**HCMC UNIVERSITY OF TECHNOLOGY AND EDUCATION**

**FACULTY FOR HIGH QUALITY TRAINING**

**INFORMATION TECHNOLOGY**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

****

**PROJECT 1 REPORT**

**EXPORT, IMPORT DATA BETWEEN**

**SQL SERVER AND MS EXCEL**

**LECTURER NAME: Dr. Nguyen Duc Khoan**

**STUDENT NAME 1: Pham Hoang Viet**

**STUDENT ID : 16110547**

**STUDENT NAME 2: Ngo Van Tu**

**STUDENT ID : 16110255**

**CLASS : 16110CL2**

**Ho Chi Minh City, December 2018LECTURER COMMENT**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**Lecturer**

*Signature*

**M.Sc. Nguyen Duc Khoan**

**CONTENT**

[CONTENT 1](#_Toc532816463)

[*1.* *Project Description* 1](#_Toc532816464)

[1.1. Objective 1](#_Toc532816465)

[1.2. User requirement 1](#_Toc532816466)

[1.3. Usecase diagram 2](#_Toc532816467)

[1.4. Usecase description tables 2](#_Toc532816468)

[*2.* *Software design* 7](#_Toc532816469)

[2.1. Technology 7](#_Toc532816470)

[2.2. Software architecture 10](#_Toc532816471)

[2.3. Design User Interface 12](#_Toc532816472)

[2.4. Design class 15](#_Toc532816473)

[2.4.1. Class diagram 15](#_Toc532816474)

[2.4.2. Class description tables 15](#_Toc532816475)

[2.5. Design database 28](#_Toc532816476)

[2.6. Configuration 28](#_Toc532816477)

[*3.* *Project Implementation* 28](#_Toc532816478)

[3.1. Environment 28](#_Toc532816479)

[3.2. Work plan 28](#_Toc532816480)

[3.3. Work assignment 30](#_Toc532816481)

[3.4. Manual 31](#_Toc532816482)

[3.4.1. Manual Login Form 31](#_Toc532816483)

[3.4.2. Manual Export data from SQL Server to Excel Form 32](#_Toc532816484)

[3.4.3. Manual Import data from Excel to SQL Server 33](#_Toc532816485)

[4. *Conclusion* 33](#_Toc532816486)

[4.1. Evaluation 33](#_Toc532816487)

[4.2. Difficulties 33](#_Toc532816488)

[4.3. Solutions 34](#_Toc532816489)

[4.4. Advantages 34](#_Toc532816490)

[4.5. Defect 34](#_Toc532816491)

[4.6. Development ideas 34](#_Toc532816492)

[4.7. Source code 34](#_Toc532816493)

[REFERENCE 35](#_Toc532816494)

**LIST OF FIGURES**

[Figure 1: Usecase diagram 2](file:///C:\Users\Admin\Desktop\nhom6_document_Updated%20(1).docx#_Toc532816495)

[Figure 2: ADO.NET architecture 8](file:///C:\Users\Admin\Desktop\nhom6_document_Updated%20(1).docx#_Toc532816496)

[Figure 3: Three-layer architecture 10](#_Toc532816497)

[Figure 4: Apply three-layer architecture in our project 12](#_Toc532816498)

[Figure 5: Login Interface 13](#_Toc532816499)

[Figure 6: Export data from SQL Server to Excel Interface 14](#_Toc532816500)

[Figure 7: Format sheet Excel Interface 14](#_Toc532816501)

[Figure 8: Import data from Excel to SQL Server Interface 15](#_Toc532816502)

[Figure 9: Class diagram 15](#_Toc532816503)

[Figure 10: Manual Login Form 31](#_Toc532816504)

[Figure 11: Manual Export data from SQL Server to Excel Form 32](#_Toc532816505)

[Figure 12: Manual Format Form 32](#_Toc532816506)

[Figure 13: Manual Import data from Excel to SQL Server 33](#_Toc532816507)

**LIST OF TABLES**

[Table 1: Usecase Log in description 2](#_Toc532816508)

[Table 2: Usecase Execute description 3](#_Toc532816509)

[Table 3: Usecase View Result description 3](#_Toc532816510)

[Table 4: Usecase Export Excel description 4](#_Toc532816511)

[Table 5: Usecase Format description 5](#_Toc532816512)

[Table 6: Usecase Import description 6](#_Toc532816513)

[Table 7: User Interface work assignment 12](#_Toc532816514)

[Table 8: List of classes are used in DataAccessLayer 16](#_Toc532816515)

[Table 9: List of classes are used in BusinessLogicLayer 16](#_Toc532816516)

[Table 10: List of classes are used in PresentationLayer 17](#_Toc532816517)

[Table 11: List of methods in DAL 17](#_Toc532816518)

[Table 12: List of methods in DAL\_Excel 18](#_Toc532816519)

[Table 13: List of methods in BLL 19](#_Toc532816520)

[Table 14: List of methods in DataBLL 20](#_Toc532816521)

[Table 15: List of methods in DBNamBLL 20](#_Toc532816522)

[Table 16: List of methods in ExportData 21](#_Toc532816523)

[Table 17: List of methods in ImportData 22](#_Toc532816524)

[Table 18: List of methods in FrmImportExport 23](#_Toc532816525)

[Table 19: List of methods in Format 25](#_Toc532816526)

[Table 20: Work plan 28](#_Toc532816527)

[Table 21: Work assignment 30](#_Toc532816528)

**PREFACE**

After nearly 8th weeks of work, our project has also been completed. Through this project, we have learned a great deal about object oriented programming application integration with Microsoft's ADO.NET technology, on how to organize classes, devide ingredient for logic. The tool to import and export data between Excel and SQL Server is really a great topic, helping users minimize the time when they want to access any table in the database, output to excel file used to report or update the database via excel file. It is an intermediate tool that has access to tables in SQL Server and provides users with some choices for excel file formatting when exporting. Although the tool still has some limitations, we will try to fix it in the future, to give users a perfect tool. Highly practical to meet the needs of users.

Finally, we do not forget to send our thanks to Master Nguyen Duc Khoan, our instructor for the project, thanks to the comments, corrections and knowledge of the teacher we have had can complete the project relatively well. We also hope to receive more comments, contributing to the product more perfect.

Sincerely thanks!

# **CONTENT**

1. ***Project Description***
   1. **Objective**

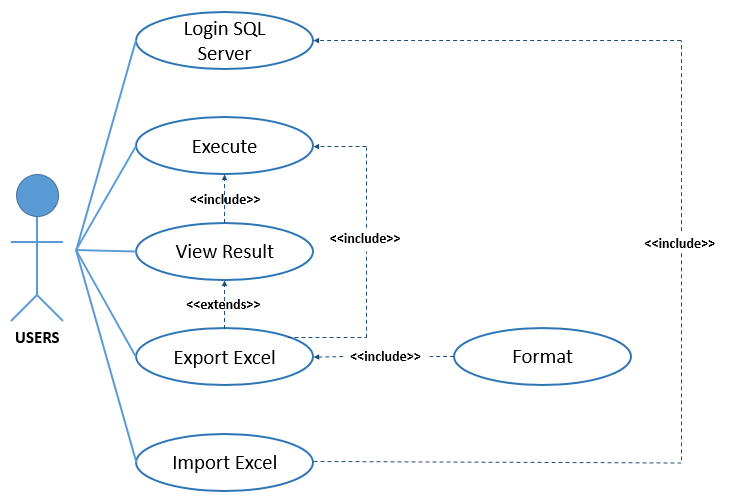
* Our application is a tool which helps users to export database from Microsoft SQL Server into Excel file and import data from Microsoft Excel into Microsoft SQL Server.
* Design user interface easy to use, professional.
  1. **User requirement**
* The application provide 3 main features:
* Users can quickly view any data table in SQL Server by executing SELECT command.
* Export database from Microsoft Server into Excel file.
* Import data from Microsoft Excel into Microsoft SQL Server
* Moreover, you can choose format after exported with many option like:
* Alignment
* Font
* Size
* Fill
* Name of sheet
* Border
  1. **Usecase diagram**

Figure 1: Usecase diagram

* 1. **Usecase description tables**

Table 1: Usecase Log in description

|  |  |  |  |
| --- | --- | --- | --- |
| Use case name | Log in | | |
| Description | Allows user to connect to SQL Server Database Management. | | |
| Actor | User | | |
| Business event | No. | Agent | System response |
|  | 1 | Access the application |  |
|  | 2 |  | Give login interface |
|  | 3 | Input information to connect to SQL Server |  |
|  | 4 | Click “Connect” button |  |
| Preconditions | Have installed DBMS Microsoft SQL Server | | |
| Condition affecting termination outcome | When connection succeeded and User click “Connect” button, show main interface.  When connection failed and show error. | | |

Table 2: Usecase Execute description

|  |  |  |  |
| --- | --- | --- | --- |
| Use case name | Execute | | |
| Description | Execute SELECT command to get data from database | | |
| Actor | User | | |
| Business event | No. | Agent | System response |
|  | 1 | Choose database name |  |
|  | 2 | Input SELECT command |  |
|  | 3 | Click “Execute” button |  |
| Preconditions | User logged in successfully and have to input SELECT command | | |
| Condition affecting termination outcome | When users don’t input any query, system will request them have to input query and show notification.  When the command isn’t “SELECT” command, system shows error report.  When the table which was accessed isn’t in database, system show error report.  When users choose database name and input correct query, data view will be showed. | | |

Table 3: Usecase View Result description

|  |  |  |  |
| --- | --- | --- | --- |
| Use case name | View Result | | |
| Description | View data table which was returned after execute query | | |
| Actor | User | | |
| Business event | No. | Agent | System response |
|  | 1 |  | Show data view |
| Preconditions | User logged in successfully and execute “SELECT” command. | | |
| Condition affecting termination outcome | When correct query was executed, data view will be showed. | | |

Table 4: Usecase Export Excel description

|  |  |  |  |
| --- | --- | --- | --- |
| Use case name | Export Excel | | |
| Description | Export data table from SQL Server into Excel file | | |
| Actor | User | | |
| Business event | No. | Agent | System response |
|  | 1 | Click “Export” button |  |
|  | 2 |  | Give Format form |
|  | 3 | Choose the options of format and click “OK” button or click “Cancel” button if not choose, or exit form. |  |
|  | 4 |  | Show Save file Dialog |
|  | 5 | Input your file name, choose Excel version, click “Save” button. |  |
|  | 6 |  | Show result box |
| Preconditions | User logged in successfully and have to execute query | | |
| Condition affecting termination outcome | If data was exported, system will show successful notification.  If data can not be exported, system will show unsuccessful notification. | | |

Table 5: Usecase Format description

|  |  |  |  |
| --- | --- | --- | --- |
| Use case name | Format | | |
| Description | Allow users to choose the options of file’s format after exported. | | |
| Actor | User | | |
| Business event | No. | Agent | System response |
|  | 1 | Choose Alignment tab to modify alignment. |  |
|  | 2 | Choose Font tab to modify font name, font style, font size and text color | Give the sample in RichTextBox |
|  | 3 | Choose Size tab to modify width, height of header and cells |  |
|  | 4 | Choose Fill tab to modify background color of header and cells | Give the sample in RichTextBox |
|  | 5 | Choose SheetName tab to input name of sheet |  |
|  | 6 | Choose border tab to modify color, style and weight of border |  |
|  | 7 | Click “OK” button | Show Save file Dialog |
|  | 8 | Click “Cancel” button | This form is close and Save file dialog will be showed |
| Preconditions | User logged in successfully, executed query and clicked “Export” button | | |
| Condition affecting termination outcome | If users modify the option of format, this change will be updated.  But not modify, format will be default. | | |

Table 6: Usecase Import description

|  |  |  |  |
| --- | --- | --- | --- |
| Use case name | Import | | |
| Description | Allows users to import data from Excel into SQL Server database | | |
| Actor | User | | |
| Business event | No. | Agent | System response |
|  | 1 | Choose database name |  |
|  | 2 | Click “Import” button |  |
|  | 3 |  | Give table name panel interface |
|  | 4 | Choose table name |  |
|  | 5 | Click “Cancel” button | Table name panel is close |
|  | 6 | Click “OK” button | Show Open file Dialog |
|  | 7 | Choose the path of file which you want to import |  |
|  | 8 | Click “OK” button |  |
|  | 9 |  | Show result box |
| Preconditions | User logged in successfully | | |
| Condition affecting termination outcome | If data was imported, system will show successful notification.  If data can not be imported, system will show unsuccessful notification, the cause may be: the file which was imported is not excel file, the file which was imported has columns different from the columns of the table in database. | | |

1. ***Software design***
   1. **Technology**

We use ADO.NET Technology for our application. ADO.NET is a [data access](https://en.wikipedia.org/wiki/Data_access) technology from the [Microsoft](https://en.wikipedia.org/wiki/Microsoft) [.NET Framework](https://en.wikipedia.org/wiki/.NET_Framework) that provides communication between relational and non-relational systems through a common set of components. ADO.NET is a set of computer software components that programmers can use to access data and data services from a database. It is a part of the [base class library](https://en.wikipedia.org/wiki/Base_Class_Library) that is included with the Microsoft .NET Framework. It is commonly used by programmers to access and modify data stored in [relational database systems](https://en.wikipedia.org/wiki/Relational_DBMS), though it can also access data in non-relational data sources.

The data provider is a set of ADO.NET classes that allow access to a particular database, execute queries, and retrieve data. Basically, a data provider is the bridge between the application and a data source.

A data provider include:

* Connection: Establish a connection with data source.
* Command: Execute query and store procedure
* DataReader: Provide a very efficient way to sequentially read data from a data source.
* DataAdapter: There are 2 ways to use. The first is to dump the data from the database into DataSet. The other way is to update the data from the DataSet and synchronize those changes to the database.

ADO.NET include 3 data provider:

* SQL Server Provider.
* OLE DB Provider.
* Oracle Provider.

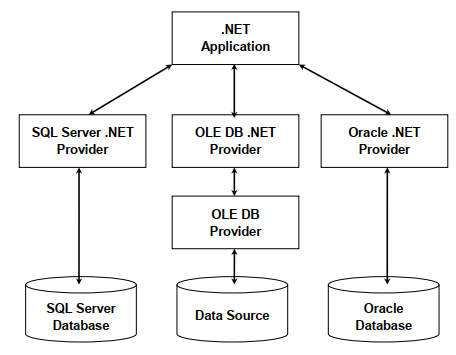


Figure 2: ADO.NET architecture

In this application, we use SQL Server Provider. SQL Server Provider include:

* SqlConnection
* SqlCommand
* SqlDataReader
* SqlDataAdapter

***SqlConnection Class***

SqlConnection Object is used to connect to a database in SQL Server.

Create SqlConnection Object:

SqlConnection cnn =  
new SqlConnection("server=localhost;database=Northwind;uid=sa;pwd=sa");

* server: Name of computer which was installed SQL Server
* database: Name of database which you want to connect
* uid: Name of database user
* pwd: Password of user

***SqlCommand Class***

Sql Command Object is used to execute a query or store procedure on the server.

Create SqlCommand Object:

* Execute a query

SqlCommand cmd = new SqlCommand();  
cmd.Connection = cnn;

cmd.CommandType = CommandType.Text;  
cmd.CommandText = "SELECT \* FROM Employees";

* Execute a store procedure

SqlCommand cmd = new SqlCommand("GetEmployees", cnn);  
cmd.CommandType = CommandType.StoredProcedure;

or:

SqlCommand cmd = cnn.CreateCommand();  
cmd.CommandType = CommandType.StoredProcedure;  
cmd.CommandText = "GetEmployees";

The methods of SqlCommand:

* **ExecuteNonQuery():** Executes the query, and does not collect any results. Generally used for queries such as INSERT, UPDATE, DELETE, CREATE, ALTER, DROP.
* **ExecuteScalar():** Executes the query, and returns a single value (from the first column of the first row).
* **ExecuteReader():** Executes the query, and returns a SqlDataReader object. When we're retrieving multiple rows and columns of data. In this case, we use the ExecuteReader method, which returns an instance of a SqlDataReader object.

***SqlDataReader Class***

SqlDataReader Object allows to reads database rows one-by-one. It reads in forward order from a SQL database.

The methods of SqlDataReader:

* **Read():** Use the Read method of the DataReader object to obtain a row from the results of the query. You can access each column of the returned row by passing the name or ordinal reference of the column to the DataReader.
* **GetInt32(), GetChar(), GetDateTime, Get...():** Get value of a column (by index) of current row, return data type as method name.
* **NextResult():** If the data returned from the SQL statement has more than two result sets, this method will move the DataReader to the next result set.
* **Close():** Close DataReader.
  1. **Software architecture**

In this project, we decided to use three-layer architecture to build application.

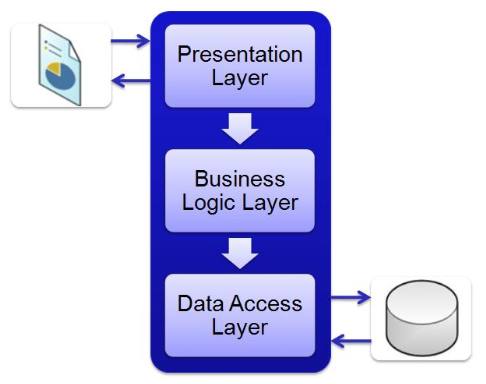


Figure 3: Three-layer architecture

Three-layer model consists three main parts:

* Presentation Layer (GUI): This layer is responsible for communicating with users. It consists of interface elements (winform, webform,…) and performs tasks such as data entry, data display, data validation before calling the Business Logic Layer (BLL).
* Business Logic Layer (BLL): This layer consists 2 tasks:
* This is where the GUI layer’s data manipulation requirements are processed, processing the data source itself from the Presentation Layer before passing it to the Data Access Layer and saving it to the database management system.
* It also examines constraints, data integrity and validity, performs computation and handles business requirements, before returning a Presentation Layer result.
* Data Access Layer (DAL): This class has the function of communicating with the database management system such as performing tasks related to data storage and retrieval (search, insert, delete, modify,…).

Advantages of 3-layer architecture:

* This division into layers makes the code more explicit. Thanks to the layering of the various function, such as interface, processing and query, instead of leaving everything in place.
* Easy maintenance when segmented, one component of the system will be easy to change. This change can be isolated in a layer, or affect the last layer without affecting the whole program.
* Easy to develop, reuse: When we want to add a function, programming in a model is easier because we have the standard to follow. And reuse when there are changes between the two environments (Winform or Webform), just change the GUI layer.
* Easy to hand: If everyone is following a standard, interacting with each other will be easier and more time-consuming.
* Easy to distribute workload: Each group, one department will receive a task in three-layer model. Such clear division will help programmers control their workload.

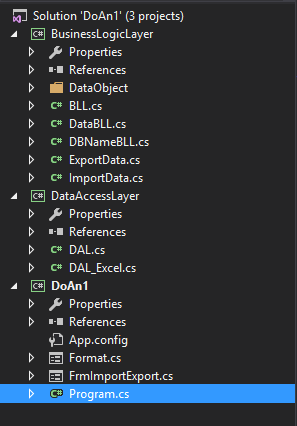


Figure 4: Apply three-layer architecture in our project

* 1. **Design User Interface**

A nice interface will provide a great user experience. However, simple manipulation is the most important.

Table 7: User Interface work assignment

|  |  |  |  |
| --- | --- | --- | --- |
| No | Interface | Designer | Purpose |
| 1 | Login to SQL Server | Ngo Van Tu | User can log in using SQL Server account. |

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | Export data from SQL Server to Excel | Pham Hoang Viet | User can choose any available database to perform to execute statement and view result in Database view |
| 3 | Format sheet Excel | Pham Hoang Viet | Provide many options for format such as: Alignment, Font, Size, Fill, SheetName, Border |
| 4 | Import data from Excel to SQL Server | Ngo Van Tu | Provide a table name panel to choose table name to import |

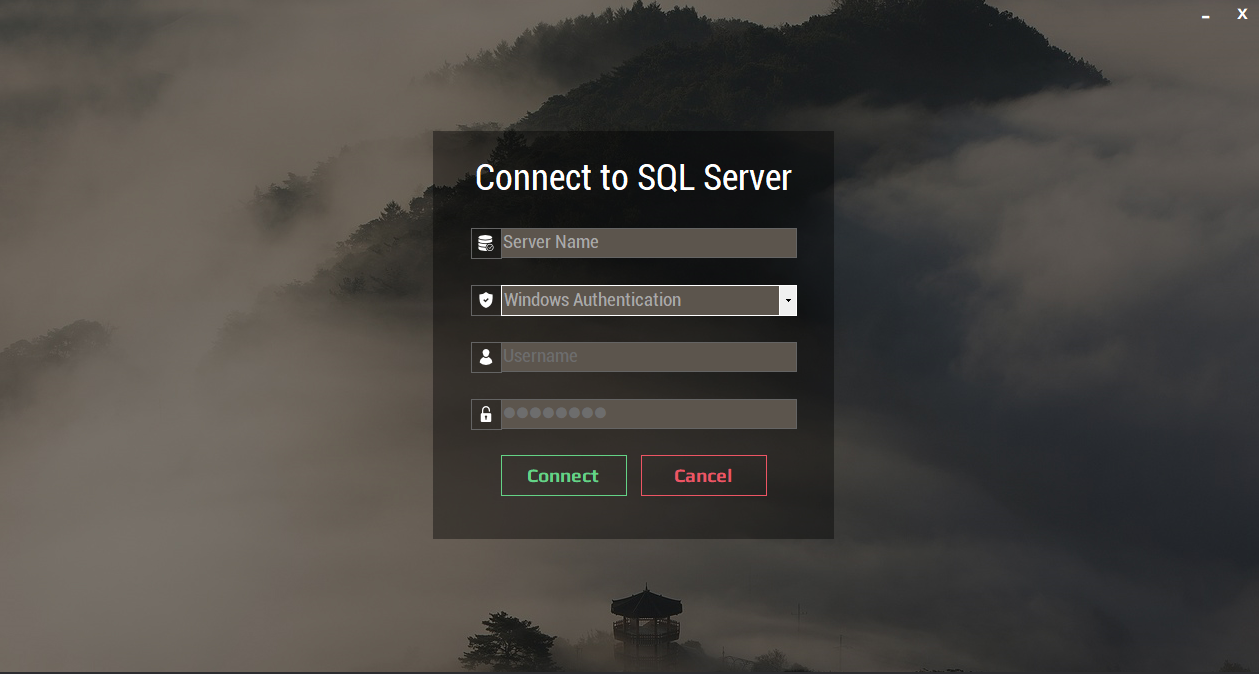
****

Figure 5: Login Interface

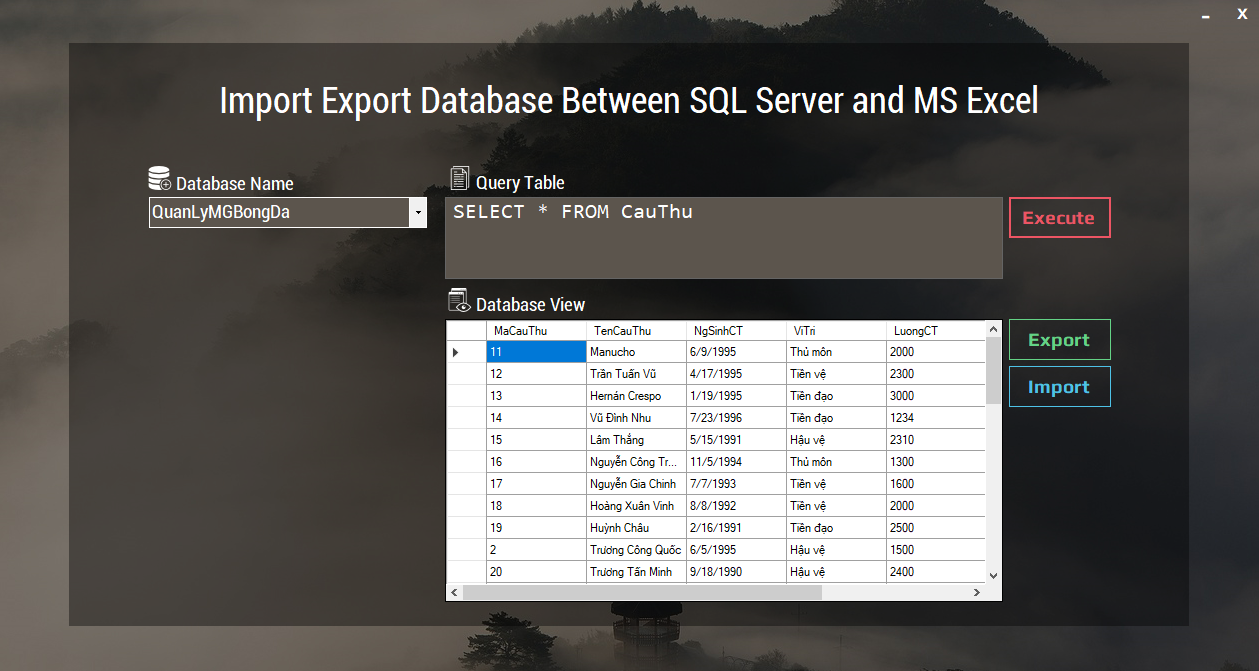


Figure 6: Export data from SQL Server to Excel Interface

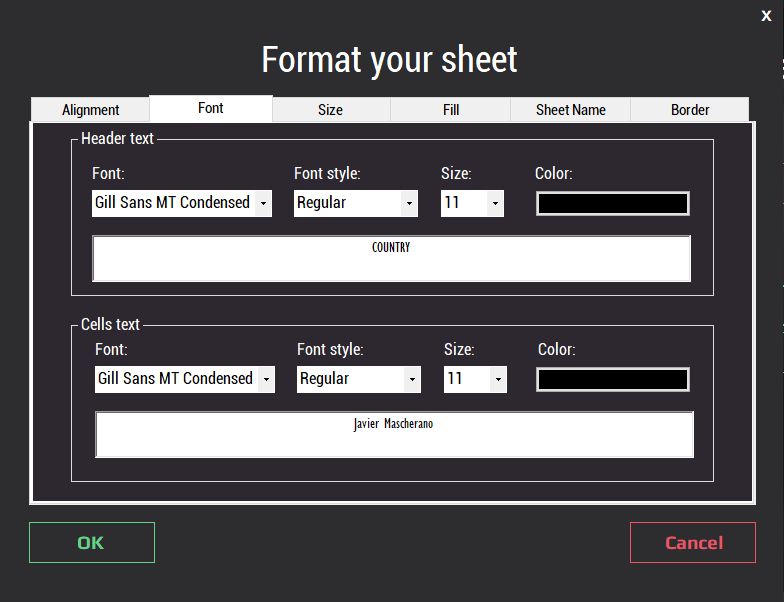


Figure 7: Format sheet Excel Interface

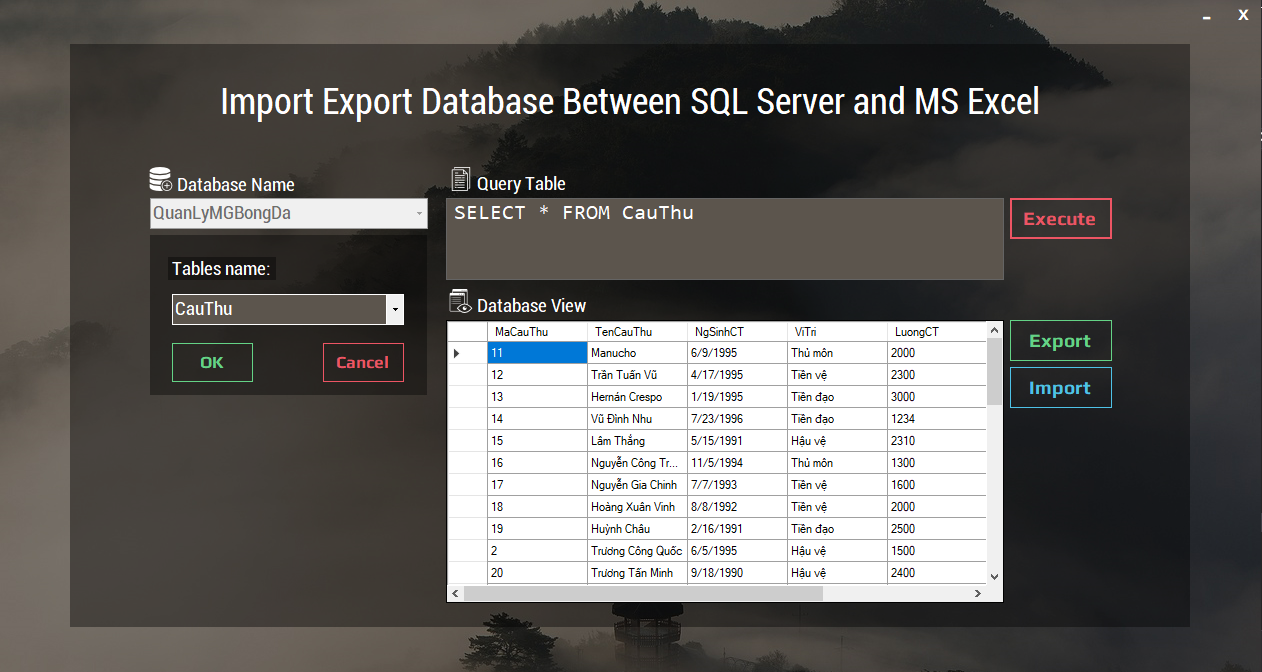


Figure 8: Import data from Excel to SQL Server Interface

* 1. **Design class**
     1. **Class diagram**

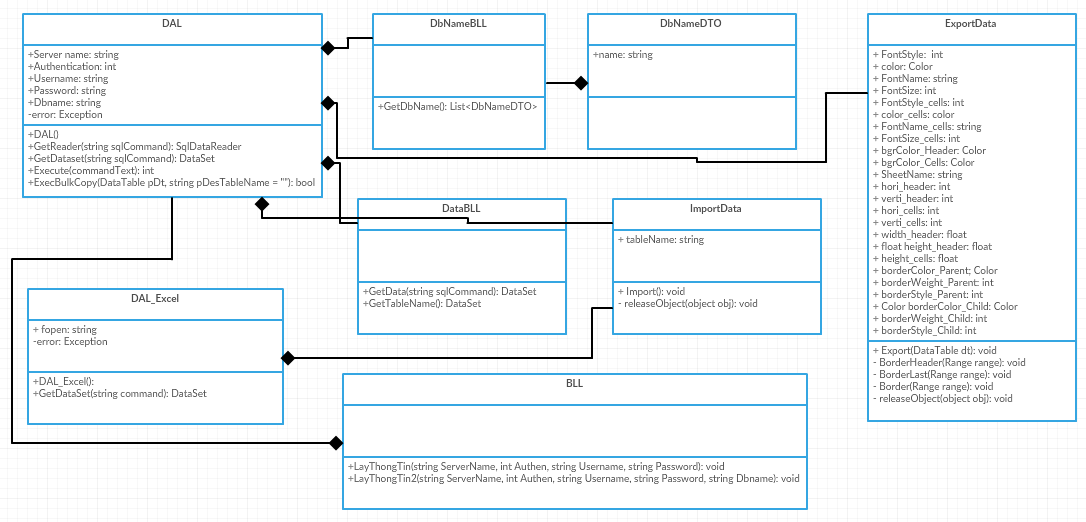
****

Figure 9: Class diagram

* + 1. **Class description tables**

**List of classes are used in the program:**

* **DataAccessLayer**

Table 8: List of classes are used in DataAccessLayer

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Class Name | Responsible | Purpose |
| 1 | DAL | Pham Hoang Viet | Connect to SQL Server by using SQL Server Provider |
| 2 | DAL\_Excel | Ngo Van Tu | Connect to Excel by using OLE DB Provider |

* **BusinessLogicLayer**

Table 9: List of classes are used in BusinessLogicLayer

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Class Name | Responsible | Purpose |
| 1 | BLL | Pham Hoang Viet | Get Information (Servername, Authentication, Username, Password) from PresentationLayer to move down DataAccessLayer |
| 2 | DataBLL | Ngo Van Tu | Get data which was asked by users, return Dataset |
| 3 | DBNameBLL | Pham Hoang Viet | Get the list of databases name in SQL Server database |
| 4 | ExportData | Pham Hoang Viet | Export data from DataTable into Excel file, provide format functions |
| 5 | ImportData | Ngo Van Tu | Import data from Excel file into database |

* **PresentationLayer**

Table 10: List of classes are used in PresentationLayer

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Class Name | Responsible | Purpose |
| 1 | FrmImportExport | Ngo Van Tu  Pham Hoang Viet | Provide Login interface and interfaces of main functions as Export, Import, Execute, View Result, Input Query, Choose Database name, Choose Table name. |
| 2 | Format | Pham Hoang Viet | Provide Format interface, include: Alignment, Font, Size, Fill, SheetName, Border |

**List of methods in classes:**

* **DataAccessLayer**
* **Methods in DAL**

Table 11: List of methods in DAL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **DAL()**  Input: none.  Output: none.  Pseudo code: none. | Create ConnectionString to connect to SQL Server Database | DAL.cs  (33) | Pham Hoang Viet |
| 2 | **GetReader(string sqlCommand)**  Input: sqlCommand  Output: reader  Pseudo code: none | Return SqlDataReader Object by execute sqlCommand using SqlCommand Object | DAL.cs  (58) | Pham Hoang Viet |
| 3 | **GetDataSet(string sqlCommand)**  Input: sqlCommand  Output: ds  Pseudo code: none | Execute sqlCommand, return a DataSet | DAL.cs  (80) | Ngo Van Tu |
| 4 | **Execute(string commandText)**  Input: commandText  Output: Result (int)  Pseudo: none | Executes the query, and does not collect any results. Generally used for queries such as Insert, Update, Delete, Create,… Returning the number of row which were affected | DAL.cs  (104) | Pham Hoang Viet |
| 5 | **ExecBulkCopy(DataTable pDt, string pDesTableName)**  Input: pDt, pDesTableName  Output:true or false  Pseudo code: none | Copy data in DataTable into a table in SQL Server database | DAL.cs  (130) | Ngo Van Tu |

* **Methods in DAL\_Excel**

Table 12: List of methods in DAL\_Excel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **DAL\_Excel()**  Input: none.  Output: none.  Pseudo code: none. | Create ConnectionString to connect to Microsoft Excel using OLE DB provider | DAL\_Excel.cs  (23) | Ngo Van Tu |
| 2 | **GetDataSet(string command)**  Input: string command  Output: ds (DataSet)  Pseudo code: none | Execute command statement, get data in Excel and fill in  DataSet | DAL\_Excel.cs  (27) | Pham Hoang Viet |

* **BusinessLogicLayer**
* **Methods in BLL**

Table 13: List of methods in BLL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **LayThongTin(string ServerName, int Authen, string Username, string Password )**  Input: ServerName, Authen, Username, Password  Output: none.  Pseudo code: none. | Get informations as ServerName, Authen, Username, Password which were inputted by user and move down DataAccessLayer to create ConnnectionString | BLL.cs  (18) | Pham Hoang Viet |
| 2 | **LayThongTin2(**  **string ServerName, int Authen, string Username, string Password, string Dbname )**  Input: ServerName, Authen, Username, Password, Dbname  Output: none.  Pseudo code: none. | Get informations as ServerName, Authen, Username, Password, Dbname which were inputted by user and move down DataAccessLayer to create ConnnectionString | BLL.cs  (27) | Ngo Van Tu |

* **Methods in DataBLL**

Table 14: List of methods in DataBLL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **GetData(string sqlCommand)**  Input: sqlCommand  Output: DataSet.  Pseudo code: none. | Get data which was asked by user and return a DataSet | DataBLL.cs  (17) | Pham Hoang Viet |
| 2 | **GetTableName()**  Input: none  Output: DataSet.  Pseudo code: none. | Get table name of specific database, and return a DataSet | DataBLL.cs  (22) | Ngo Van Tu |

* **Methods in DBNameBLL**

Table 15: List of methods in DBNamBLL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **GetDbName()**  Input: none  Output: listDbName (List<DbNameDTO>)  Pseudo code: none. | Get database name and fill in DataReader, after that, fill in list of database name, return listDbName | DBNameBLL.cs  (16) | Ngo Van Tu |

* **Methods in ExportData**

Table 16: List of methods in ExportData

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **Export(DataTable dt)**  Input: dt  Output: none  Pseudo code: none. | Export data in DataTable into Excel, provide format excel file function | ExportData.cs  (47) | Pham Hoang Viet |
| 2 | **BorderHeader(**  **Range range)**  Input: range  Output: none  Pseudo code: none | Create border for header | ExportData.cs  (268) | Pham Hoang Viet |
| 3 | **BorderLast(**  **Range range)**  Input: range  Output: none  Pseudo code: none | Create border for the last row | ExportData.cs  (292) | Pham Hoang Viet |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | **Border(**  **Range range)**  Input: range  Output: none  Pseudo code: none | Create border for the other rows | ExportData.cs  (326) | Pham Hoang Viet |
| 5 | **releaseObject(**  **object obj)**  Input: obj  Output: none  Pseudo code: none | Release objects which were created | ExportData.cs  (371) | Pham Hoang Viet |

* **Methods in ImportData**

Table 17: List of methods in ImportData

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **Import()**  Input: none  Output: none  Pseudo code: none. | Get data from Microsoft Excel, fill in DataSet, after that, copy data from DataSet into SQL Server Database | ImportData.cs  (18) | Ngo Van Tu |
| 2 | **releaseObject(**  **object obj)**  Input: obj  Output: none  Pseudo code: none | Release objects which were created | ImportData.cs  (46) | Ngo Van Tu |

* **PresentationLayer**
* **Methods in FrmImportExport**

Table 18: List of methods in FrmImportExport

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **FrmImportExport\_Load(object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none. | Show Login panel | FrmImport  Export.cs  (27) | Ngo Van Tu  Pham Hoang Viet |
| 2 | **btnConnect\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Call LayThongTin() method in BLL, show list of database name which was returned on combobox | FrmImport  Export.cs  (54) | Pham Hoang Viet |
| 3 | **btnCancel\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Show nofitication, close form | FrmImport  Export.cs  (300) | Pham Hoang Viet |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | **btnExecute\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Call LayThongTin2() method in BLL, show datatable which was returned on datagridview | FrmImport  Export.cs  (309) | Pham Hoang Viet |
| 5 | **btnExport\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Show Format form, call LayThongTin2() method in BLL, call GetData() method in DataBLL to get data and fill in dataset. After that, call Export() method in ExportData to export data from dataset into Excel. | FrmImport  Export.cs  (404) | Pham Hoang Viet |
| 6 | **btnImport\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Show table name panel, call LayThongTin2() method in BLL, call GetTableName() method in DataBLL to get list of table name of the specific database. After that, show it on combobox. | FrmImport  Export.cs  (443) | Ngo Van Tu |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | **btnOK\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Call Import() method in ImportData class to import data from Excel into SQL Server database. | FrmImport  Export.cs  (465) | Ngo Van Tu |

* **Methods in Format**

Table 19: List of methods in Format

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Method | Purpose | File name,  Line | Responsible |
| 1 | **Format\_Load(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Create list of options of format like: font, size, style, alignment, weight,… | Format.cs  (92) | Ngo Van Tu |
| 2 | **FormatHeader()**  Input: none  Output: none  Pseudo code: none | Create the samples to show on RichTextBox for Header. | Format.cs  (184) | Pham Hoang Viet |
| 3 | **FormatCells()**  Input: none  Output: none  Pseudo code: none | Create the samples to show on RichTextBox for Cells. | Format.cs  (219) | Pham Hoang Viet |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | **btnOK\_Click(**  **object sender, EventArgs e)**  Input: sender, e  Output: none  Pseudo code: none | Call SaveFormat() method and close this form. | Format.cs  (317) | Ngo Van Tu |
| 5 | **SaveFormat(string fontName, int fontStyle, int fontSize, Color color, string fontName\_cells, int fontStyle\_cells, int fontSize\_cells, Color color\_cells,**  **Color bgrColor\_Header, Color bgrColor\_Cells,**  **string sheetName,**  **int hori\_header, int verti\_header, int hori\_cells, int verti\_cells, float width\_header, float height\_header, float height\_cells,**  **Color borderColor\_ Parent, int borderWeight\_ Parent, int borderStyle\_Parent, Color borderColor\_Child, int borderWeight\_**  **Child, int borderStyle\_Child)**  Input: fontName, fontStyle, fontSize, color, fontName\_Cells, fontStyle\_Cells, fontSize\_Cells, color\_Cells, bgrColor\_Header, bgrColor\_Cells, sheetName, hori\_header, verti\_header, hori\_cells, verti\_cells, width\_header, height\_header, height\_cells,  borderColor\_Parent,  borderWeight\_  Parent,  borderStyle\_Parent,  borderColor\_Child,  borderWeight\_  Child,  borderStyle\_Child,  Output: none  Pseudo code: none | Get values which were inputted by users, move down ExportData Class to process. | Format.cs  (355) | Pham Hoang Viet |

* 1. **Design database**

All available databases in Microsoft SQL Server.

* 1. **Configuration**

Your computer have to be installed Microsoft SQL Server and Microsoft Excel.

1. ***Project Implementation***
   1. **Environment**

- Programming language: C#.  
- NET Framework: Microsoft .NET Framework v4.6.1.  
- Database Management System: SQL Server.

* Integrated Development Environment: Visual Studio 2017.
  1. **Work plan**

Table 20: Work plan

|  |  |  |
| --- | --- | --- |
| Student’s name | Evaluate contribution | Task work |
| Ngo Van Tu  Pham Hoang Viet | 100% | Design UI |
| Ngo Van Tu  Pham Hoang Viet | 100% | Design Class |
| Ngo Van Tu  Pham Hoang Viet | 100% | Coding |
| Ngo Van Tu  Pham Hoang Viet | 100% | Test app and Debug |
| Ngo Van Tu  Pham Hoang Viet |  | Write Report |

* 1. **Work assignment**

Table 21: Work assignment

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No. | Goal | Schedule | | | | | | | |
| 1 | Describe the requirements of the project. | **o** | **o** |  |  |  |  |  |  |
| 2 | Use case diagram | **o** | **o** |  |  |  |  |  |  |
| 3 | Design class |  | **o** | **o** | **o** |  |  |  |  |
| 4 | Learn about the functions to work with Excel in ADO.NET |  | **o** | **o** |  |  |  |  |  |
| 5 | Design UI |  | **o** | **o** | o |  |  |  |  |
| 6 | Implement UI |  |  | **o** | **o** |  |  |  |  |
| 7 | Implement classes |  |  | **o** | **o** | **o** |  |  |  |
| 8 | Connect SQL Server Database and Fill combobox with Databases’s name |  |  | **o** | **o** | **o** |  |  |  |
| 9 | Execute SQL command in Textbox and show on Datagridview |  |  | **o** | **o** | **o** |  |  |  |
| 10 | Export data |  |  | **o** | **o** | **o** |  |  |  |
| 11 | Format excel file after exported |  |  |  | **o** | **o** | **o** |  |  |
| 12 | Import data |  |  |  | **o** | **o** | **o** |  |  |
| 13 | Fill in TreeView with Database and Export multiple tables into a Excel file |  |  |  |  |  |  |  |  |
| 14 | Testing. |  |  |  |  |  | **o** | **o** |  |
| 15 | Report. |  |  |  |  |  |  | **o** |  |
| Day | | 02/11/2018 | 09/11/2018 | 16/11/2018 | 23/11/2018 | 30/11/2018 | 07/12/2018 | 14/12/2018 | 21/12/2018 |
| Week | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Note | | **o** – Begin  **o** – Complete 50%  **o** – Complete 100% | | | | | | | |

* 1. **Manual**
     1. **Manual Login Form**

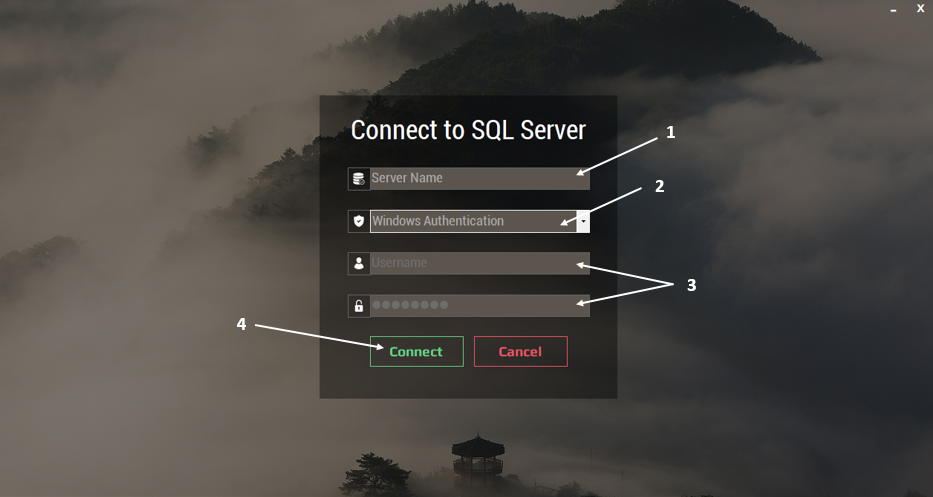
****

Figure 10: Manual Login Form

Note: Here, we will use account login SQL Server.

* Step 1: Type server name (computer name).
* Step 2: Select privilege login. Here, include 2 privilege:

+ Windows Authentication: If user select this privilege, you don’t need perform step 3. The end, select step 4.

+ SQL Server Authentication: If user select this privilege, you need perform step 3. The end, select step 4.

* Step 3: Type your username and type your password.
* Step 4: Connect.
  + 1. **Manual Export data from SQL Server to Excel Form**

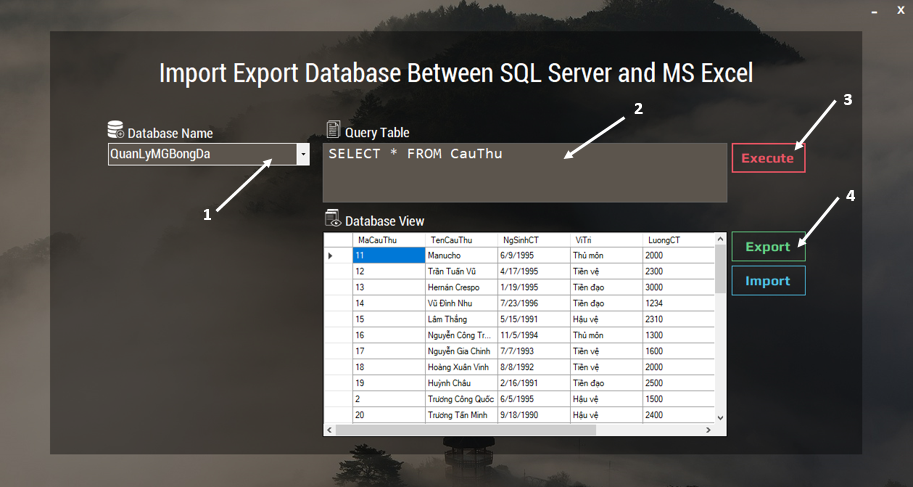


Figure 11: Manual Export data from SQL Server to Excel Form

* Step 1: Select Database Name.
* Step 2: Type the statement in query table.
* Step 3: Select Execute, data after perform Step 2 will appear in Database View.
* Step 4: Select Export, Format form interface is appear.

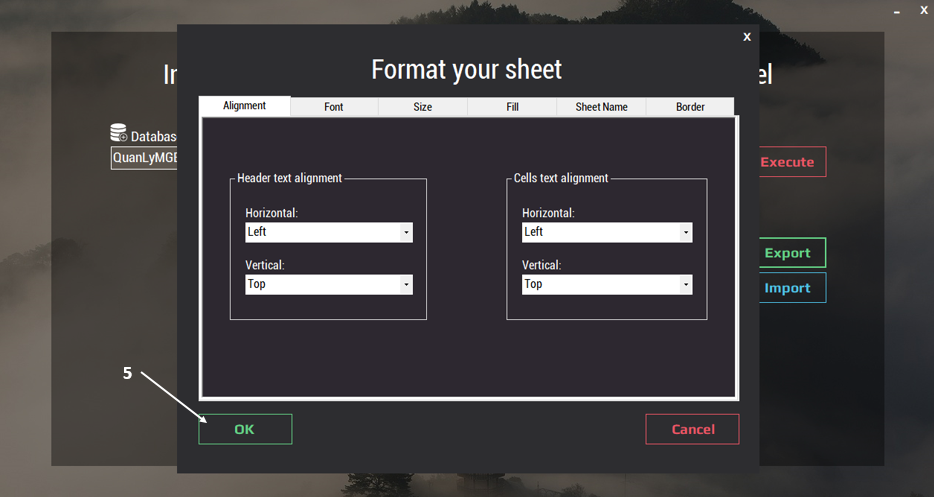


Figure 12: Manual Format Form

+ If user don’t modify any options in Format form, the application will initialize default value.

+ User can modify format of sheet after exported with many options in tabs such as: Alignment, Font, Size, Fill, SheetName, Border. After modifying format of sheet , user perform Step 5.

* Step 5: Select OK and Save file Dialog will appear.
  + 1. **Manual Import data from Excel to SQL Server**

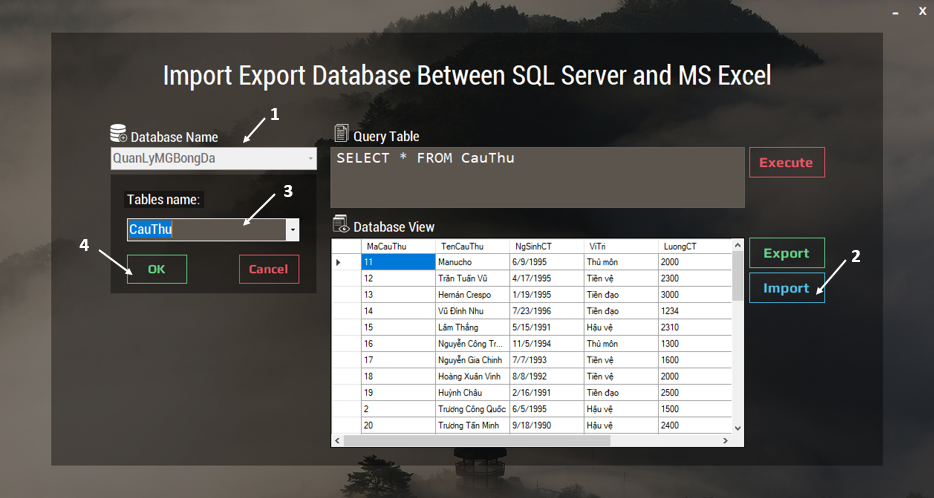
****

Figure 13: Manual Import data from Excel to SQL Server

* Step 1: Select database name.
* Step 2: Select Import, after select import, tables name appear.
* Step 3: Select table name need import.
* Step 4: Select OK and Open file Dialog will appear.

1. ***Conclusion***
   1. **Evaluation**

* Satisfying 90% project of teachers' instructional requirements.
* The application is modeled three layers in the projectand integration with Object Oriented Programming.
* Can using for many version Microsoft Excel from 2003 to current.
  1. **Difficulties**
* Not meet some requirements of the project.
* Not create a tree view load database name and load tables when connected SQL Server.
* Apply not good three layers model and Object Oriented Programming in project.
  1. **Solutions**
* Review the knowledge Windows Programming, Object Oriented Programming, Database, …
* Learn and research concept on internet.
  1. **Advantages**
* Support for users up data to Excel form SQL Server.
* User interface simple and easy using
* Permission user format sheet before export data from SQL Server to Excel.
  1. **Defect**
* Manipulation operate of the application is not good.
* Can not export data from SQL Server to excel at many table.
* Unformatted data type when export data from SQL Server to Excel.
* Can not import data if that data existed in SQL Server.
* Slow speed when export if large data.
  1. **Development ideas**
* Try optimize application.
* Overcome defect.
* Design many better new function
* Using many database management system.
  1. **Source code**

*https://github.com/hoangviet290398/Project1?fbclid=IwAR2ulKW8BSZw6SZy2P7Z\_ClgYAk0MnRv4BSQRXLr-wIj2HbGOyXh6HOOqnw*

# **REFERENCE**

*[1] Khoan Nguyen. "ADO NET Tutorial 1." Internet: https://www.youtube.com/watch?v=sDhuKpcoepk&t=2202s&list=PLZrXMCpcckDX\_sApCzQ\_btfIaWSB3qzLO&index=2, Aug. 17, 2018.*

*[2] Khoan Nguyen. "ADO NET Tutorial 2." Internet: https://www.youtube.com/watch?v=QSwr98p8ohk&index=2&list=PLZrXMCpcckDX\_sApCzQ\_btfIaWSB3qzLO, Aug. 17, 2018.*

*[3] huatrung. "MÔ HÌNH 3 LỚP (3 – LAYER) CÓ GÌ HAY ?" Internet: https://techtalk.vn/mo-hinh-3-lop-co-gi-hay.html, Mar. 22, 2017.*

*[4] Dương @le.anh.duong. "Thao tác với file Excel." Internet: https://viblo.asia/p/thao-tac-voi-file-excel-pVYRPj4lG4ng, Nov. 30, 2015.*

*[5] Trần Duy Thanh. "C#-Xuất dữ liệu SQL Server ra Excel để báo cáo." Internet: https://duythanhcse.wordpress.com/2018/08/13/c-xuat-du-lieu-sql-server-ra-excel-de-bao-cao/, Aug. 13, 2018.*

*[6] Mr. Cùi Bắp. "Import dữ liệu từ Excel và cập nhật lên SQL." Internet: https://laptrinhvb.net/bai-viet/chuyen-de-csharp/Import-du-lieu-tu-Excel-va-cap-nhat-len-SQL/9d66e044fe280399.html, Oct. 26, 2015.*