ElectricityBill.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BillAutomation    //DO NOT change the namespace name
{
   public class ElectricityBill        //DO NOT change the class name
   {
     private String consumerNumber;
       private String consumerName;
       private int unitsConsumed;
       private double billAmount;

       public string ConsumerNumber
       {
          get { return this.consumerNumber; }
          set
          {
             if (value.Substring(0, 2).Equals("EB"))
             {
                this.consumerNumber = value;
             }
             else
                throw new FormatException("Invalid Consumer Number");
```

```
        }
    }
    //public string ConsumerNumber { get; set; }
    public string ConsumerName { get; set; }
    public int UnitsConsumed { get; set; }
    public double BillAmount { get; set; }
  }
}
```

ElectricityBoard.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BillAutomation
{
  public class ElectricityBoard
  {
```

```
    public SqlConnection SqlCon { get; set; }

    public ElectricityBoard() {}

    public void AddBill(ElectricityBill eb)
    {
        try
        {
            string querystring = "insert into ElectricityBill
values('"+eb.ConsumerNumber+"','"+eb.ConsumerName+"',"+eb.UnitsConsumed+","+eb.BillAmount+")"
;
            SqlCon.Open();
            SqlCommand cmd = new SqlCommand(querystring, SqlCon);
            SqlDataReader reader = cmd.ExecuteReader();
            SqlCon.Close();
        }
        catch (Exception e)
        {
            Console.WriteLine("Error : " + e.Message);
        }
    }

    public void CalculateBill(ElectricityBill ebill)
    {
        if (ebill.UnitsConsumed <= 100)
            ebill.BillAmount = 0;
        else if (ebill.UnitsConsumed > 100 && ebill.UnitsConsumed <= 300){
            int temp=ebill.UnitsConsumed-100;
            ebill.BillAmount = temp * 1.5;
```

```
    }
    else if (ebill.UnitsConsumed > 300 && ebill.UnitsConsumed <= 600){


        int temp200=ebill.UnitsConsumed-100;
        int temp300=temp200-200;
        ebill.BillAmount=(200*1.5)+(temp300*3.5);
    }
    else if (ebill.UnitsConsumed > 600 && ebill.UnitsConsumed <= 1000){


        int temp200=ebill.UnitsConsumed-100;
        int temp400=temp200-500;

        ebill.BillAmount=(200*1.5)+(300*3.5)+(temp400*5.5);
    }
    else if (ebill.UnitsConsumed > 1000){

         int temp200=ebill.UnitsConsumed-100;
        int temp400=temp200-900;

        ebill.BillAmount=(200*1.5)+(300*3.5)+(400*5.5)+(temp400*7.5);
    }
}

public List<ElectricityBill> Generate_N_BillDetails(int num)
{
    try
    {
```

```
            string querystring = "Select TOP "+num+" * from ElectricityBill ORDER BY consumer_number
desc";
            SqlCon.Open();
            SqlCommand cmd = new SqlCommand(querystring, SqlCon);
            SqlDataReader reader = cmd.ExecuteReader();

            List<ElectricityBill> l1 = new List<ElectricityBill>();

            while (reader.Read())
            {
               ElectricityBill eb1 = new ElectricityBill();
               eb1.ConsumerNumber = reader[0].ToString();
               eb1.ConsumerName = reader[1].ToString();
               eb1.UnitsConsumed = (int)reader[2];
               eb1.BillAmount = (double)reader[3];
               l1.Add(eb1);
            }

            SqlCon.Close();
            return l1;
         }
         catch (Exception e)
         {
            Console.WriteLine("Error 1: " + e.Message);
         }
         return null;
      }
   }
}
```

DBHandler.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Configuration;

namespace BillAutomation
{
   public class DBHandler
   {public DBHandler(){}
      public SqlConnection GetConnection()
      {
         SqlConnection con = null;
         String connection = ConfigurationManager.ConnectionStrings["MyCon"].ConnectionString;
         con = new SqlConnection(connection);
         return con;

      }
   }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using System.Collections;
using System.Data;
using System.Configuration;

namespace BillAutomation        //DO NOT change the namespace name
{
   public class Program       //DO NOT change the class name
   {

      static void Main(string[] args)  //DO NOT change the 'Main' method signature
      {ElectricityBoard eb = null;
         DBHandler db = new DBHandler();
         SqlConnection con = db.GetConnection();
         List<ElectricityBill> l2 = new List<ElectricityBill>();
         Console.WriteLine("Enter Number of Bills To Be Added : ");
         int totBill = Convert.ToInt32(Console.ReadLine());
```

```
for (int cnt = 0; cnt < totBill; cnt++)
{
   Console.WriteLine("Enter Consumer Number : ");
   String conNo = Console.ReadLine();
   Console.WriteLine("Enter Consumer Name : ");
   String name = Console.ReadLine();
   Console.WriteLine("Enter Units Consumed : ");
   int units = Convert.ToInt32(Console.ReadLine());

   ElectricityBill ebill = new ElectricityBill();
   try
   {
      ebill.ConsumerNumber = conNo;
   }
   catch (FormatException e)
   {
      Console.WriteLine(e);
   }
   ebill.ConsumerName = name;
   ebill.UnitsConsumed = units;

   eb = new ElectricityBoard();
   eb.SqlCon = con;
   eb.CalculateBill(ebill);
   eb.AddBill(ebill);
   l2.Add(ebill);
}
Console.WriteLine();
Console.Write("Enter Last 'N' Number of Bills To Generate : ");
```

```
        int num = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine();
        foreach(var p in l2)
        {
           Console.WriteLine(((ElectricityBill)p).ConsumerNumber);
           Console.WriteLine(((ElectricityBill)p).ConsumerName);
           Console.WriteLine(((ElectricityBill)p).UnitsConsumed);
           Console.WriteLine("Bill Amount: "+((ElectricityBill)p).BillAmount);
        }

        List<ElectricityBill> l1=eb.Generate_N_BillDetails(num);
        Console.WriteLine("Details of Bill Generation");
        foreach(var ie in l1)
        {
           Console.WriteLine("EB Bill for " + ((ElectricityBill)ie).ConsumerName +" is "+
((ElectricityBill)ie).BillAmount);
        }
      }
   }

          public class BillValidator{     //DO NOT change the class name

     public String ValidateUnitsConsumed(int UnitsConsumed)     //DO NOT change the method
signature
     {
        //Implement code here
        if(UnitsConsumed<0)
          {
           return "Given units is invalid";
```

```
            }else{
                return"";
            }
        }
    }
}
```