Fakulti Sains Komputer dan Teknologi Maklumat
Universiti Tun Hussein Onn Malaysia

LABORATORY 7

This laboratory exercise is about ASP.NET connection to Microsoft SQL Server database.

A. Setting Up

(a) Database Installation

We will be using Microsoft SQL Server Express Edition 2017 in this lab. Those who have not downloaded and installed the software, you may download it from here:

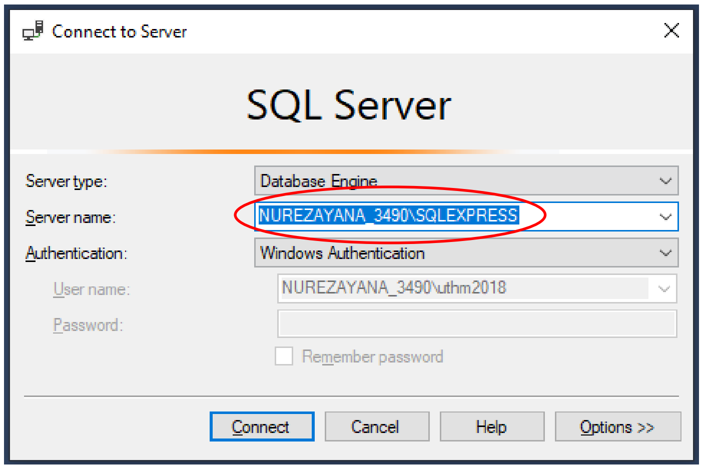
- SQL Server 2017 Express

This version of SQL Server is free and includes SQL Server Management Studio while more current versions may split between the Management Studio and the SQL Server itself.

*During the installation, make sure that you pick LocalDB in Feature Selection. Once done, you may open SQL Server Management Studio and it is ready to use.

(b) Using Microsoft SQL Server 2017

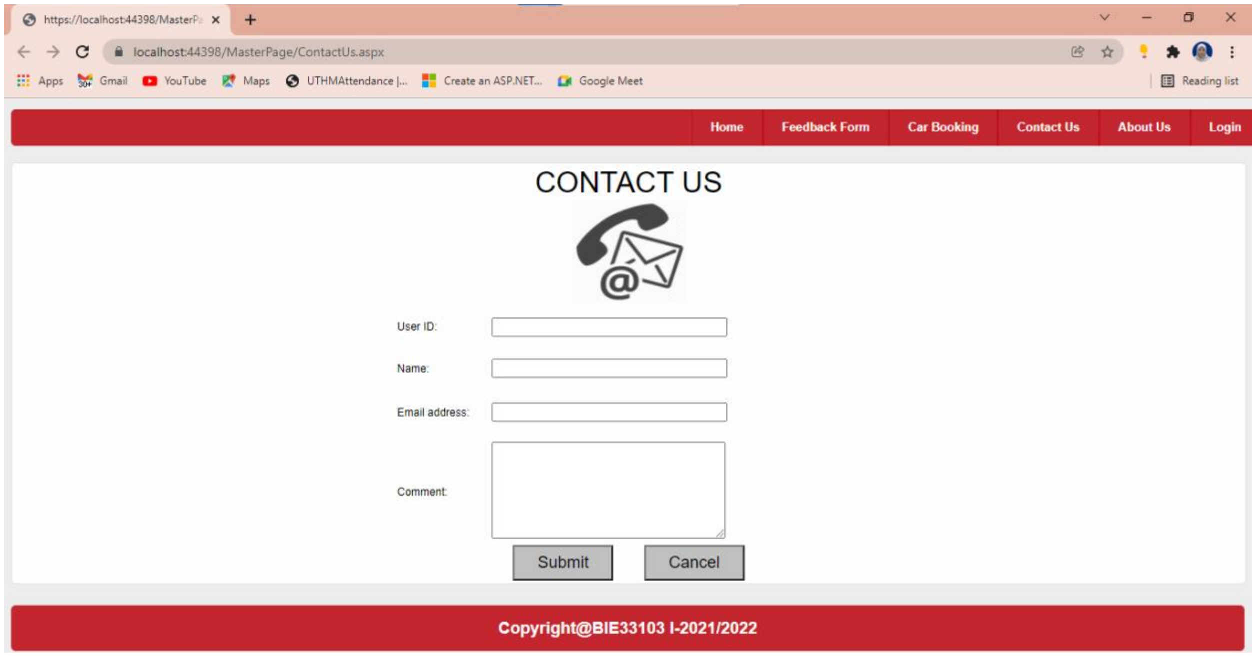
One basic thing that we must do on Microsoft SQL Server software is to name our *server*.



Server type: Database Engine
Server name: NUREZAYANA_3490\SQLEXPRESS
Authentication: Windows Authentication
User name: NUREZAYANA_3490\uthm2018
Password:
Remember password

B. Creating, connecting to database and accessing the database

The following page is taken from Laboratory 5. Assume that, user will key in data into the form and upon clicking the Submit button, data will be stored into database.

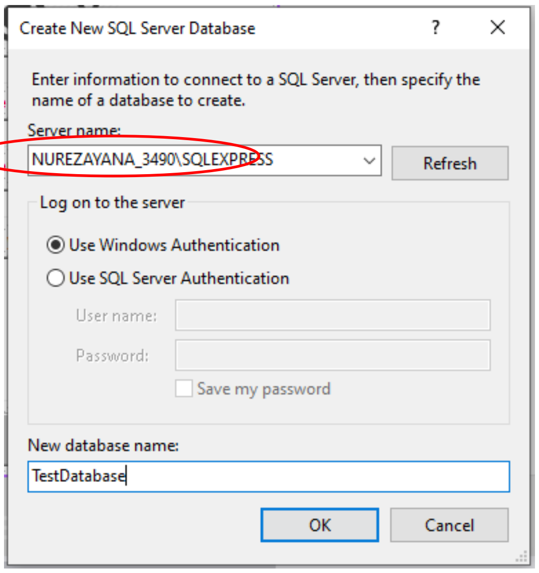


(a) Creating a database

Let's create a database name *TestDatabase* and a table inside the database name *contactus.*

There are few different ways to create a database across Visual Studio and Microsoft SQL Server. One way is to create it inside Visual Studio (I think this is less hassle for easier connection).

1. To create a database:
   - View Server Explorer
   - Right click on Data Connections
   - Click on Create New SQL Server Database and fill in your server name and your database name. Click OK.


Create New SQL Server Database

Enter information to connect to a SQL Server, then specify the name of a database to create.

Server name: NUREZAYANA_3490\SQLEXPRESS

Log on to the server

(●) Use Windows Authentication

( ) Use SQL Server Authentication

User name:

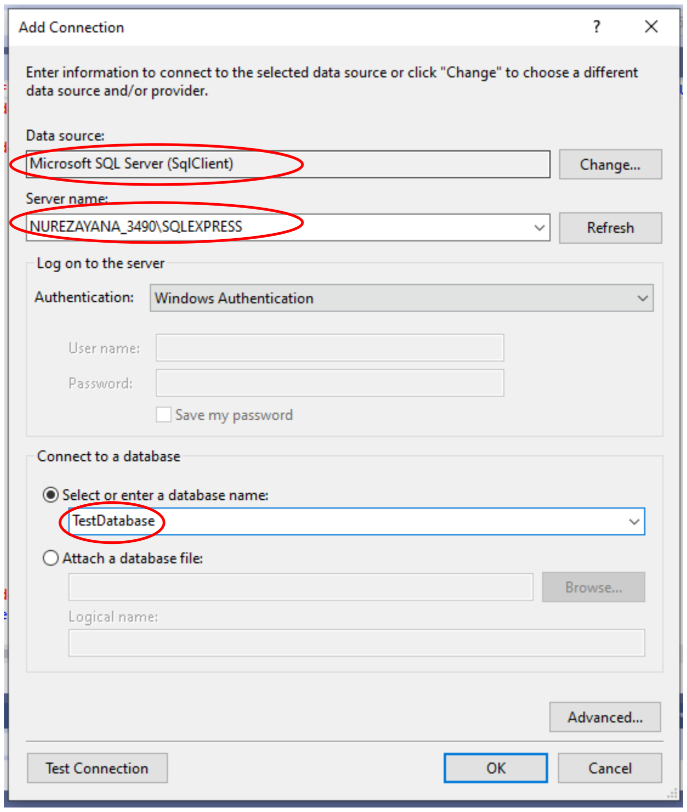
Password:

Save my password

New database name: TestDatabase

-

- Or, if you have created a new database table in Microsoft SQL Server Management Tool, you can add connection by right click on Data Connections and click on Add Connection.


Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (SqlClient)

Server name: NUREZAYANA_3490\SQLEXPRESS

Log on to the server

Authentication: Windows Authentication

User name:

Password:

Save my password

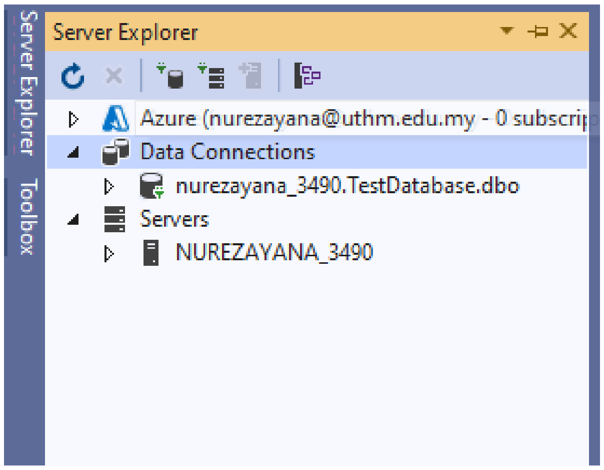
Connect to a database

(●) Select or enter a database name: TestDatabase

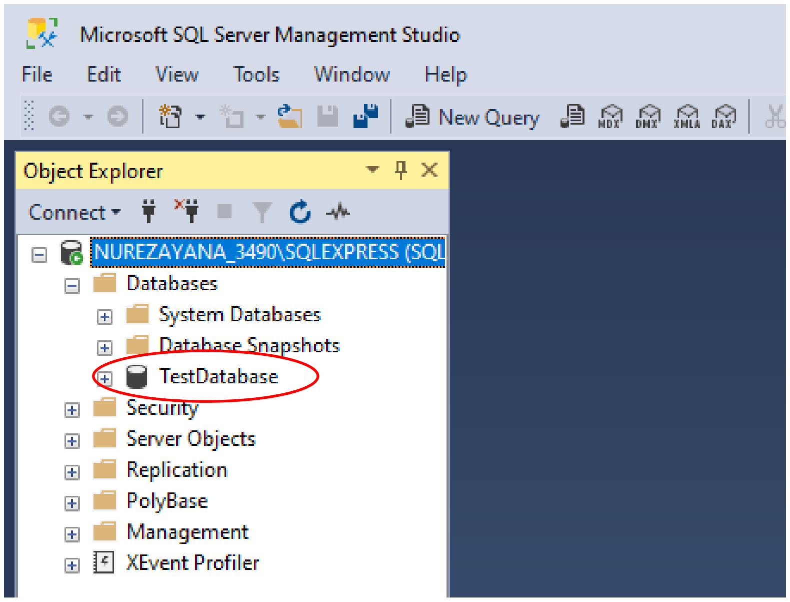
( ) Attach a database file:

Logical name:

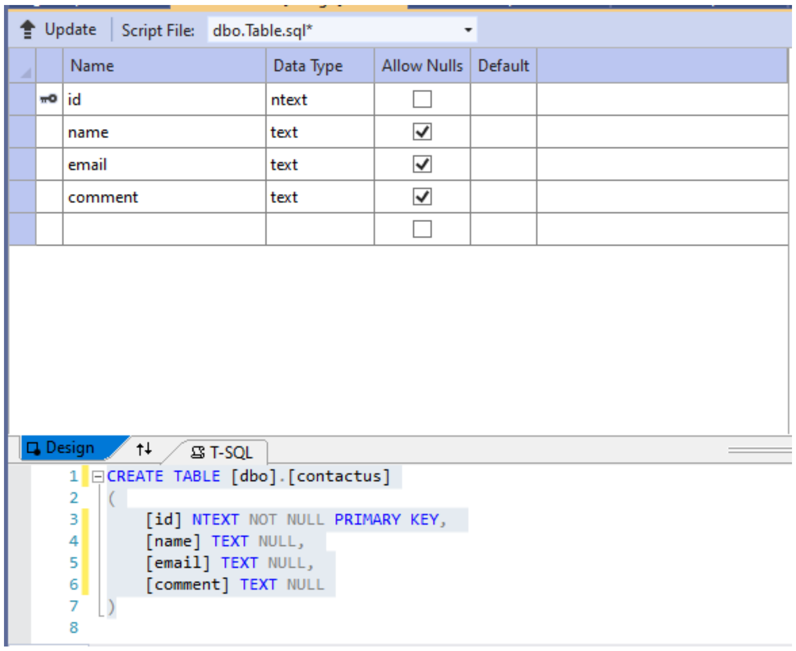
- A new database connection will be created as in the sample below:


Server Explorer
Toolbox
Azure (nurezayana@uthm.edu.my - 0 subscrip
Data Connections
nurezayana_3490.TestDatabase.dbo
Servers
NUREZAYANA_3490

2. Now you can switch alternately between Visual Studio and Microsoft SQL Server Management Tool to manage the database.
If we go back to Microsoft SQL Server Management Tool, by clicking on Databases, we can view our created database just now.



3. Next, right click on Tables → Table, we can already create a new table as below. Save the table as *contactus.*



Name	Data Type	Allow Nulls	Default
id	ntext	☐	
name	text	✓	
email	text	✓	
comment	text	✓	
		☐	

```
CREATE TABLE [dbo].[contactus]
(
    [id] NTEXT NOT NULL PRIMARY KEY,
    [name] TEXT NULL,
    [email] TEXT NULL,
    [comment] TEXT NULL
)
```

Now, our *TestDatabase* database is ready!

(2) Database Connection

As we mention, upon clicking submit button, data will be send to the database.

```
<asp:Button ID="Button_Submit" runat="server" OnClick="ButtonSubmit_Click" Text="Submit" />
```

There are five **basic steps** for the whole thing to work:

1. Make a connection
   To make a connection with database, ADO.NET provides a class named SqlConnection. So, we will create an object of this class and will pass the connection string. So that we need to import the namespace of:

```
using System.Data;
using System.Data.SqlClient;
```

Following is the sample.

```
SqlConnection con = new SqlConnection("Data Source=.;Initial Catalog =
DatabaseConnectivity; Trusted_Connection=true;");
```

2. Open connection

```
con.Open();
```

3. Prepare Command

   To prepare a command, ADO.NET gives us a class named SqlCommand which we will use as below:

```
SqlCommand com = new SqlCommand(); // Create a object of SqlCommand
class
com.Connection = con; //Pass the connection object to Command
```

4. Execute Your Command

```
com.ExecuteNonQuery();
```

5. Close connection

```
con.Close();
```

As for our case study today, `ButtonSubmit_Click()` will contain the following code:

```
protected void ButtonSubmit_Click(object sender, EventArgs e)
    {
        string ID = txtID.Text;
        string Name = txtName.Text;
        string Email = txtemail.Text;
        string Comment = txtComment.Text;

        SqlConnection con = new SqlConnection
            ("Data Source =.\\SQLEXPRESS; Initial Catalog = TestDatabase;
             Integrated Security = True; Pooling = False");

        SqlCommand com = new SqlCommand();

        try
        {

            con.Open();

            //Response.Write("Successful\n");

            SqlDataAdapter cmd = new SqlDataAdapter();// Create a object of
SqlDataAdapter class
            cmd.InsertCommand = new SqlCommand("INSERT INTO contactUs
            VALUES (@id, @name, @email, @comment) ", con); //Pass the
            connection object to cmd


            cmd.InsertCommand.Parameters.Add("@id", SqlDbType.NText).Value = ID;
            cmd.InsertCommand.Parameters.Add("@name", SqlDbType.Text).Value = Name;
            cmd.InsertCommand.Parameters.Add("@email", SqlDbType.Text).Value = Email;
            cmd.InsertCommand.Parameters.Add("@comment", SqlDbType.Text).Value =
Comment;

            cmd.InsertCommand.ExecuteNonQuery();  //to execute the SQL command

        }
        catch (Exception ex)
        {
            Response.Write(ex.ToString());
```

Assigning values from our TextBox to a local variables ID, Name, now and Comment.

Step 1

Step 3

Step 2

Step 4

```
        }
        finally
        {
            con.Close();

        }

    }
```
Step 5

At **Step 4,** we write our SQL command to insert data into contactUs.

As you can see, this code to assign the value for specific column inside the table:

```
        cmd.InsertCommand.Parameters.Add("@id", SqlDbType.NText).Value = ID;
```

where, `@id` referring to `id` in Table contactus , while `ID` is our local variable.

****NOTE:**
At **Step 4,** there are various ways of writing the code. Another way is by writing it like this:

```
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "INSERT INTO contactUs VALUES ('"+ID+"', '"+Name+"', '"+Email+"',
    '"+Comment+"')";
```

Exercise:

1. Upon clicking Submit button, pop-up a dialog box to get confirmation whether user wants to proceed.

2. Create a table named Login for Login page. Read data from table Login to check whether ID and password that have been input are correct. You may use `SqlDataReader` to perform the exercise.

* `SqlDataReader` class is use for data reader

Instruction for submission:

- Your lab report must be in pdf.
- Copy your code program in asp.net, C# code in codebehind and screenshot the output displayed in the browser.
- Submission at **AUTHOR** (Tab Individual Activities).
- All work is to be done on an individual basis.
- Duration: 1 week only.