Unit - 4

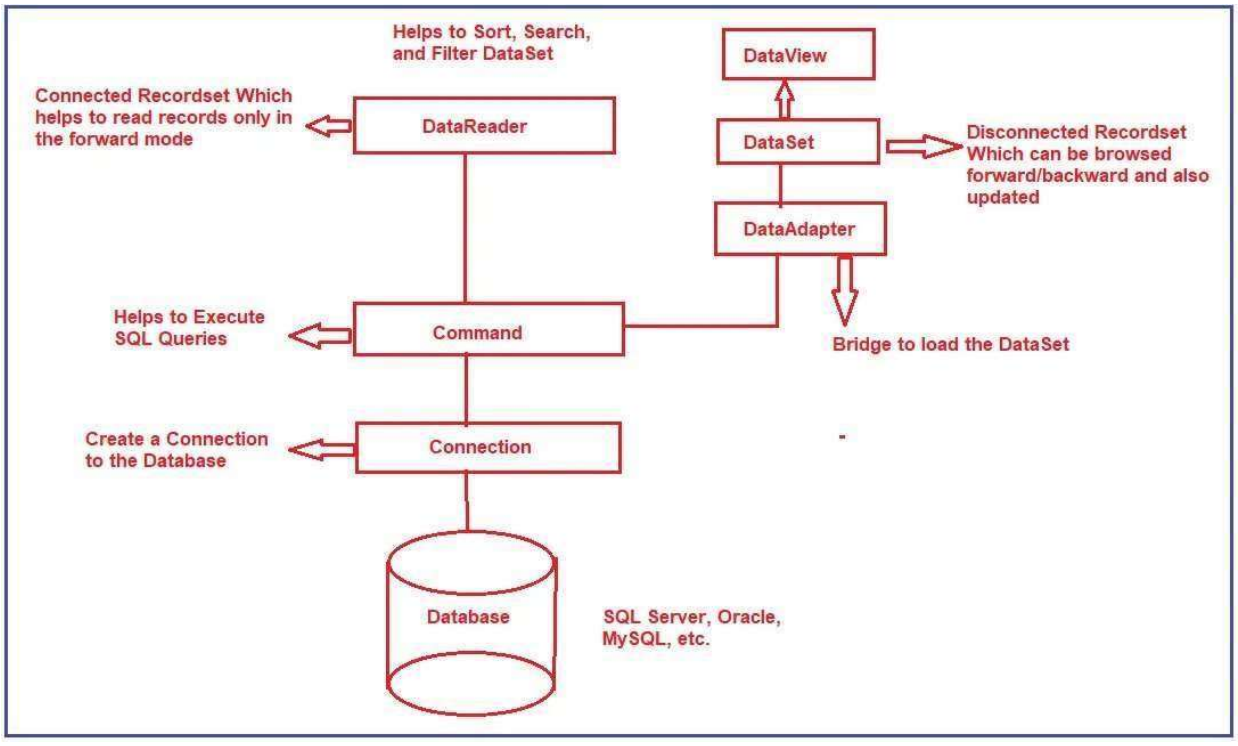
ADO.NET & Database

4.1 ADO.NET-Activex Data Object :

It is a module of .Net Framework which is used to establish connection between application and data sources. Data sources can be such as SQL Server and XML. ADO.NET consists of classes that can be used to connect, retrieve, insert and delete data.

The ADO.NET architecture comprises six important components. They are as follows:

1. Connection
2. Command
3. DataReader
4. DataAdapter
5. DataSet
6. DataView



Connection: The connection object is the first important component of your application. It is required to connect to a backend database that may be SQL Server, Oracle, MySQL, etc. You must have two things to create a connection object. Your database Machine name or IP address or someplace where it is stored is where it is.

Command: The second important component is the command object. When we discuss databases such as SQL Server, Oracle, MySQL, then speak SQL, it is the command object that we use to create SQL queries. After you create your SQL queries, you can execute them over the connection using the command object.

DataReader: The ADO.NET SqlDataReader class in asp.net is used to read data from the SQL Server database in the most efficient manner.

DataAdapter: The DataAdapter in ADO.NET acts as a bridge between a DataSet object and a SQL Server database, facilitating the retrieval and saving of data.

DataSet: The DataSet represents a subset of the database in memory. That means the ADO.NET DataSet is a collection of tables containing data in memory in tabular format.

DataView Class: A DataView enables you to modify the appearance of the data stored in a DataTable, a data-binding skill that is frequently employed in data-view applications. You may alter the sort order of data in a table or filter it based on row state or on a filter expression using a DataView.

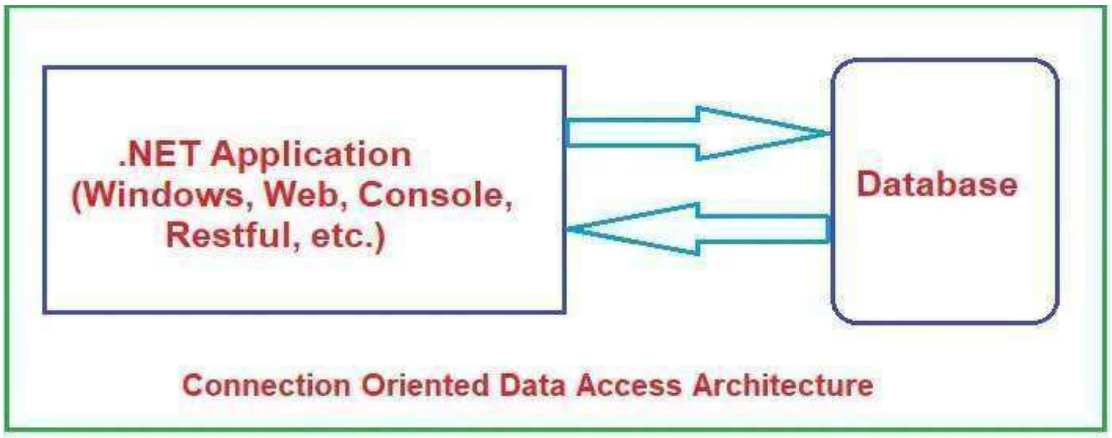
4.2 Types of Architecture to Access the Data using ADO.NET:

The Architecture supported by ADO.NET for communicating with data sources is categorized into two models. They are as follows:

Connected Oriented Architecture
Disconnected Oriented Architecture

So, ADO.NET supports both Connection-Oriented Architectures as well as Disconnection-Oriented Architecture. Depending upon the functionality or business requirement of an application, we can make it either Connection- Oriented or Disconnection-Oriented. Even, it is also possible to use both Architectures together in a single .NET application to communicate with different data sources.

ADO.NET Connection-Oriented Data Access Architecture:
In the case of Connection Oriented Data Access Architecture, always an open and active connection is required in between the .NET Application and the database. An example is Data Reader and when we are accessing the data from the database, the Data Reader object requires an active and open connection to access the data, If the connection is closed then we cannot access the data from the database and in that case, we will get the runtime error.



Command:
Command class behaves as a bridge between the front end and back end. It contains the query which has to perform from the front end and back end and also it contains an object of Connection Class:
Example: SqlCommand cmd=new SqlCommand("Query which has to perform",Connection Object);

DataReader :

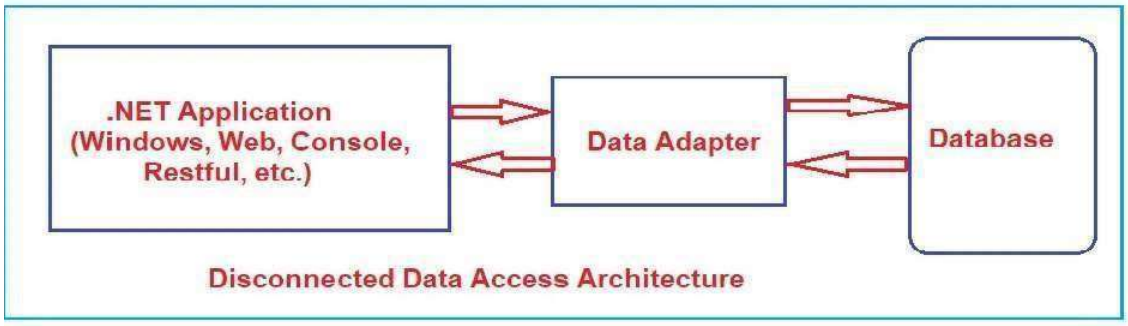
DataReader class is used to read the data from the database. It works in forward only and reads the only mode and requires the connection for the complete time. That is why we use it in connected architecture. The forward only feature makes it an efficient way to read data. Thus we can say, DataReader is connection-oriented and requires an active connection while reading the data.

In order to make an object of the DataReader class, we never use the new keyword instead we call the ExecuteReader() of the command object. For e.g.Here cmd.ExecuteReader() executes the command and creates the instance of DataReader class and loads the instance with data.

Example:SqlCommand cmd= new SqlCommand("Select * from Table"); SqlDatareader rdr=cmd.ExecuteReader(cmd);

ADO.NET Disconnection-Oriented Data Access Architecture:

In the case of Disconnection Oriented Data Access Architecture, always an open and active connection is not required in between the .NET Application and the database. In this architecture, Connectivity is required only to read the data from the database and to update the data within the database. An example is DataAdapter and DataSet or DataTable classes. Here, using the DataAdapter object and using an active and open connection, we can retrieve the data from the database and store the data in a DataSet or DataTable. The DataSets or DataTables are in-memory objects or you can say they store the data temporarily within .NET Application. Then whenever required in our .NET Application, we can fetch the data from the dataset or data table and process the data. Here, we can modify the data, we can insert new data, can delete the data from within the dataset or data tables. So, while processing the data within the.NET Application using DataSet or Datatable, we do not require an active and open connection.



Data Adapter:

A DataAdapter is a key component of ADO.NET, a technology in the Microsoft.NET framework for connecting to and working with databases. The DataAdapter serves as a bridge between a dataset (or DataTable) and a data source, typically a relational database. Its primary purpose is to facilitate the exchange of data between a dataset and the underlying data store, which is often a database like Microsoft SQL Server, Oracle, or others. The main functions of a DataAdapter include:

The DataAdapter helps in mapping the data in the dataset to the structure of the underlying data source. It assists in translating between the dataset's structure and the database's table schema.

Data Set :
In asp, a DataSet is an in-memory, disconnected data structure provided by ADO.NET for working with tabular data. It is part of the .NET Framework's data access technology and is commonly used to store, manipulate, and represent data retrieved from various data sources, such as databases.

Data row:
In asp, a DataRow is a fundamental component of the ADO.NET library and represents a single row of data within a DataTable object. A DataTable is a tabular structure that holds data in a column-and-row format, and a DataRow is essentially one row of data within that table.

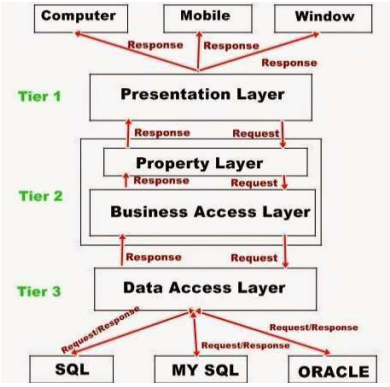
Data column:
In asp, a Data Column is a fundamental component of the ADO.NET library and is used to define and manage the schema of a column within a DataTable object.It is individual column itself.

Data relation:
In asp, a DataRelation is a component of the ADO.NET library that represents a relationship between two DataTable objects in a dataset. It defines how rows in one table are related to rows in another table, typically based on common columns or keys. DataRelation objects are used to establish and maintain the structure of related data within a dataset.

Data view: In asp, a Data View is a component of ADO.NET that provides a customized view or filter of the data contained within a Data Table. It allows you to perform various operations on the data, such as filtering, sorting, and searching, without modifying the underlying data. Data View objects are often used when you want to present a subset of data from a Data Table to the user interface or perform operations on a filtered view of the data.

4.3 3 Tier Architecture:

Three-layer architecture is dividing the project into three layers that are *User interface layer*, *business layer* and data(database) layer where we separate UI, logic, and data in three divisions. Suppose the user wants to change the UI from windows to the phone than he has to only make change in UI layer, other Layers are not affected by this change.



1.) Presentation Layer:-

Presentation Layer is nothing but it is a user interface which every user see on your computer, mobile and window screen. You can say, designing part of any application is known as Presentation Layer. The User can post input and get output on your presentation Layer only. In asp.net .axpx file is known as a Presentationlayer.

2.) Business Access Layer:-

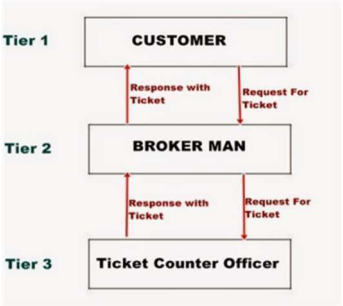
Business Access Layer is act as mediator Layer between Presentation layer and Data Access layer. This layer is used to transfer the data between Presentation Layer and Data Access Layer. This layer is mainly used for Validations and calculations purpose. Every validations and calculations of data are held on that layer only. I have also implemented Property layer or Entity Layer concepts in Business Access Layer. It is optional layer if you are working on small projects. But if you are working on large projects then you have to include this layer in your 3 Tier Architecture Applications. It is used to enhance the security and prevent to brokering the application.

3.) DataAccess Layer :-

This Layer only communicate with Business Access Layer. Data Access Layer contains the methods that helps Business Access Layer. Business layer class's methods call the Data Access Layer Class methods to perform some required action with database such as insertion,deletion,updation etc. All database related connection codes are written in this layer only such as sql query, stored procedure etc.

Real Life Example:-

Suppose if you want to book tatkal ticket then what will you do ? You will hire a third party man that is called mediator or Broker man.You can easily talk with this Broker man but you can't talk with Ticket counter officers. The Broker man can easily talk with Ticket counter officers as well as users.
Same Funda and activity followed in 3 Tier Architecture Applications. You can relate it as follows:-
User (You) ---> Presentation Layer
Broker man ---> Business Access Layer
Ticket Counter Officer ---> Data Access Layer



Why does we use 3 Tier Architecture:-

There are following reasons to use 3 Tier Architecture in our asp.net application as given below:

- To increase the security in Application.
- To easily maintain the application.
- To easily modify or change in application.
- To reduce the server over load.
- To reduce the loading time of application.

Advantages of 3 Tier Architecture:-

- Each layer always uses your separate codes, so it is easy to modify the codes.
- It is helpful to easily maintain and understand the large projects.
- You can easily change your graphical Environment.
- It is more secure because any users can't access the database directly.
- You can easily change any layer codes without affecting other two layers.
- It is a more consistence application.

Disadvantage of 3 Tier Architecture:-

- It takes more time to build.
- Many people's face problems because they haven't good knowledge in oops concepts and other c# programming such as class,object,property etc.
- 3 Tier Architecture is more complex to build.

4.4 Controls Which is used for display data:

1. GridView Control :

- GridView control is used to display whole table data on web page. In GridView control each column define a filed or title, while each rows represents a record or data.
- The GridView control dipaly data with rows and columns wise, we can display whole table in GridView and we also display only required columns from table in GridView control in asp.net.
- In GridView control we can easily do sorting, paging and inline editing.
- Also we can perform editing and deleting operation on displayed data with GridView control.

Example:

```
protected void btnview_Click(object sender, EventArgs e)
{
SqlConnection SQLConn = new SqlConnection("Data Source=.\\SQLEXPRESS;Initial
Catalog='Blog';Integrated Security=True");
SqlDataAdapter SQLAdapter = new SqlDataAdapter("Select * from UserMst", SQLConn);
DataTable DT = new DataTable();

SQLAdapter.Fill(DT);

GridView1.DataSource = DT;
GridView1.DataBind();
}
```

2. Details View Control :

DetailsView control used to display a single database record of table layout in ASP.Net. Means it works with a single data item at a time. DetailsView control used to display record, insert, update, and delete Records from database.

Example:

```
private void BindDetailsView()
{

SqlConnection SQLConn = new SqlConnection("Data Source=COMPUTER_1\\SQLEXPRESS;Initial
Catalog=BOOK;Integrated Security=True");
SqlDataAdapter SQLAdapter = new SqlDataAdapter("select * from UserMst", SQLConn);
DataTable DT = new DataTable();
SQLAdapter.Fill(DT);
DetailsView1.DataSource = DT;
```

```
DetailsView1.DataBind();
}
```

3. Repeater Control:

The Repeater control as the name suggests will repeat its content and hence it can be used to display data.

HeaderTemplate – The content of this template will not be repeated and will be placed in the top most position i.e. head section of the Repeater control.

ItemTemplate – The content of this template will be repeated for each record present in its DataSource.

AlternatingItemTemplate – AlternatingItemTemplate is used for adding alternate items. It is used along with ItemTemplate, generally for displaying a different design for alternating items.

SeparatorTemplate - This template is used to add a separator between two items of the Repeater control.

FooterTemplate – The content of this template will not be repeated and will be placed in the bottom most position i.e. footer section of the Repeater control.

Example :

```
rotected void Page_Load(object sender, EventArgs e)
{
   if (!this.IsPostBack)
   {
      this.BindRepeater();
   }
}
 private void BindRepeater()
{
   string constr = ConfigurationManager.ConnectionStrings["constr"].ConnectionString;
   using (SqlConnection con = new SqlConnection(constr))
   {
      using (SqlCommand cmd = new SqlCommand("SELECT TOP 10 * FROM Customers", con))
      {
         using (SqlDataAdapter sda = new SqlDataAdapter(cmd))
         {
            DataTable dt = new DataTable();
            sda.Fill(dt);
            rptCustomers.DataSource = dt;
            rptCustomers.DataBind();
         }
      }
   }
}
```

4. DataList Control :

The ASP.NET DataList control is a light weight server side control that works as a container for data items. It is used to display data into a list format to the web pages.

It displays data from the data source. The data source can be either a DataTable or a table from database.

Example :

```
protected void Page_Load(object sender, EventArgs e)
      {
DataTable table = new DataTable();
table.Columns.Add("ID");
         table.Columns.Add("Name");
table.Columns.Add("Email");
         table.Rows.Add("101", "Sachin Kumar", "sachin@example.com");
         table.Rows.Add("102", "Peter", "peter@example.com");
         DataList1.DataSource = table;
         DataList1.DataBind();
 }
```

5. FormView Control :

In ASP.NET, a "FormView" control is a data-bound control used to display data from a single record at a time. It is commonly used when you want to display detailed information about a single item from a data source, such as a database.

Example :

```
<asp:FormView ID="FormView1" runat="server" DataSourceID="SqlDataSource1">
  <ItemTemplate>
    <table>
      <tr>
        <td>Product ID:</td>
        <td><%# Eval("ProductID") %></td>
      </tr>
      <tr>
        <td>Product Name:</td>
        <td><%# Eval("ProductName") %></td>
      </tr>
      <tr>
        <td>Unit Price:</td>
        <td><%# Eval("UnitPrice", "{0:C}") %></td>
      </tr>
    </table>
  </ItemTemplate>
```

```
</asp:FormView>

<asp:SqlDataSource ID="SqlDataSource1" runat="server"
   ConnectionString="<%$ ConnectionStrings:YourConnectionString %>"
   SelectCommand="SELECT ProductID, ProductName, UnitPrice FROM Products WHERE
ProductID = @ProductID">
   <SelectParameters>
      <asp:QueryStringParameter Name="ProductID" QueryStringField="id" />
   </SelectParameters>
</asp:SqlDataSource>
```

Comman Properties for DataList View , GridView, DetailsView ,FormView,Repeater Controls

Properties	Description
DataSource	All these controls have a DataSource property, which is used to bind data from a data source such as a database, XML file, or object data source.
PageSize	For controls like GridView, DataList, and Repeater, this property specifies the number of records displayed per page when paging is enabled.
AllowPaging	A boolean property that enables or disables paging for the control.
AutoGenerateColumns	A boolean property that indicates whether columns should be generated automatically based on the data source schema.