FACTFILE:
GCE SOFTWARE SYSTEMS DEVELOPMENT

Rewarding Learning

Appendix 1

Add Customer details from AddCustomer form to the table in database

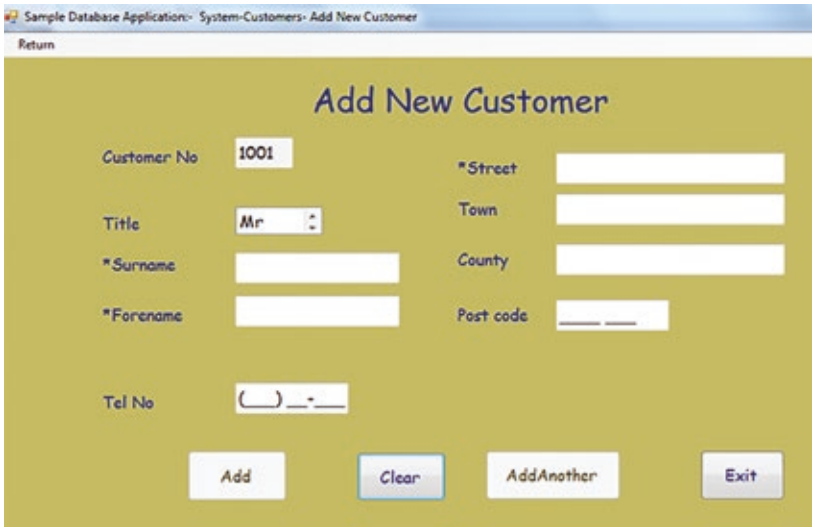

Table 1.1 – AddCustomer Form

Aside from usual libraries you must include the SQL library

```
using System.Data.SqlClient;

namespace ProjSampleDatabase
{
    public partial class FrmCustomerAdd : Form
    {
        SqlDataAdapter daCustomer;
        DataSet dsSampleDatabase = new DataSet();
        SqlCommandBuilder cmdBCustomer;
        DataRow drCustomer;
        String cnstr, sqlCustomer;
        int nextCustomerNo;
```

```
        public FrmOwnerAdd()
        {
            InitializeComponent();      // builds the form design

           //hold application name once in Properties and use on all forms
            this.Text = Properties.Resources.appStr;

        }//end FrmOwnerAdd

private void FrmOwnerAdd_Load(object sender, EventArgs e)
{
     // disable Add / AddAnother buttons
            btnAddAnother.Enabled = false;
            btnAdd.Enabled = false;


     // connect to database and create dataset and table to hold customer
     details
     // Use Properties to hold value for maintenance
            cnstr = Properties.Resources.connectStr;
     // cnstr = @"Data Source = localhost\sqlexpress;
     // Initial Catalog =SampleDatabase; Integrated Security = true";

            sqlCustomer = @"select * from Customers";
            daCustomer = new SqlDataAdapter(sqlCustomer, cnstr);
            cmdBCustomer = new SqlCommandBuilder(daCustomer);
            daCustomer.FillSchema(dsSampleDatabase, SchemaType.Source, "Customer");
            daCustomer.Fill(dsSampleDatabase, "Customer");
            int noRows = dsSampleDatabase.Tables["Customer"].Rows.Count;
            if (noRows == 0)
                nextCustomerNo = 1000;
            else
            {
              drCustomer = dsSampleDatabase.Tables["Customer"].Rows[noRows - 1];
              nextCustomerNo  = int.Parse(drCustomer["CustomerNo"].ToString()) + 1;

            }// end else
            txtCustomerNo.Text = nextCustomerNo.ToString();
     }// end FrmOwnerAdd_Load
```

```
//validation methods
    private bool validString(Control txt,int min, int max)
    {
       bool ok = true;
       TextBox txtB = (TextBox)txt;
       if (String.IsNullOrEmpty(txtB.Text))
       {
           ok = false;
           errP.SetError(txtB, " required field - must enter data");
       }// end if

       else if (txtB.Text.Length < min || txtB.Text.Length > max)
       {
            ok = false;
            errP.SetError(txtB, " must have minimum of "+ min+ " chars and a
            maximum of "+ max);
       }// end else if


        return ok;
     }// end validString

    private bool validLetters(Control txt)
    {
       bool ok = true;
       TextBox txtB = (TextBox)txt;
       for (int x = 0; x < txtB.Text.Length; x++)
       {   // if( txtB.Text[x])
         //  complete to check for letters  and possible embedded hyphen

       }// end for
       return ok;
    }// end validLetters
```

```
// button ADD click event
private void btnAdd_Click(object sender, EventArgs e)
   {
   // simple add using validation method checking existence and min/max length
   // of a string ( used for surname, forename and street)
   // sets error provider against invalid field

       errP.Clear();
       Boolean ok = true;

       if (lstTitle.SelectedIndex == -1)
       {
           ok = false;
           errP.SetError(lstTitle, "Must select Title");
       }// end if

       if (!validString(txtForename, 3, 15))
              ok = false;
       else
              if (!validLetters(txtForename))
                     ok = false;
       if (!validString(txtSurname, 3, 15))
              ok = false;
       if (!validString(txtAddress1, 5, 20))
              ok = false;

      // include try - catch here
       if (ok)
       {
              drCustomer = dsSampleDatabase.Tables["Customer"].NewRow();

              drCustomer["CustomerNo"] = int.Parse(txtCustomerNo.Text);
              drCustomer["Title"] = lstTitle.SelectedItem.ToString();
              drCustomer["Forename"] = txtForename.Text.Trim();
              drCustomer["Surname"] = txtSurname.Text.Trim();
              drCustomer["Address1"] = txtAddress1.Text.Trim();
              drCustomer["Address2"] = txtAddress2.Text.Trim();
              drCustomer["Address3"] = txtAddress3.Text.Trim();
              drCustomer["Postcode"] = txtPostcode.Text.Trim();
              drCustomer["TelNo"] = txtTelNo.Text.Trim();

              dsSampleDatabase.Tables["Customer"].Rows.Add(drCustomer);
              daCustomer.Update(dsSampleDatabase, "Customer");

              MessageBox.Show("Customer Added");
              btnAddAnother.Enabled = true;
              btnAdd.Enabled = false;
       }// end if
    }// end btnAdd_Click

  }// end FrmCustomerAdd
}// end ProjSampleDatabase
```

An advanced ADD would use the class Customer to validate the fields and throw a customised Exception which would be caught in the ADD. The catch would set the error provider with a specific error message

Customised Exception

```
class CustomerException : Exception
    {

        public CustomerException(string message)
            : base(message)
        {   }

    }// end class
```

Class Customer

```
class Customer
    {
        private int customerNo;
        private String title;
        private String surname;
        private String forename;
        private String street;
        private String town;
        private String county;
        private String postcode;
        private String telNo;

        public Customer()
    {
            customerNo=0;
            title=null;
            surname=null;
            forename=null;
            street=null;
            town=null;
            county=null;
            postcode=null;
            telNo=null;
    }// end Customer

    public Customer(int customerNo, String title, String surname, String forename,
    String street,String town, String county, String postcode, String telNo)
    {
             CustomerNo=customerNo;
             Title= title;
             Surname= surname;
             Forename= forename;
             Street= street;
             Town= town;
             County= county;
             Postcode= postcode;
             TelNo= telNo;
    }// end Customer

    public int CustomerNo
    {
```

```
        get { return customerNo; }
        set { customerNo = value; }

    }// end CustomerNo
     public String Title
     {
        get { return title; }
        set { title = value; }

     }// end Title
     public String Surname
     {
        get { return surname; }
        set
     {//implement validation / throw new exception
            String str = value;
            if (validString(str, 3, 15).CompareTo ("ok")!=0)
                throw new CustomerException("Invalid Surname - Please check");
            else
                surname  = value;
     }// end set

     }// end Surname

     public String Forename
     {
        get { return forename ; }
        set { forename = value; }

     }// end Forename

     public String Street
     {
        get { return street ; }
        set { street  = value; }

     }// end Street

     public String Town
     {
        get { return town ; }
        set { town  = value; }

     }// end Town

     public String County
     {
        get { return county ; }
        set { county  = value; }

     }// end County

     public String Postcode
     {
        get { return postcode ; }
        set { postcode  = value; }

     }// end Postcode
```

```
        public String TelNo
        {
            get { return telNo ; }
            set { telNo = value; }

        }// end TelNo

        public String toString()
        {   //basic - extend for all fields
            return String.Format("\n {0:d4}  {1} {2} {3} {4} {5}", customerNo,
                    surname, forename, street, postcode, telNo);
        }// end toString

        private String validString(String str, int min, int max)
        {
            String message = "ok";

            if (String.IsNullOrEmpty(str))
            {
                message = " Required field - must enter data";
            }// end if
            else if (str.Length < min || str.Length > max)
            {
            message= " must have minimum of " + min + " chars and a maximum of " +max;
            }// end else if

            return message;
        }// end validString

        }// end class Customer
}// this seems to be an extra bracket!!!
```

Revised Click event for ADD button

```
private void btnAdd_Click(object sender, EventArgs e)
{
// add using customer class validation for the surname - implement for other fields
// try-catch used for each individual field
// could use around all fields but only first error found would be shown by error
// provider

        errP.Clear();
        Boolean ok = true;
        Customer c = new Customer();

        if(lstTitle.SelectedIndex == -1)
        {
           ok = false;
           errP.SetError(lstTitle, "Must select Title");
        }// end if
        if (!validString(txtForename, 3, 15))
           ok = false;
        else
        if (!validLetters(txtForename))
           ok = false;
        try{
        c.Surname = txtSurname.Text.Trim();
        }// end try
        catch (CustomerException ex)
        {
           ok = false;
           errP.SetError(txtSurname, ex.Message);
        }// end catch
        if (!validString(txtStreet, 5, 20))
           ok = false;


        if (ok)
        {
                drCustomer = dsSampleDatabase.Tables["Customer"].NewRow();

                drCustomer["CustomerNo"] = int.Parse(txtCustomerNo.Text);
                drCustomer["Title"] = lstTitle.SelectedItem.ToString();
                drCustomer["Forename"] = txtForename.Text.Trim();
                drCustomer["Surname"] = c.Surname;
                drCustomer["Address1"] = txtStreet.Text.Trim();
                drCustomer["Address2"] = txtTown.Text.Trim();
                drCustomer["Address3"] = txtCounty.Text.Trim();
                drCustomer["Postcode"] = txtPostcode.Text.Trim();
                drCustomer["TelNo"] = txtTelNo.Text.Trim();

                dsSampleDatabase.Tables["Customer"].Rows.Add(drCustomer);
                daCustomer.Update(dsSampleDatabase, "Customer");

                MessageBox.Show("Customer Added");
                btnAddAnother.Enabled = true;
                btnAdd.Enabled = false;
        }// end if
}// end btnAdd_Click
```