếp của một ứnng dụng với một web service được thực hiện qua thông điệp SOAP sử dụng network protocol HTTP. Ứng dụng sẽ đặc tả yêu cầu trong SOAP message và thông qua network protocol gởi đến cho web service. Web service sẽ nhận và phân tích yêu cầu sau đó trả về kết quả thích hợp.



Hình 2.2.2.1: Messaging using SOAP

Hình 2.2.2.2 mô tả cấu trúc một thông điệp SOAP. Một thông điệp SOAP bao gồm các thành phần sau:

* Protocol Header: Cho biết thông tin về các chuẩn giao thức được sử dụng.
* SOAP Envelop: Thông tin chính của message bao gồm:
  + SOAP Header: Chứa các SOAP header.
  + SOAP body: Thông tin về name và data được đặc tả dưới dạng XML. Ngoài ra còn có trường lỗi được dùng để gởi các web service exception.



*Hình 2.2.2.2 SOAP message structure*

Sau đây là ví dụ về SOAP message, trong đó message request được sử dụng để yêu cầu web service thực hiện method ‘int doubleAnInteger (int numberToDouble)’ nhằm nhân đôi một số nguyên. Messge request sẽ truyền đối số 123(số nguyên cần nhân đôi) trong phần SOAP body:

* Request

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>

<SOAP-ENV:Envelope

SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"

xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:SOAP-ENC=<http://schemas.xmlsoap.org/soap/encoding/> xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/1999/XMLSchema">

<SOAP-ENV:Body>

<ns1:doubleAnInteger xmlns:ns1="urn:MySoapServices">

<param1 xsi:type="xsd:int">**123**</param1>

</ns1:doubleAnInteger>

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>

* Response

<?xml version="1.0" encoding="UTF-8" ?>

<SOAP-ENV:Envelope

xmlns:SOAP-ENV=<http://schemas.xmlsoap.org/soap/envelope/>

xmlns:xsi=<http://www.w3.org/1999/XMLSchema-instance>

xmlns:xsd="http://www.w3.org/1999/XMLSchema">

<SOAP-ENV:Body>

<ns1:doubleAnIntegerResponse xmlns:ns1="urn:MySoapServices"

SOAP-ENV: encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

<return xsi:type="xsd:int">246</return>

</ns1:doubleAnIntegerResponse>

</SOAP-ENV:Body>

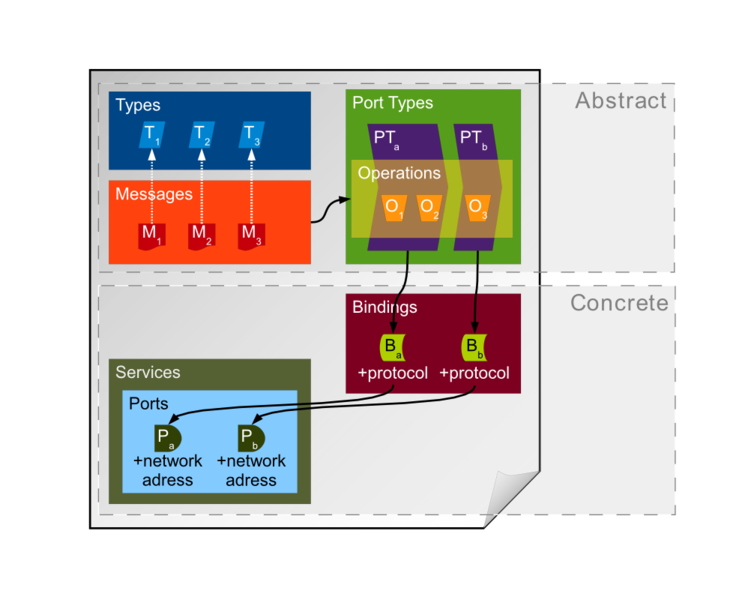
</SOAP-ENV:Envelope>

Message response mang thông tin kết quả trả về trong phần body. Kết quả của phép nhân đôi số nguyên 123 (được truyền bởi message request) là 246.

### Web Service Description Language – WSDL

Việc đặc tả chức năng, interface giao tiếp của một web service được thực hiện dựa vào Web Service Description Language (WSDL). Việc đặc tả này bao gồm protocol được sử dụng, cấu trúc và định dạng dữ liệu dùng để giao tiếp...

Hình 2.2.3.1 mô tả các thành phần cơ bản của một file WSDL dùng để đặc tả một web service.

* Services: Chứa các method có thể được sử dụng thông qua các web protocol.
* Ports: Địa chỉ dùng để kết nối đến web service. Thông thường, ports được mô tả bằng một HTTP URL. 

*Hình 2.2.3.1: Overview of WSDL*

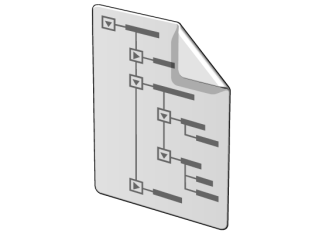
* Port Types: định nghĩa một web service, các tác vụ mà service cung cấp và định dạng các thông điệp được sử dụng để khởi động các tác vụ này.
* Operations: Mỗi operation có thể được xem như một method hay một lời gọi hàm trong các ngôn ngữ lập trình cổ điển.
* Binding: chỉ định port type, các operation, SOAP binding stype (RPC/Document), SOAP protocol được dùng.
* Message: Mỗi message tương ứng với một operation và chứa các thông tin cần thiết để thực thi operation đó. Mỗi message có một name duy nhất và một hay nhiều logical part. Các logical part được phân biệt với nhau qua name và có thể lưu trữ các tham số cần cho operation.
* Element: Được định nghĩa trong Types. Mỗi element có một name duy nhất và kiểu dữ liệu. Element được dùng để đặc tả dữ liệu dùng trong message. Element có thể đặc tả các dữ liệu đơn giản (string, integer) hay phức tạp hơn như array, struct, ...
* XSD file: Các element thường được định nghĩa trong các XML Schema Definition (XSD) file. XSD file có thể ở trong cùng file WSDL hoặc ở file riêng biệt.

1. PHÂN TÍCH THIẾT KẾ CHƯƠNG TRÌNH

# Kiến trúc hệ thống

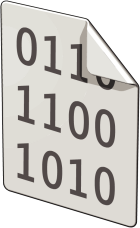


Client

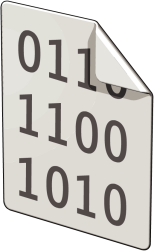
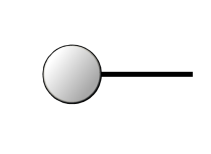
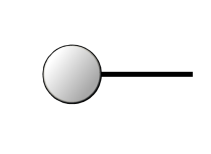
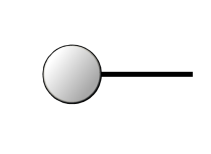


Service Host

Configuration



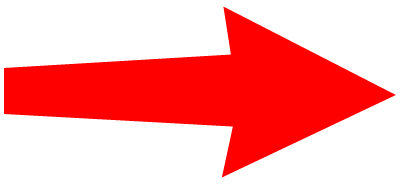
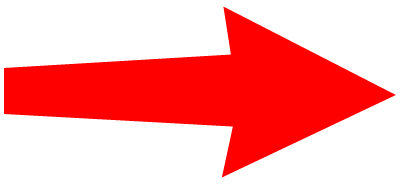
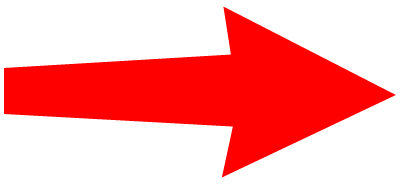
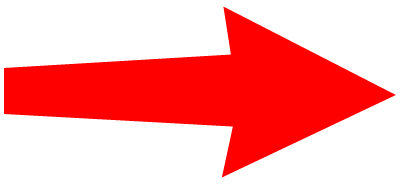
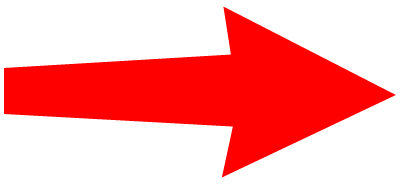
Host code



WCF Service Library

Contract

Implementation



*Hình: Hệ thống gồm 3 phần chính: Client, Service Host và WCF Service Library.*

* Client: là ứng dụng phía người sử dụng, có thể là windows application hoặc web applicaion.
* Service Host: quản lý cấu hình host.
* WCF Service Library: là thư viện cung cấp các chức năng và xử lý dữ liệu

## Nhóm chức năng hỗ trợ người dùng

### CheckLogin

**Mô tả chức năng**

Cho phép đăng nhập hệ thống, yêu cầu bắt buộc để thực hiện các tác vụ kế tiếp.

[OperationContract]

 bool CheckLogin(string username, string password, int mode);

**Đối tượng sử dụng**

Sinh viên, người thân sinh viên, giảng viên

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| username | char(6) | Username là mã số sinh viên, mã giảng viên, bắt buộc đủ 6 ký tự |
| password | varchar(50) | Mật khẩu để đăng nhập, tối thiểu 6 ký tự, tối đa 50 ký tự, không chấp nhận chuỗi unicode |
| mode | int | Phân biệt người dùng đang đăng nhập hệ thống. |

**Output**

**bool** - trả về true hoặc false, tương ứng đăng nhập thành công hay thất bại

### ChangePassword

**a. Mô tả chức năng**

Cho phép người dùng thay đổi mật khẩu đăng nhập.

[[OperationContract]

bool ChangePassword(string username, string old\_password, string new\_password, int mode);

**b. Đối tượng sử dụng**

Sinh viên, người thân sinh viên, giảng viên

**c. Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| username | char(6) | Username là mã số sinh viên, mã giảng viên, bắt buộc đủ 6 ký tự |
| old\_password | varchar(50) | Mật khẩu đăng nhập đang sử dụng |
| new\_password | varchar(50) | Mật khẩu đăng nhập mới |
| mode | int | Phân biệt người dùng đang đăng nhập hệ thống. |

**d. Output**

bool - trả về true hoặc false, tương ứng việc thay đổi mật khẩu thành công hay thất bại.

### Search\_Subject

**Mô tả**

Hỗ trợ người dùng tìm kiếm môn đã học trước đây theo Mã môn học hoặc tên môn học, và xem kết quả môn học đó.

[OperationContract]

Subject[] Search\_Subject(string keyword);

**Đối tượng sử dụng**

Sinh viên, Người nhà, Giảng viên

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| keyword | string | Từ khóa tìm kiếm, gồm mã môn học hoặc tên môn học |

**Output**

Subject[] - trả vể mảng các phần tử Subject → danh sách môn học

**Data Contract:** Subject

### Send\_Feedback

**Mô tả**

Cho phép người dùng gửi phản hồi về server.

[OperationContract(IsOneWay = true)]

void Send\_Feedback(string username, string subject, string content);

**Đối tượng sử dụng**

Sinh viên, Người nhà, Giảng viên

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| username | char(6) | Username là mã số sinh viên, mã giảng viên, bắt buộc đủ 6 ký tự |
| subject | nvarchar(50) | Tiêu đề |
| content | nvarchar(500) | Nội dung phản hồi |

**Output**

Sử dụng phương thức IsOneWay cho việc gửi phàn hồi về server.

## Nhóm contracts quản lý thông tin cá nhân

### Get\_Info\_Student

**Mô tả**

Cung cấp chi tiết thông tin cá nhân của sinh viên theo mã số sinh viên. Khi người dùng muốn kiểm tra thông tin của sinh viên đó.

[OperationContract]

Student Get\_Info\_Stu(string code\_student);

**Đối tượng sử dụng**

Những đối tượng cần xem thông tin của sinh viên bao gồm ‘Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Chuỗi MSSV |

**Output**

Object “Student” với dữ liệu tương ứng với mã số sinh viên (code\_student) truyền vào

**Data Contract:** Student

### Update\_Info\_Student ()

**Mô tả chức năng**

Cung cấp chi tiết thông tin cá nhân của sinh viên

[OperationContract]

[FaultContract(typeof(InfoFault[]))]

bool Update\_Info\_Stu(Student Stu);

**Đối tượng sử dụng**

Những đối tượng cần xem thông tin của sinh viên bao gồm ‘Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| student\_item | Student | Data Contract |

**Output**

bool - trả về true hoặc false, tương ứng cập nhật dữ liệu thành công hay thất bại.

**Fault Contract**

Hệ thống trả về lỗi “InfoFault” nếu thông tin gửi lên gây lỗi trong các trường hợp:

- Thiếu các dữ liệu bắt buộc

- Truyền sai định dạng ngày tháng, kiểu số, email.

### Get\_Info\_Lec

**a) Mô tả chức năng**

Cung cấp chi tiết thông tin cá nhân của giảng viên

[OperationContract]

Lecturer Get\_Info\_Lec(string code\_lecturer);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:** (không cần tham số)

**d) Output**

Lecturer - trả vể đối tượng Lecturer, thông tin giảng viên

### Update\_Info\_Lec

**a) Mô tả chức năng**

Cho phép thay đổi một số thông tin cá nhân.

[OperationContract]

[FaultContract(typeof(InfoFault[]))]

bool Update\_Info\_Lec(Lecturer lec);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:** [DataContract] Lecturer

**d) Output**

bool - trả về true hoặc false, tương ứng thay đổi thông tin cá nhân thành công hay thất bại.

1. **Fault Contract**

Hệ thống trả về lỗi “InfoFault” nếu thông tin gửi lên gây lỗi trong các trường hợp:

- Thiếu các dữ liệu bắt buộc

- Truyền sai định dạng ngày tháng, kiểu số, email.

## Nhóm contracts cung cấp điểm sinh viên

### Get\_Mark (Overloading, Name = “Mark\_Subject\_Single”)

**Mô tả chức năng**

Cung cấp điểm của 1 môn học của sinh viên ứng với mã sinh viên và mã môn học truyền lên.

[OperationContract(Name = "Info\_Stu\_Sem\_Only")]

Student\_Semester Info\_Semester(string code\_student, string semester);

**Đối tượng sử dụng**

Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc |
| code\_subject | char | Mã môn học, bắt buộc |

**Output**

Mark[] - trả vể mảng các phần tử Mark, một môn học có thể học nhiều học kỳ (học lại, học cải thiện…)

**Data Contract:** Mark

### Get\_Mark (Overloading, Name = “Mark\_Subject\_All”)

**Mô tả chức năng**

Cung cấp bảng điểm mới nhất của của sinh viên (qua tất cả học kì)

[OperationContract(Name = "Info\_Stu\_Sem\_All")]

Student\_Semester[] Info\_Semester(string code\_student);

**Đối tượng sử dụng**

Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc |

**Output**

Mark[] - trả vể mảng các phần tử Mark, toàn bộ kết quả của tất cả các môn đã học.

### Info\_Semester (Overloading, Name = “Info\_Stu\_Sem\_Only”)

**Mô tả chức năng**

Cung cấp thông tin kết quả của một học kỳ cụ thể gồm có: số tín chỉ đã đăng ký, tín chỉ đạt được, và trung bình của học kỳ đó

[OperationContract(Name = "Info\_Stu\_Sem\_Only")]

Student\_Semester Info\_Semester(string code\_student, string semester);

**Đối tượng sử dụng**

Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc |
| semester | char | Mã học kì |

**Output**

Student\_Semester – thông tin kết quả của học kỳ

**Datacontract:** Student\_Semester

### Info\_Semester (Overloading, Name = “Info\_Stu\_Sem\_All”)

**Mô tả chức năng**

Cung cấp thông tin kết quả của tất cả các học kỳ đã học gồm có: số tín chỉ đã đăng ký, tín chỉ đạt được, và trung bình của học kỳ đó

[OperationContract(Name = "Info\_Stu\_Sem\_All")]

Student\_Semester[] Info\_Semester(string code\_student);

**Đối tượng sử dụng**

Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc |

**Output**

Student\_Semester[] – mảng các phần tử Student\_Semester

**Datacontract:** Student\_Semester

### Get\_Avg\_Semester

**Mô tả chức năng**

Cung cấp thông tin điểm trung bình của một học kỳ cụ thể.

[OperationContract]

float Get\_Avg\_Semester (string code\_student, string semester);

**Đối tượng sử dụng**

Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc |
| semester | char | Mã học kì |

**Output**

Float

### Get\_Avg\_Cumulative

**Mô tả chức năng**

Cung cấp thông tin điểm trung bình tích lũy của sinh viên.

[OperationContract]

float Get\_Avg\_Cumulative (string code\_student);

**Đối tượng sử dụng**

Sinh viên, người thân, giảng viên, phòng đào tạo’

**Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc |

**Output**

Float

## Nhóm contracts giành cho giảng viên

### Lec\_Sub\_Single

**a) Mô tả chức năng**

Cung cấp danh sách các môn học đã giảng dạy trong một học kỳ nào đó.

[OperationContract(Name = "Lec\_Sub\_Single")]

Subject[] Lec\_Sub(string code\_lecturer, string semester);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên, bắt buộc |
| semester | char | Mã học kì |

**d) Output**

Subject[] - trả vể mảng các phần tử Subject → danh sách môn học

### Lec\_Sub\_All

**a) Mô tả chức năng**

Cung cấp danh sách các môn học đã giảng dạy trong tất cả các học kỳ.

[OperationContract(Name = "Lec\_Sub\_All")]

Subject[] Lec\_Sub(string code\_lecturer);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên, bắt buộc |

**d) Output**

Subject[] - trả vể mảng các phần tử Subject → danh sách môn học

### List\_Class

**a) Mô tả chức năng**

Cung cấp danh sách một lớp môn học do giảng viên dạy trong một học kỳ cụ thể, bao gồm cả bảng điểm học tập.

[OperationContract]

List\_Stu\_Class[] List\_Class(string code\_lecturer, string code\_subject, string semester);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên, bắt buộc |
| code\_subject | varchar(20) | Mã môn học |
| semester | char | Mã học kì |

**d) Output**

List\_Stu\_Class[] - trả vể mảng các phần tử List\_Stu\_Class → danh sách sinh viên lớp môn học, và bảng điểm

### UploadResource

**a) Mô tả chức năng**

Cho phép giảng viên upload bảng điểm của lớp môn học do mình giảng dạy về server cho phòng đào tạo nhập điểm vào hệ thống.

[OperationContract]

bool UploadResource(string resName, byte[] data);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| resName | varchar(50) | Tên file upload |
| data | byte[] | Gói dữ liệu dạng byte để truyền tải về server. |

**d) Output**

bool – true: upload thành công, false: upload thất bại

### DownloadResource

**a) Mô tả chức năng**

Cho phép giảng viên download bảng điểm của lớp môn học do mình giảng dạy.

[OperationContract]

byte[] DownloadResource(string resName);

**b) Đối tượng sử dụng**

Giảng viên

**c) Input:**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| resName | varchar(50) | Tên file upload |

**d) Output**

byte[] – trả về gói dữ liệu dạng byte

## Nhóm contracts cung cấp thông tin chung

### List\_Semester

**a) Mô tả chức năng**

Cung cấp danh sách học kỳ phục vụ cho client xây dựng ứng dụng, vì danh sách học kỳ có thể thêm xóa, sửa bởi người quản lý dịch vụ.

[OperationContract]

string[] List\_Semester();

**b) Đối tượng sử dụng**

Sinh viên, người thân sinh viên, giảng viên

**c) Input:** không cần tham số

**d) Output**

string[] - trả vể mảng các phần tử string tên học kỳ → danh sách học kỳ

### List\_Subject

**a) Mô tả chức năng**

Cung cấp danh sách môn học phục vụ cho client xây dựng ứng dụng, vì danh sách môn học có thể thêm, xóa, sửa bởi người quản lý dịch vụ.

[OperationContract]

Subject[] List\_Subject();

**b) Đối tượng sử dụng**

Sinh viên, người thân sinh viên, giảng viên

**c) Input: (**Không cần tham số)

**d) Output**

Subject[] - trả vể mảng các phần tử Subject → danh sách môn học

**e) DataContract:** Subject

## Nhóm contracts cung cấp dữ liệu thống kê

### Info\_Statistic\_Stu\_General

**a) Mô tả chức năng**

Thống kê thông tin tổng quát sinh viên

[OperationContract]

Statistic\_Student Info\_Statistic\_Stu\_General(string code\_student);

**b) Đối tượng sử dụng**

Sinh viên, Người nhà sinh viên

**c) Input**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_student | char(6) | Mã sinh viên, bắt buộc đủ 6 ký tự |

**d) Output**

Statistic\_Student - trả về thông tin thống kê

**e) DataContract:** Statistic\_Student

### Get\_Min\_Mark

**Mô tả chức năng**

Thống kê theo điểm thi thấp nhất của 1 môn học. Cung cấp cho Client 1 điểm thi thấp nhất của môn học trong học kì.

[OperationContract]

float Get\_Min\_Mark(string code\_lecturer, string code\_subject, string semester);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |

**Output**

Float – Trả về 1 giá trị kiểu float là điểm của một sinh viên trong môn học.

### Get\_Max\_Mark

**Mô tả chức năng**

Thống kê theo điểm thi cao nhất của 1 môn học. Cung cấp cho Client 1 điểm thi cao nhất của môn học trong học kì.

[OperationContract]

float Get\_Max\_Mark(string code\_lecturer, string code\_subject, string semester);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |

**Output**

Float – Trả về 1 giá trị kiểu float là điểm của một sinh viên trong môn học.

### Get\_Avg\_Mark

**Mô tả chức năng**

Thống kê theo điểm thi trung bình của 1 môn học. Cung cấp cho Client 1 điểm thi trung bình của môn học trong học kì.

[OperationContract]

float Get\_Avg\_Mark(string code\_lecturer, string code\_subject, string semester);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |

**Output**

Float – Trả về 1 giá trị kiểu float là điểm của một sinh viên trong môn học.

### Class\_Stu\_Total.

**Mô tả chức năng**

Thống kê sỉ số lớp môn học. Cung cấp cho Client số lượng sinh viên tham gia lớp môn học.

[OperationContract]

int Class\_Stu\_Total(string code\_lecturer, string code\_subject, string semester);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |

**Output**

Int – Số lượng sinh viên tham gia học môn học này.

### Statistic\_Mark\_Equal

**Mô tả chức năng**

Cung cấp cho Client số lượng điểm của một môn học bằng với số điểm Client mong muốn. Thống kê điểm của một môn học (Chính xác với điểm truyền vào)

[OperationContract]

int Statistic\_Mark\_Equal(string code\_lecturer, string code\_subject, string semester, float mark);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì, điểm điều kiện.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |
| mark | float | Điểm môn học |

**Output**

Int – Số lượng điểm trong lớp môn học bằng với điểm điều kiện truyền vào.

### Statistic\_Mark\_Distance

**Mô tả chức năng**

Cung cấp cho Client số lượng điểm của một môn học nằm trong khoảng điểm mà Client mong muốn. Thống kê điểm của một môn học (Nằm trong khoảng điểm truyền vào)

[OperationContract]

int Statistic\_Mark\_Distance(string code\_lecturer, string code\_subject, string semester, float from, float to);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì, điểm điều kiện bắt đầu, điểm điều kiện kết thúc.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |
| from | float | Điểm môn học bắt đầu |
| to | float | Điểm môn học kết thúc |

**Output**

Int – Số lượng điểm trong lớp môn học nằm trong khoảng điểm điều kiện truyền vào.

### Statistic\_Mark\_Less

**Mô tả chức năng**

Cung cấp cho Client số lượng điểm của một môn học nhỏ hơn số điểm mà Client mong muốn. Thống kê điểm của một môn học (Nhỏ hơn điểm truyền vào)

[OperationContract]

int Statistic\_Mark\_Less(string code\_lecturer, string code\_subject, string semester, float mark);

**Đối tượng sử dụng**

Giảng viên.

**Input**

Mã giảng viên, mã môn học, tên học kì, điểm điều kiện.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |
| mark | float | Điểm môn học |

**Output**

Int – Số lượng điểm trong lớp môn học nhỏ hơn điểm điều kiện truyền vào.

### Statistic\_Mark\_Greater

**Mô tả chức năng**

Cung cấp cho Client số lượng điểm của một môn học lớn hơn số điểm mà Client mong muốn. Thống kê điểm của một môn học (Lớn hơn điểm truyền vào)

[OperationContract]

int Statistic\_Mark\_Greater(string code\_lecturer, string code\_subject, string semester, float mark);

**Đối tượng sử dụng**

Giảng viên

**Input**

Mã giảng viên, mã môn học, tên học kì, điểm điều kiện.

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| code\_lecturer | char(6) | Mã giảng viên bắt buộc phải 6 kí tự |
| code\_subject | varchar(20) | Mã môn học. |
| semester | varchar(20) | Tên học kì |
| mark | float | Điểm môn học |

**Output**

Int – Số lượng điểm trong lớp môn học lớn hơn điểm điều kiện truyền vào.

# Datacontract

[DataContract]

public class Subject

{

private string semester;

private string code\_sub;

private string name\_sub;

private int soTC;

private int sotiet;

[DataMember]

public string Semester

{

get { return semester; }

set { semester = value; }

}

[DataMember]

public string Code\_sub

{

get { return code\_sub; }

set { code\_sub = value; }

}

[DataMember]

public string Name\_sub

{

get { return name\_sub; }

set { name\_sub = value; }

}

[DataMember]

public int SoTC

{

get { return soTC; }

set { soTC = value; }

}

[DataMember]

public int Sotiet

{

get { return sotiet; }

set { sotiet = value; }

}

}

[DataContract]

public class List\_Stu\_Class

{

private string mssv;

private string hoten;

private float diemKT;

private float diemThi;

private float TB\_h10;

private string TB\_hchu;

private float TB\_h4;

private bool tinhtrang;

private string loai;

[DataMember]

public string Mssv

{

get { return mssv; }

set { mssv = value; }

}

[DataMember]

public string Hoten

{

get { return hoten; }

set { hoten = value; }

}

[DataMember]

public float DiemKT

{

get { return diemKT; }

set { diemKT = value; }

}

[DataMember]

public float DiemThi

{

get { return diemThi; }

set { diemThi = value; }

}

[DataMember]

public float DTB\_h10

{

get { return TB\_h10; }

set { TB\_h10 = value; }

}

[DataMember]

public string DTB\_hchu

{

get { return TB\_hchu; }

set { TB\_hchu = value; }

}

[DataMember]

public float DTB\_h4

{

get { return TB\_h4; }

set { TB\_h4 = value; }

}

[DataMember]

public bool Tinhtrang

{

get { return tinhtrang; }

set { tinhtrang = value; }

}

[DataMember]

public string Loai

{

get { return loai; }

set { loai = value; }

}

}

[DataContract]

public class Statistic\_Mark

{

private int total;

private int less\_than\_5;

private int distance\_5\_8;

private int distance\_8\_10;

private float min\_mark;

private float max\_mark;

private float avg\_mark;

[DataMember]

public int Total

{

get { return total; }

set { total = value; }

}

[DataMember]

public int Less\_than\_5

{

get { return less\_than\_5; }

set { less\_than\_5 = value; }

}

[DataMember]

public int Distance\_5\_8

{

get { return distance\_5\_8; }

set { distance\_5\_8 = value; }

}

[DataMember]

public int Distance\_8\_10

{

get { return distance\_8\_10; }

set { distance\_8\_10 = value; }

}

[DataMember]

public float Min\_mark

{

get { return min\_mark; }

set { min\_mark = value; }

}

[DataMember]

public float Max\_mark

{

get { return max\_mark; }

set { max\_mark = value; }

}

[DataMember]

public float Avg\_mark

{

get { return avg\_mark; }

set { avg\_mark = value; }

}

}

[DataContract]

public class Member

{

private string code;

private string fullname;

private string \_class;

[DataMember]

public string Code

{

get { return code; }

set { code = value; }

}

[DataMember]

public string FullName

{

get { return fullname; }

set { fullname = value; }

}

[DataMember]

public string \_Class

{

get { return \_class; }

set { \_class = value; }

}

public Member(string \_code, string \_fullname, string \_\_class)

{

this.code = \_code;

this.fullname = \_fullname;

this.\_Class = \_\_class;

}

}

//-------------------------------------------------------------------------------

[DataContract]

public class Student

{

private string mssv;

private string hoten;

private int gioitinh;

private string ngaysinh;

private string noisinh;

private string lop;

private string nganh;

private string khoa;

private string heDT;

private string khoahoc;

private int tongTC\_DT;

private string diachi;

private string dienthoai;

private string email;

private string dantoc;

private string tongiao;

private string quoctich;

private string hotencha;

private string nghenghiepcha;

private string hotenme;

private string nghenghiepme;

[DataMember]

public string Mssv

{

get { return mssv; }

set { mssv = value; }

}

[DataMember]

public string Hoten

{

get { return hoten; }

set { hoten = value; }

}

[DataMember]

public int Gioitinh

{

get { return gioitinh; }

set { gioitinh = value; }

}

[DataMember]

public string Ngaysinh

{

get { return ngaysinh; }

set { ngaysinh = value; }

}

[DataMember]

public string Noisinh

{

get { return noisinh; }

set { noisinh = value; }

}

[DataMember]

public string Lop

{

get { return lop; }

set { lop = value; }

}

[DataMember]

public string Nganh

{

get { return nganh; }

set { nganh = value; }

}

[DataMember]

public string Khoa

{

get { return khoa; }

set { khoa = value; }

}

[DataMember]

public string HeDT

{

get { return heDT; }

set { heDT = value; }

}

[DataMember]

public string Khoahoc

{

get { return khoahoc; }

set { khoahoc = value; }

}

[DataMember]

public int TongTC

{

get { return tongTC\_DT; }

set { tongTC\_DT = value; }

}

[DataMember]

public string Diachi

{

get { return diachi; }

set { diachi = value; }

}

[DataMember]

public string Dienthoai

{

get { return dienthoai; }

set { dienthoai = value; }

}

[DataMember]

public string Email

{

get { return email; }

set { email = value; }

}

[DataMember]

public string Dantoc

{

get { return dantoc; }

set { dantoc = value; }

}

[DataMember]

public string Tongiao

{

get { return tongiao; }

set { tongiao = value; }

}

[DataMember]

public string Quoctich

{

get { return quoctich; }

set { quoctich = value; }

}

[DataMember]

public string Hotencha

{

get { return hotencha; }

set { hotencha = value; }

}

[DataMember]

public string Nghenghiepcha

{

get { return nghenghiepcha; }

set { nghenghiepcha = value; }

}

[DataMember]

public string Hotenme

{

get { return hotenme; }

set { hotenme = value; }

}

[DataMember]

public string Nghenghiepme

{

get { return nghenghiepme; }

set { nghenghiepme = value; }

}

}

//-----------------------------------------------------------------

[DataContract]

public class Mark

{

private string maMH;

private string tenMH;

private int soTC;

private int soTiet;

private float diemKT;

private float diemThi;

private float TB\_h10;

private string TB\_hchu;

private float TB\_h4;

private bool tinhtrang;

private string loai;

private string hocky;

private string giangvien;

[DataMember]

public int SoTC

{

get { return soTC; }

set { soTC = value; }

}

[DataMember]

public int SoTiet

{

get { return soTiet; }

set { soTiet = value; }

}

[DataMember]

public string MaMH

{

get { return maMH; }

set { maMH = value; }

}

[DataMember]

public string TenMH

{

get { return tenMH; }

set { tenMH = value; }

}

[DataMember]

public float DiemKT

{

get { return diemKT; }

set { diemKT = value; }

}

[DataMember]

public float DiemThi

{

get { return diemThi; }

set { diemThi = value; }

}

[DataMember]

public float DTB\_h10

{

get { return TB\_h10; }

set { TB\_h10 = value; }

}

[DataMember]

public string DTB\_hchu

{

get { return TB\_hchu; }

set { TB\_hchu = value; }

}

[DataMember]

public float DTB\_h4

{

get { return TB\_h4; }

set { TB\_h4 = value; }

}

[DataMember]

public bool Tinhtrang

{

get { return tinhtrang; }

set { tinhtrang = value; }

}

[DataMember]

public string Loai

{

get { return loai; }

set { loai = value; }

}

[DataMember]

public string Hocky

{

get { return hocky; }

set { hocky = value; }

}

[DataMember]

public string Giangvien

{

get { return giangvien; }

set { giangvien = value; }

}

}

//------------------------------------------------------

[DataContract]

public class Student\_Semester

{

private string mssv;

private string hocky;

private float tbhk;

private int tc\_dk\_hk;

private int tc\_dat\_hk;

[DataMember]

public string Mssv

{

get { return mssv; }

set { mssv = value; }

}

[DataMember]

public string Hocky

{

get { return hocky; }

set { hocky = value; }

}

[DataMember]

public float TBHK

{

get { return tbhk; }

set { tbhk = value; }

}

[DataMember]

public int TC\_dat\_HK

{

get { return tc\_dat\_hk; }

set { tc\_dat\_hk = value; }

}

[DataMember]

public int TC\_dk\_HK

{

get { return tc\_dk\_hk; }

set { tc\_dk\_hk = value; }

}

}

//-------------------------------------------------------------------------------

[DataContract]

public class Statistic\_Student

{

private int tongMH;

private int tongMH\_no;

private int tongTC\_dk;

private int tongTC\_dat;

private float tb\_tichluy;

private int tong\_xuatsac;

private int tong\_gioi;

private int tong\_kha;

private int tong\_trungbinh;

private int tong\_kem;

public Statistic\_Student()

{

this.tongMH = 0;

this.tongMH\_no = 0;

this.tongTC\_dk = 0;

this.tongTC\_dat = 0;

this.tb\_tichluy = 0;

this.tong\_gioi = 0;

this.tong\_kha = 0;

this.tong\_trungbinh = 0;

this.tong\_kem = 0;

}

[DataMember]

public int TongMH

{

get { return tongMH; }

set { tongMH = value; }

}

[DataMember]

public int TongMH\_no

{

get { return tongMH\_no; }

set { tongMH\_no = value; }

}

[DataMember]

public int TongTC\_dk

{

get { return tongTC\_dk; }

set { tongTC\_dk = value; }

}

[DataMember]

public int TongTC\_dat

{

get { return tongTC\_dat; }

set { tongTC\_dat = value; }

}

[DataMember]

public float TB\_tichluy

{

get { return tb\_tichluy; }

set { tb\_tichluy = value; }

}

[DataMember]

public int Tong\_xuatsac

{

get { return tong\_xuatsac; }

set { tong\_xuatsac = value; }

}

[DataMember]

public int Tong\_gioi

{

get { return tong\_gioi; }

set { tong\_gioi = value; }

}

[DataMember]

public int Tong\_kha

{

get { return tong\_kha; }

set { tong\_kha = value; }

}

[DataMember]

public int Tong\_trungbinh

{

get { return tong\_trungbinh; }

set { tong\_trungbinh = value; }

}

[DataMember]

public int Tong\_kem

{

get { return tong\_kem; }

set { tong\_kem = value; }

}

}

//------------------------------------------------------

[DataContract]

public class Feedback

{

private string thoigan;

private string tieude;

private string noidung;

private string traloi;

private bool trangthai;

[DataMember]

public string Thoigian

{

get { return thoigan; }

set { thoigan = value; }

}

[DataMember]

public string Tieude

{

get { return tieude; }

set { tieude = value; }

}

[DataMember]

public string Noidung

{

get { return noidung; }

set { noidung = value; }

}

[DataMember]

public string Traloi

{

get { return traloi; }

set { traloi = value; }

}

[DataMember]

public bool Trangthai

{

get { return trangthai; }

set { trangthai = value; }

}

}

//-------------------------------------------------------------------------------

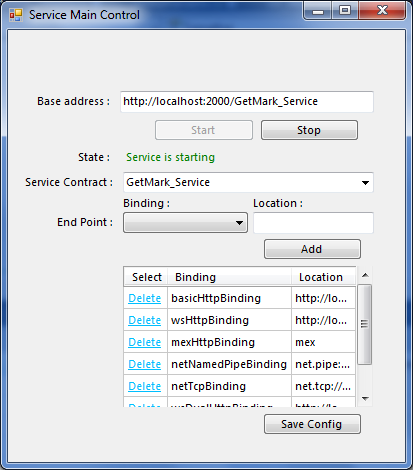
# Cơ sở dữ liệu



# Hiện thực chương trình

## Service host

Đảm nhận việc khởi chạy webservice, đồng thời cấu hình binding và endpoint. Người quản trị hệ thống có thể quản lý các địa chỉ cung cho người dùng, thêm xóa sửa các endpoint và lưu cấu hình vào file config.



*Hình 4.1.1 Giao diện Điều khiển Service*

## Client

### Đăng nhập, phân quyền

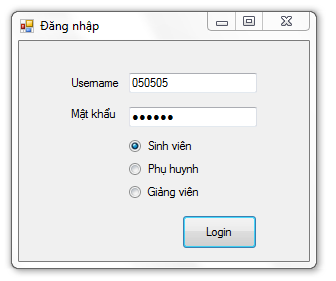
Đối tuợng người sử dụng bao gồm: Sinh viên, giảng viên, người nhà sinh viên.

Sinh viên và người nhà dung chung một account nhưng với 2 mật khẩu độc lập.

Người nhà đuợc quyền thay đổi mật khẩu của họ nhưng không được cập nhật profile của sinh viên.

Sinh viên được quyền thay đổi profile của mình.

Giảng viên có thể xem điểm của các sinh viên của lớp môn học. Có quyền tự cập nhật profile.

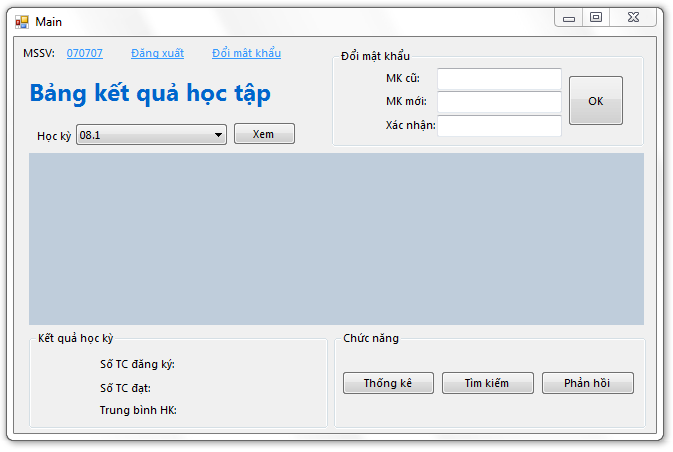


*Hình 4.2.1 Giao diện Login*

### Thay đổi mật khẩu

Sau khi đăng nhập thành công, trên giao diện chính, người dùng click “Đổi mật khẩu” → enable nhóm các control cho người người thao tác, người dùng nhập vào mật khẩu hiện tại, và mật khẩu mới cùng với xác nhận → Chọn “OK”:

* Gửi dữ liệu về server xử lý và trả lời
* Mật khẩu hiện tại không đúng → Thông báo



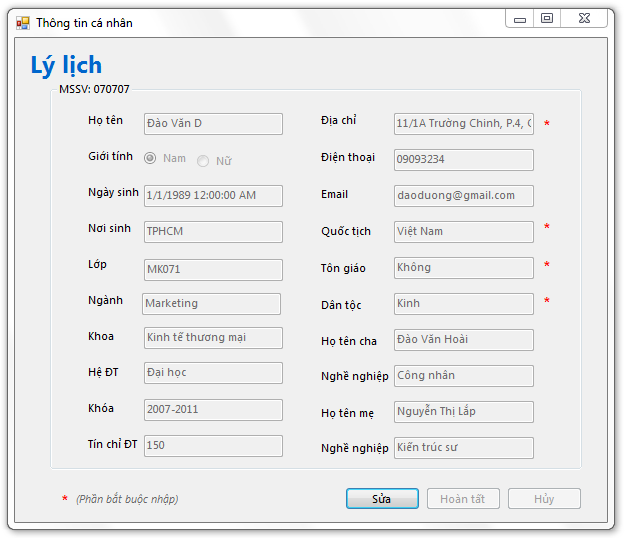
*Hình 4.2.2 Giao diện thay đổi mật khẩu(góc phải-trên)*

### Quản lý profile

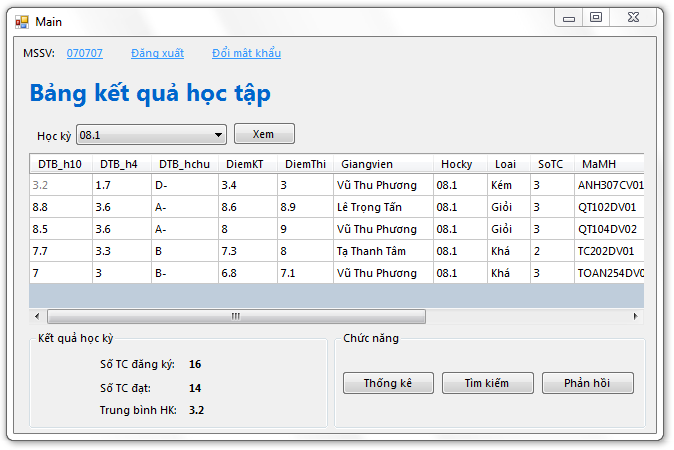
Sau khi đăng nhập, các user có quyền sử dụng chức năng này(bao gồm: sinh viên, giảng viên) sẽ chọn button Profile.

Từ giao diện ‘Profile’ người dung bấm button ‘Sửa’→ enable các field đuợc phép sửa (bao gồm: địa chỉ, điện thoại, email, quốc tịch, tôn giáo, dân tộc, tên cha, mẹ….) và disable các field không được phép(gồm: họ tên, mã số, giới tính, lớp, ngành, khóa, hệ đào tạo…)→ Chọn ‘Hoàn tất’→Kiểm tra dữ liệu:

* Nếu hợp lệ (các field yêu cầu bắt buộc:địa chỉ, quốc tịch, tôn giáo, dân tộc… và các field đúng dạng: điện thoại, email) → Gửi dữ liệu lên server, server thông báo kết quả
* Không hợp lệ → Thông báo yêu cầu nhập lại.

*Hình 4.2.3 Giao diện Profile*

### Xem kết quả

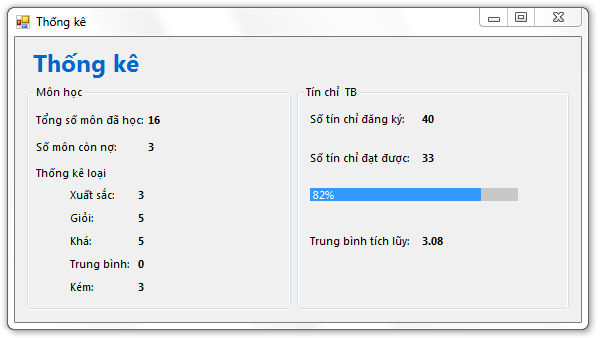


*Hình 4.2.4 Giao diện Bảng kết quả*

Sau khi đăng nhập, hệ thống mặc định chuyển đến giao diện hiển thị điểm sinh viên.

Người dùng chọn học kỳ cần xem trong danh sách đổ xuống → Bấm button ‘Xem’.

### Thống kê tiến trình học tập



*Hình 4.2.5 Giao diện thống kê*

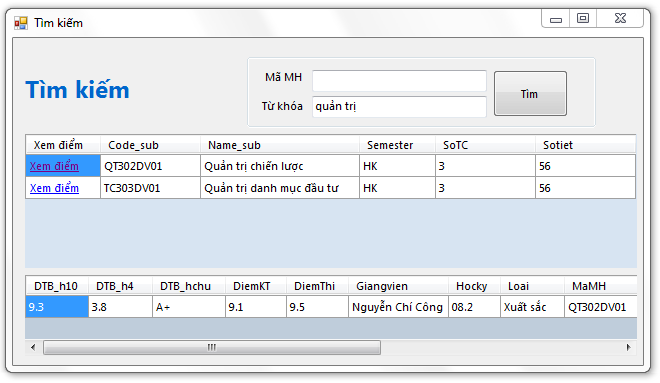
Sau khi đăng nhập, từ giao diện xem điểm sinh viên, người dùng chọn button ‘Thống kê’ → Giao diện ‘Thống kê’ hiển thị: số liệu thống kê của tổng số môn đã học, thống kê theo loại, số tín chỉ đạt đuợc trên tổng tín chỉ đăng kí.

### Tìm kiếm

Hỗ trợ người dùng (sinh viên, người nhà) tìm kiếm môn đã học theo mã môn học hoặc từ khóa tên môn học. Người dùng nhập mã môn học hoặc từ khó tên môn học trên giao diện, click “Tìm”

Gửi dữ liệu lên server và nhận kết quả trả về.

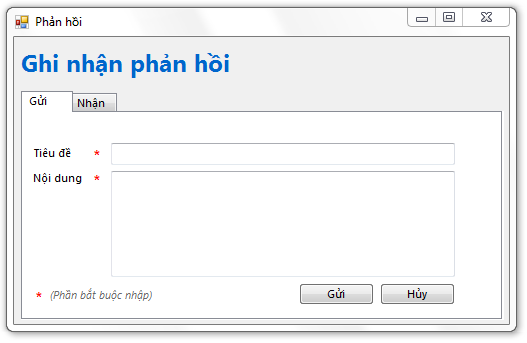
Danh sách tìm thấy hiện lên giao diện, người dùng có thể xem kết quả học tập của các môn.



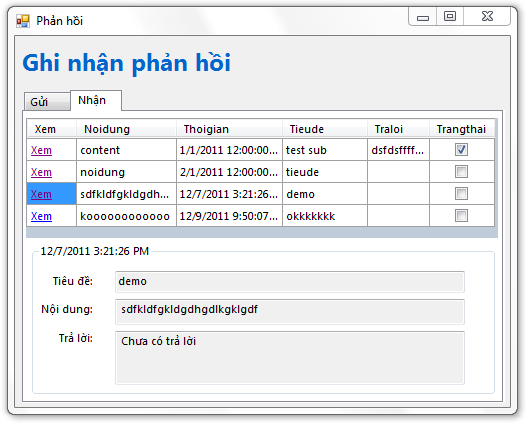
*Hình 4.2.6 Giao diện Tìm kiếm*

### Gửi phản hồi

Cho phép người dùng (sinh viên, phụ huynh, giảng viên) gửi phản hồi về người quản trị hệ thống về các vấn đề liên quan, bao gồm “Tiêu đề” và “Nội dung”, khi có trả lời từ người quản trị, người dùng có thê xem câu trả lời hay tin nhấn.



*Hình 4.2.7a Giao diện Gửi phản hồi*

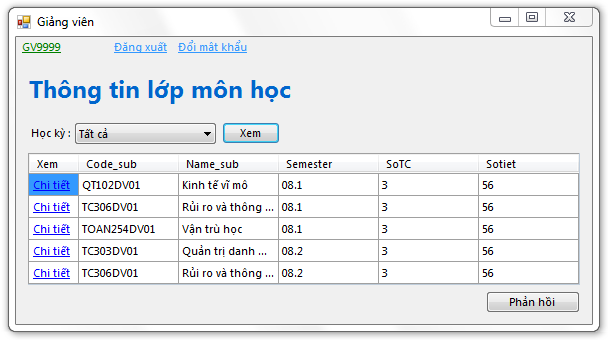


*Hình 4.2.7b Giao diện Nhận phản hồi*

### Thông tin lớp môn học

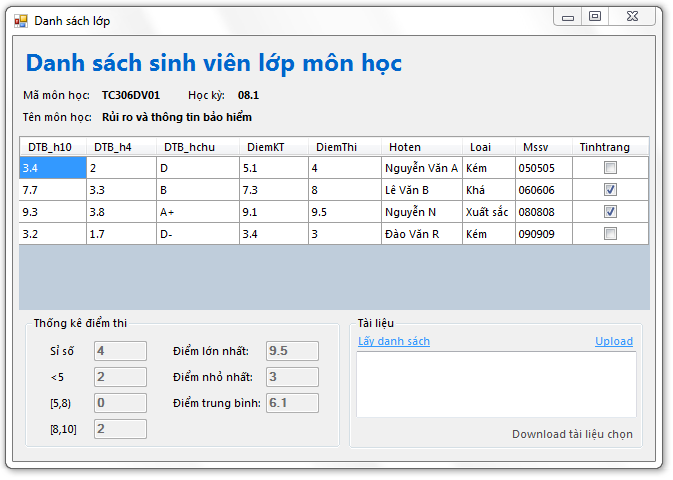
Cho phép giảng viên xem thông tin về lớp môn học do mình giảng dạy, gồm danh sách lớp, kết quả học tập, các thống kế về điểm thi

Từ giao diện, chọn học kỳ cần xem, danh sách các môn học do giảng viên dạy sẽ hiển thị, cho phép giảng viên xem thông tin lớp môn học.



*Hình 4.2.8a Giao diện Thông tin lớp môn học*

Click “Xem” trên mỗi dòng, form thông tin lớp môn học hiển thị



*Hình 4.2.8b Giao diện Danh sách SV - lớp MH*

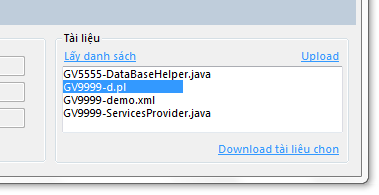
### Tải lên – Tải về tài liệu

Cho phép giảng viên upload 1 tài liệu lên server để người quản trị dùng làm dữ liệu nhập liệu:

Từ giao diện, chọn ”Upload” →Chọn file cần upload→OK

Cho phép giảng viên download tài liệu của mình đã upload:

Từ giao diện, chọn ”Lấy danh sách” →Chọn file cần download trong danh sách trả về→”Download tài liệu này”



*Hình 4.2.9 Giao diện Download-Upload tài liệu*

1. KẾT LUẬN

# Ưu điểm

* Hỗ trợ đa dạng các binding: basicHttpBinding, wsHttpBinding, wsDualHttpBinding, netTcpBinding, netNamedPipeBinding,
* 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 điểm môn học
* Hổ trợ tốt việc thống kê

# Khuyết điểm

* Xử lý các ngoại lệ chưa chặt chẽ.
* Chức năng client chưa phong phú
* Các quy tắc nghiệp vụ chưa hoàn chỉnh
* Tốc độ đáp ứng của service chưa cao

# Hướng phát triển

* Nâng cao tốc độ xử lý.
* Kiểm tra chặc chẽ các quy tắc nghiệp vụ, đảm bảo tính đúng đắn của dữ liệu.
* 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 dữ liệu.

1. Cài đặt và sử dụng

# Cài đặt

* Attach database vào hệ quản trị cơ sở dữ liệu SQL Server bằng file GetMark\_Service.mdf và GetMark\_Service\_log.ldf

(folder …\GroupOneProject\Database).

* Kiểm tra chuỗi kết nối tại file connect.cs

string strcon = @"Data Source=.\SQLEXPRESS;Initial Catalog=GetMark\_Service; Integrated Security=True";

# Sử dụng

* Build “ServiceHost\_Form” → success → chạy file ServiceHost\_Form.exe trong folder …\GroupOneProject\ServiceHost\_Form\bin\Debug
* Cấu hình Host theo ý muốn và lưu vào file config (lưu ý: client cũng được cấu hình proxy trong file Proxy.cs với các loại binding phổ biến, và các chức năng gửi request đến server đang sử dụng binding WSHttpBingding)
* Start service host
* Start Debugging chương trình Client
* Đăng nhập:
* Sinh viên:

Username: 070707

Password: 070707

* Phụ huynh

Username: 070707

Password: 123456

* Giảng viên

Username: GV9999

Passwprd: GV9999