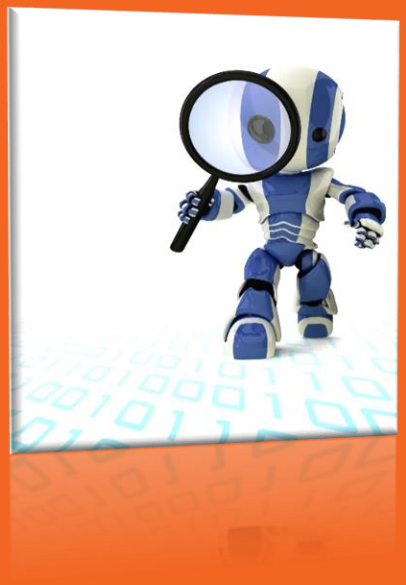
FPT Education

FPT POLYTECHNIC



LẬP TRÌNH C# 3

BÀI 7: MÔ HÌNH TỔ CHỨC DỰ ÁN PHẦN MỀM

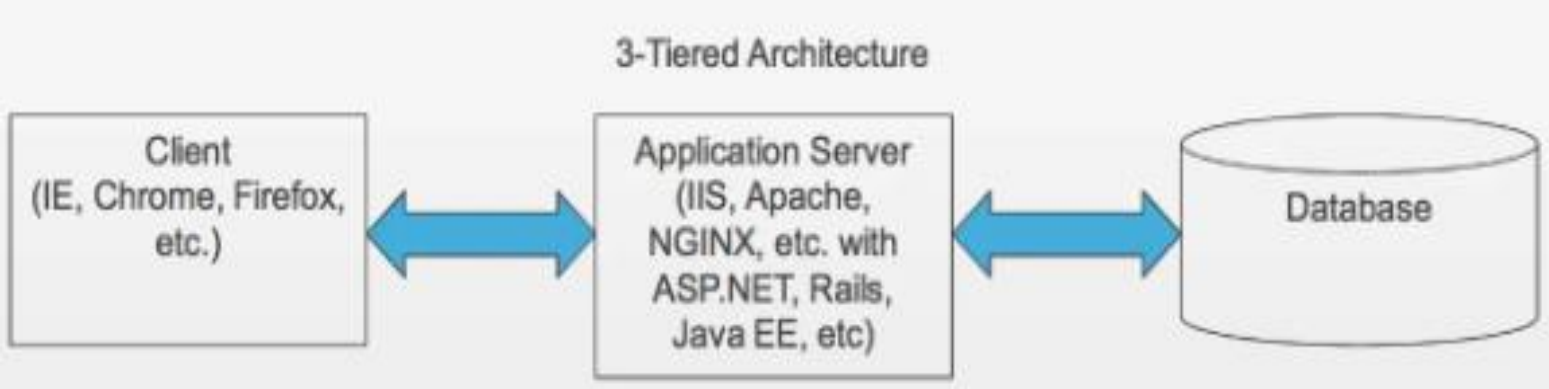
www.poly.edu.vn

Mục tiêu

- Mô hình cấu trúc dự án
- 3 layer và database first

MÔ HÌNH 3-TIERS

- 3-tiers là một kiến trúc kiểu client/server mà trong đó giao diện người dùng (UI-user interface), các quy tắc xử lý (BR-business rule hay BL-business logic), và việc lưu trữ dữ liệu được phát triển như những module độc lập, và hầu hết là được duy trì trên các nền tảng độc lập, và mô hình 3 tầng (3-tiers) được coi là một kiến trúc phần mềm và là một mẫu thiết kế.



MÔ HÌNH 3-TIERS

- Đây là kiến trúc triển khai ứng dụng ở mức vật lý, Kiến trúc gồm 3 module chính và riêng biệt
  - Tầng Presentation: hiển thị các thành phần giao diện để tương tác với người dùng như tiếp nhận thông tin, thông báo lỗi, …
  - Tầng Business Logic: thực hiện các hành động nghiệp vụ của phần mềm như tính toán, đánh giá tính hợp lệ của thông tin, … Tầng này còn di chuyển, xử lí thông tin giữa 2 tầng trên dưới
  - Tầng Data: nơi lưu trữ và trích xuất dữ liệu từ các hệ quản trị CSDL hay các file trong hệ thống. Cho phép tầng Business logic thực hiện các truy vấn dữ liệu

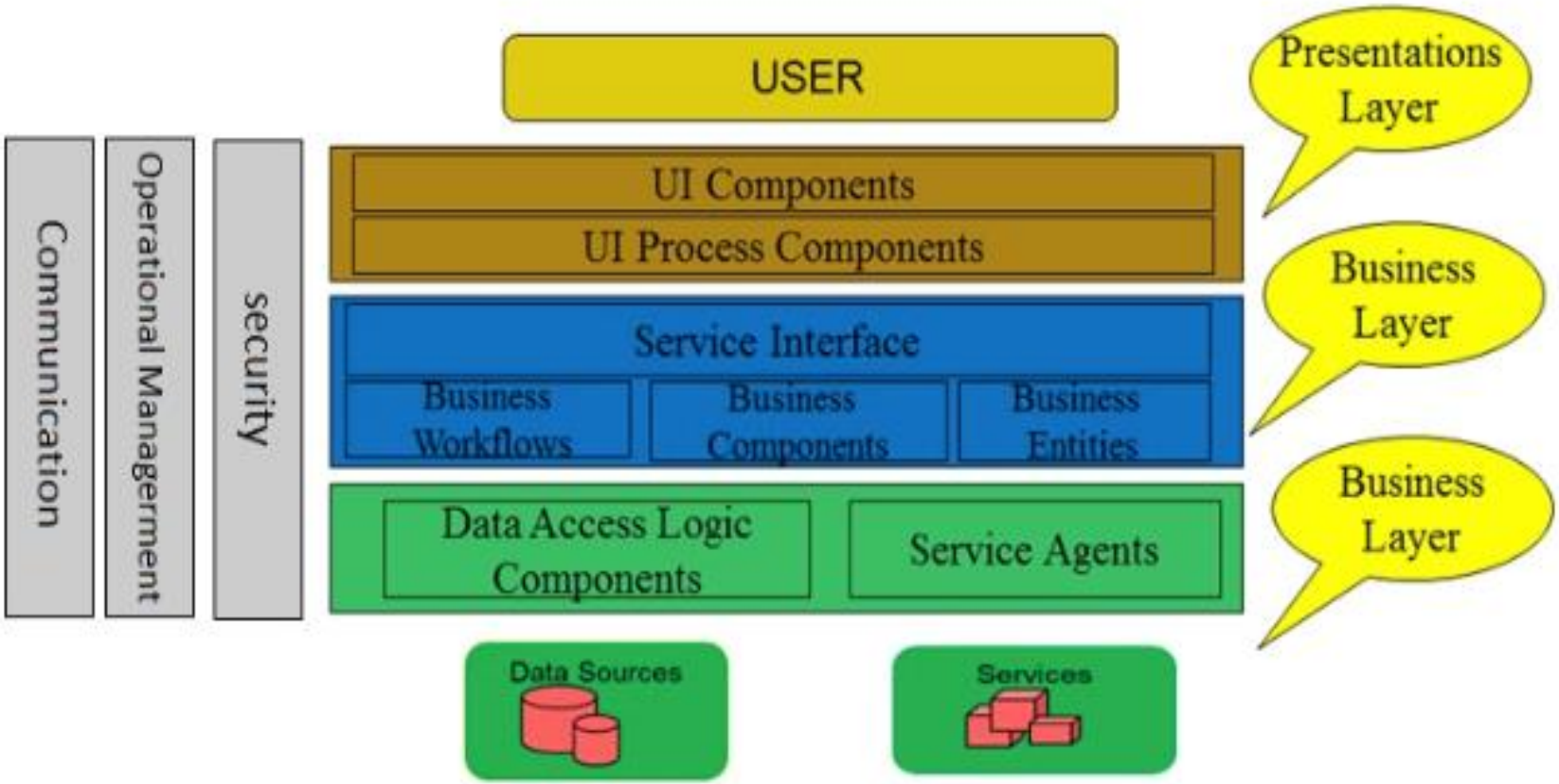
MÔ HÌNH 3-TIERS

- Ưu điểm:Dễ dàng mở rộng, thay đổi quy mô của hệ thống: Khi cần tải lớn, người quản trị có thể dễ dàng thêm các máy chủ vào nhóm, hoặc lấy bớt ra trong trường hợp ngược lại
- Nhược điểm:
  - Việc truyền dữ liệu giữa các tầng sẽ chậm hơn vì phải truyền giữa các tiến trình khác nhau (IPC), dữ liệu cần phải được đóng gói -> truyền đi -> mở gói trước khi có thể dùng được.
  - Việc phát triển ứng dụng phức tạp hơn

MÔ HÌNH 3 LỚP (3-LAYERS)

- Không như 3-Tiers có tính vật lý, 3-Layers có tính logic (mỗi layer có 1 công việc) và là 1 thành phần của 3-Tiers. Gồm 3 lớp chính:
  - Graphic User Interface (GUI): Thành phần giao diện, là các form của chương trình tương tác với người sử dụng.
  - Business Logic Layer (BLL): Xử lý các nghiệp vụ của chương trình như tính toán, xử lý hợp lệ và toàn vẹn về mặt dữ liệu.
  - Data Access Layer (DAL): Tầng giao tiếp với các hệ quản trị CSDL
  - Trong 1 số trường hợp vì lượng thông tin gởi nhiều ta có thể dùng Data Tranfer Object (DTO) để chuyển đối tượng hoặc danh sách đối tượng giữa các tầng với nhau cho tiện dụng.

MÔ HÌNH 3 LỚP (3-LAYERS)



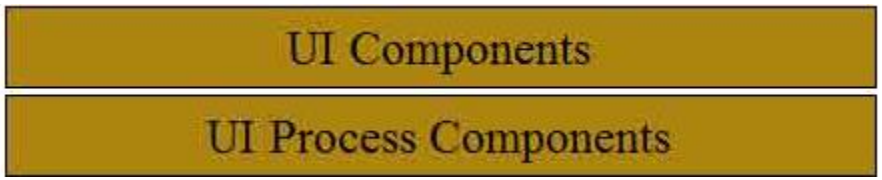
MÔ HÌNH 3 LỚP (3-LAYERS)

- Presentation Layer (GUI)
  - UI Components : gồm các thành phần tạo nên giao diện của ứng dụng (GUI). Chúng chịu trách nhiệm thu nhận và hiển thị dữ liệu cho người dùng… Ví dụ : textbox, button, combobox, …
  - UI Process Components : là thành phần chịu trách nhiệm quản lý các quá trình chuyển đổi giữa các UI… Ví dụ : Sắp xếp quá trình kiểm tra thông tin khách hàng:
    - 1.Hiển thị màn hình tra cứu ID
    - 2.Hiển thị màn hình thông tin chi tiết khách hàng tương ứng
    - 3.Hiển thị màn hình liên lạc với khách hàng.

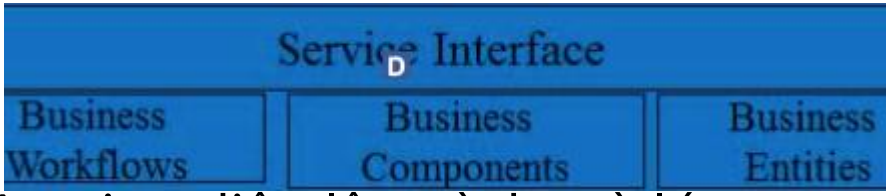
MÔ HÌNH 3 LỚP (3-LAYERS)

Presentation Layer (GUI)

- UI Components : gồm các thành phần tạo nên giao diện của ứng dụng (GUI). Chúng chịu trách nhiệm thu nhận và hiển thị dữ liệu cho người dùng… Ví dụ : textbox, button, combobox, …
- UI Process Components : là thành phần chịu trách nhiệm quản lý các quá trình chuyển đổi giữa các UI… Ví dụ : Sắp xếp quá trình kiểm tra thông tin khách hàng:
  - 1.Hiển thị màn hình tra cứu ID
  - 2.Hiển thị màn hình thông tin chi tiết khách hàng tương ứng
  - 3.Hiển thị màn hình liên lạc với khách hàng.



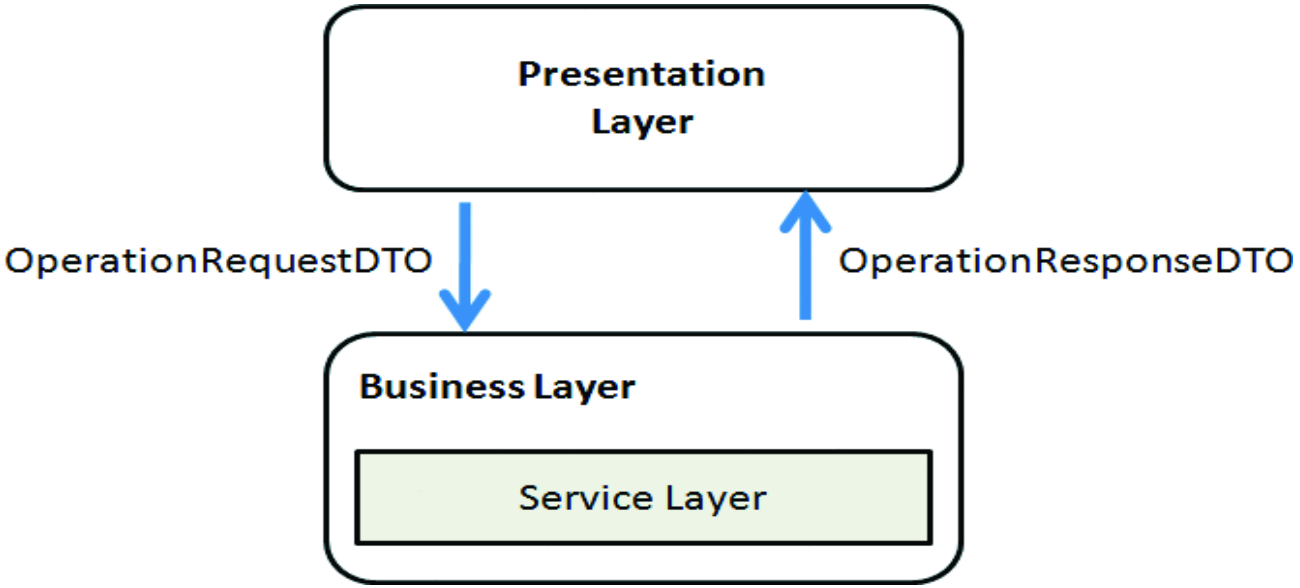
MÔ HÌNH 3 LỚP (3-LAYERS)



- Bussiness Layer (BLL):
  - Service Interface : là thành phần giao diện lập trình mà lớp này cung cấp cho lớp Presentation sử dụng.
  - Bussiness Workflows : chịu trách nhiệm xác định và điều phối các quy trình nghiệp vụ gồm nhiều bước và kéo dài.
    - Ví dụ : Thực hiện mua một đơn hàng trên tiki qua nhiều bước : kiểm tra gói hàng còn không?, tính tổng chi phí, cho phép giao dịch và sắp xếp việc giao hàng.
  - Bussiness Components : chịu trách nhiệm kiểm tra các quy tắc nghiệp vụ, ràng buộc logic và thực hiện các công việc . Các thành phần này cũng thực hiện các dịch vụ mà Service Interface cung cấp và Business Workflows sẽ sử dụng nó
    - Ví dụ : Tiếp tục ví dụ ở trên. Bạn sẽ cần một Bussiness Component để kiểm tra gói hàng có khả dụng không ? hay một component để tính tổng chi phí,...

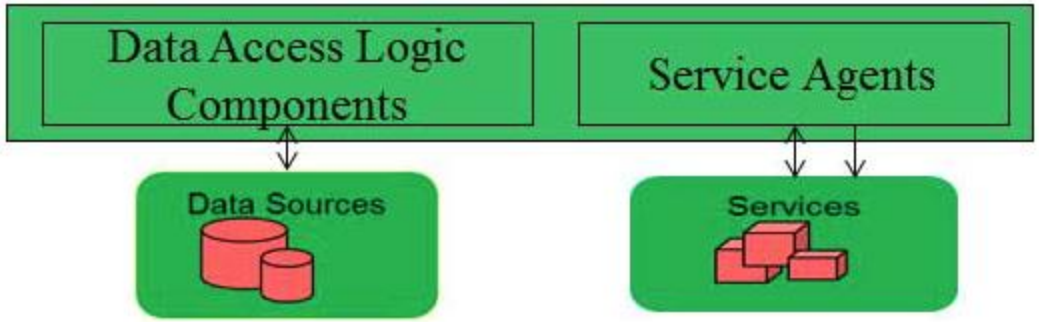
MÔ HÌNH 3 LỚP (3-LAYERS)

- Bussiness Layer (BLL):
  - Bussiness Entities : thường được sử dụng như Data Transfer Objects (DTO) . Bạn có thể sử dụng để truyền dữ liệu giữa các lớp (Presentation và Data Layer). Chúng thường là cấu trúc dữ liệu (DataSets, XML,…) hay các lớp đối tượng đã được tùy chỉnh
    - Ví dụ : tạo 1 class Student lưu trữ các dữ liệu về tên, ngày sinh, ID, lớp



MÔ HÌNH 3 LỚP (3-LAYERS)

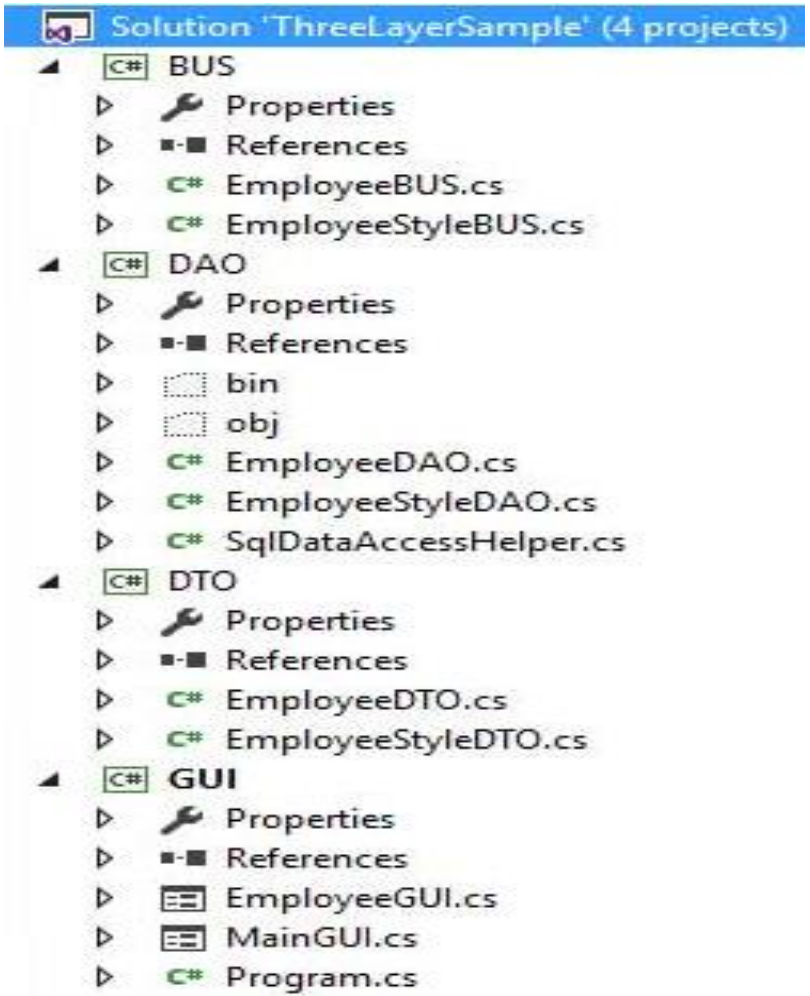
Data Layer (DAL)



- Data Access Logic Components : chịu trách nhiệm chính lưu trữ và truy xuất dữ liệu từ các nguồn dữ liệu (Data Sources) như XML, file system,... Hơn nữa còn tạo thuận lợi cho việc dễ cấu hình và bảo trì.
- Service Agents : giúp bạn gọi và tương tác với các dịch vụ từ bên ngoài một cách dễ dàng và đơn giản.

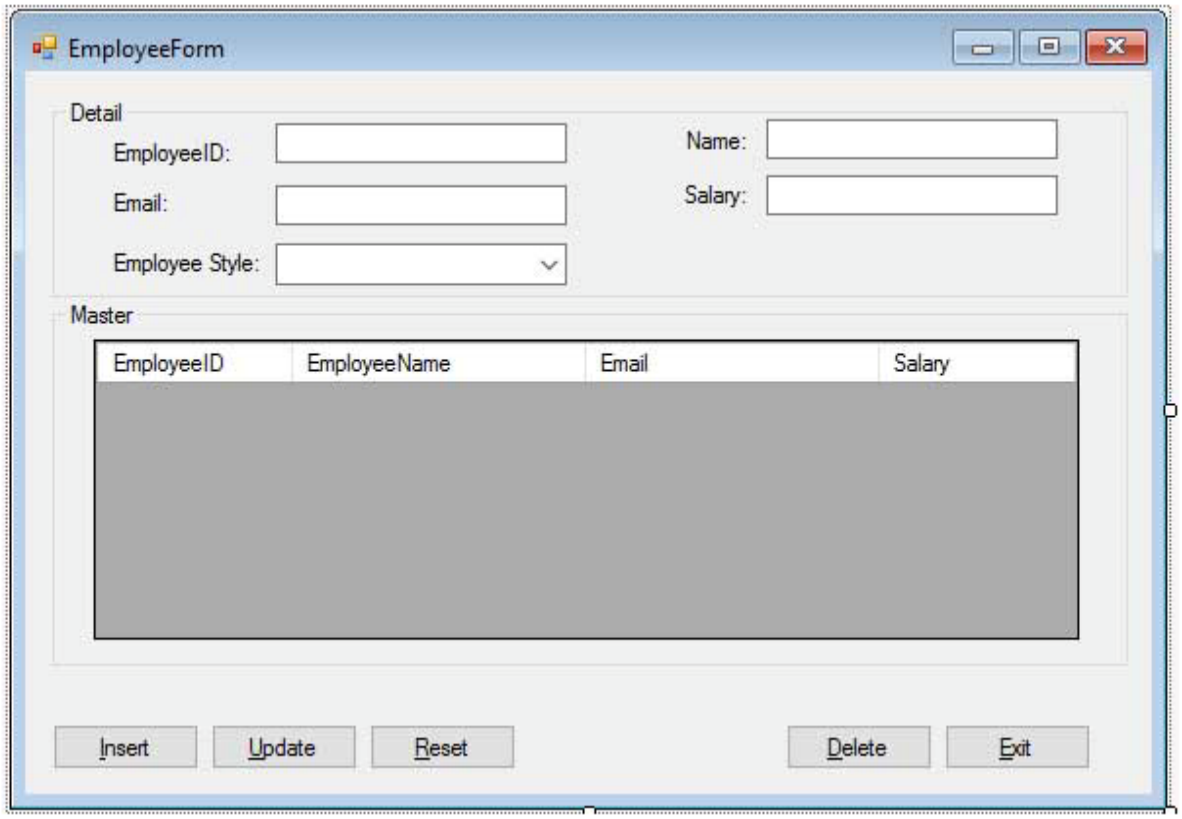
MÔ HÌNH 3 LỚP (3-LAYERS)

- Cấu trúc mô hình 3 lớp: Để hiểu rõ về cấu trúc và cách xây dựng của mô hình 3 lớp, chúng ta cùng tham khảo một ví dụ về mô hình quản lí công nhân gồm các lớp BUS, DAO, GUI
- Đối với 3 Project DTO, Business và Data Access chúng ta tạo theo Class Library và Add Reference vào project GUI



MÔ HÌNH 3 LỚP (3-LAYERS)

- GUI gồm các button insert, update, reset ,delete ,exit .Người dùng sẽ giao tiếp với màn hình giao diện:



MÔ HÌNH 3 LỚP (3-LAYERS)

- Lớp DTO, đây không phải là layer, đây chỉ là 1 gói dữ liệu đươc trao đổi giữa các lớp. Gói dữ liệu này được xây dựng dưới dạng lớp đối tượng

```
namespace DTO
{
    public class EmployeeDTO
    {
        #region Atrributes
        private String _employeeID;
        private String _name;
        private String _email;
        private float _salary;
        private int _employeeStyle;
        #endregion
        //.....
```

MÔ HÌNH 3 LỚP (3-LAYERS)

- Các nghiệp vụ xử lý chính được đặt ở lớp BUS (hay là BLL) gồm các nghiệp vụ insert, update, delete, retrieve

```
namespace BUS
{
    public class EmployeeBUS
    {
        #region 1. Inserting
        public static bool InsertEmployee(EmployeeDTO emp)
        {
            if (EmployeeDAO.CheckEmployeeByID(emp.EmployeeID)==true
                &amp;&amp;EmployeeStyleDAO.CheckEmployeeStyleByID(emp.EmployeeStyle)==f
alse)
            {
                return false;
            }
            return EmployeeDAO.InsertEmployee(emp);
        }
        #endregionhttps://techtalk.vn/wp-admin/post-new.php#
        //2. Updating
        //3. Deleting
        //4. Retrieving
    }
}
```

MÔ HÌNH 3 LỚP (3-LAYERS)

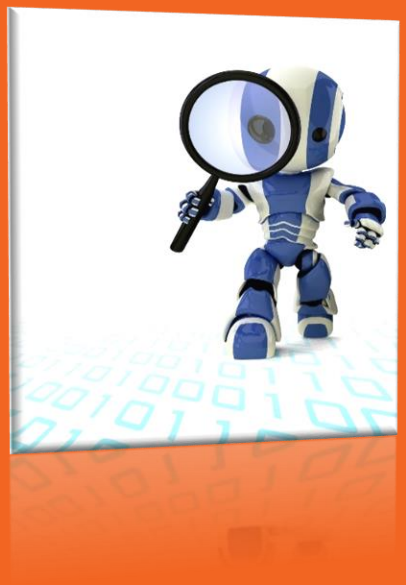
- Lớp DAO(hay là DAL). Truy vấn đến cơ sở dữ liệu

```
public class EmployeeDAO
{
    #region 1. Inserting
    public static bool InsertEmployee(EmployeeDTO emp)
    {
        bool result=false;
        try
        {
            // Create List Sql Parameter
            List sqlParams = new List();
            sqlParams.Add(new SqlParameter("@EmployeeID", emp.EmployeeID));
            sqlParams.Add(new SqlParameter("@Name", emp.Name));
            sqlParams.Add(new SqlParameter("@Email", emp.Email));
            sqlParams.Add(new SqlParameter("@Salary", emp.Salary));
            sqlParams.Add(new SqlParameter("@EmployeeStyle", emp.EmployeeStyle));
            // Call Store Procedure
            int n = SqlDataAccessHelper.ExecuteNoneQuery("spInsertEmployee", sqlPar
ams);
            if (n == 1)
                result = true;
        }
        catch (Exception ex)
        {
            throw ex;
        }
        return result;
    }
#endregion
    // 2. Updating
    // 3. Deleting
    // 4. Retrieving
```


DEMO
-Ứng dụng mô hình 3 lớp viết
Form đăng ký thành viên

FPT Education

FPT POLYTECHNIC



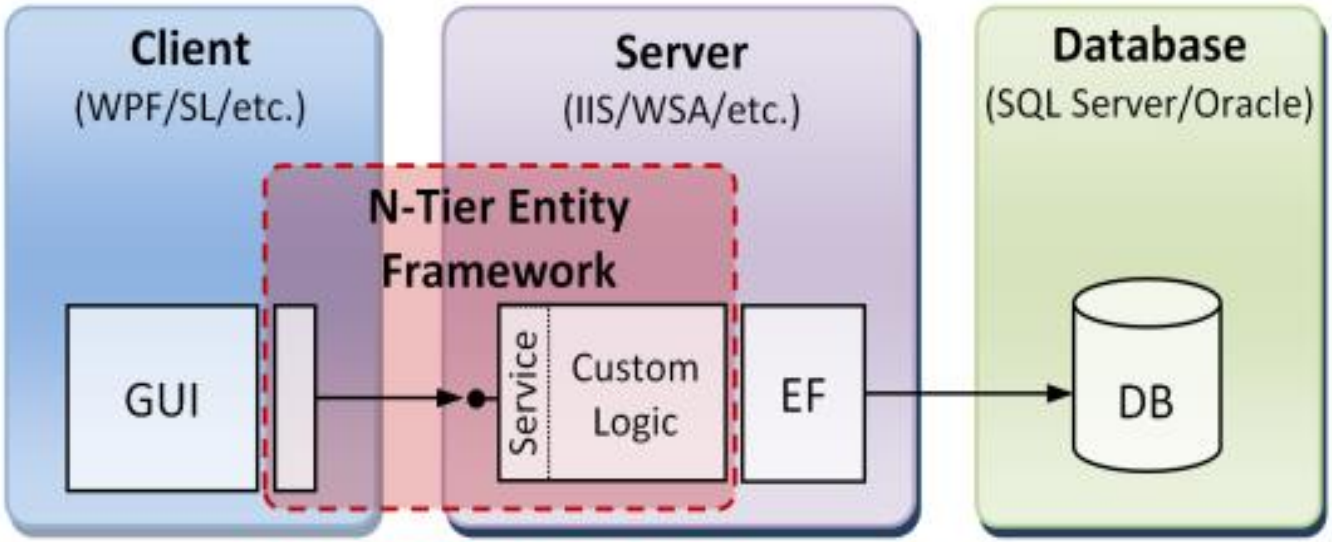
LẬP TRÌNH C# 3

BÀI 7: MÔ HÌNH TỔ CHỨC DỰ ÁN PHẦN MỀM (P2)

www.poly.edu.vn

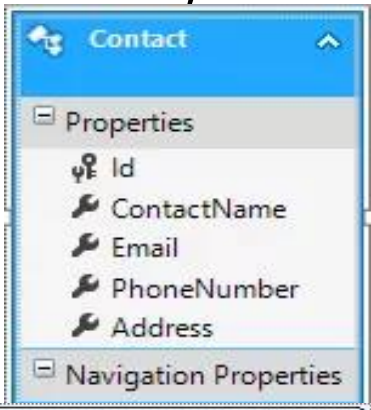
3 LAYER VÀ ENTITY FRAMEWORK

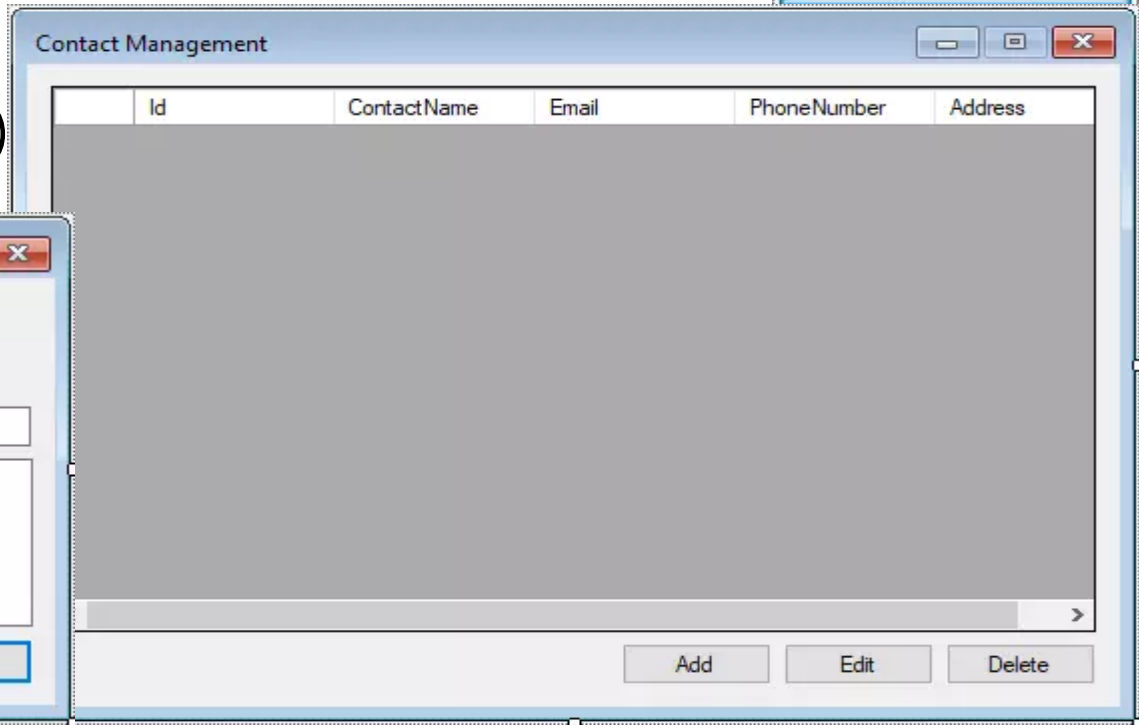
- Sự kết hợp 3 layer và Entity Framework mang lại nhiều lợi ích cho dự án
  - Cấu trúc dự án rõ rang
  - Thuận tiện cho việc thay đổi loại cơ sở dữ liệu của dự án
  - Hỗ trợ các thao tác xử lý dữ liệu nhanh

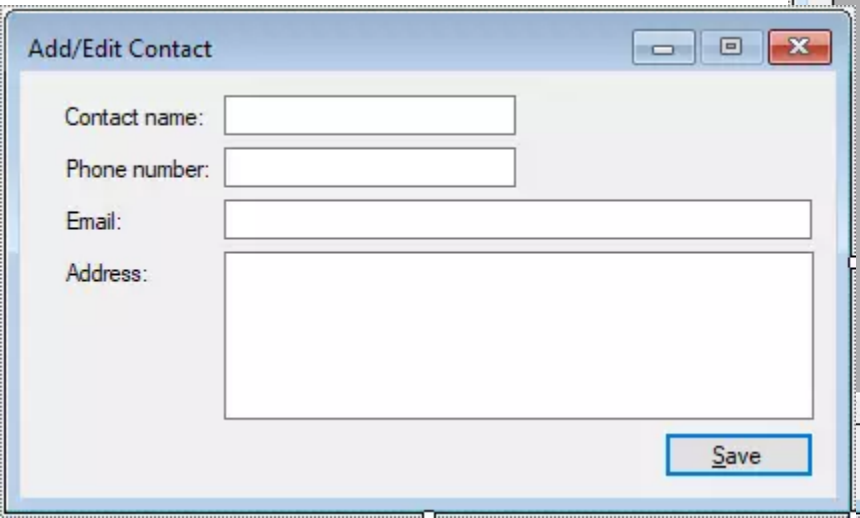


3 LAYER VÀ DATABASE FIRST

- Ví dụ ứng dụng quản lý Contact (ContactName, Email, PhoneNumber, Address)
  - Thiết kế database với các thông tin trên
  - Tạo project model và tạo EF model
  - Tạo form quản lý (form1)
  - Tạo form thêm (frmAddEditContact)







3 LAYER VÀ DATABASE FIRST

- Tạo interface tên"IContactRepository" quy định các thành mà lớp DAL phải hiện thực (tạo mới project loại library tên DataLayer)

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

namespace DataLayer
{
    public interface IContactRepository
    {
        List<Contact> GetAll();
        Contact GetById(int id);
        Contact Insert(Contact obj);
        void Update(Contact obj);
        void Delete(Contact obj);
    }
}
```

- Tạo lớp ContactRepository hiện thực interface "IContactRepository " (tầng tương tác với csdl)

```
namespace DataLayer
{
    public class ContactRepository : IContactRepository
    {
        public void Delete(Contact obj)
        {
            using(ContactEntities db = new ContactEntities())
            {
                db.Contacts.Attach(obj);
                db.Contacts.Remove(obj);
                db.SaveChanges();
            }
        }

        public List<Contact> GetAll()
        {
            using (ContactEntities db = new ContactEntities())
            {
                return db.Contacts.ToList();
            }
        }

        public Contact GetById(int id)
        {
            using (ContactEntities db = new ContactEntities())
            {
                return db.Contacts.Find(id);
            }
        }
```

```
        public Contact Insert(Contact obj)
        {
            using (ContactEntities db = new ContactEntities())
            {
                db.Contacts.Add(obj);
                db.SaveChanges();
                return obj;
            }
        }

        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();
            }
        }
```

3 LAYER VÀ DATABASE FIRST

- Tạo project BusinessLayer (tầng BUS) có tên class "ContactServices" xử lý các nghiệp vụ

```
namespace BusinessLayer
{
    public static class ContactServices
    {
        static IContactRepository repository;

        static ContactServices()
        {
            repository = new ContactRepository();
        }

        public static List<Contact> GetAll()
        {
            return repository.GetAll();
        }

        public static Contact GetById(int id)
        {
            return repository.GetById(id);
        }
```

```
        public static Contact Insert(Contact obj)
        {
            return repository.Insert(obj);
        }

        public static void Update(Contact obj)
        {
            repository.Update(obj);
        }

        public static void Delete(Contact obj)
        {
            repository.Delete(obj);
        }
```

3 LAYER VÀ DATABASE FIRST

- Xử lý tầng GUI (form1)

```
namespace EF3Tiers
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            //Call business layer to get data
            contactBindingSource.DataSource = ContactServices.GetAll();
        }

        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(contactBindingSource.Current as Contact))
    {
        if (frm.ShowDialog() == DialogResult.OK)
            contactBindingSource.DataSource = ContactServices.GetAll();
    }
}

private void btnDelete_Click(object sender, EventArgs e)
{
    if (contactBindingSource.Current == null)
        return;
    if(MessageBox.Show("Are you sure want to delete this record ?", "Message", MessageBoxButtons.Y
    {
        ContactServices.Delete(contactBindingSource.Current as Contact);
        contactBindingSource.RemoveCurrent();
    }
}
```

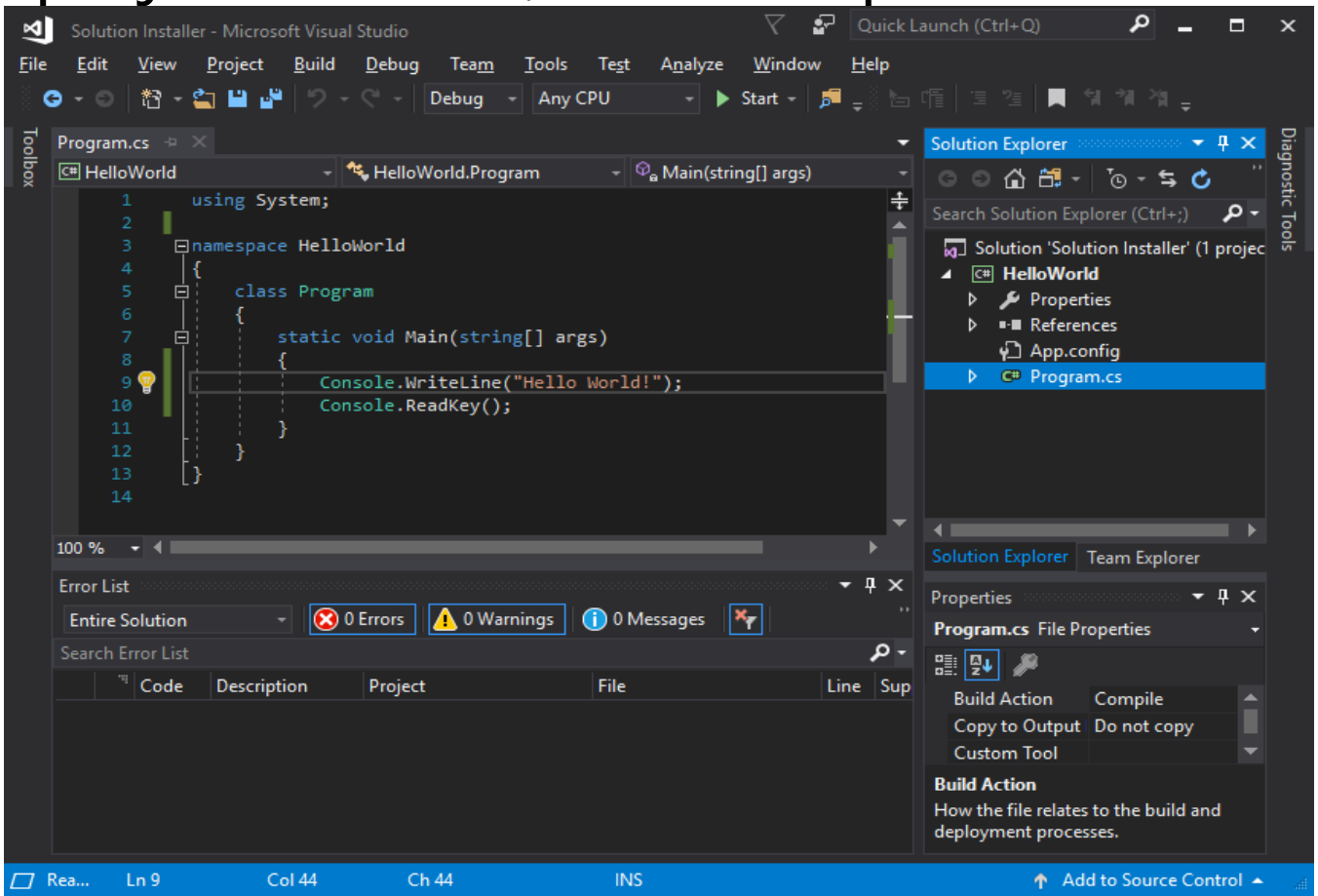
Xử lý tầng GUI (frmAddEditContact)

```
namespace EF3Tiers
{
    public partial class frmAddEditContact : Form
    {
        bool IsNew;
        public frmAddEditContact(Contact obj)
        {
            InitializeComponent();
            if (obj == null)
            {
                contactBindingSource.DataSource = new Contact();
                IsNew = true;
            }
            else
            {
                contactBindingSource.DataSource = obj;
                IsNew = false;
            }
        }

        private void frmAddEditContact_FormClosing(object sender, FormClosingEventArgs e)
        {
            if (DialogResult == DialogResult.OK)
            {
                if (string.IsNullOrEmpty(txtContactName.Text))
                {
                    MessageBox.Show("Please enter your contact name.", "Message", MessageBoxButi
                    txtContactName.Focus();
                    e.Cancel = true;
                    return;
                }
                if (IsNew)
                    ContactServices.Insert(contactBindingSource.Current as Contact);
                else
                    ContactServices.Update(contactBindingSource.Current as Contact);
            }
        }
    }
```

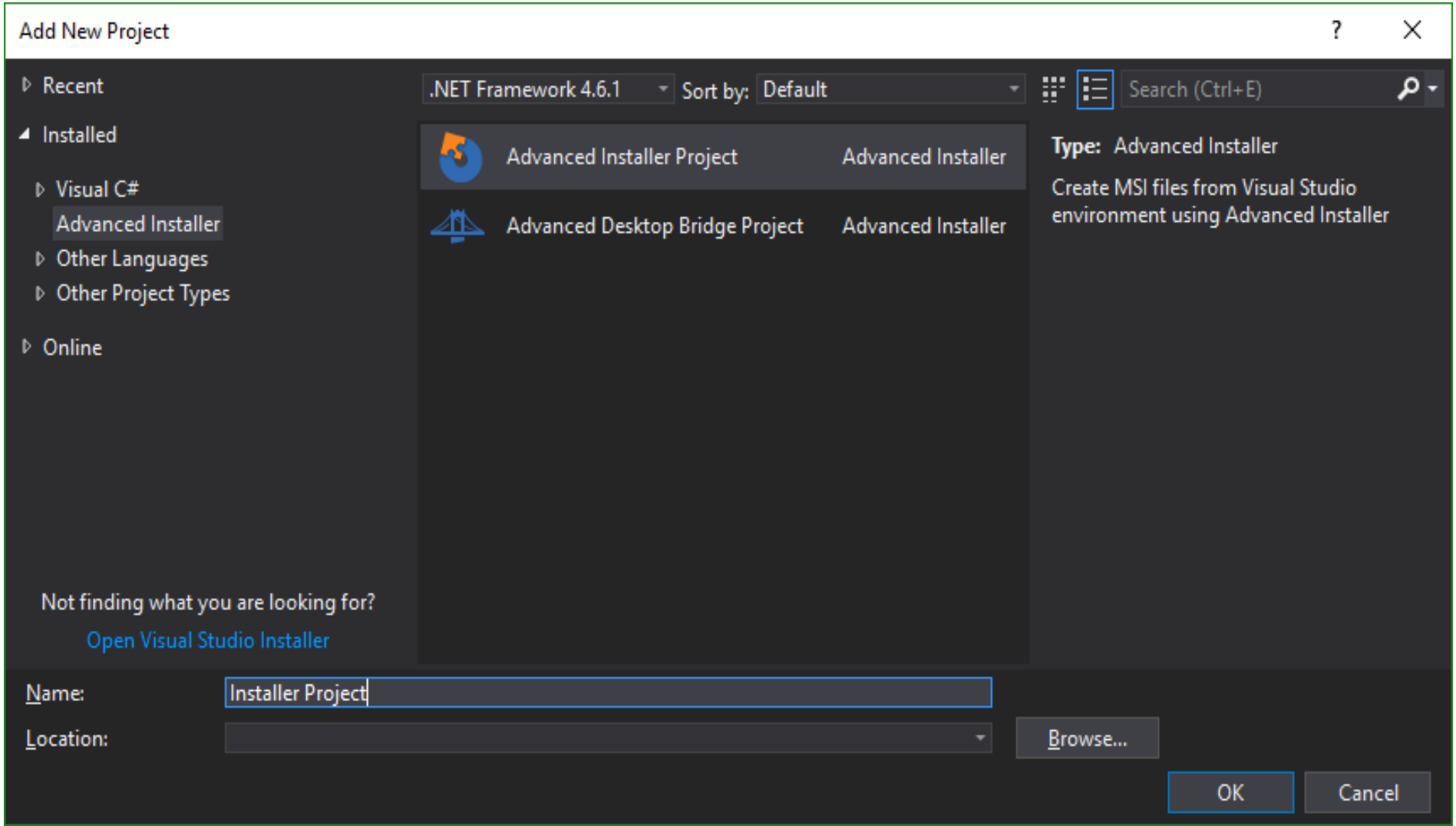
TẠO FILE SETUP

- Sử dụng Advanced Installer Project có sẵn trong công cụ Vs
- Mở project muốn tạo file setup



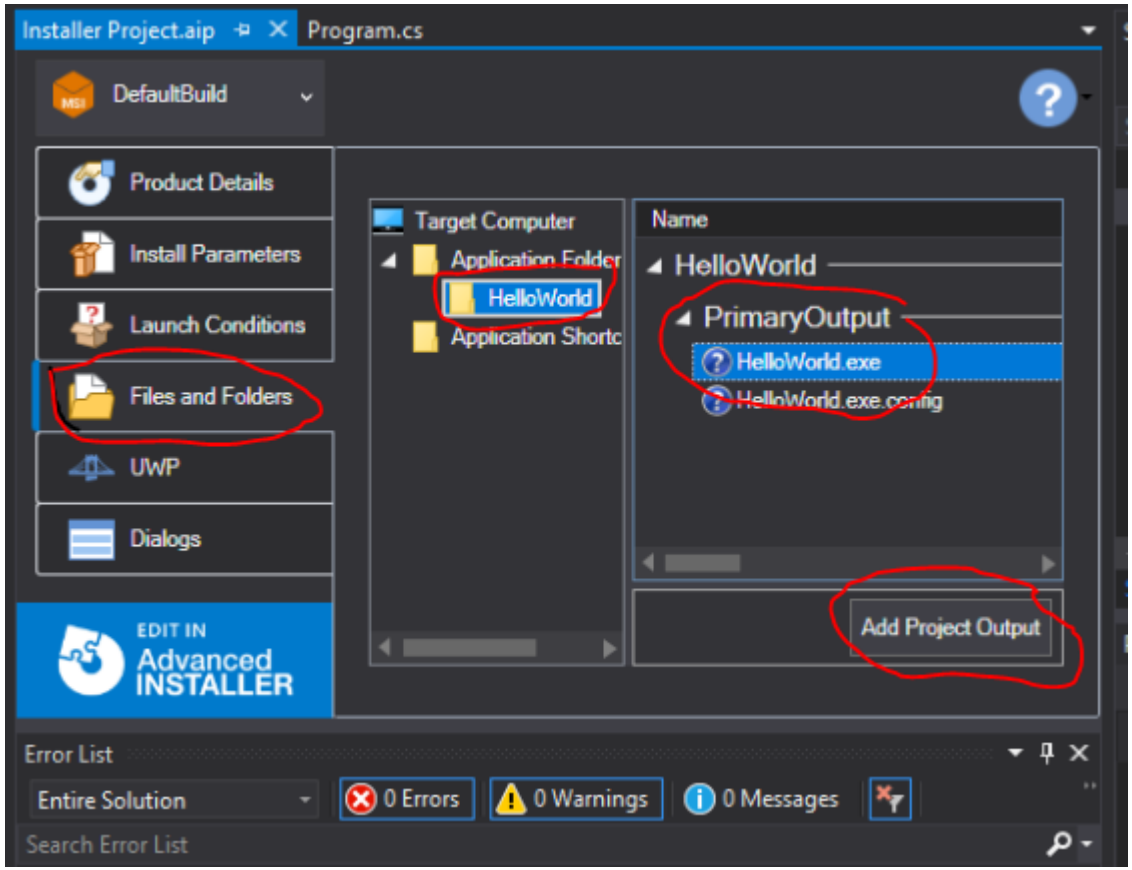
TẠO FILE SETUP

- Tạo mới project loại Advanced Installer Project



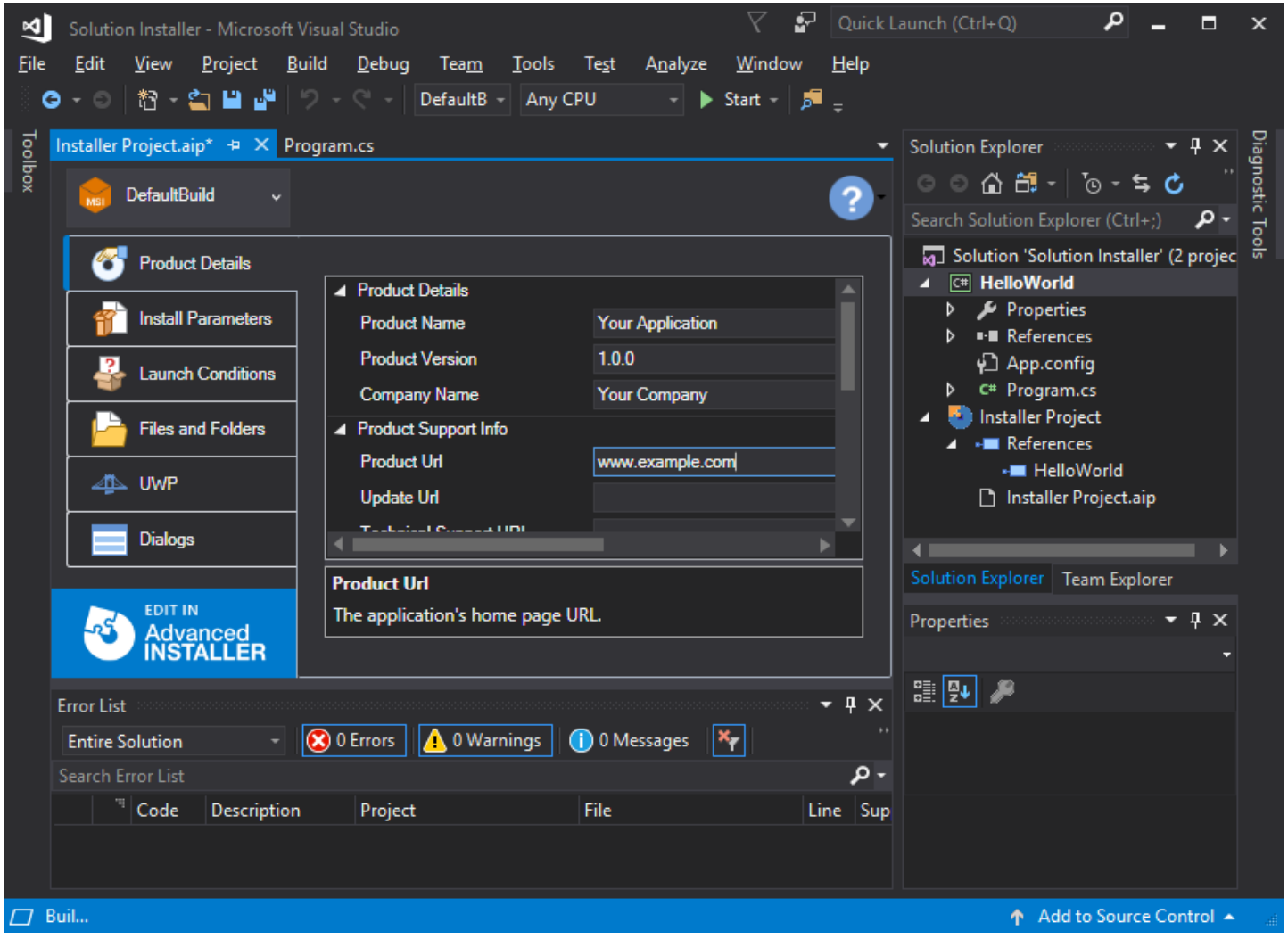
TẠO FILE SETUP

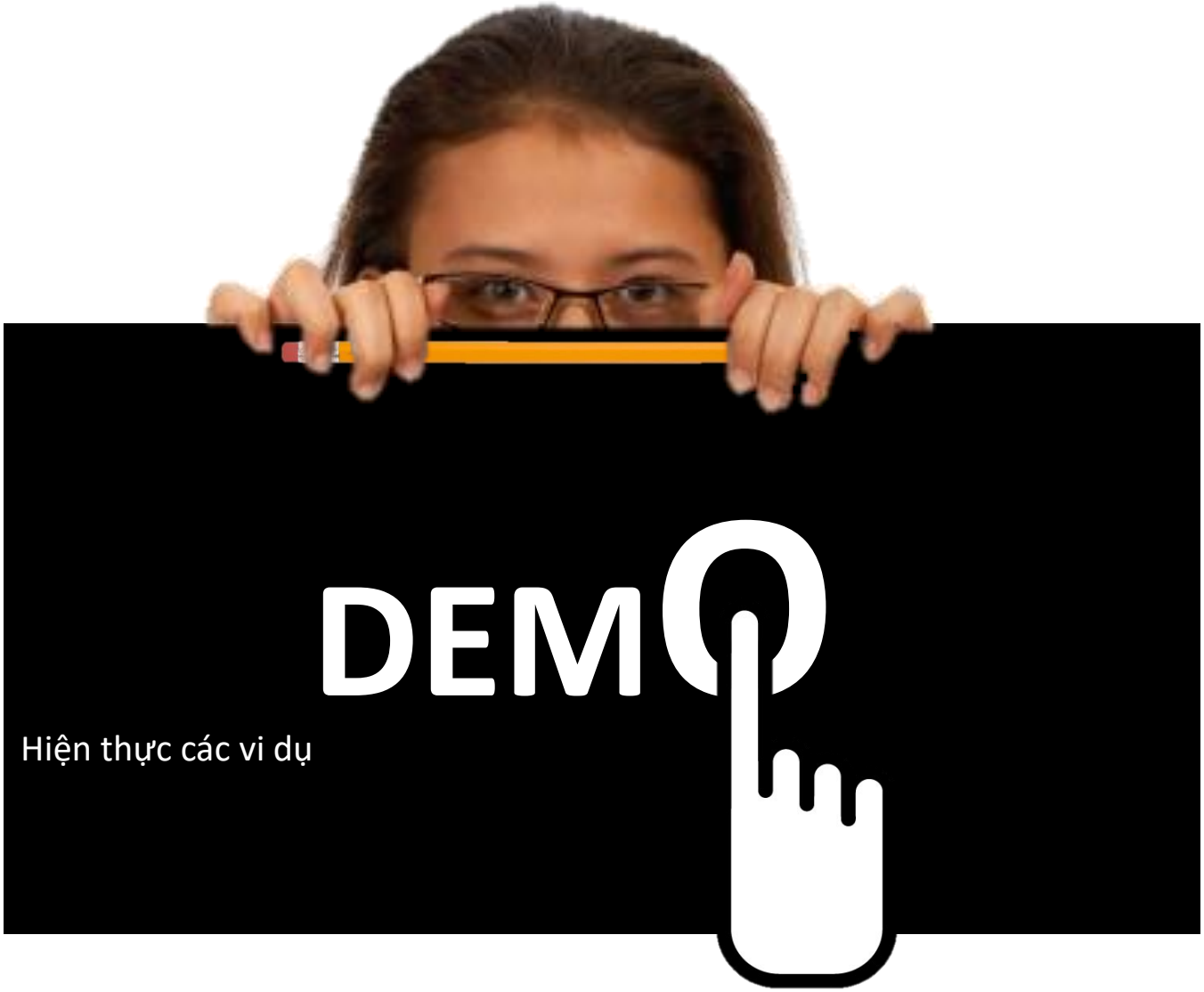
- Chọn "Primary Output" and "References" trong tab Installer Project.api



TẠO FILE SETUP

- Cập nhật các thông tin cho tập tin cài đặt




DEMO
Hiện thực các vi dụ

Tổng kết bài học

- Mô hình cấu trúc dự án
- 3 layer và database first

KẾT THÚC