|  |
| --- |
| **Ministry of education and training** |
| Software Design Description |
| FPT Management Trainning Process |
|  |
| |  |  | | --- | --- | | **FMTP Team** | | | **Group Members** | Nguyễn Huỳnh Long – 00571 – LongNH00571  Trần Minh Trực – 00570 – TrucTM00570  Phạm Duy Khương – 00576 – KhuongPD00576  Nguyễn Anh Tuấn – 00751 – TuanNA00751 | | **Supervisor** | Instructor Kiều Trọng Khánh | | **Ext Supervisor** | N/A | | **Capstone Project code** | FMTP | |
| **Ho Chi Minh, 14th Feb, 2012** |

**Record of Changes**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Change Item** | **Description** | **By** | **Version** |
| 18/02/2012 | All | Create the document | LongNH | 0.1 |
| 23/02/2012 | Detail Design Description | Add | LongNH,  KhuongPD,  TrucTM,  TuanNA | 0.2 |
| 23/02/2012 | Database Design | Modify | TrucTM | 1.0 |

# Table of Contents

[Table of Contents 2](#_Toc317782223)

[I. Design Overview 4](#_Toc317782224)

[II. System Architectural Design 5](#_Toc317782225)

[1. Choice of System Architecture: Three Tiers Architecture 5](#_Toc317782226)

[2. Discussion of Alternative Designs 5](#_Toc317782227)

[3. Description of System Interface 5](#_Toc317782228)

[III. Web Application Component Diagram 6](#_Toc317782229)

[1. FMTP.Common Component 6](#_Toc317782230)

[2. FMTP.Resources Component 7](#_Toc317782231)

[3. FMTP.DataAccess Layer Component 7](#_Toc317782232)

[4. FMTP.Business Layer Component 7](#_Toc317782233)

[5. FMTP.Web Component 8](#_Toc317782234)

[IV. Detailed Design Description 8](#_Toc317782235)

[1. <User> Login 8](#_Toc317782236)

[2. <User>Forgot password 11](#_Toc317782237)

[3. <Staff> Add Student Score 16](#_Toc317782238)

[4. <System> Grading 19](#_Toc317782239)

[5. <Student> View Private Information 20](#_Toc317782240)

[6. <Student, Staff> Edit Student’s Information 22](#_Toc317782241)

[7. <Student, Staff> Export All Collection Reports 24](#_Toc317782242)

[8. <Staff> Auto Suggest Semester 27](#_Toc317782243)

[9. <Staff> Auto Suggest Subject 32](#_Toc317782244)

[10. <Staff> Auto Suggest Class 36](#_Toc317782247)

[11. <Staff> View List Students and Grade by Filter Conditions 41](#_Toc317782249)

[12. <Staff> Export Collection of Reports 45](#_Toc317782252)

[13. <Staff> Auto Suggest Student Name 49](#_Toc317782253)

[14. <Staff> View List Subjects of Student 52](#_Toc317782254)

[15. <Staff> Edit Staff Personal Information 55](#_Toc317782255)

[16. <System> Send Report E-Mail for Student 58](#_Toc317782256)

[17. <Staff>Import Excel Data File 60](#_Toc317782257)

[V. Database design 66](#_Toc317782258)

[1. ERD 66](#_Toc317782259)

[2. Tables 67](#_Toc317782260)

[3. Table Detail 68](#_Toc317782261)

[VI. References 72](#_Toc317782262)

## Design Overview

This document describes the technical and UI design of the FMTP system. It includes the architectural design and the detailed design of common functions and business functions. It also includes the design of database model, and overall design of user interface.

The architectural design describes the overall architecture of the system, and the architecture of each main component and subsystem. It will describe the patterns being used, the role of each component and the role of the system in the working environment.

The detailed design describes static and dynamic structure for each component and function. It includes class diagrams, class explanations, and sequence diagrams of the main use cases. The detailed design uses notations of .NET framework 4.0 and C# 4.0 as they are the framework and language for developing the system.

The database design describes the relationship between entities, and details of each entity. It uses notations of SQL Server 2008 as it is the database server for developing the system.

The user interface design describes the layout of the system, and some design for the screens.

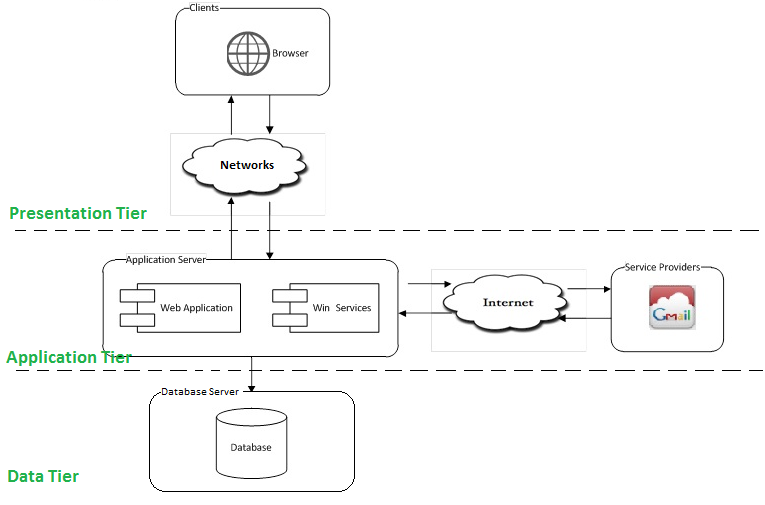
Overview:

* Section II: gives an overall description of the system architecture design. It gives what architecture is chosen and its model that is presented in graphic
* Section III: gives component diagrams that describe the connection and integration of the system.
* Section IV: gives the detail design description include class diagram, class explanation, sequence diagram... to details the application functions in details
* Section V: describe Graphic User Interface Screen with the layouts of system.
* Section VI: describe an ERD with many entity and database design.

## System Architectural Design

### Choice of System Architecture: Three Tiers Architecture

*(Three Tiers Architecture Diagram)*



### Discussion of Alternative Designs

### Description of System Interface

The system architecture will be based on the **three tiers** as the main architecture that emphasizes separation of system components to allow for a distributed system that is scalable portable and extendable. It also makes use of the Internet and common web protocols to deliver its services to the users. The main entities of the above diagram will be described as following:

• **Presentation Tier (Clients):** These include internet browsers running on personal computers, or some targeting mobile devices. The web browsers will interact with the Web Application in Application Server via normal protocols (HTTP, HTTPS…).

• **Application Server Tier:** This is the server of the system using the administrator object and some securities to manage such as maintaining, configuration and runs on IIS web server. It includes two main components: Web Application and Win Services.

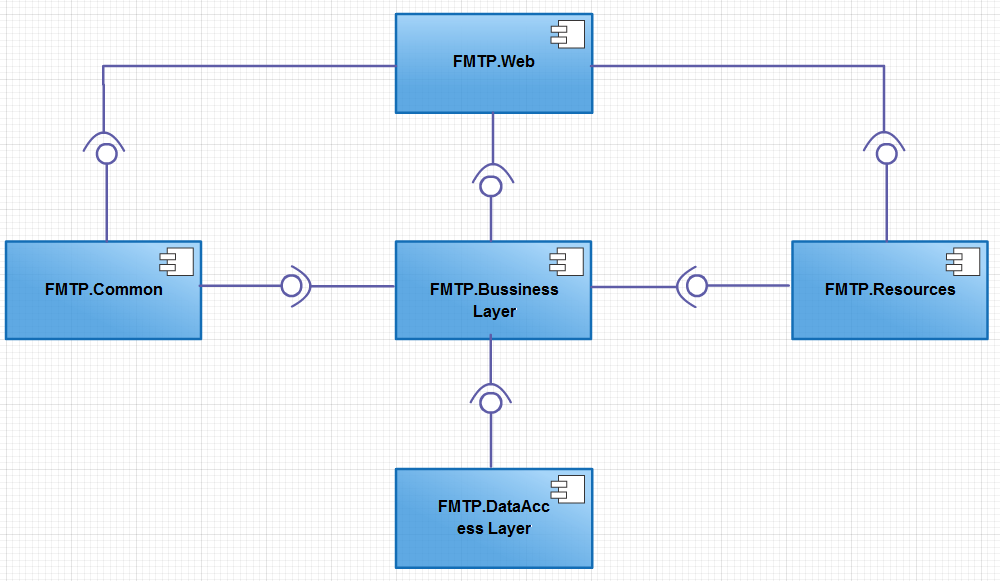
* **The** **Web Application** is responsible for creating dynamic web pages using ASP.Net 2.0, HTML, AJAX, CSS combined with JavaScript technologies. Besides that, Web Application also provides the functions of middleware which is deployed in the application server to do businesses logic tasks and access database tasks. This provides the flexibility in maintaining and portable, especially providing services in 24/7. So, it is developed using **Three Layers** architecture framework.
* **The** **Win Service** provides for auto e-mailing report. It is noted that this component will not be included in this project’s scope. However the system architecture must be extensible for adding such a component in the future.

• **Data Tier (Database):** the DBMS is used to manipulate the data and constraints to the application progress. This is the central data storage for the system. It may be running on a separate server to the Application Server, or in the same machine.

• **Service Providers:** these are third party service providers that the application may need services from. For example the application may need Google Mail services from Google for Win Services to send e-mail…

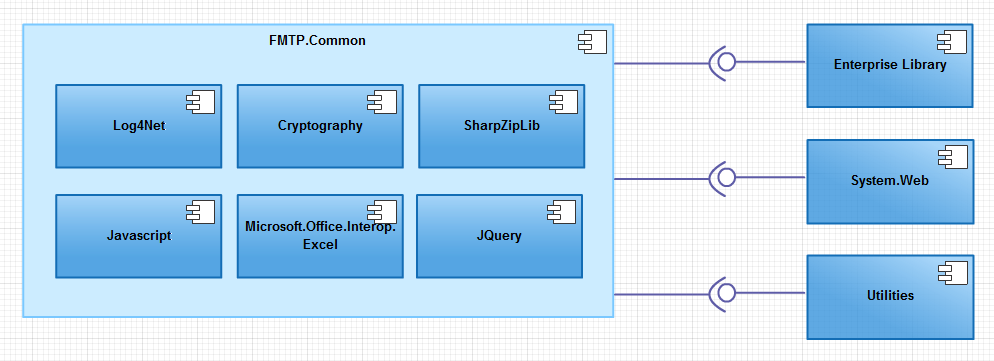
## Web Application Component Diagram

The architecture for the FMTP web application is **Three Layers Architecture** and will be depicted as the following diagram:



### FMTP.Common Component

This component contains common logic that can be used throughout the application. It includes for example mechanism for logging, caching, cryptography, and other utilities such as java script classes, css...



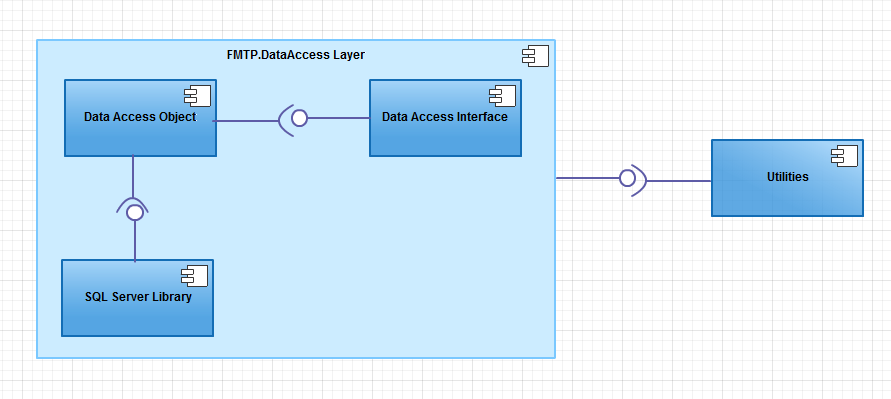
### FMTP.Resources Component

This component contains static resources for the system. It includes localized strings, messages, images, and report template...

### FMTP.DataAccess Layer Component

This component contains all data access objects which is responsible for accessing data in database system component and exchange data information with database for the system. It includes data access classes, data access interface classes…

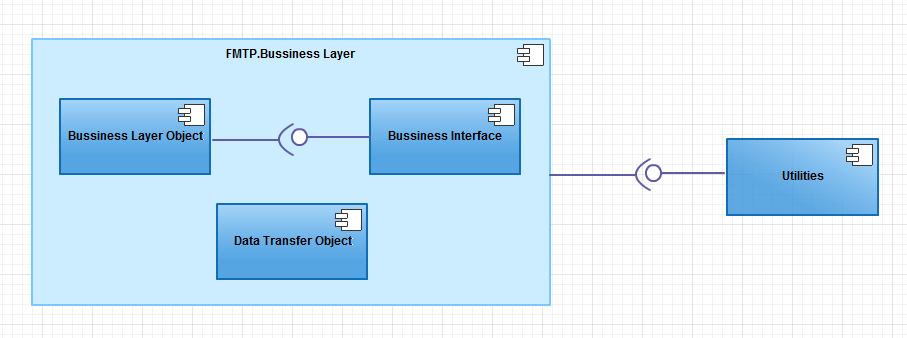
The aim and challenge for the design is that this component will be application type independent. That is this component is not tied only to web application but can be used for other .NET application types if necessary.



### FMTP.Business Layer Component

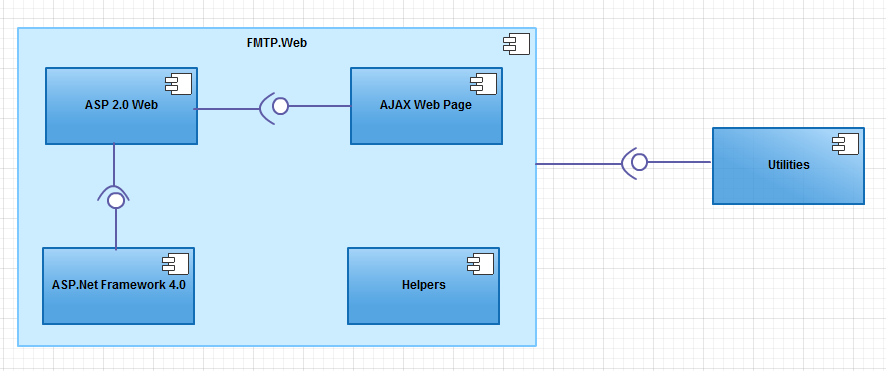
This component contains all business logic for the system. It includes business layer object classes, data transfer object classes, business interface class…

The aim and challenge for the design is that this component will be application type independent. That is this component is not tied only to web application but can be used for other .NET application types if necessary.



### FMTP.Web Component

This component is responsible for delivering the front end to the users and accepting their inputs. It will be developed using ASP.NET 2.0 and .Net framework 4.0. This Web Component is designed and implemented using the following attributes:

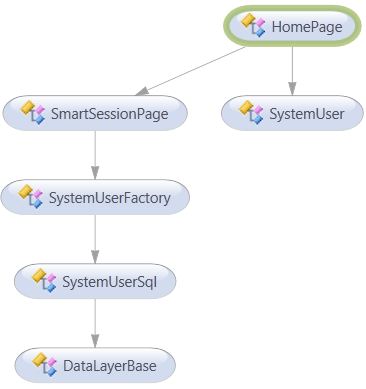
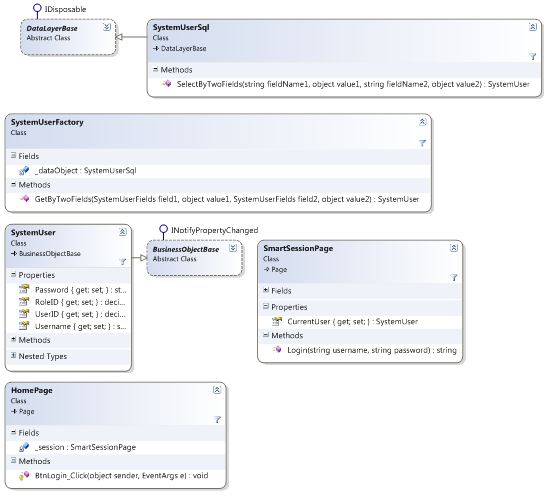


## Detailed Design Description

### <User> Login

#### Class Diagram

Fig\_1.1.1 Login Class Diagram



Fig\_1.1.2 Login Class DiagramDetail

#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which implement get a connection object to connect to database and inherits IDisposable Interface class to use method Dispose connection when doesn’t use.

##### **SystemUserSql Class (Data Access Layer)**

**Attributes: N/A**

**Methods:** SystemUser SelectByTwoFields(string fieldName1, object value1, string fieldName2, object value2): This method is used to call procedure for select object SystemUser based on two columns.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName1** | Semester | The name of column1 in DB |
| **value1** | Object | The looking value of fieldname1 |
| **fieldName2** | Semester | The name of column1 in DB |
| **value2** | Object | The looking value of fieldname2 |
| **Return** | SystemUser | SystemUser Object |

##### **SystemUserFactory Class (Business Layer)**

**Attributes: N/A**

**Methods:** SystemUser GetByTwoFields(SystemUser.SystemUserFields field1, object value1, SystemUser.SystemUserFields field2, object value2): This method is used to call method SelectByTwoFields from SystemUserSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **field1** | SystemUserFields | The name of column1 in DB |
| **value1** | Object | The looking value of field1 |
| **field1** | SystemUserFields | The name of column2 in DB |
| **value2** | Object | The looking value of field2 |
| **Return** | SystemUser | SystemUser Object |

##### **SystemUser Class (Business Layer)**

Data Transfer Object class, uses to store SystemUser’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **Password** | string | public | Password of login user |
| **RoleID** | int | public | Role of login user |
| **UserID** | decimal | public | Index in DB of this user |
| **Username** | string | public | Username of login user |

**Methods: N/A**

##### **BussinessObjectBase Class (Bussiness Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **SmartSessionPage Class (Common layer)**

This Class is used for store session of this application include some attributes and methods. This session describe attributes CurrentUser and method login.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **CurrentUser** | SystemUser | public | Object for storing session current user |

**Methods:** string Login(string username, string password). This method is used for checking validation of System User when login into system by call GetByTwoFields method from business layer. If login User has been validated, method will store this user into CurrentUser.

**Parameters and Return**

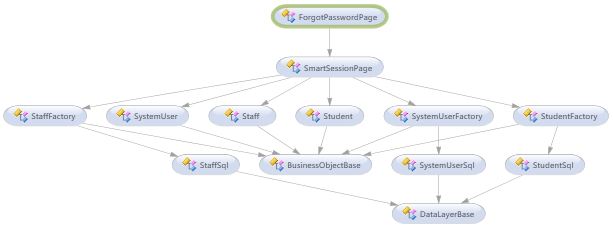
|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **username** | string | Value of username when users input value |
| **password** | string | Value of password when users input value |

##### **HomePage Class (Web Layer)**

This is the login page for user login into system. Method btnLogin\_Click will call method Login from SmartSessionPage for checking and redirect to the right page

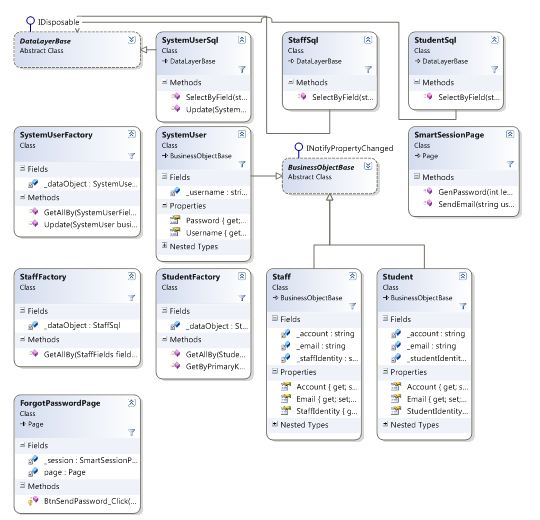
### <User>Forgot password

#### Class Diagram



Fig\_2.1.1 Forgot Class Diagram

Fig\_2.1.2 Forgot Class DiagramDetail



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Reference [**1.2.1**](#_DataLayerBase_Class_(Data)

##### **SystemUserSql Class (Data Access Layer)**

Reference [**1.2.2**](#_SystemUserSql_Class_(Data)

**Methods:**

1/ SelectByField(string fieldName, object value): This method calls procedure SelectByFiend from DB for getting SystemUser by basing on column name and value of this column

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName** | string | Column name |
| **value** | object | Value uses for looking |
| **Return** | bool | Return SystemUser object |

2/ Update(SystemUser businessObject): Update new information of system user into DB. In this session, update will update password of this SystemUser.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | SystemUser | This object contains information for updating DB |
| **Return** | bool | Method will return true of update successful |

##### **StaffSql Class (Data Access Layer)**

**Methods:** SelectByField(string fieldName, object value): This method is used to call procedure for select object Staff based on fieldName column.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName** | string | The column name in DB |
| **value** | object | Value for looking |
| **Return** | Staff | Return the Staff object |

##### **StudentSql Class (Data Access Layer)**

**Methods:** SelectByField(string fieldName, object value): This method is used to call procedure for select object Staff based on fieldName column.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName** | string | The column name in DB |
| **value** | object | Value for looking |
| **Return** | Staff | Return the Staff object |

##### **SystemUserFactory Class (Business Layer)**

**Attributes: N/A**

**Methods:**

1/ SystemUser GetAllBy(SystemUser.SystemUserFields fieldName, object value):This method is used to call method SelectByField from SystemUserSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName** | SystemUserFields | The name of column in DB |
| **value** | object | The looking value of field |
| **Return** | SystemUser | SystemUser Object |

2/ SystemUser Update(SystemUser businessObject): This method is used to call method Update from SystemUserSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | SystemUser | SystemUser Object |
| **Return** | SystemUser | Return true if update successfully |

##### **StaffUserFactory Class (Business Layer)**

**Attributes: N/A**

**Methods:** GetAllBy(Staff.StaffFields fieldName, object value):This method is used to call method SelectByField from StaffSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName** | StaffFields | The name of column in DB |
| **value** | object | The looking value of field |
| **Return** | Staff | Staff Object |

##### **StudentUserFactory Class (Business Layer)**

**Attributes: N/A**

**Methods:**

1/ GetAllBy(Student. StudentFields fieldName, object value):This method is used to call method SelectByField from StudentSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **fieldName** | StudentFields | The name of column in DB |
| **value** | object | The looking value of field |
| **Return** | Student | Student Object |

##### **SystemUser Class (Business Layer)**

Data Transfer Object class, uses to store SystemUser’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **Password** | string | public | Password of login user |
| **Username** | string | public | Username of login user |

**Methods: N/A**

##### **Staff Class (Business Layer)**

Data Transfer Object class, uses to store Staff’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **Account** | string | public | Account of staff |
| **Email** | string | public | Email of Staff |
| **StaffIdentify** | string | public | Identify number of Staff |

**Methods: N/A**

##### **Student Class (Business Layer)**

Data Transfer Object class, uses to store Student’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **Account** | string | public | Account of student |
| **Email** | string | public | Email of student |
| **StudentIdentify** | string | public | Identify number of student |

**Methods: N/A**

##### **BussinessObjectBase Class (Bussiness Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **SmartSessionPage Class (Common layer)**

This class use to execute send password method and gen auto password.

**Methods:**

1/ string GenPassword(int lenght): This method is used to gen Random password.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **lenght** | int | Expect length of password |
| **Return** | string | Password string after generate |

2/ SendEmail(string username, string identify): Method will get username and identify from input of user. After that, method will look for object in DB and send email.

**Parameters and Return**

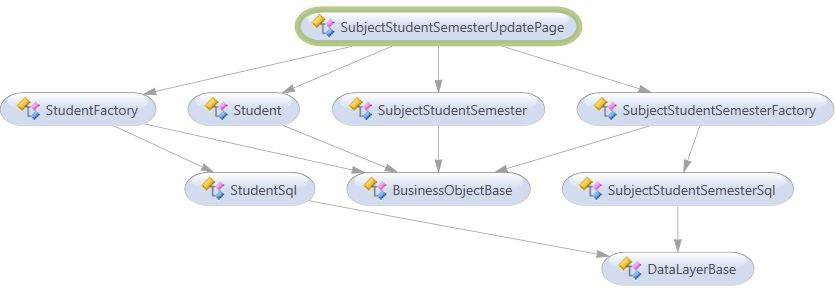
|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **username** | string | Username value |
| **identify** | string | Identify value |
| **Return** | string | Return value after send email.  “1”. Incorrect username and password  “2”. Unable to connect DB  “Email”. Return email. |

##### **HomePage Class (Web Layer)**

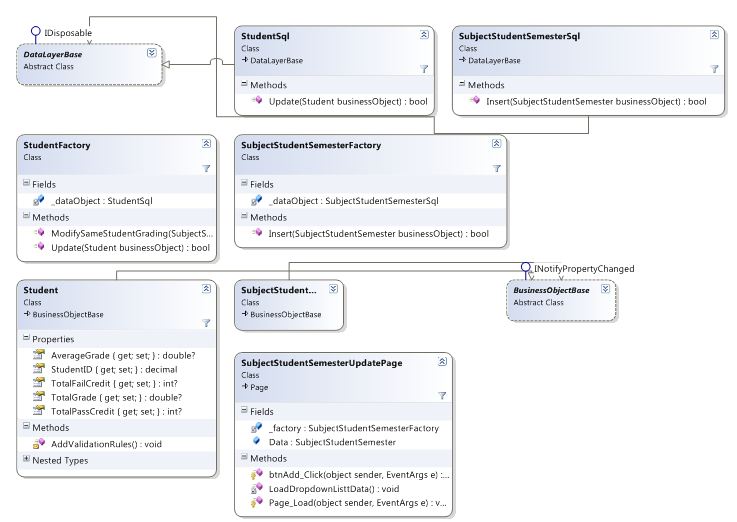
This is the forgot password page for user reset new password. Method BtnSendPassword\_Click will call method SendPassword from SmartSessionPage for checking and sending password.

### <Staff> Add Student Score

#### Class Diagram



Fig\_3.1.1 Add Student Grade Diagram



Fig\_3.1.2 Add Student Grade DiagramDetail

#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which implement get a connection object to connect to database and inherits IDisposable Interface class to use method Dispose connection when doesn’t use.

##### **StudentSql Class (Data Access Layer)**

**Methods:** Update(Student businessObject): Update new information of system user into DB. In this session, update will update grade and credit of this Student.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Student | This object contains information for updating DB |
| **Return** | bool | Method will return true of update successful |

##### **SubjectStudentSemesterSql Class (Data Access Layer)**

**Methods:** Insert(SubjectStudentSemester businessObject): Update new information of system user into DB.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Student | This object contains information for updating DB |
| **Return** | bool | Method will return true of insert successful |

##### **StudentFactory Class (Business Layer)**

**Attributes: N/A**

**Methods:**

1/ Update(Student businessObject):This method is used to call method Update from StudentSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Student | Object Student for update |
| **Return** | bool | Method will return true if update successfully |

2/ ModifySameStudentGrading(SubjectStudentSemester oldObject, SubjectStudentSemester newObject): This method is used calculate and update grade for student

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **oldObject** | SubjectStudentSemester | oldObject store previous SubjectStudentSemester that student has learned |
| **newObject** | SubjectStudentSemester | newObject store new SubjectStudentSemester of student |
| **Return** | int | Report if method executes successfully or not, or errors occur. |

##### **SubjectStudentSemesterFactory Class (Business Layer)**

**Attributes: N/A**

**Methods:** Insert(SubjectStudentSemester businessObject):This method is used to call method insert from SubjectStudentSemsterSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | SubjectStudentSemester | Object Student for insert |
| **Return** | bool | Method will return true if insert successfully |

##### **BussinessObjectBase Class (Bussiness Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **Student Class (Business Layer)**

Data Transfer Object class, uses to store Student’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **StudentID** | Decimal | public | Identify for student |
| **AverageGrade** | double | public | Average grade of student |
| **TotalGreditPass** | Int | public | Number of credit that student has passed |
| **TotalGreditPass** | Int | public | Number of credit that student has passed |
| **TotalGrade** | double | public | Total grade of student |

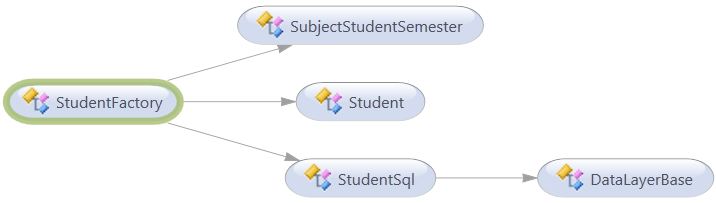
##### **SubjectStudentSemester Class (Business Layer)**

Data Transfer Object class, uses to store SubjectStudentSemester’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

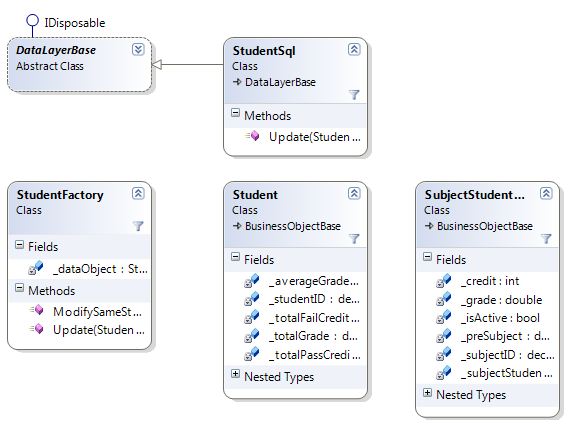
**Attributes: reference DB design.**

### <System> Grading

#### Class Diagram



Fig\_4.1.1 Grading Class Diagram



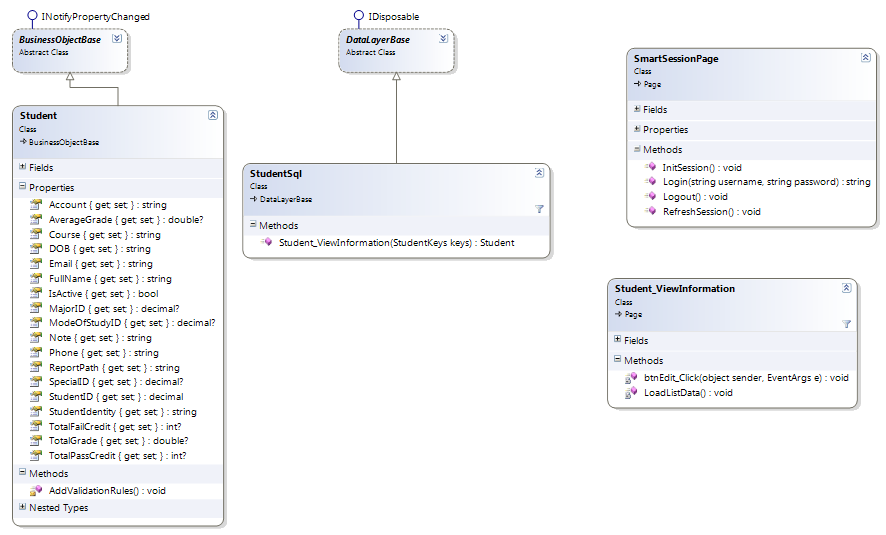
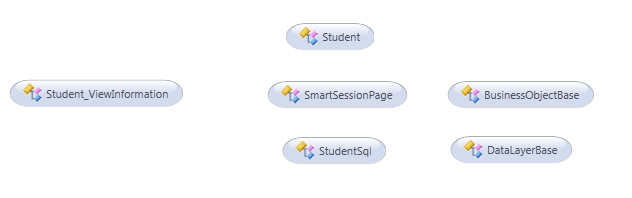
Fig\_4.1.2 Grading Class DiagramDetail

#### Class Explanations

[**Reference 3.2**](#_Class_Explanations)

### <Student> View Private Information

#### Class Diagram



Fig\_5.1.1 View Private Information class diagram

Fig\_5.1.2 View Private Information class diagram detail

#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which implement get a connection object to connect to database and inherits IDisposable Interface class to use method Dispose connection when doesn’t use.

##### **StudentSql Class (Data Access Layer)**

**Attributes: N/A   
 Methods:**

1/ Student Student\_ViewStudentInformation(StudentKeys keys): This method uses to get student’s information by student’s id.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| Parameter/Return | Type | Description |
| keys | decimal | ID of student which to view |
| return | Student | Student object contains student’s information. |

##### **Student Class (Business Layer)**

Data Transfer Object class, uses to store Student’s information in system and transfer among “Business Layer”, “DataAccess layer”, and “Presentation Layer”.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| \_studentiID | decimal | public | ID of student |
| \_account | string | public | Account of student |
| \_fullname | string | public | Full name of student |
| \_dOB | string | public | Date of birth of student |
| \_email | string | public | Email of student |
| \_studentIdentity | decimal | public | Identity of student |
| \_phone | string | public | Phone number of student |
| \_majorID | decimal | public | ID of major |
| \_specialID | decimal | public | ID of special |
| \_modeOfStudyID | decimal | public | ID of mode of study |
| \_totalGrade | double | public | Total grade of student |
| \_totalPassCredit | int | public | Total credit pass |
| \_totalFailCredit | int | public | Total credit fail |
| \_averageGrade | double | public | Average grade of student |
| \_course | string | public | The subject’s credit Course of student |
| \_isActive | bool | public | Status of account |
| \_note | string | public | Others note |
| \_reportPath | string | public | Path of report |

**Methods:**

1/ Void AddValidationRules() : add rules for each data fields.

##### **Student\_ViewInformation Class (Web Layer)**

A Page Class contains HTML markup and .NET code. The Page show details student’s information and allows student to move to update information page for updating new data.

##### **SmartSessionPage Class (Common Layer)**

Contain session information of the account.

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

#### Sequence Diagram

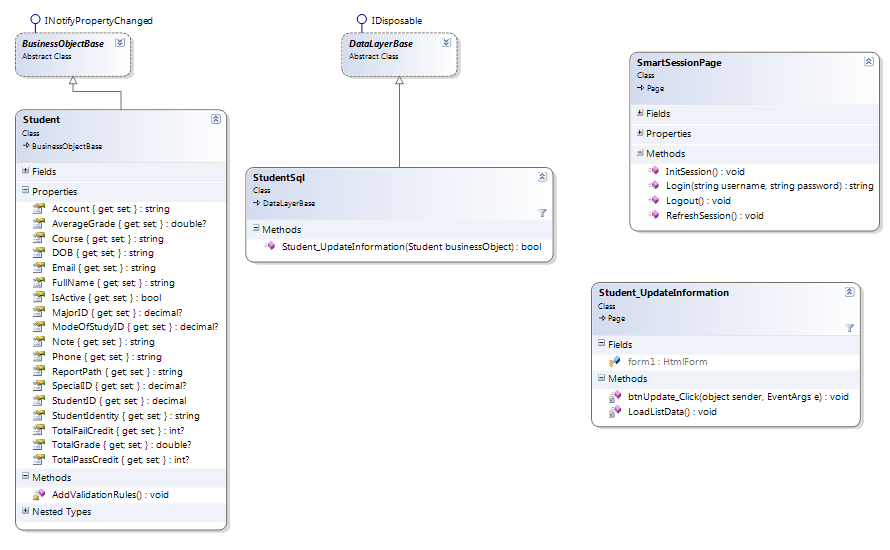
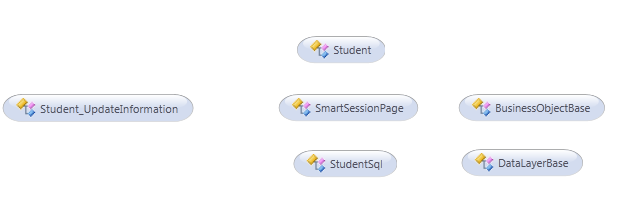
Refer to section C.II.2.37.3 in this document.

#### User Interface Design

Refer to section C.II.2.37.2 in this document.

### <Student, Staff> Edit Student’s Information

#### Class Diagram



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which implement get a connection object to connect to database and inherits IDisposable Interface class to use method Dispose connection when doesn’t use.

##### **StudentSql Class (Data Access Layer)**

**Attributes: N/A**

**Methods:**

1/ Bool Student\_UpdateStudentInformation(Student businessObject). This method uses to update student’s information.

Parameters and Return

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Student | ID of student which to view |
| **return** | bool | Return true/false if the update is or not successful. |

##### **Student Class (Business Layer)**

Data Transfer Object class, uses to store Student’s information in system and transfer among “Business Layer”, “DataAccess layer”, and “Presentation Layer”.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_studentiID** | decimal | public | ID of student |
| **\_account** | string | public | Account of student |
| **\_fullname** | string | public | Full name of student |
| **\_dOB** | string | public | Date of birth of student |
| **\_email** | string | public | Email of student |
| **\_studentIdentity** | decimal | public | Identity of student |
| **\_phone** | string | public | Phone number of student |
| **\_majorID** | decimal | public | ID of major |
| **\_specialID** | decimal | public | ID of special |
| **\_modeOfStudyID** | decimal | public | ID of mode of study |
| **\_totalGrade** | double | public | Total grade of student |
| **\_totalPassCredit** | int | public | Total credit pass |
| **\_totalFailCredit** | int | public | Total credit fail |
| **\_averageGrade** | double | public | Average grade of student |
| **\_course** | string | public | The subject’s credit Course of student |
| **\_isActive** | bool | public | Status of account |
| **\_note** | string | public | Others note |
| **\_reportPath** | string | public | Path of report |

**Methods:**

1/ Void AddValidationRules() : add rules for each data fields.

##### **Student\_UpdateInformation Class (Web Layer)**

A Page Class contains HTML markup and .NET code which uses to update new student’s information.

##### **SmartSessionPage Class (Common Layer)**

Contain session information of the account.

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

#### Sequence Diagram

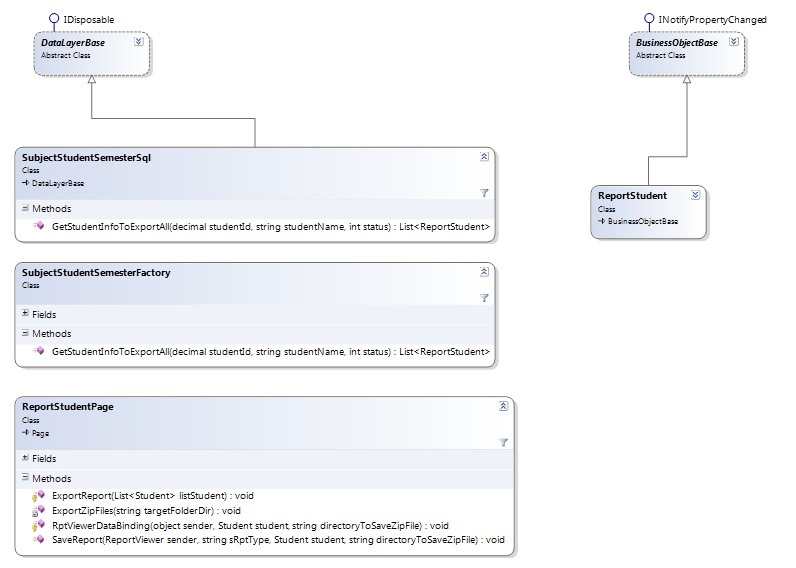
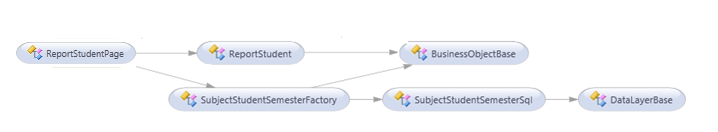
Refer to section C.II.2.38.3, C.II.2.45.8.3 in this document.

#### User Interface Design

Refer to section C.II.2.38.2, C.II.2.45.8.2 in this document.

### <Student, Staff> Export All Collection Reports

#### Class diagram



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which implement get a connection object to connect to database and inherits IDisposable Interface class to use method Dispose connection when doesn’t use.

##### **SubjectStudentSemesterSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/List<ReportStudent> GetStudentInfoToExportAll(decimal studentId,string studentName. int status)

This method uses to access to Database and search all student grades.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **studentId** | decimal | The Student ID uses to filter |
| **studentName** | string | Name of student |
| **status** | int | The Status use to filter |
| **return** | List<ReportStudent> | List of Student grades’s info result. |

##### **SubjectStudentSemesterFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SubjectStudentSemesterSql | Private | SubjectStudentSemesterSql object to perform ReportStudent operations |

**Methods:**

1/List<ReportStudent> GetStudentInfoToExportAll(decimal studentId, string studentName, int status)

This method uses to search all student grades’ information which is appropriate with the semester, status for exporting by using GetStudentInfoToExport function in SubjectStudentSemesterSql class.

##### **ReportStudentPage Class (Web Layer)**

1/void RptViewerDataBinding(object sender, Student student, string directoryToSaveZipFile)

This method uses to make contents of report files

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sender** | Object |  |
| **student** | Student | The student information to export |
| **directoryToSaveZipFile** | string | Directory path to save rar file report |
| **return** | void |  |

2/SaveReport(ReportViewer sender, string sRptType, Student student, string directoryToSaveZipFile)

This method uses to create file report excel for student and save to server’s directory

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sender** | ReportViewer | ReportViewer object uses to render report content |
| **sRptType** | string | File type of the report (Excel) |
| **student** | Student | The student information to export |
| **directoryToSaveZipFile** | string | The directory path of file rar report |
| **return** | void |  |

3/void ExportZipFiles(string targetFolderDir)

This method uses to create rar file report from excel files which was saved before at Server’s folder in targetFolderDir path and send to user for download.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **targetFolderDir** | string | The directory path of file rar report |
| **return** | void |  |

##### **ReportStudent Class (Business Layer)**

Data Transfer Object class, uses to store ReportStudent’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

#### Sequence Diagram

Refer to section C.II.2.45.7.3, C.II.2.44.3 in this document.

#### User Interface Design

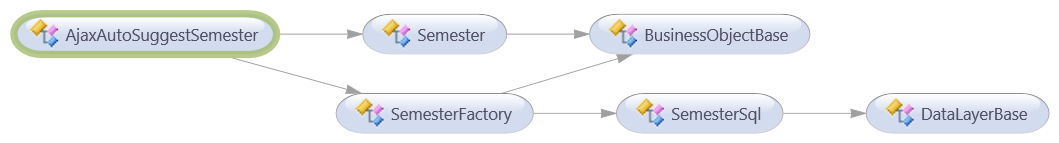
Refer to section C.II.2.45.7.3, C.II.2.44.2 in this document.

### <Staff> Auto Suggest Semester



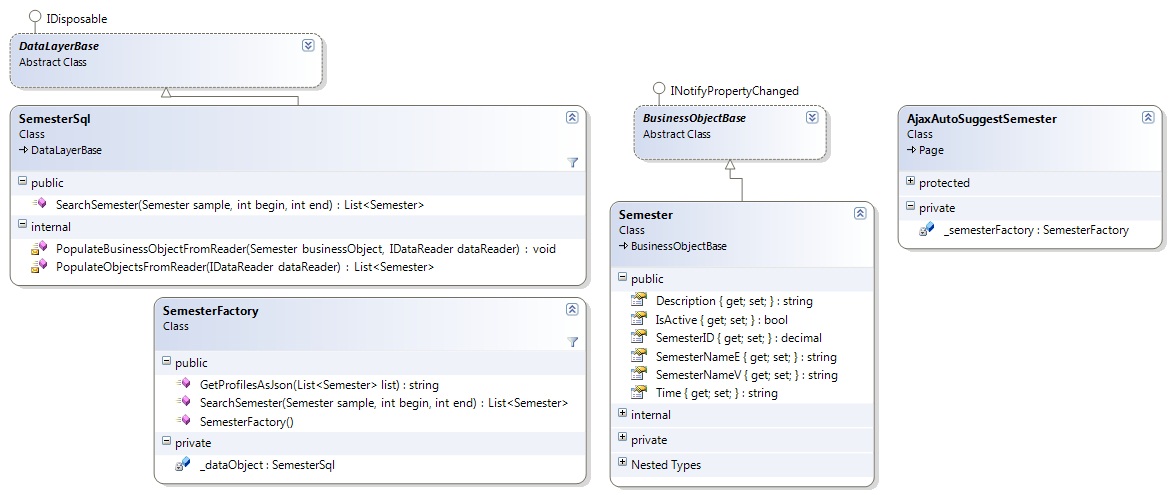
#### Class Diagram

This part is the detailed design for the following use cases: Auto Suggest Semester



Fig\_8.1.1 Auto Suggest Semester Class diagram

Fig\_8.1.2 Auto Suggest Semester Class diagram detail



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which is implemented to get database connect object to connect to database and inherit Idisposable Interface class to use method Dispose connection when unused.

##### **SemesterSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/ List<Semester> SearchSemester(Semester smt, int begin, int end)

This method uses to access to Database and search all semester fields which equals with search ‘sample’ condition and select from ‘begin’ field to ‘end’ filed and return this list of semesters to system.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **Sample** | Semester | The Semester object which contains the search conditions inside such as: Semester Name… |
| **Begin** | int | The begin row to select in list results |
| **End** | int | The row row to select in list results |
| **Return** | List<Semester> | List of Semesters result. |

2/ List<Semester> PopulateObjectsFromReader(IdataReader dataReader)

This method uses to get Semester Object from dataReader after access information in database and add to List Semester.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **dataReader** | IdataReader | The DataReader which contains information of list Semester objects which are gotten from database. |
| **Return** | List<Semester> | List of Semesters result. |

3/ void PopulateBussinessObjectFromReader(Semester bussinessObject, IdataReader dataReader)

This method uses to read dataReader object’s value into Semester object’s properties.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **bussinessObject** | Semester | Semester object |
| **dataReader** | IdataReader | The DataReader which contains information of list Semester objects which are gotten from database. |
| **Return** | void | Return nothing |

##### **SemesterFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SemesterSql | Private | SemesterSql object to perform Semester operations |

**Methods:**

1/ SemesterFactory()

This is constructor of SemesterFactory class.

2/ string GetProfileAsJson(List<Semester> list)

This method uses to make List of semester objects become JSON string to use as a list of javascript objects

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **list** | List<Semester> | The list of Semester objects |
| **return** | String | JSON string of list Semesters. |

3/ List<Semester> SearchSemester(Semester sample, int begin, int end)

This method uses to search semesters by using SearchSemester function in SemesterSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Semester | The Semester object which contains the search conditions inside such as: Semester Name… |
| **begin** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **return** | List<Semester> | List of Semesters result. |

##### **BussinessObjectBase Class (Bussiness Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **Semester Class (Bussiness Layer)**

Data Transfer Object class, uses to store semester’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

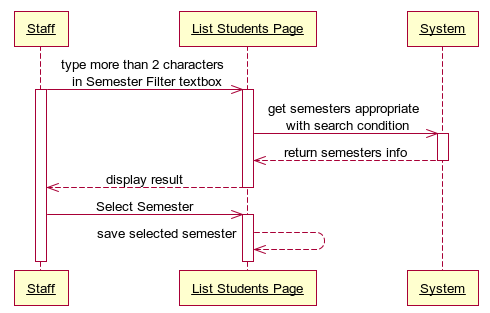
**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **SemesterID** | decimal | public | ID of Semester |
| **SemesterNameE** | string | public | Semester Name in English |
| **SemesterNameV** | string | public | Semester Name in Vietnamese |
| **Description** | string | public | Description of semester |
| **Time** | string | public | The periods time of this semester |
| **IsActive** | bool | public | The status of this semester |

**Methods: N/A**

##### **AjaxAutoSuggestSemester Class (Web Layer)**

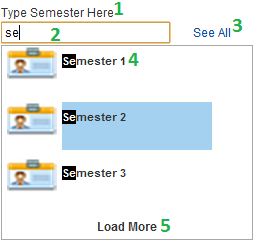
Ajax Page Class which contains HTML markup and .NET code to render details semester’s information.



#### Sequence Diagram

#### User Interface Design

##### **Screen Images**



##### **Objects and Actions**

**Data Fields Definition**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field name** | **Description** | **Read**  **Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| (1) Semester label | A label contents “Type Semester Here” | N/A | N/A | Label | Text | Fixed |
| (2)Semester text field | The textbox for staff to type name of semester he/she wants to choose. | N | N | Textbox | Text | 200 |
| (4) Semester Result label | A label contents the semester name result of suggesting. | N/A | N/A | Label | Text | Fixed |

**Buttons / Hyper Links**

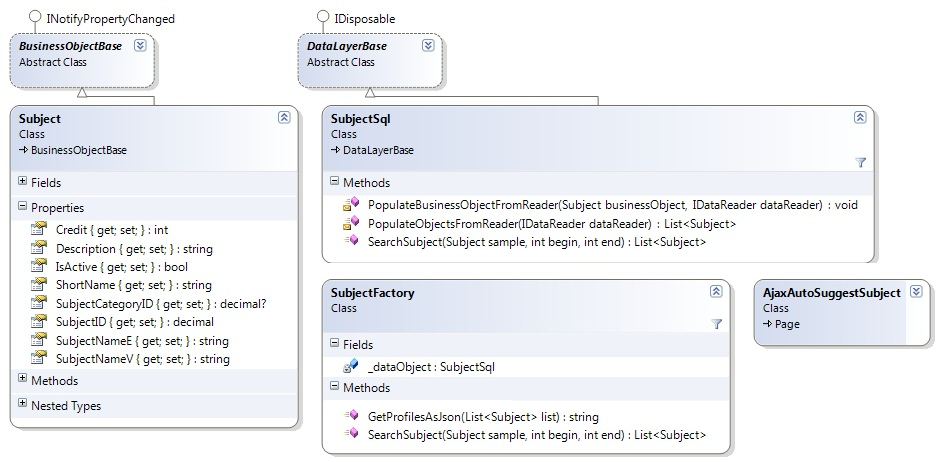
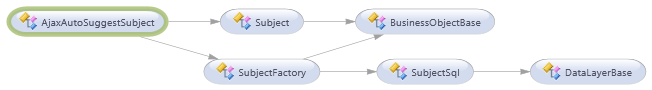
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **Description** | **Available Criteria** | **Validations** | **Outcome** |
| (3) See All Hyperlink Button | The button will allows the system to search all of the semesters and return into List Box result | Always available | No | The semester results will be loaded and display into the List Box Semester Result for staff to choose. |
| (5) Load More Hyperlink Button | The button will allow the system to load next three semester results into List Box result. | Only available when the result of semester in return list is more than 3. | No | The next 3 semester results will be loaded and display into the List Box Semester Result for staff to choose. |

### <Staff> Auto Suggest Subject



#### Class Diagram

This part is the detailed design for the following use cases: Auto Suggest Subject



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **Subject Sql Class (Data Access Layer)**

**Attributes: N/A**   
**Methods:**

1/ List<Subject> SearchSubject(Subject sample, int begin, int end)

This method uses to access to Database and search all Subject fields which equals with search ‘sample’ condition and select from ‘begin’ field to ‘end’ filed and return this list of Subjects to system.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Subject | The Subject object which contains the search conditions inside such as: Subject Name… |
| **begin** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **return** | List<Subject> | List of Subjects result. |

2/ List<Subject> PopulateObjectsFromReader(IdataReader dataReader)

This method uses to get Subject Object from dataReader after access information in database and add to List Subject.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **dataReader** | IdataReader | The DataReader which contains information of list Subject objects which are gotten from database. |
| **return** | List<Subject> | List of Subjects result. |

3/ void PopulateBussinessObjectFromReader(Subject bussinessObject, IdataReader dataReader)

This method uses to read dataReader object’s value into Subject object’s properties.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **bussinessObject** | Subject | Subject object |
| **dataReader** | IdataReader | The DataReader which contains information of list Subject objects which are gotten from database. |
| **return** | void | Return nothing |

##### **Subject Factory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SubjectSql | Private | SubjectSql object to perform Subject operations |

**Methods:**

1/ string GetProfileAsJson(List<Subject> list)

This method uses to make List of Subject objects become JSON string to use as a list of javascript objects

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **list** | List<Subject> | The list of Subject objects |
| **return** | String | JSON string of list Subjects. |

2/ List<Subject> SearchSubject(Subject sample, int begin, int end)

This method uses to search Subjects by using SearchSubject function in SubjectSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Subject | The Subject object which contains the search conditions inside such as: Subject Name… |
| **begin** | Int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **return** | List<Subject> | List of Subjects result. |

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **Subject Class (Business Layer)**

Data Transfer Object class, uses to store Subject’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

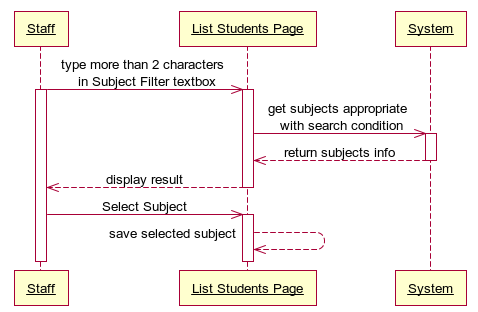
|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **SubjectID** | decimal | public | ID of Subject |
| **SubjectNameE** | string | public | Subject Name in English |
| **SubjectNameV** | string | public | Subject Name in Vietnamese |
| **Description** | string | public | Description of Subject |
| **Credit** | int | public | The subject’s credit |
| **ShortName** | string | public | Short name of the subject |
| **SubjectCategoryID** | decimal | public | The category ID of Subject |
| **IsActive** | bool | public | The status of this Subject |

**Methods: N/A**

##### **AjaxAutoSuggest Subject Class (Web Layer)**

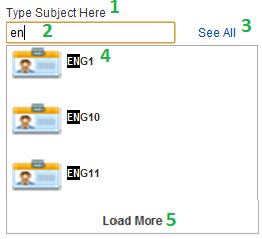
Ajax Page Class which contains HTML markup and .NET code to render details Subject‘s information.

#### Sequence Diagram



#### User Interface Design

##### **Screen Images**



##### **Objects and Actions**

**Data Fields Definition**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field name** | **Description** | **Read**  **Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| (1) Subject label | A label contents “Type Subject Here” | N/A | N/A | Label | Text | Fixed |
| (2) Subject text field | The textbox for staff to type name of subject he/she wants to choose. | N | N | Textbox | Text | 200 |
| (4) Subject Result label | A label contents the subject name result of suggesting. | N/A | N/A | Label | Text | Fixed |

**Buttons / Hyper Links**

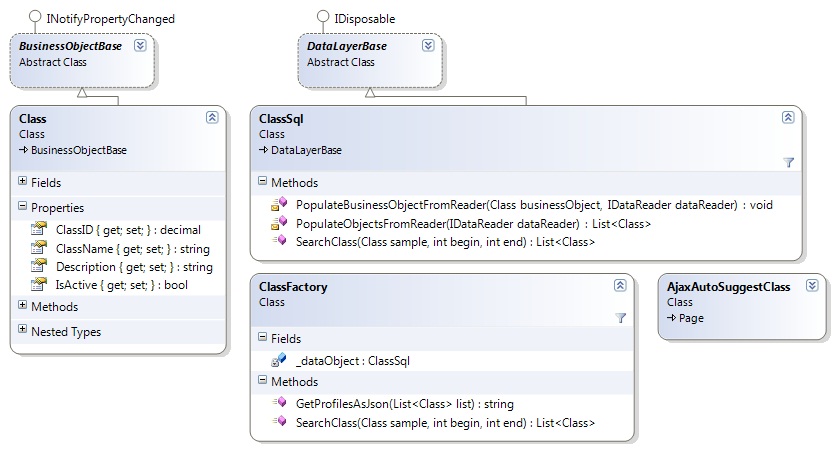
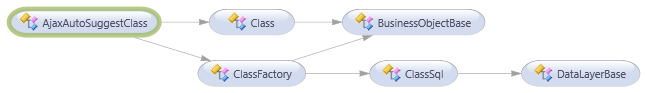
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **Description** | **Available Criteria** | **Validations** | **Outcome** |
| (3) See All Hyperlink Button | The button will allows the system to search all of the subjects and return into List Box result | Always available | No | The subject results will be loaded and display into the List Box Subject Result for staff to choose. |
| (5) Load More Hyperlink Button | The button will allow the system to load next three subject results into List Box result. | Only available when the result of subject in return list is more than 3. | No | The next 3 subject results will be loaded and display into the List Box Subject Result for staff to choose. |

### <Staff> Auto Suggest Class



#### Class Diagram

This part is the detailed design for the following use cases: Auto Suggest Class



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **ClassSql Class (Data Access Layer)**

**Attributes: N/A**   
**Methods:**

1/ List<Class> SearchClass(Class sample, int begin, int end)

This method uses to access to Database and search all Class fields which equals with search ‘sample’ condition and select from ‘begin’ field to ‘end’ filed and return this list of Classes to system.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Class | The Class object which contains the search conditions inside such as: Class Name… |
| **begin** | int | The begin row to select in list results |
| **end** | int | The row end to select in list results |
| **return** | List<Class> | List of Classes result. |

2/ List<Class> PopulateObjectsFromReader(IdataReader dataReader)

This method uses to get Class Object from dataReader after access information in database and add to List Class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **dataReader** | IdataReader | The DataReader which contains information of list Class objects which are gotten from database. |
| **return** | List<Class> | List of Classes result. |

3/ void PopulateBusinessObjectFromReader(Class businessObject, IdataReader dataReader)

This method uses to read dataReader object’s value into Class object’s properties.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Class | Class object |
| **dataReader** | IdataReader | The DataReader which contains information of list Class objects which are gotten from database. |
| **return** | void | Return nothing |

##### **ClassFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | ClassSql | Private | ClassSql object to perform Class operations |

**Methods:**

1/ string GetProfileAsJson(List<Class> list)

This method uses to make List of Class objects become JSON string to use as a list of javascript objects

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **list** | List<Class> | The list of Class objects |
| **return** | String | JSON string of list Classes. |

2/ List<Class> SearchClass(Class sample, int begin, int end)

This method uses to search Classes by using SearchClass function in ClassSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Class | The Class object which contains the search conditions inside such as: Class Name… |
| **begin** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **return** | List<Class> | List of Classes result. |

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **Class Class (Business Layer)**

Data Transfer Object class, uses to store Class’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

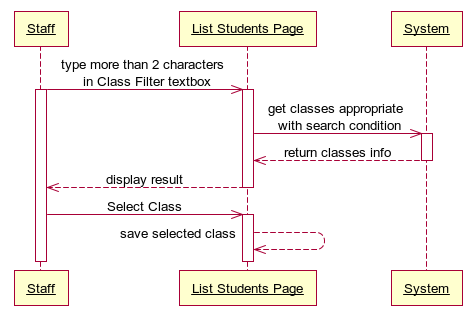
|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **ClassID** | decimal | public | ID of Class |
| **ClassName** | string | public | Name of class |
| **Description** | string | public | Description of class |
| **IsActive** | bool | public | The status of this class |

**Methods: N/A**

##### **AjaxAutoSuggest Class Class (Web Layer)**

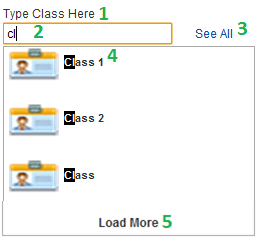
Ajax Page Class which contains HTML markup and .NET code to render details Class‘s information.

#### Sequence Diagram



#### User Interface Design

##### **Screen Images**



##### **Objects and Actions**

**Data Fields Definition**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field name** | **Description** | **Read**  **Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| (1) Class label | A label contents “Type Class Here” | N/A | N/A | Label | Text | Fixed |
| (2) Class text field | The textbox for staff to type name of Class he/she wants to choose. | N | N | Textbox | Text | 200 |
| (4) Class Result label | A label contents the Class name result of suggesting. | N/A | N/A | Label | Text | Fixed |

**Buttons / Hyper Links**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **Description** | **Available Criteria** | **Validations** | **Outcome** |
| (3) See All Hyperlink Button | The button will allows the system to search all of the Classes and return into List Box result | Always available | No | The Class results will be loaded and display into the List Box Class Result for staff to choose. |
| (5) Load More Hyperlink Button | The button will allow the system to load next three Class results into List Box result. | Only available when the result of Class in return list is more than 3. | No | The next 3 Class results will be loaded and display into the List Box Class Result for staff to choose. |

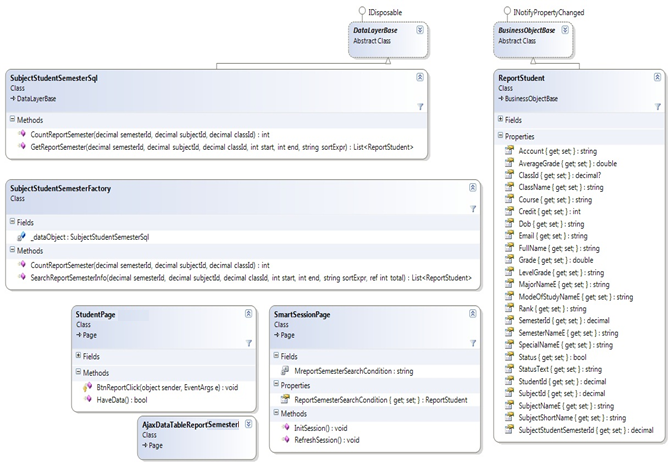
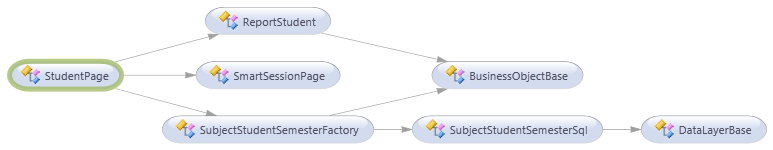
### <Staff> View List Students and Grade by Filter Conditions



#### Class Diagram

This part is the detailed design for the following use cases:

* Filter List Students By Semester
* Filter List Students By Subject
* Filter List Students By Class



#### Class Explanations

##### **SubjectStudentSemesterSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/ int CountReportSemester(decimal semesterId, decimal subjectId, decimal classId)

This method uses to access to Database and count the numbers of all students who studied the filter subject of the filter semester in the filter class and return the number in int type.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **semesterId** | decimal | The Semester ID uses to filter |
| **subjectId** | decimal | The Subject ID uses to filter |
| **classId** | decimal | The Class ID uses to filter |
| **Return** | int | The number of students appropriate with filter conditions. |

2/ List<ReportStudent> GetReportSemester(decimal semesterId, decimal subjectId, decimal classId, int start, int end, string sortExpr)

This method uses to get all Students in system whose information appropriate with filter conditions such as semester, subject, and class and return the List of ReportStudent objects.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **semesterId** | decimal | The Semester ID uses to filter |
| **subjectId** | decimal | The Subject ID uses to filter |
| **classId** | decimal | The Class ID uses to filter |
| **start** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **sortExpr** | string | The sort expression variable uses to sort table result. |
| **return** | List<ReportStudent> | List of Student Objects result. |

##### **SubjectStudentSemesterFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SubjectStudentSemesterSql | Private | SubjectStudentSemesterSql object to perform ReportStudent operations |

**Methods:**

1/ int CountReportSemester(decimal semesterId, decimal subjectId, decimal classId)

This method uses to count the numbers of all students who studied the filter subject of the filter semester in the filter class by using CountReportSemester function in SubjectStudentSemesterSql class and return the number of students in int type.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **semesterId** | Decimal | The Semester ID uses to filter |
| **subjectId** | Decimal | The Subject ID uses to filter |
| **classId** | decimal | The Class ID uses to filter |
| **return** | int | The number of students appropriate with filter conditions. |

2/ List<ReportStudent> SearchReportSemesterInfo(decimal semesterId, decimal subjectId, decimal classId, int start, int end, string sortExpr, ref int total)

This method uses to search list of students by using GetReportSemester function in SubjectStudentSemesterSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **semesterId** | decimal | The Semester ID uses to filter |
| **subjectId** | decimal | The Subject ID uses to filter |
| **classId** | decimal | The Class ID uses to filter |
| **start** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **sortExpr** | string | The sort expression variable uses to sort table result. |
| **total** | ref int | The reference variable total number of results after filter by conditions |
| **return** | List<ReportStudent> | List of Student Objects result. |

##### **ReportStudent Class (Bussiness Layer)**

Data Transfer Object class, uses to store ReportStudent’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **StudentID** | decimal | Public | ID of student |
| **FullName** | string | Public | Full name of student |
| **DOB** | string | Public | Date of birth of student |
| **Email** | string | Public | Email of student |
| **Course** | string | Public | Course of student |
| **SemesterID** | decimal | Public | ID of semester which student has studied |
| **SemesterNameE** | string | Public | English Name of semester which student has studied |
| **SubjectID** | decimal | public | ID of subject which student has studied |
| **SubjectNameE** | string | public | English Name of subject which student has studied |
| **ClassID** | decimal | public | ID of class which student has studied |
| **ClassName** | string | public | Name of class which student has studied |
| **MajorNameE** | string | public | English Name of major |
| **SpecialNameE** | string | public | English Name of special |
| **ModeOfStudyNameE** | string | public | English Name of mode study |
| **Credit** | int | public | The subject’s credit |
| **Grade** | double | public | The subject’s grade of student |
| **LevelGrade** | string | public | The grade level of student |
| **Rank** | string | public | The subject’s rank of student |
| **Status** | bool | public | The subject’s status of student |
| **StatusText** | string | public |  |
| **SubjectShortName** | string | public | The short name of subject |
| **AverageGrade** | double | public | The average grade of all subjects of student |
| **Account** | string | public | The account of student |
| **SubjectStudentSemesterID** | decimal | public |  |

**Methods: N/A**

##### **SmartSessionPage Class (Common Layer)**

Class uses for storing session variables.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **ReportSemesterSearchCondition** | ReportStudent | public | ReportStudent object |

**Methods:**

1/ void InitSession()

This method uses to initiate all session’s variables when needed.

2/ void RefreshSession()

This method uses to refresh all session’s variables when needed.

##### **AjaxDataTableReportSemester Class (Web Layer)**

Ajax Page Class serves as an Ajax page which contains .NET code to render details student’s information for ReportStudentPage class.

##### **StudentPage Class (Web Layer)**

The Report Student Page class which contains HTML markup, css, javascript code, and .NET code to render all students information for end user.

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **BusinessObjectBase Class (Bussiness Layer)**

Refer to section D.IV.45.2.4 in this document.

#### Sequence Diagram

Refer to section C.II.2.45.2.3, C.II.2.45.4.3, and C.II.2.45.6.3 in this document.

#### User Interface Design

##### **Screen Images**

Refer to section C.II.2.45.2.2, C.II.2.45.4.2, and C.II.2.45.6.2 in this document.

##### **Objects and Actions**

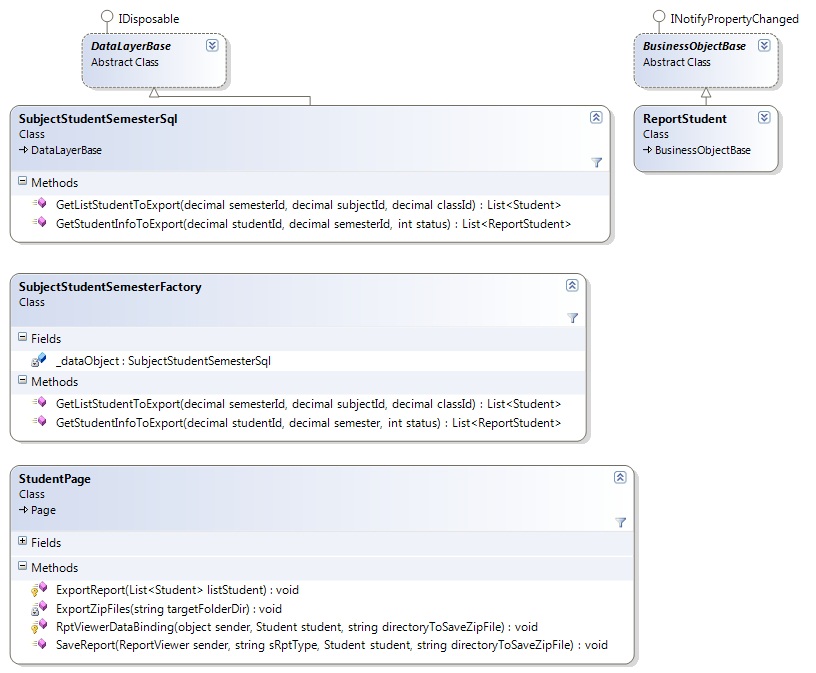
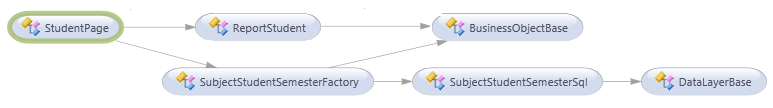
Refer to section C.II.2.45.2.2, C.II.2.45.4.2, and C.II.2.45.6.2 in this document.

### <Staff> Export Collection of Reports



#### Class Diagram

This part is the detailed design for the following use cases: Export Collection of Report



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **SubjectStudentSemesterSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/List<Student> GetListStudentToExport(decimal semesterId, decimal subjectId, decimal classId)

This method uses to access to Database and search all student have information appropriate with the semester, subject, and class conditions to prepare for exporting.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **semesterId** | decimal | The Semester ID uses to filter |
| **subjectId** | decimal | The Subject ID uses to filter |
| **classId** | decimal | The Class ID uses to filter |
| **return** | List<Student> | List of Students result. |

2/List<ReportStudent> GetStudentInfoToExport(decimal studentId, decimal semesterId, int status)

This method uses to access to Database and search all student grades’ information which is appropriate with the semester, status for exporting.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **studentId** | decimal | The Student ID uses to filter |
| **semester** | decimal | The Semester ID uses to filter |
| **status** | int | The Status use to filter |
| **return** | List<ReportStudent> | List of Student grades’s info result. |

##### **SubjectStudentSemesterFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SubjectStudentSemesterSql | Private | SubjectStudentSemesterSql object to perform ReportStudent operations |

**Methods:**

1/List<Student> GetListStudentToExport(decimal semesterId, decimal subjectId, decimal classId)

This method uses to search all student have information appropriate with the semester, subject, and class conditions to prepare for exporting by using GetListStudentToExport function in SubjectStudentSemesterSql class.

2/List<ReportStudent> GetStudentInfoToExport(decimal studentId, decimal semesterId, int status)

This method uses to search all student grades’ information which is appropriate with the semester, status for exporting by using GetStudentInfoToExport function in SubjectStudentSemesterSql class.

##### **StudentPage Class (Web Layer)**

1/void ExportReport(List<Student> listStudent)

This method uses to go through list students and get each student in list to export grade information

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **listStudent** | List<Student> | The list students need to be exported report |
| **return** | void |  |

2/void RptViewerDataBinding(object sender, Student student, string directoryToSaveZipFile)

This method uses to make contents of report files

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sender** | Object |  |
| **student** | Student | The student information to export |
| **directoryToSaveZipFile** | string | Directory path to save rar file report |
| **return** | void |  |

3/SaveReport(ReportViewer sender, string sRptType, Student student, string directoryToSaveZipFile)

This method uses to create file report excel for student and save to server’s directory

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sender** | ReportViewer | ReportViewer object uses to render report content |
| **sRptType** | string | File type of the report (Excel) |
| **student** | Student | The student information to export |
| **directoryToSaveZipFile** | string | The directory path of file rar report |
| **return** | void |  |

4/void ExportZipFiles(string targetFolderDir)

This method uses to create rar file report from excel files which was saved before at Server’s folder in targetFolderDir path and send to user for download.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **targetFolderDir** | string | The directory path of file rar report |
| **return** | void |  |

##### **ReportStudent Class (Bussiness Layer)**

Data Transfer Object class, uses to store ReportStudent’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

##### **BusinessObjectBase Class (Bussiness Layer)**

Refer to section D.IV.45.2.4 in this document.

#### Sequence Diagram

Refer to section C.II.2.45.7.3 in this document.

#### User Interface Design

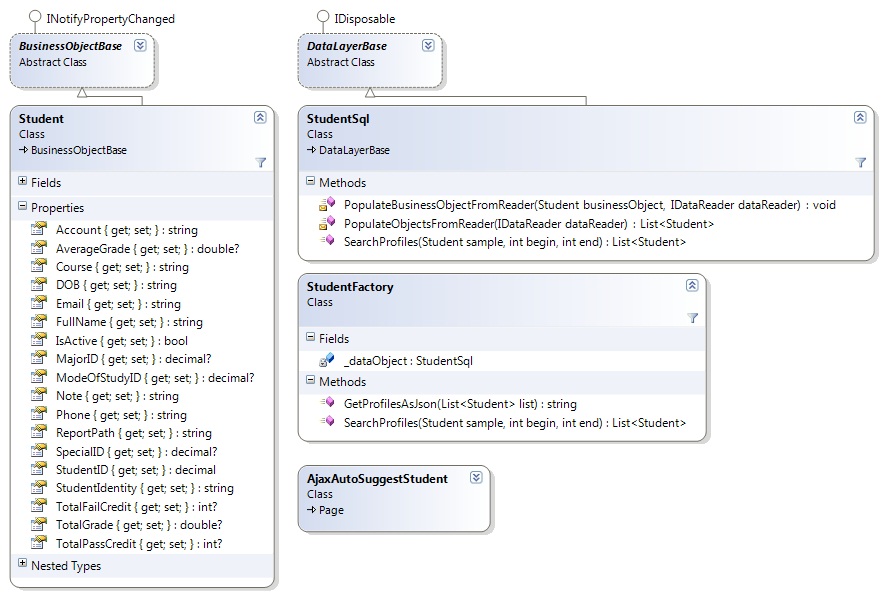
##### **Screen Images**

Refer to section C.II.2.45.7.2 in this document.

##### **Objects and Actions**

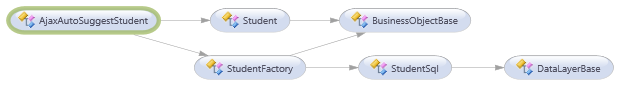
Refer to section C.II.2.45.7.2 in this document.

### <Staff> Auto Suggest Student Name



#### Class Diagram

This part is the detailed design for the following use cases: Auto Suggest Student Name



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **StudentSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/ List<Student> Search Student (Student sample, int begin, int end)

This method uses to access to Database and search all Student fields which equals with search ‘sample’ condition and select from ‘begin’ field to ‘end’ filed and return this list of Students to system.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Student | The Student object which contains the search conditions inside such as: Student Name… |
| **begin** | int | The begin row to select in list results |
| **end** | int | The row end to select in list results |
| **return** | List<Student> | List of Students result. |

2/ List<Student> PopulateObjectsFromReader(IdataReader dataReader)

This method uses to get Student Object from dataReader after access information in database and add to List Student.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **dataReader** | IdataReader | The DataReader which contains information of list Student objects which are gotten from database. |
| **return** | List<Student> | List of Students result. |

3/ void PopulateBusinessObjectFromReader(Student businessObject, IdataReader dataReader)

This method uses to read dataReader object’s value into Student object’s properties.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Student | Student object |
| **dataReader** | IdataReader | The DataReader which contains information of list Student objects which are gotten from database. |
| **return** | void | Return nothing |

##### **StudentFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | StudentSql | Private | StudentSql object to perform Student operations |

**Methods:**

1/ string GetProfileAsJson(List<Student> list)

This method uses to make List of Student objects become JSON string to use as a list of javascript objects

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **list** | List<Student> | The list of Student objects |
| **return** | String | JSON string of list Students. |

2/ List<Student> SearchStudent(Student sample, int begin, int end)

This method uses to search Students by using SearchStudent function in StudentSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sample** | Student | The Student object which contains the search conditions inside such as: Student Name… |
| **begin** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **return** | List<Student> | List of Students result. |

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **Student Class (Business Layer)**

Data Transfer Object class, uses to store Class’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **StudentID** | decimal | public | ID of Student |
| **FullName** | string | public | Name of Student |
| **DOB** | string | public | Date of Birth of Student |
| **Email** | string | public | The email of this Student |

**Methods: N/A**

##### **AjaxAutoSuggestStudent Class (Web Layer)**

Ajax Page Class which contains HTML markup and .NET code to render details Student ‘s information.

#### Sequence Diagram

Refer to section C.II.2.46.1.3 in this document

#### User Interface Design

##### **Screen Images**

Refer to section C.II.2.46.1.2 in this document.

##### **Objects and Actions**

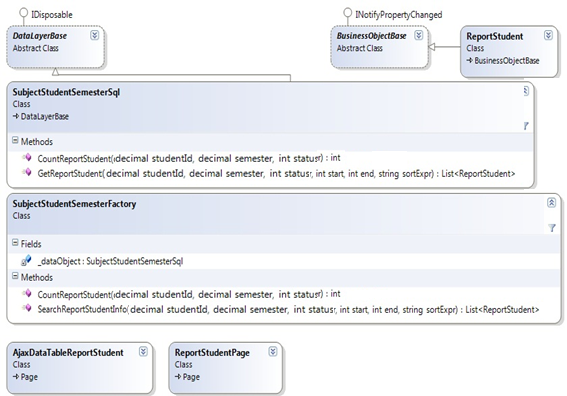
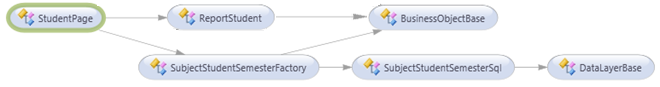
Refer to section C.II.2.46.1.2 in this document

### <Staff> View List Subjects of Student

#### Class Diagram

This part is the detailed design for the following use cases:

* Filter List Subjects of Students By Student Name
* Filter List Subjects of Students By Semester
* Filter List Subjects of Students By Subject’s status



#### Class Explanations

##### **SubjectStudentSemesterSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/ int CountReportStudent(decimal studentId, decimal semester, int status)

This method uses to access to Database and count the numbers of all student’s subjects which is studied by this student in the filter semester and has filter status and return the number in int type.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **studentId** | decimal | The Student ID uses to filter |
| **semester** | decimal | The Semester ID uses to filter |
| **status** | int | The Status use to filter |
| **return** | int | Number of count result |

2/ List<ReportStudent> GetReportStudent(decimal studentId, decimal semester, int status, int start, int end, string sortExpr)

This method uses to get all student’s subjects in system which’s information appropriate with filter conditions such as semester, status and return the List of ReportStudent objects.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **studentId** | decimal | The Student ID uses to filter |
| **semester** | decimal | The Semester ID uses to filter |
| **status** | int | The Status use to filter |
| **start** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **sortExpr** | string | The sort expression variable uses to sort table result. |
| **return** | List<ReportStudent> | List of Student Objects result. |

##### **SubjectStudentSemesterFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SubjectStudentSemesterSql | Private | SubjectStudentSemesterSql object to perform ReportStudent operations |

**Methods:**

1/ int CountReportStudent(decimal studentId, decimal semester, int status)

This method uses to access to Database and count the numbers of all student’s subjects which is studied by this student in the filter semester and has filter status by using methos in SubjectStudentSemesterSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **studentId** | decimal | The Student ID uses to filter |
| **semester** | decimal | The Semester ID uses to filter |
| **status** | int | The Status use to filter |
| **return** | int | Number of count result |

2/ List<ReportStudent> SearchReportStudentInfo(decimal studentId, decimal semester, int status, int start, int end, string sortExpr, ref int total)

This method uses to search list of student’s subjects by using method in SubjectStudentSemesterSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **studentId** | decimal | The Student ID uses to filter |
| **semester** | decimal | The Semester ID uses to filter |
| **status** | int | The Status use to filter |
| **start** | int | The begin row to select in list results |
| **end** | int | The end row to select in list results |
| **sortExpr** | string | The sort expression variable uses to sort table result. |
| **total** | ref int | The reference variable total number of results after filter by conditions |
| **return** | List<ReportStudent> | List of subjects belong to this student result. |

##### **ReportStudent Class (Bussiness Layer)**

Data Transfer Object class, uses to store ReportStudent’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

Refer to section D.IV.48.2.3

**Methods: N/A**

##### **AjaxDataTableReportStudent Class (Web Layer)**

Ajax Page Class serves as an Ajax page which contains .NET code to render details student’s subjects information for ReportStudentPage class.

##### **ReportStudentPage Class (Web Layer)**

The Report Student Page class which contains HTML markup, css, javascript code, and .NET codes to render all students’ subjects information for end user.

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **BusinessObjectBase Class (Bussiness Layer)**

Refer to section D.IV.45.2.4 in this document.

#### Sequence Diagram

Refer to section C.II.2.46.2.3, C.II.2.46.4.3, and C.II.2.46.5.3 in this document.

#### User Interface Design

##### **Screen Images**

Refer to section C.II.2.46.2.2, C.II.2.46.4.2, and C.II.2.46.5.2 in this document.

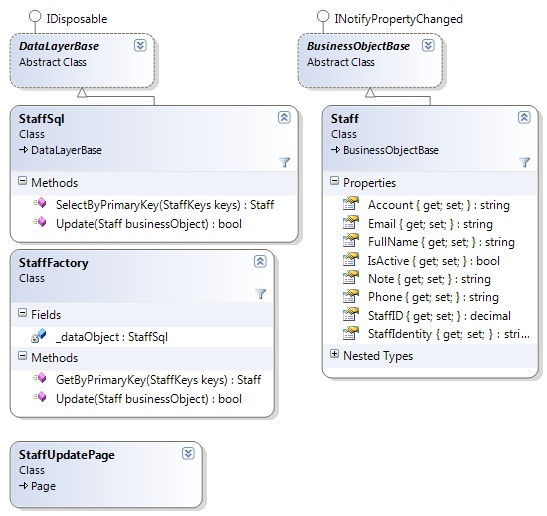
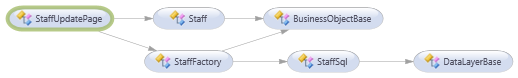
##### **Objects and Actions**

Refer to section C.II.2.46.2.2, C.II.2.46.4.2, and C.II.2.46.5.2 in this document.

### <Staff> Edit Staff Personal Information

#### Class Diagram

This part is the detailed design for the following use cases: Edit Staff Personal Information



#### Class Explanations

##### **DataLayerBase Class (Data Access Layer)**

Refer to section D.IV.45.2.1 in this document.

##### **StaffSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/ Saff SelectByPrimaryKey (StaffKey keys)

This method uses to access to Database and get staff’s information to display after.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **keys** | StaffKey | Object key thay stores StaffID inside |
| **return** | Staff | Object Staff stores staff information |

2/ bool Update(Staff businessObject)

This method uses to update staff information into database.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Staff | Object Staff stores staff information to update. |
| **return** | bool | Return success or fail. |

##### **StaffFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | StaffSql | Private | StaffSql object to perform Staff operations |

**Methods:**

1/ Staff GetByPrimaryKey(StaffKey keys)

This method uses to get staff’s information by using method in StaffSql class to display after.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **keys** | StaffKey | Object key thay stores StaffID inside |
| **return** | Staff | Object Staff stores staff information |

2/ bool Update(Staff businessObject)

This method uses to update staff information into database by using method in StaffSql class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **businessObject** | Staff | Object Staff stores staff information to update. |
| **return** | bool | Return success or fail. |

##### **BusinessObjectBase Class (Business Layer)**

Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

##### **Staff Class (Business Layer)**

Data Transfer Object class, uses to store Staff’s information in system and transfer among Business Layer, DataAccess layer, and Presentation Layer.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **StaffID** | decimal | public | ID of Staff |
| **FullName** | string | public | Name of Staff |
| **Account** | string | public | Account of Staff |
| **Email** | string | public | The email of this Staff |
| **Phone** | string | public | Phone number of staff |
| **StaffIdentity** | string | public | Identity number of staff |
| **Note** | string | public | Some note about staff |
| **IsActive** | bool | public | Status of staff |

**Methods: N/A**

##### **StaffUpdatePage Class (Web Layer)**

Ajax Page Class which contains HTML markup and .NET code to render details Staff‘s information.

#### Sequence Diagram

Refer to section C.II.2.47.3 in this document

#### User Interface Design

##### **Screen Images**

Refer to section C.II.2.47.2 in this document.

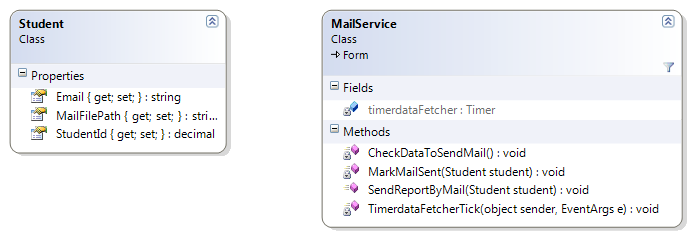
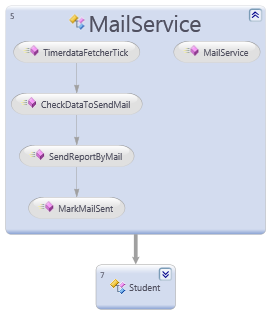
##### **Objects and Actions**

Refer to section C.II.2.47.2 in this document

### <System> Send Report E-Mail for Student

#### Class Diagram

This part is the detailed design for the following use cases: Auto Report E-Mail



#### Class Explanations

##### **MailService Class**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **timerdataFetcher** | Timer | Private | Timer Object of C# uses for auto raise events |

**Methods:**

1/ void CheckDataToSendMail()

This method uses to access into database and check data to send email If has any.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **return** | void | Nothing |

2/ void SendReportByMail(Student student)

This method uses to send email with attaching file report for student if method CheckDataToSendMail() return having data.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **student** | Student | The Student object which contains the Student information includes email, report file, and student ID. |
| **return** | void | Nothing |

3/ void MarkMailSent(Student student)

This method uses to access into database and mark the data report has been sent.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **student** | Student | The Student object which contains the Student information includes email, report file, and student ID. |
| **return** | void | Nothing |

4/ void TimerdataFetcherTick(Object sender, EvenArds e)

This method uses for call method auto checking data in database after every 10 minutes.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **return** | void | Nothing |

##### **Student Class**

Data Transfer Object student is used to store student’s information in system.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **StudentID** | decimal | public | ID of Student |
| **Email** | string | public | The email of this Student |
| **MailFilePath** | string | public | The location path where report file of this student is located in. |

**Methods: N/A**

#### Sequence Diagram

Refer to section C.II.2.48.3 in this document

#### User Interface Design

##### **Screen Images**

Refer to section C.II.2.48.2 in this document.

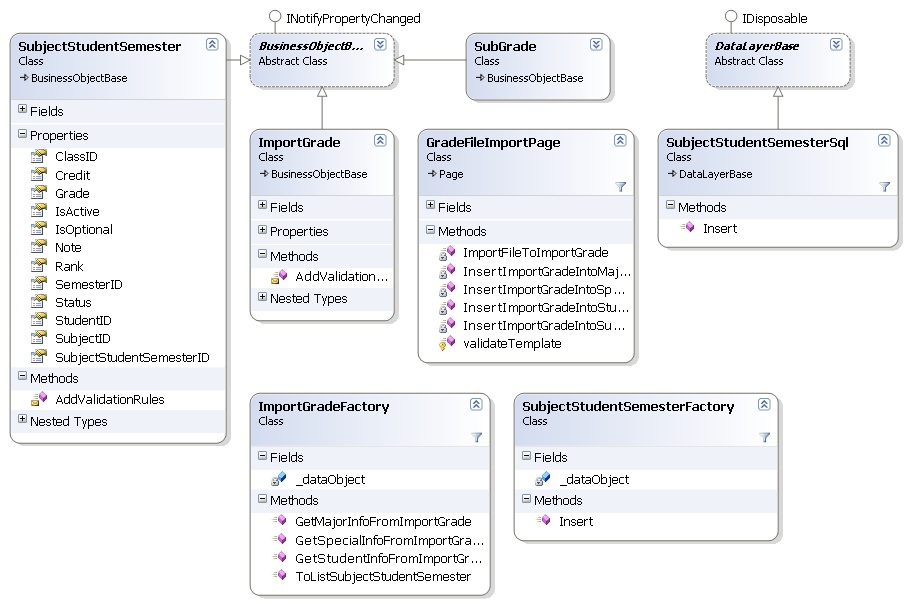
##### **Objects and Actions**

Refer to section C.II.2.48.2 in this document

### <Staff>Import Excel Data File

#### Class Diagram

#### Class Explanations



##### **DataLayerBase Class (Data Access Layer)**

Abstract class for data access layer classes, which implement get a connection object to connect to database and inherits IDisposable Interface class to use method Dispose connection when doesn’t use.

##### **ImportGrade Class (Business Layer)**

ImportGrade Class take the mission as Data Transfer Objecs, uses to store Import Grade information in system and transfer among “Business Layer”, “DataAccess layer”, and “Presentation Layer”. Each record of ImportGrade class presents data of an excel row of import file, means each record of ImportGrade show all the grades of indicated student.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_studentiID** | decimal | public | ID of student |
| **\_account** | string | public | Account of student |
| **\_fullname** | string | public | Full name of student |
| **\_dOB** | string | public | Date of birth of student |
| **\_email** | string | public | Email of student |
| **\_studentIdentity** | decimal | public | Identity of student |
| **\_phone** | string | public | Phone number of student |
| **\_majorID** | decimal | public | ID of major |
| **\_specialID** | decimal | public | ID of special |
| **\_modeOfStudyID** | decimal | public | ID of mode of study |
| **\_subGrade** | List<SubGrade> | public | List that contain subgrade information |

##### **ImportGradeFactory Class (Business Layer)**

**Methods:**

1/ List<SubjectStudentSemester> ToListSubjectStudentSemester (ImportGrade importGrade)

This method uses to create a list of SubjectStudentSemester to be import into SubjectStudentSemester table. Each Subrade in ImportGrade object will create a corresponding SubjectStudentSemester object.

2/ Student GetStudentInfoFromImportGrade(ImportGrade importGrade)

This method uses to create Student object that contains student information. If the student is not existed in the database, it will be import into database using StudentFactory and StudentSql Classes.

3/ Major Get MajorInfoFromImportGrade(ImportGrade importGrade)

This method uses to create Major object that contains major information. If the major is not existed in the database, it will be import into database using MajorFactory and MajorSql Classes.

4/ Special Get Special InfoFromImportGrade(ImportGrade importGrade)

This method uses to create Special object that contains special information. If the special is not existed in the database, it will be import into database using Special Factory and Special Sql Classes.

##### **SubGrade Class (Business Layer)**

SubGrade Class use to store student grade information of a subject of student in concrete semester and class.

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| \_subjectID | decimal | public | ID of student |
| \_semesterID | decimal | public | SemesterID of the grade |
| \_classID | decimal | public | ClassID of the grade |
| **\_credit** | int | public | Credit cost to study this subject |
| **\_grade** | Double | public | Score of student when study this subject on this semester and in this class |

##### **SubjectStudentSemester Class (Data Access Layer)**

Reference 44.2.2

##### **SubjectStudentSemesterSql Class (Data Access Layer)**

**Attributes: N/A   
Methods:**

1/void InsertStudentSubjectSemester (SubjectStudentSemester sss)

This method uses to insert a StudentSubjectSemester datarow into database. This function’s calling when user click on “Import Grade”, happens after the data has been validated.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **sss** | SubjectStudentSemester | SubjectStudentSemester Object to be insert into database |

##### **SubjectStudentSemesterFactory Class (Business Layer)**

**Attributes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **\_ dataObject** | SubjectStudentSemesterSql | Private | SubjectStudentSemesterSql object to perform ImportSubjectStudentSemester operations |

**Methods:**

1/ void InsertSubjectStudentSemester (SubjectStudentSemester sss)

This method uses to insert student grades’ information which is appropriate with the semester, status for importing by using InsertSubjectStudentSemester function in SubjectStudentSemesterSql class.

##### **ImportGradePage Class (Web Layer)**

1/List<ImportGrade> ImportFileToImportGrade(string filepath)

This method scan all rows of excel sheet used range to import into ImportGrade classes for used. Each row will be scanned following this rule:

* Student info field in the ImportGrade classes will be filled in first.

Ada

* A loop will run over each 4 columns to make subgrade and add into the ImportGrade class

This ImportGrade class will be add to returned list

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **filepath** | string | The URL of excel file that user want to import from local computer |
| **return** | void |  |

**Exception and handler**

* Could not find specific files: the filepath url is wrong, check first
* Datatype mismatch: throw exception, return null
* Nullable exception: throw exception, return null

2/void InsertImportGradeIntoSubjectStudentSemester(ImportGrade importGrade)

This method first get List of SubjectStudentSemester object that get by ToListSubjectStudentSemester method of ImportGradeFactoryClass. Then, each SubjectStudentSemester object in this list will be import into database using InsertStudentSubjectSemester of SubjectStudentSemesterFactory Class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **importGrade** | ImportGrade | ImportGrade object, get by a row of data import file |
| **return** | void |  |

3/void InsertImportGradeIntoStudent(ImportGrade importGrade)

This method first get Student object that get by GetStudentInfoFromImportGrade method of ImportGradeFactoryClass. Then, this Student will be check if existed in database. If not, this Student object will be inserted into database using Insert method of StudentFactory Class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **importGrade** | ImportGrade | ImportGrade object, get by a row of data import file |
| **return** | void |  |

4/void InsertImportGradeIntoMajor(ImportGrade importGrade)

This method first get Major object that get by GetMajorInfoFromImportGrade method of ImportGradeFactoryClass. Then, this Major will be check if existed in database. If not, this Major object will be inserted into database using Insert method of MajorFactory Class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **importGrade** | ImportGrade | ImportGrade object, get by a row of data import file |
| **return** | void |  |

5/void InsertImportGradeIntoSpecial(ImportGrade importGrade)

This method first get Special object that get by GetSpecialInfoFromImportGrade method of ImportGradeFactoryClass. Then, this Special will be check if existed in database. If not, this Special object will be inserted into database using Insert method of SpecialFactory Class.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **importGrade** | ImportGrade | ImportGrade object, get by a row of data import file |
| **return** | void |  |

6/boolean ValidateTemplate(WorkSheet workSheet)

This method validate import file by check if header columns is corrected.

**Parameters and Return**

|  |  |  |
| --- | --- | --- |
| **Parameter/Return** | **Type** | **Description** |
| **workSheet** | WorkSheet | The import sheet that user selected |
| **return** | Boolean | True: The selected excel file is validated  False: The selected excel file is not validated. Throw exception. |

##### **BusinessObjectBase Class (Business Layer)**

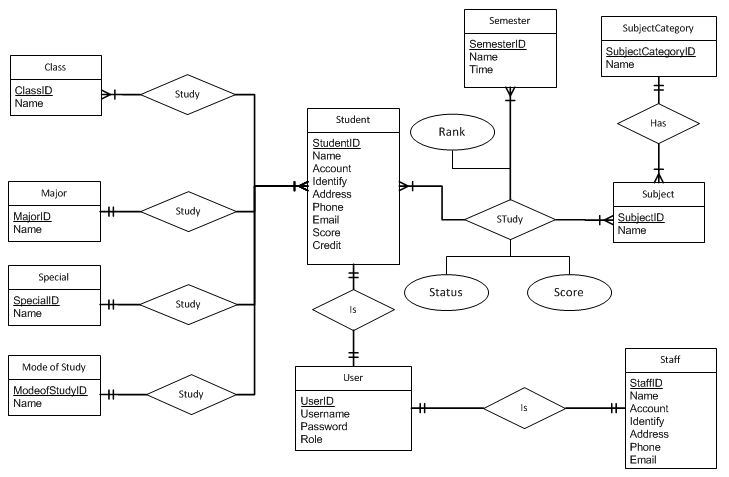
Abstract class for Data Transfer Object layer classes, which implement validation for data types in business object classes.

#### Sequence Diagram

Refer to section C.II.2.57.11 in this document.

#### User Interface Design

Refer to section C.II.2.57.11 in this document.


























































## Database design

### ERD

Fig 1. ERD

### Tables



Fig\_2 Tables

|  |  |  |
| --- | --- | --- |
| **No.** | **Table Name** | **Description** |
| **1** | **SystemUser** | Storing information about user of this system. Such as: Staff, Student, Admin |
| **2** | **Student** | Storing information of student |
| **3** | **Staff** | Storing information of Staff |
| **4** | **Semester** | Storing information of semester |
| **5** | **Subject** | Storing information of subject |
| **6** | **Class** | Storing information of Class |
| **7** | **Major** | Storing information of Major |
| **8** | **Special** | Storing information of special |
| **9** | **ModeOfStudy** | Storing information of ModeOfStudy |
| **10** | **SubjectStudentSemester** | Storing information about grade of Student |
| **11** | **SubjectCategory** | Storing information of category of Subject |

### Table Detail



#### SystemUser table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **UserID** | numeric(18, 0) |  |  | PK |  | Identify for User |
| **Username** | varchar(200) |  |  | FK |  | Username |
| **RoleID** | numeric(18, 0) |  |  |  |  | Identify role for this user |
| **Password** | varchar(200) |  |  |  |  | Password |
| **Note** | nvarchar(MAX) | x |  |  |  | Note |
| **IsActive** | bit |  |  |  |  | Identify this user is active or not |

#### Student table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **StudentID** | numeric(18, 0) |  |  | PK |  | Identify of student |
| **Account** | varchar(200) |  | **x** |  |  | Account for login into system |
| **FullName** | nvarchar(200) |  |  |  |  | Full name |
| **DOB** | varchar(20) | x |  |  |  | Date of Birthday |
| **Email** | varchar(200) |  |  |  |  | Email |
| **StudentIdentity** | varchar(20) |  |  |  |  | Student number id |
| **Phone** | varchar(20) |  |  |  |  | Phone |
| **MajorID** | numeric(18, 0) |  |  | FK |  | Major |
| **SpecialID** | numeric(18, 0) |  |  | FK |  | Special |
| **ModeOfStudyID** | numeric(18, 0) |  |  | FK |  | Mode |
| **TotalGrade** | float |  |  |  |  | Total grade of student |
| **TotalPassCredit** | int |  |  |  |  | Total pass credit |
| **TotalFailCredit** | int |  |  |  |  | Total fail credit |
| **AverageGrade** | float |  |  |  |  | Average Grade of student |
| **Course** | varchar(20) | x |  |  |  | Score of this student |
| **IsActive** | bit | x |  |  |  | Identify this student is active or not |
| **Note** | nvarchar(MAX) | x |  |  |  | Note |
| **ReportPath** | nvarchar(MAX) | x |  |  |  | Report path |

#### Staff table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **StaffID** | numeric(18,0) |  |  | PK |  | Identify of staff |
| **Account** | varchar(200) |  | **x** |  |  | Account for login into system |
| **Fullname** | varchar(200) |  |  |  |  | Full name |
| **Email** | varchar(200) | X |  |  |  | Email |
| **StaffIdentity** | varchar(20) | X |  |  |  | Staff number id |
| **Phone** | varchar(20) | X |  |  |  | Phone |
| **Note** | nvarchar(MAX) | X |  |  |  | Note |
| **IsActive** | bit |  |  |  |  | Identify this Staff is active or not |

#### Semester table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **SemesterID** | numeric(18,0) |  |  | PK |  | Identify for semester |
| **SemesterNameV** | nvarchar(200) | X |  |  |  | Name in Vietnamese |
| **SemesterNameE** | varchar(200) |  |  |  |  | Name in English |
| **StartTime** | varchar(200) | X |  |  |  | Start time of semester |
| **EndTime** | varchar(200) | X |  |  |  | End time of semester |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Identify this Semester is active or not |

#### Subject table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **SubjectID** | numeric(18,0) |  |  | PK |  | Identify for this subject |
| **SubjectNameV** | nvarchar(200) | X |  |  |  | Name in Vietnamese |
| **SubjectNameE** | varchar(200) |  |  |  |  | Name in English |
| **ShortName** | varchar(200) | X |  |  |  | Short name for this subject |
| **IsOptional** | bool |  |  |  |  | This subject is selective, unselective… |
| **Credit** | int |  |  |  |  | Credit of this subject |
| **SubjectCategoryID** | numeric(18,0) | X |  | FK |  | Subject category that subject is included |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Identify this Subject is active or not |

#### Class table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **ClassID** | numeric(18,0) |  |  | PK |  | Identify of this class |
| **ClassName** | varchar(200) |  |  |  |  | Name of class |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Identify this class is active or not |

#### Major table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **MajorID** | numeric(18,0) |  |  | PK |  | Identify |
| **MajorNameV** | nvarchar(200) | X |  |  |  | Name in Vietnamese |
| **MajorNameE** | varchar(200) |  |  |  |  | Name in English |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Active status |

#### Special table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **SpecialID** | numeric(18,0) |  |  | PK |  | Identify |
| **SpecialNameV** | nvarchar(200) | X |  |  |  | Name in Vietnamese |
| **SpecialNameE** | varchar(200) |  |  |  |  | Name in English |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Active status |

#### ModeOfStudy table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **ModeOfStudyID** | numeric(18,0) |  |  | PK |  | Identify |
| **ModeOfStudyNameV** | nvarchar(200) | X |  |  |  | Name in Vietnamese |
| **ModeOfStudyNameE** | varchar(200) |  |  |  |  | Name in English |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Active status |

#### SubjectCategory table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **CategoryID** | numeric(18,0) |  |  | PK |  | Identify |
| **CategoryNameV** | nvarchar(200) | X |  |  |  | Name in Vietnamese |
| **CategoryNameE** | varchar(200) |  |  |  |  | Name in English |
| **Description** | nvarchar(MAX) | X |  |  |  | Description |
| **IsActive** | bit |  |  |  |  | Active status |

#### SubjectStudentSemester table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Null** | **Unique** | **P/F Key** | **Default** | **Description** |
| **SubjectStudentSemesterID** | numeric(18, 0) |  |  | PK |  | Identify |
| **SubjectID** | numeric(18, 0) |  |  | FK |  | Reference value from Subject |
| **StudentID** | numeric(18, 0) |  |  | FK |  | Reference value from Student |
| **SemesterID** | numeric(18, 0) |  |  | FK |  | Reference value from Semester |
| **Credit** | int |  |  |  |  | Credit of this subject |
| **Grade** | float |  |  |  |  | Grade of this subject |
| **Rank** | varchar(20) |  |  |  |  | Rank of this subject |
| **ClassID** | numeric(18, 0) | x |  | FK |  | Reference value from Class |
| **Status** | int |  |  |  |  | Status of this subject |
| **IsOptional** | int |  |  |  |  | This subject is selective or not selective or… |
| **PreSubject** | numeric(18, 0) | x |  |  |  | Previous subject of this subject |
| **Note** | nvarchar(MAX) | x |  |  |  | Note |
| **IsActive** | bit |  |  |  |  | Active status |

## References

[2] MSDN, ' UML Class Diagrams: Guidelines ', viewed 18 February 2012

<http://msdn.microsoft.com/en-us/library/dd409416.aspx>

[3] MSDN, ' UML Sequence Diagrams: Guidelines ', viewed 18 February 2012

<http://msdn.microsoft.com/en-us/library/dd409389.aspx>

[4] Wikipedia, ' Dependency Injection ', viewed 18 February 2012

<http://en.wikipedia.org/wiki/Dependency_injection>