<http://tutoriz.com/Thread-N-TIER-LAYER-ARCHITECTURE-IN-ASP-NET-DAL-BLL-BO-PL-C>

[3 tier architecture example in asp.net with C#](http://www.aspdotnet-suresh.com/2010/05/introduction-to-3-tier-architecture-in_17.html)

By: Suresh Dasari May 17, 2010

Categories: [3-TierArchitecture](http://www.aspdotnet-suresh.com/search/label/3-TierArchitecture), [Asp.net](http://www.aspdotnet-suresh.com/search/label/Asp.net)

**Introduction**

Here I will explain about uses of 3-Tier architecture and how to create or implement 3-tier architecture for our project in asp.net

**Description**

1.    What is the use of 3-tier architecture and why we go for that architecture?

2.    First we need to know what 3-Tier architecture is.

3.    How to create 3-Tier architecture for our project?

**Uses of 3-Tier Architecture**

1.    To make application more understandable.

2.    Easy to maintain, easy to modify application and we can maintain good look of architecture.

3.    If we use this 3-Tier application we can maintain our application in consistency manner.

Basically 3-Tier architecture contains 3 layers

1.    Application Layer **or** Presentation Layer

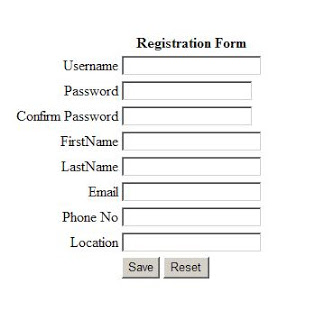
2.    Business Access Layer(**BAL**) **or** Business Logic Layer(**BLL**)

3.    Data Access Layer(**DAL**)

Here I will explain each layer with simple example that is User Registration

**Application Layer**or**Presentation Layer**

Presentation layer contains UI part of our application i.e., our **aspx** pages or input is taken from the user. This layer mainly used for design purpose and get or set the data back and forth. Here I have designed my registration aspx page like this

[](http://1.bp.blogspot.com/_B28NJpJ61hA/TNF1uxEb8DI/AAAAAAAAAOU/eOnxOWB6OLQ/s1600/UserRegistrationform.JPG)

This is Presentation Layer for our project Design your page like this and double click on button save now in code behind we need to write statements to insert data into database this entire process related to Business Logic Layer and Data Access Layer.

Now we will discuss about Business Access Layer or Business Logic Layer

**Business Access Layer**(BAL)or **Business Logic Layer** (BLL)

This layer contains our business logic, calculations related with the data like insert data, retrieve data and validating the data. This acts as a interface between Application layer and Data Access Layer

Now I will explain this business logic layer with my sample

I have already finished form design (Application Layer) now I need to insert user details into database if user click on button save. Here user entering details regarding Username, password, Firstname, Lastname, Email, phone no, Location. I need to insert all these 7 parameters to database. Here we are placing all of our database actions into data access layer (**DAL**) in this case we need to pass all these 7 parameters to data access layers.

In this situation we will write one function and we will pass these 7 parameters to function like this

String Username= **InserDetails**(string Username, string Password, string Email, string Firstname, string Lastname, string phnno, string Location)

If we need this functionality in another button click there also we need to declare the parameters like**string Username, string Password** like this rite. If we place all these parameters into one place and use these parameters to pass values from application layer to data access layer by using single object to whenever we require how much coding will reduce think about it for this reason we will create **entity layer** or **property layer**this layer comes under sub of group of our Business Logic layer

Don't get confuse just follow my instructions enough

How we have to create entity layer it is very simple

Right click on your project web application---> select add new item ----> select class file in wizard --->give name as **BEL.CS**because here I am using this name click ok

 Open the **BEL.CS**class file declare the parameters like this in entity layer

Don’t worry about code it’s very simple for looking it’s very big nothing is there just parameters declaration that’s all check I have declared whatever the parameters I need to pass to data access layer I have declared those parameters only

**BEL.CS**

|  |
| --- |
| #region Variables  /// <summary>  /// User Registration Variables  /// </summary>  private string \_UserName;  private string \_Password;  private string \_FirstName;  private string \_LastName;  private string \_Email;  private string \_Phoneno;  private string \_Location;  private string \_Created\_By;  #endregion  /// <summary>  /// Gets or sets the <b>\_UserName</b> attribute value.  /// </summary>  /// <value>The <b>\_UserName</b> attribute value.</value>  public string UserName  {  get  {  return \_UserName;  }  set  {  \_UserName = value;  }  }  /// <summary>  /// Gets or sets the <b>\_Password</b> attribute value.  /// </summary>  /// <value>The <b>\_Password</b> attribute value.</value>  public string Password  {  get  {  return \_Password;  }  set  {  \_Password = value;  }  }  /// <summary>  /// Gets or sets the <b>\_FirstName</b> attribute value.  /// </summary>  /// <value>The <b>\_FirstName</b> attribute value.</value>  public string FirstName  {  get  {  return \_FirstName;  }  set  {  \_FirstName = value;  }  }  /// <summary>  /// Gets or sets the <b>\_LastName</b> attribute value.  /// </summary>  /// <value>The <b>\_LastName</b> attribute value.</value>  public string LastName  {  get  {  return \_LastName;  }  set  {  \_LastName = value;  }  }  /// <summary>  /// Gets or sets the <b>\_Email</b> attribute value.  /// </summary>  /// <value>The <b>\_Email</b> attribute value.</value>  public string Email  {  get  {  return \_Email;  }  set  {  \_Email = value;  }  }  /// <summary>  /// Gets or sets the <b>\_Phoneno</b> attribute value.  /// </summary>  /// <value>The <b>\_Phoneno</b> attribute value.</value>  public string Phoneno  {  get  {  return \_Phoneno;  }  set  {  \_Phoneno = value;  }  }  /// <summary>  /// Gets or sets the <b>\_Location</b> attribute value.  /// </summary>  /// <value>The <b>\_Location</b> attribute value.</value>  public string Location  {  get  {  return \_Location;  }  set  {  \_Location = value;  }  }  /// <summary>  /// Gets or sets the <b>\_Created\_By</b> attribute value.  /// </summary>  /// <value>The <b>\_Created\_By</b> attribute value.</value>  public string Created\_By  {  get  {  return \_Created\_By;  }  set  {  \_Created\_By = value;  } |

Our parameters declaration is finished now I need to create Business logic layer how I have create it follow same process for add one class file now give name called **BLL.CS**. Here one point don’t forget this layer will act as only mediator between application layer and data access layer based on this assume what this layer contains. Now I am writing the following **BLL.CS**(Business Logic layer)

|  |
| --- |
| #region Insert UserInformationDetails  /// <summary>  /// Insert UserDetails  /// </summary>  /// <param name="objUserBEL"></param>  /// <returns></returns>  public string InsertUserDetails(BEL objUserDetails)  {  DAL objUserDAL = new DAL();  try  {  return objUserDAL.InsertUserInformation(objUserDetails);  }  catch (Exception ex)  {  throw ex;  }  finally  {  objUserDAL = null;  }  }  #endregion  Here if you observe above code you will get doubt regarding these  what is  BEL objUserDetails  DAL objUserDAL = new DAL(); |

and how this method comes

return objUserDAL.InsertUserInformation(objUserDetails);

Here BEL objUserDetails means we already created one class file called BEL.CS with some parameters have you got it now I am passing all these parameters to Data access Layer by simply create one object for our BEL class file

What is about these statements I will explain about it in data access layer

DAL objUserDAL = new DAL();

return objUserDAL.InsertUserInformation(objUserDetails);

this DAL related our Data access layer. Check below information to know about that function and Data access layer

**Data Access Layer**(DAL)

Data Access Layer contains methods to connect with database and to perform insert,update,delete,get data from database based on our input data

I think it’s to much data now directly I will enter into DAL

Create one more class file like same as above process and give name as **DAL.CS**

Write the following code in DAL class file

|  |
| --- |
| //SQL Connection string  string ConnectionString = ConfigurationManager.AppSettings["LocalConnection"].ToString();  #region Insert User Details  /// <summary>  /// Insert Job Details  /// </summary>  /// <param name="objBELJobs"></param>  /// <returns></returns>  public string InsertUserInformation(BEL objBELUserDetails)  {  SqlConnection con = new SqlConnection(ConnectionString);  con.Open();  SqlCommand cmd = new SqlCommand("sp\_userinformation", con);  cmd.CommandType = CommandType.StoredProcedure;  try  {  cmd.Parameters.AddWithValue("@UserName",objBELUserDetails.UserName);  cmd.Parameters.AddWithValue("@Password", objBELUserDetails.Password);  cmd.Parameters.AddWithValue("@FirstName", objBELUserDetails.FirstName);  cmd.Parameters.AddWithValue("@LastName", objBELUserDetails.LastName);  cmd.Parameters.AddWithValue("@Email", objBELUserDetails.Email);  cmd.Parameters.AddWithValue("@PhoneNo", objBELUserDetails.Phoneno);  cmd.Parameters.AddWithValue("@Location", objBELUserDetails.Location);  cmd.Parameters.AddWithValue("@Created\_By", objBELUserDetails.Created\_By);  cmd.Parameters.Add("@ERROR", SqlDbType.Char, 500);  cmd.Parameters["@ERROR"].Direction = ParameterDirection.Output;  cmd.ExecuteNonQuery();  string strMessage = (string) cmd.Parameters["@ERROR"].Value;  con.Close();  return strMessage;  }  catch (Exception ex)  {  throw ex;  }  finally  {  cmd.Dispose();  con.Close();  con.Dispose();  }  }  #endregion |

Here if you observe above functionality I am getting all the parameters by simply creating BELobjBELUserDetails. If we create one entity file we can access all parameters through out our project by simply creation of one object for that **entity class** based on this we can reduce redundancy of code and increase re usability

Observe above code have u seen this function before? in **BLL.CS**i said i will explain it later got it in**DAL.CS** I have created one function **InsertUserInformation**and using this one in **BLL.CS**by simply creating one object of **DAL** in **BLL.CS**.

Here you will get one doubt that is why **BLL.CS** we can use this **DAL.CS** directly into our code behind  we already discuss Business logic layer provide interface between DAL and Application layer by using this we can maintain consistency to our application.

Now our **Business Logic Layer** is ready and our **Data access layer** is ready now how we can use this in our application layer write following code in your save button click like this

|  |
| --- |
| protected void btnsubmit\_Click(object sender, EventArgs e)  {  string Output = string.Empty;  if (txtpwd.Text == txtcnmpwd.Text)  {  BEL objUserBEL = new BEL();  objUserBEL.UserName = txtuser.Text;  objUserBEL.Password = txtpwd.Text;  objUserBEL.FirstName = txtfname.Text;  objUserBEL.LastName = txtlname.Text;  objUserBEL.Email = txtEmail.Text;  objUserBEL.Phoneno = txtphone.Text;  objUserBEL.Location = txtlocation.Text;  objUserBEL.Created\_By = txtuser.Text;  BLL objUserBLL = new BLL();  Output = objUserBLL.InsertUserDetails(objUserBEL);  }  else  {  Page.RegisterStartupScript("UserMsg", "<Script language='javascript'>alert('" + "Password mismatch" +"');</script>");  }  lblErrorMsg.Text = Output;  } |

Here if you observe I am passing all parameters using this BEL(Entity Layer) and we are calling the method**InsertUserDetails** by using this BLL(Business Logic Layer)

Now run your applciation test with debugger you can get idea clearly.

I hope it helps you.

**Download sample code attached**

[](http://www.box.net/shared/3hy1li0714)