

MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Thiết kế ứng dụng với mô hình 3 layer
- ✓ Kết hợp mô hình 3 layer và entity framework

PHẦN I

Bài 1 (4 điểm)

a/Sử dụng mô hình 3 lớp kết hợp ADO.Net viết ứng dụng : “Quản lý thành viên”

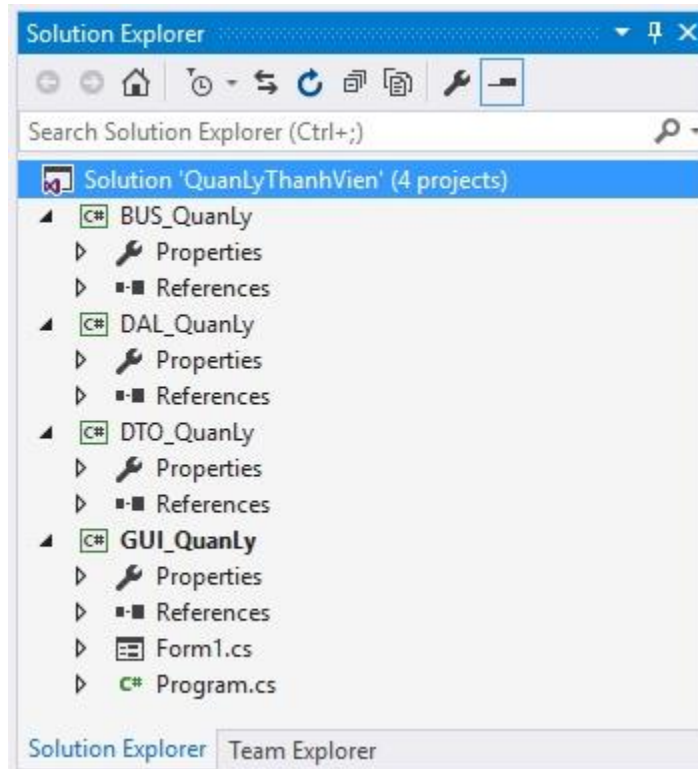
TV_ID	TV_NAME	TV_PHONE	TV_EMAIL
1	SethPhat	0000	SethPhat.com
2	123123	121	asdfasdf11

CSDL với table ThanhVien như sau:

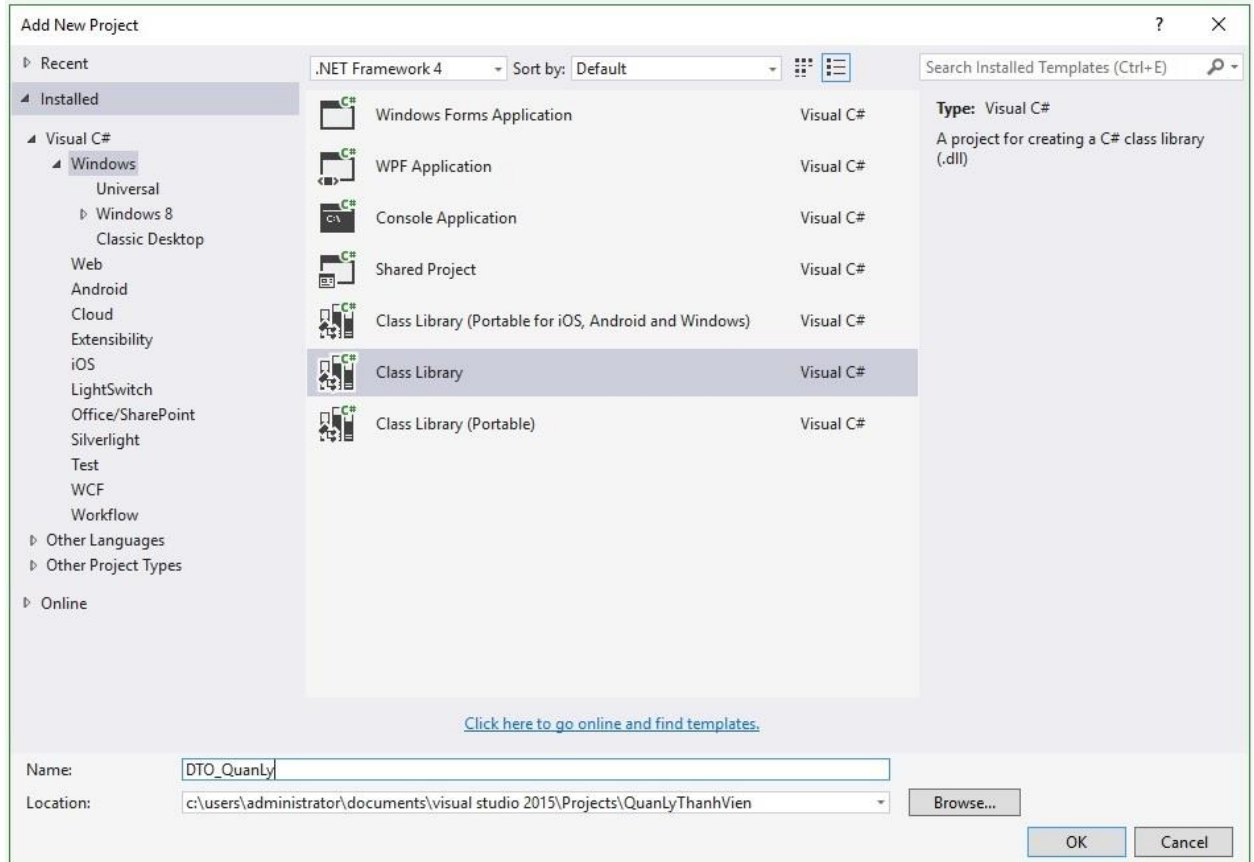
```
CREATE TABLE THANHVIEN (  
    TV_ID INT NOT NULL PRIMARY KEY IDENTITY,  
    TV_NAME NVARCHAR(30) NOT NULL,  
    TV_PHONE VARCHAR(11) NOT NULL,  
    TV_EMAIL VARCHAR(50) NOT NULL  
)
```

Hướng dẫn:

- Tạo dự án có cấu trúc:

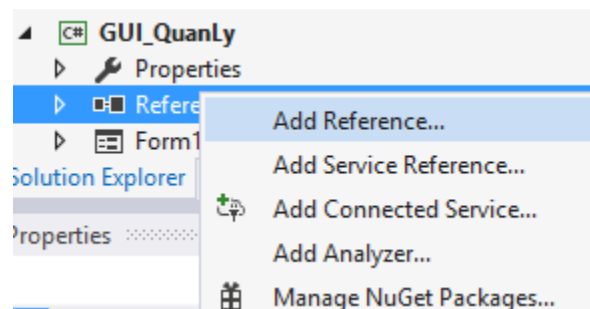


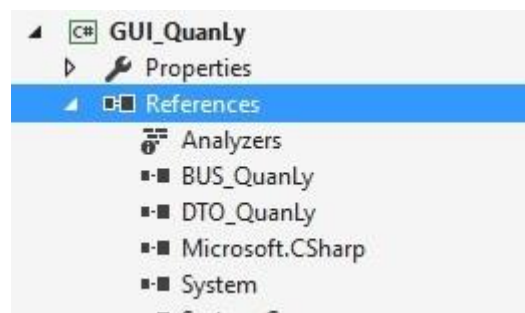
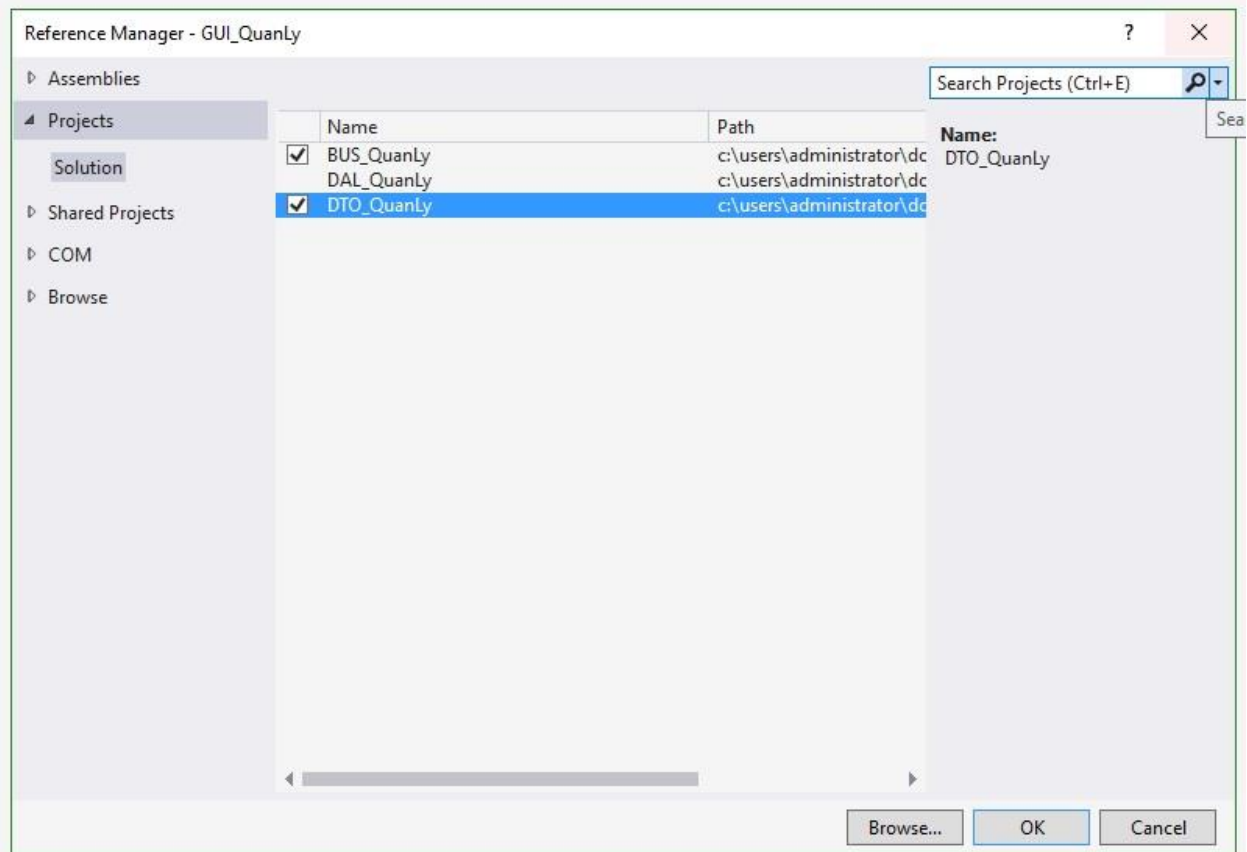
Trong đó 3 Project DTO, Business và Data Access chúng ta tạo theo Class Library



- Cần liên kết các project:
 - GUI liên kết tới Business Layer và DTO.
 - Business Layer liên kết tới được Data Access và DTO.
 - Data Access chỉ liên kết tới DTO

Sử dụng tính năng References => Add Reference để liên kết, ví dụ o GUI liên kết tới Business Layer và DTO:





Tương tự sinh viên thực hiện liên kết cho BUS và DAL.

- Viết mã lệnh cho project DTO, tạo DTO_ThanhVien.cs bên trong project và thực hiện viết các properties, hàm khởi tạo

```
namespace DTO_QuanLy
{
    public class DTO_ThanhVien
    {
        private int _THANHVIEN_ID;
        private string _THANHVIEN_NAME;
        private string _THANHVIEN_PHONE;
        private string _THANHVIEN_EMAIL;

        /* ===== GETTER/SETTER ===== */
        public int THANHVIEN_ID
        {
            get
            {
                return _THANHVIEN_ID;
            }

            set
            {
                _THANHVIEN_ID = value;
            }
        }

        public string THANHVIEN_NAME
        {
            get
            {
                return _THANHVIEN_NAME;
            }

            set
            {
                _THANHVIEN_NAME = value;
            }
        }

        public string THANHVIEN_PHONE
        {
            get
            {
                return _THANHVIEN_PHONE;
            }

            set
            {
                _THANHVIEN_PHONE = value;
            }
        }

        public string THANHVIEN_EMAIL
        {
            get
            {
                return _THANHVIEN_EMAIL;
            }

            set
            {
                _THANHVIEN_EMAIL = value;
            }
        }

        /* === Constructor === */
        public DTO_ThanhVien()
        {
        }

        public DTO_ThanhVien(int id, string name, string phone, string email)
        {
            this.THANHVIEN_ID = id;
            this.THANHVIEN_EMAIL = email;
            this.THANHVIEN_NAME = name;
            this.THANHVIEN_PHONE = phone;
        }
    }
}
```

- Viết mã lệnh cho project DAL_QuanLy gồm 2 class:
 - DBConnect.cs: thực hiện kết nối với database, sinh viên lưu ý chỉnh sửa chuỗi connection phù hợp với hệ quản trị sql mà sv đang dùng

```
using System.Data.SqlClient;

namespace DAL_QuanLy
{
    public class DBConnect
    {
        protected SqlConnection _conn = new SqlConnection("Data
Source=ADMINISTRATOR\\SQLEXPRESS;Initial Catalog=ThanhVien;Integrated Security=True");
    }
}
```

```
}
```

- DAL_ThanhVien.cs: xử lý các nghiệp vụ Lấy tất cả, Thêm, Xóa, Sửa

```
➤ namespace DAL_QuanLy
➤ {
➤     public class DAL_ThanhVien : DBConnect
➤     {
➤         /// <summary>
➤         /// Get toàn bộ thành viên
➤         /// </summary>
➤         /// <returns></returns>
➤         public DataTable getThanhVien()
➤         {
➤             SqlDataAdapter da = new SqlDataAdapter("SELECT * FROM THANHVIEN",
➤ _conn);
➤             DataTable dtThanhvien = new DataTable();
➤             da.Fill(dtThanhvien);
➤             return dtThanhvien;
➤         }
➤
➤         /// <summary>
➤         /// Thêm thành viên
➤         /// </summary>
➤         /// <param name="tv"></param>
➤         /// <returns></returns>
➤         public bool themThanhVien(DTO_ThanhVien tv)
➤         {
➤             try
➤             {
➤                 // Ket noi
➤                 _conn.Open();
➤
➤                 // Query string - vì mình để TV_ID là identity (giá trị tự
➤ tăng dần) nên ko cần phải insert ID
➤                 string SQL = string.Format("INSERT INTO THANHVIEN(TV_NAME,
➤ TV_PHONE, TV_EMAIL) VALUES ('{0}', '{1}', '{2}')" , tv.THANHVIEN_NAME,
➤ tv.THANHVIEN_PHONE, tv.THANHVIEN_EMAIL);
➤
➤                 // Command (mặc định command type = text nên chúng ta khỏi phải
➤ làm gì nhiều).
➤                 SqlCommand cmd = new SqlCommand(SQL, _conn);
➤
➤                 // Query và kiểm tra
➤                 if (cmd.ExecuteNonQuery() > 0)
➤                     return true;
➤             }
➤             catch (Exception e)
➤             {
➤
➤             }
➤             finally
```

```

>         {
>             // Dong ket noi
>             _conn.Close();
>         }
>
>         return false;
>     }
>
>     /// <summary>
>     /// Sửa thành viên
>     /// </summary>
>     /// <param name="tv"></param>
>     /// <returns></returns>
>     public bool suaThanhVien(DTO_ThanhVien tv)
>     {
>         try
>         {
>             // Ket noi
>             _conn.Open();
>
>             // Query string
>             string SQL = string.Format("UPDATE THANHVIEN SET TV_NAME =
> '{0}', TV_PHONE = '{1}', TV_EMAIL = '{2}' WHERE TV_ID = {3}",
> tv.THANHVIEN_NAME, tv.THANHVIEN_PHONE, tv.THANHVIEN_EMAIL, tv.THANHVIEN_ID);
>
>             // Command (mặc định command type = text nên chúng ta khỏi phải
> làm gì nhiều).
>             SqlCommand cmd = new SqlCommand(SQL, _conn);
>
>             // Query và kiểm tra
>             if (cmd.ExecuteNonQuery() > 0)
>                 return true;
>
>         }
>         catch (Exception e)
>         {
>
>         }
>         finally
>         {
>             // Dong ket noi
>             _conn.Close();
>         }
>
>         return false;
>     }
>
>     /// <summary>
>     /// Xóa thành viên
>     /// </summary>
>     /// <param name="tv"></param>
>     /// <returns></returns>
>     public bool xoaThanhVien(int TV_ID)
>     {

```

```

➤      try
➤      {
➤          // Ket noi
➤          _conn.Open();
➤
➤          // Query string - vì xóa chỉ cần ID nên chúng ta ko cần 1 DTO,
ID là đủ
➤          string SQL = string.Format("DELETE FROM THANHVIEN WHERE TV_ID
= {0})", TV_ID);
➤
➤          // Command (mặc định command type = text nên chúng ta khỏi phải
làm gì nhiều).
➤          SqlCommand cmd = new SqlCommand(SQL, _conn);
➤
➤          // Query và kiểm tra
➤          if (cmd.ExecuteNonQuery() > 0)
➤              return true;
➤
➤      }
➤      catch (Exception e)
➤      {
➤      }
➤      finally
➤      {
➤          // Dong ket noi
➤          _conn.Close();
➤      }
➤
➤      return false;
➤    }
➤  }

```

- Viết mã lệnh cho project BUS_QuanLy, tạo lớp BUS_ThanhVien.cs với các phương thức chỉ cần gọi lên DAL và trả về tương ứng cho GUI


```
namespace BUS_QuanLy
{
    public class BUS_ThanhVien
    {
        DAL_ThanhVien dalThanhVien = new DAL_ThanhVien();

        public DataTable getThanhVien()
        {
            return dalThanhVien.getThanhVien();
        }

        public bool themThanhVien(DTO_ThanhVien tv)
        {
            return dalThanhVien.themThanhVien(tv);
        }

        public bool suaThanhVien(DTO_ThanhVien tv)
        {
            return dalThanhVien.suaThanhVien(tv);
        }

        public bool xoaThanhVien(int TV_ID)
        {
            return dalThanhVien.xoaThanhVien(TV_ID);
        }
    }
}
```

- Xử lý project GUI_QuanLy, bắt sự kiện khi người dùng click các button thì thực hiện các phương thức tương ứng

```
➤ namespace GUI_QuanLy
➤ {
➤     public partial class GUI_ThanhVien : Form
➤     {
➤         BUS_ThanhVien busTV = new BUS_ThanhVien();
➤
➤         public GUI_ThanhVien()
➤         {
➤             InitializeComponent();
➤         }
➤     }
➤ }
```

```

➤     private void btnExit_Click(object sender, EventArgs e)
➤     {
➤         Application.Exit();
➤     }
➤
➤     private void btnAdd_Click(object sender, EventArgs e)
➤     {
➤         if (txtEmail.Text != "" && txtName.Text != "" && txtSDT.Text !=
➤         "")
➤         {
➤             // Tạo DTO
➤             DTO_ThanhVien tv = new DTO_ThanhVien(0, txtName.Text,
txtSDT.Text, txtEmail.Text); // Vì ID tự tăng nên để ID số gì cũng dc
➤
➤             // Thêm
➤             if (busTV.themThanhVien(tv))
➤             {
➤                 MessageBox.Show("Thêm thành công");
➤                 dgvTV.DataSource = busTV.getThanhVien(); // refresh
datagridview
➤             }
➤             else
➤             {
➤                 MessageBox.Show("Thêm ko thành công");
➤             }
➤         }
➤         else
➤         {
➤             MessageBox.Show("Xin hãy nhập đầy đủ");
➤         }
➤     }
➤
➤     private void GUI_ThanhVien_Load(object sender, EventArgs e)
➤     {
➤         dgvTV.DataSource = busTV.getThanhVien(); // get thanh vien
➤     }
➤

```

```

➤ private void btnEdit_Click(object sender, EventArgs e)
➤ {
➤     // Kiểm tra nếu có chọn table rồi
➤     if (dgvTV.SelectedRows.Count > 0)
➤     {
➤         if (txtEmail.Text != "" && txtName.Text != "" && txtSDT.Text
➤ != "")
➤         {
➤             // Lấy row hiện tại
➤             DataGridViewRow row = dgvTV.SelectedRows[0];
➤             int ID = Convert.ToInt16(row.Cells[0].Value.ToString());
➤
➤             // Tạo DTO
➤             DTO_ThanhVien tv = new DTO_ThanhVien(ID, txtName.Text,
➤ txtSDT.Text, txtEmail.Text); // Vì ID tự tăng nên để ID số gì cũng dc
➤
➤             // Sửa
➤             if (busTV.suaThanhVien(tv))
➤             {
➤                 MessageBox.Show("Sửa thành công");
➤                 dgvTV.DataSource = busTV.getThanhVien(); // refresh
➤ datagridview
➤             }
➤             else
➤             {
➤                 MessageBox.Show("Sửa ko thành công");
➤             }
➤         }
➤         else
➤         {
➤             MessageBox.Show("Xin hãy nhập đầy đủ");
➤         }
➤     }
➤     else
➤     {
➤         MessageBox.Show("Hãy chọn thành viên muốn sửa");
➤     }
➤ }

```

```

➤    }
➤
➤    private void dgvTV_Click(object sender, EventArgs e)
➤    {
➤        // Lấy row hiện tại
➤        DataGridViewRow row = dgvTV.SelectedRows[0];
➤
➤        // Chuyển giá trị lên form
➤        txtName.Text = row.Cells[1].Value.ToString();
➤        txtSDT.Text = row.Cells[2].Value.ToString();
➤        txtEmail.Text = row.Cells[3].Value.ToString();
➤    }
➤
➤    private void btnDelete_Click(object sender, EventArgs e)
➤    {
➤        // Kiểm tra nếu có chọn table rồi
➤        if (dgvTV.SelectedRows.Count > 0)
➤        {
➤
➤            // Lấy row hiện tại
➤            DataGridViewRow row = dgvTV.SelectedRows[0];
➤            int ID = Convert.ToInt16(row.Cells[0].Value.ToString());
➤
➤            // Xóa
➤            if (busTV.xoaThanhVien(ID))
➤            {
➤                MessageBox.Show("Xóa thành công");
➤                dgvTV.DataSource = busTV.getThanhVien(); // refresh
datagridview
➤            }
➤            else
➤            {
➤                MessageBox.Show("Xóa ko thành công");
➤            }
➤        }
➤    }

```

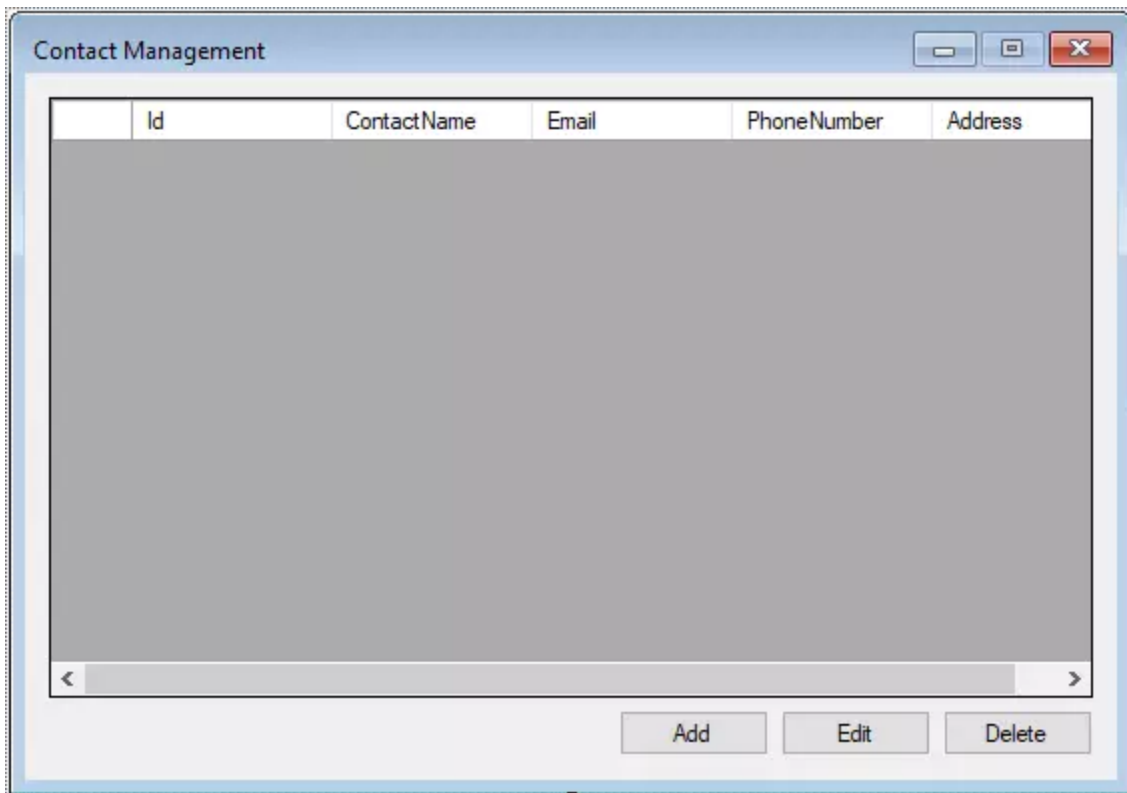
```
>         else
>         {
>             MessageBox.Show("Hãy chọn thành viên muốn xóa");
>         }
>     }
> }
```

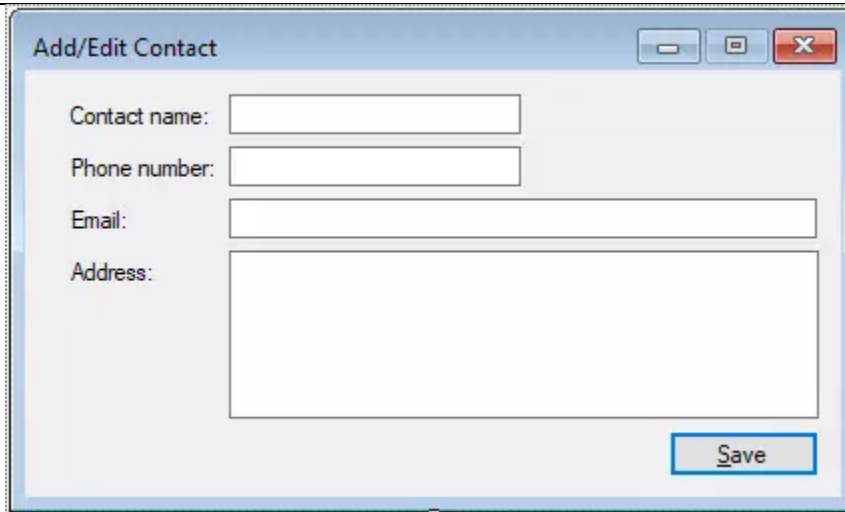
PHẦN 2

Bài 2 (4 điểm)

Xây dựng ứng dụng quản lý Contacts kết hợp sử dụng EntityFramework (DatabaseFirst), Hướng dẫn:

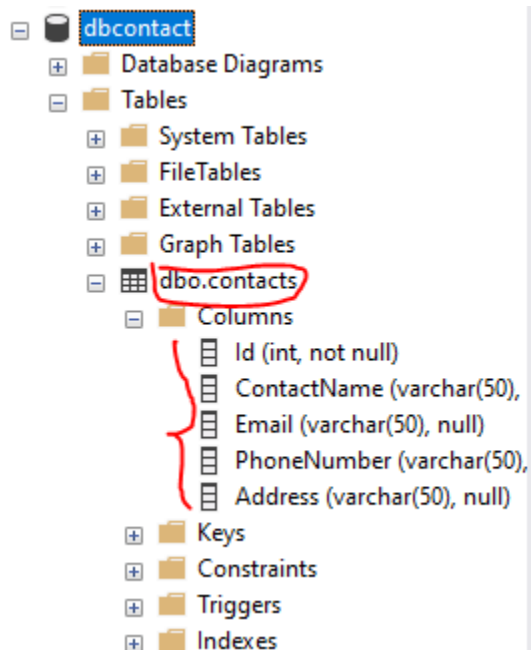
- Tạo dự án winform và thiết kế form Contact Management, form frmAddEditContact



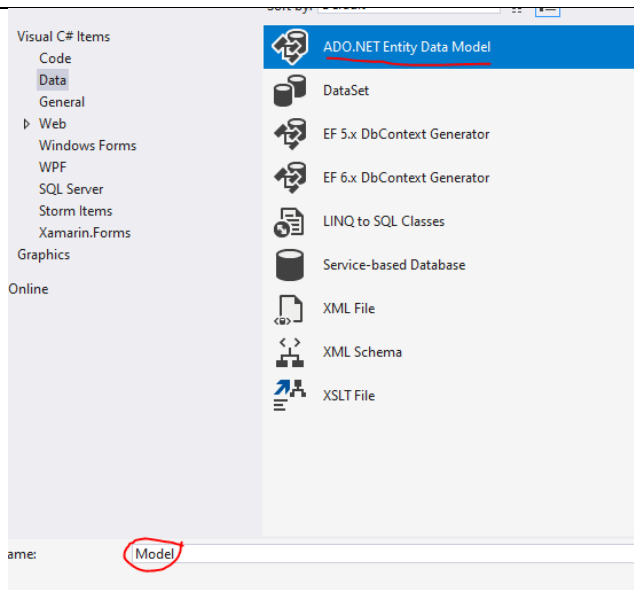


A Windows-style dialog box titled "Add/Edit Contact". It contains four input fields: "Contact name:", "Phone number:", "Email:", and "Address:". The "Address:" field is a larger text area. A "Save" button is located at the bottom right of the dialog.

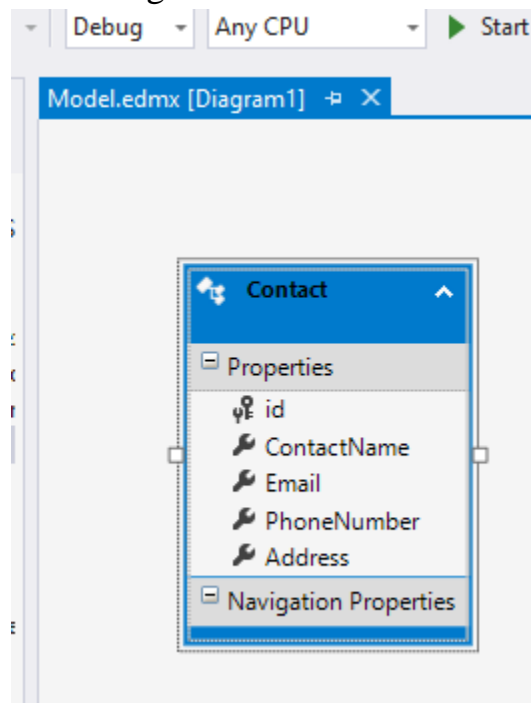
- Thiết kế csdl dbcontact có table contacts chứa các column Id tự tăng, các column còn lại như hình



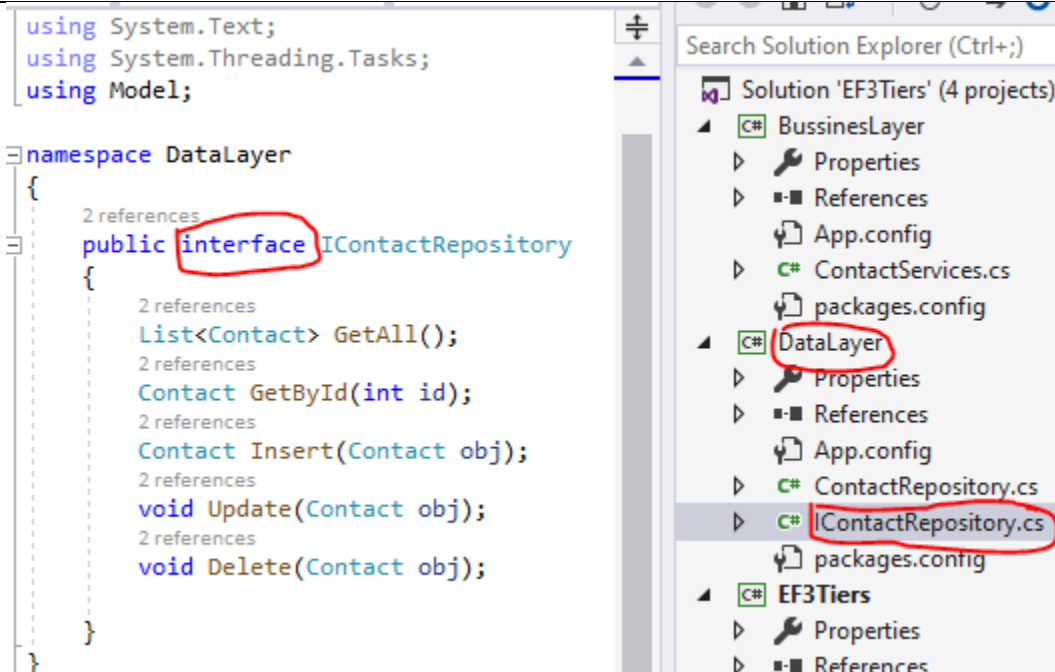
- Thêm project Model vào dự án, trong project này tạo EF model tên "Model" thông qua ADO.NET Entity Data Model



Kết quả sau khi tạo thành công Model.edmx



- Tạo DataLayer project, bên trong tạo interface IContactRepository đảm nhiệm công việc tương tác giữa các nghiệp vụ và dữ liệu



- Trong project Datalayer tạo class ContactRepository hiện thực interface IContactRepository xử lý các nghiệp vụ

```
namespace DataLayer
{
    1 reference
    public class ContactRepository : IContactRepository
    {
        2 references
        public void Delete(Contact obj)
        {
            using (ContactEntities db = new ContactEntities())
            {
                db.Contacts.Attach(obj);
                db.Contacts.Remove(obj);
                db.SaveChanges();
            }
        }
        2 references
        public List<Contact> GetAll()
        {
            using (ContactEntities db = new ContactEntities())
            {
                return db.Contacts.ToList();
            }
        }
    }
    2 references
}
```



```

public Contact GetById(int id)
{
    using (ContactEntities db = new ContactEntities())
    {
        return db.Contacts.Find(id);
    }
}
2 references
public Contact Insert(Contact obj)
{
    using (ContactEntities db = new ContactEntities())
    {
        db.Contacts.Add(obj);
        db.SaveChanges();
        return obj;
    }
}
2 references
public void Update(Contact obj)
{
    using (ContactEntities db = new ContactEntities())
    {
        db.Contacts.Attach(obj);
        db.Entry(obj).State = System.Data.Entity.EntityState.Modified;
        db.SaveChanges();
    }
}
}

```

- Tạo project BusinessLayer chứa class ContactServices và các phương thức gọi đến lớp ContactRepository

```

namespace BussinesLayer
{
    7 references
    public static class ContactServices
    {
        static IContactRepository repository;
        0 references
        static ContactServices()
        {
            repository = new ContactRepository();
        }
        3 references
        public static List<Contact> GetAll()
        {
            return repository.GetAll();
        }
        0 references
        public static Contact GetById(int id)
        {
            return repository.GetById(id);
        }
    }
}

```

```

public static Contact Insert(Contact obj)
{
    return repository.Insert(obj);
}
1 reference
public static void Update(Contact obj)
{
    repository.Update(obj);
}
1 reference
public static void Delete(Contact obj)
{
    repository.Delete(obj);
}
}

```

- Xử lý các sự kiện trên form Contact Management

```

public partial class Form1 : Form
{
    1 reference
    public Form1()
    {
        InitializeComponent();
    }
    1 reference
    private void Form1_Load(object sender, EventArgs e)
    {
        contactBindingSource.DataSource = ContactServices.GetAll();
    }
    1 reference
    private void btnAdd_Click(object sender, EventArgs e)
    {
        using (frmAddEditContact frm = new frmAddEditContact(null))
        {
            if (frm.ShowDialog() == DialogResult.OK)
                contactBindingSource.DataSource = ContactServices.GetAll();
        }
    }
    private void btnEdit_Click(object sender, EventArgs e)
    {
        if (contactBindingSource.Current == null)
            return;
        using (frmAddEditContact frm = new frmAddEditContact(cont
        {
            if (frm.ShowDialog() == DialogResult.OK)
                contactBindingSource.DataSource = ContactServices
        }
    }
    1 reference
    private void btnDelete_Click(object sender, EventArgs e)
    {
        if (contactBindingSource.Current == null)
            return;
        if (MessageBox.Show("Are you sure delete", "Message",
            MessageBoxButtons.YesNo, MessageBoxIcon.Question) == 
        {
            ContactServices.Delete(contactBindingSource.Current a
            contactBindingSource.RemoveCurrent();
        }
    }
}

```

➤ Xử lý form frmAddEditContact

```

public partial class frmAddEditContact : Form
{
    bool IsNew;
    2 references
    public frmAddEditContact(Contact obj){
        InitializeComponent();
        if(obj == null)
        {
            contactBindingSource.DataSource = new Contact();
            IsNew = true;
        }
        else
        {
            contactBindingSource.DataSource = obj;
            IsNew = false;
        }
    }
    1 reference
    private void frmAddEditContact_FormClosing(object sender, FormClosingEventArgs e){
        if(DialogResult == DialogResult.OK)
        {
            if(string.IsNullOrEmpty(txtContactName.Text))
            {
                MessageBox.Show("{Please enter your contact name.", "Messeage",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
                txtContactName.Focus();
                e.Cancel = true;
                return;
            }
            if(IsNew)
                ContactServices.Insert(contactBindingSource.Current as Contact);
            else
                ContactServices.Update(contactBindingSource.Current as Contact);
        }
    }
}

```

➤ Chạy kiểm tra kết quả

Add/Edit Contact

Contact Name:

Phone Number:

Email:

Address:

Contacts Management

	id	ContactName	Email	PhoneNumber	Address
▶	2	Fpoly	Fpoly@edu.vn	0978038559	778/b1 nguyễn k...

