

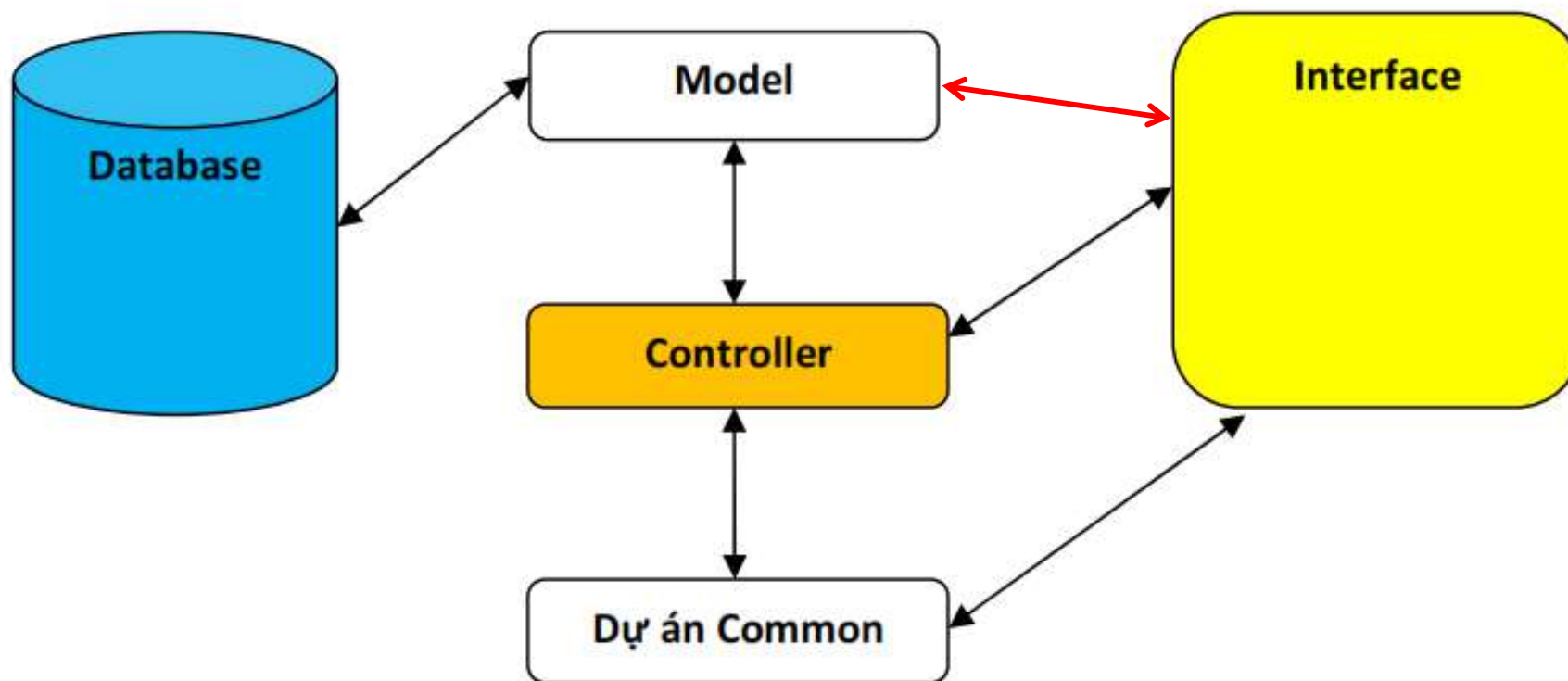
# Integrate Entity Framework to Web Project

Phần này mô tả cách tích hợp EF vào dự án Web trên thực tế.

EF nên đặt ở đâu, sử dụng như thế nào, dựa vào EF phát triển thêm một số phương thức phù hợp với yêu cầu.

# Integrate EF to Web Project

EF nên là dự án thư viện (DLL) riêng biệt và được nhúng vào ứng dụng.



# Integrate EF to Web Project

Trong mô hình trên (lưu ý không phải là MVC)

- **Model** chính là dự án EF (DLL)
- **Controller** kế thừa Model để phát triển các phương thức tùy biến (DLL)
- **Common** là dự án thư viện lưu trữ các phần khác liên quan đến dự án (DLL)
- **Interface** chính là dự án Web

Tùy theo yêu cầu cách đặt tên có thể khác nhau.

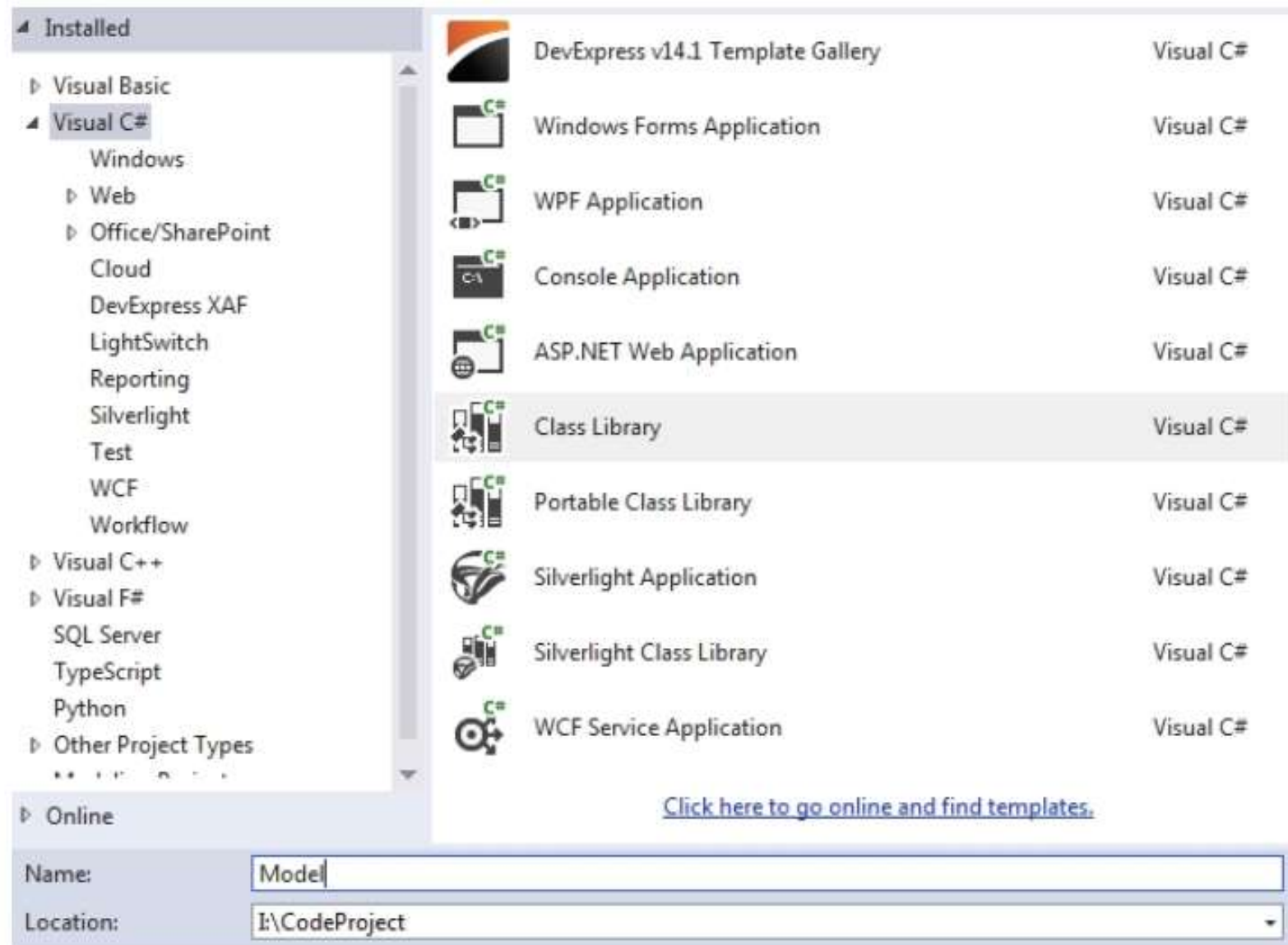
# Integrate EF to Web Project

## 1. Tạo dự án thư viện Model

Tạo dự án tên là CodeProject, trong dự án đặt tên dự án con là Model dạng dự án thư viện.

Gieo mã nguồn từ database theo dạng **Database First**, đặt tên Context là **EF**.

# Integrate EF to Web Project



# Integrate EF to Web Project

Trong tập tin **EF.Context.cs** chứa nội dung:

<pre>namespace Model {     using System;     using System.Data.Entity;     using System.ComponentModel.DataAnnotations.Schema;     using System.Linq;      public partial class EF : DbContext     {         public EF()             : base("name=EF")         {         }          public virtual DbSet&lt;Article&gt; Articles { get; set; }         public virtual DbSet&lt;Category&gt; Categories { get; set; }         public virtual DbSet&lt;Link&gt; Links { get; set; }          protected override void OnModelCreating(DbModelBuilder modelBuilder)         {         }     } }</pre>	<p>//Không gian dự án</p> <p>//Kế thừa Dbcontext để tương tác với database</p> <p>//Các bảng dữ liệu trong database tương ứng với các lớp.</p>
---	--

# Integrate EF to Web Project

Như vậy dự án Model đã tạo thành từ Entity Framework, dựa vào LINQ lập trình viên có thể thực hiện nhiều thao tác với database như:

- **Add, Delete, Insert, Update, Select**
- **và nhiều thao tác khác ....**

Tuy nhiên để tối ưu mã nguồn thì lập trình viên nên viết 1 dự án khác để tái sử dụng code tương tác cho nhiều trường hợp khác nhau.

# Integrate EF to Web Project

## 2. Tạo dự án Controller

Nếu lập trình viên vẫn có thói quen dùng truy vấn sql thay vì **LINQ** thì có thể dùng Dynamic **LINQ** để hỗ trợ 1 phần cách code cũ.

Tuy nhiên, không khuyến cáo dùng sql trên mô hình EF.




# Integrate EF to Web Project

Cài đặt Dynamic LINQ bằng cách vào Tools -> Library Package Manager -> Package Manager Console, gõ lệnh:

```
PM> Install-Package System.Linq.Dynamic.Library
```

# Integrate EF to Web Project

Ở database có bảng tên là Link chứa các liên kết  
**Link (LinkID, LinkName, LinkDescription, LinkParentID)** với mô tả

	Column Name	Data Type	Allow Nulls
	LinkID	int	<input type="checkbox"/>
	LinkName	nvarchar(50)	<input checked="" type="checkbox"/>
	LinkDescription	nvarchar(50)	<input checked="" type="checkbox"/>
	LinkURL	nvarchar(50)	<input checked="" type="checkbox"/>
	LinkParentID	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

# Integrate EF to Web Project

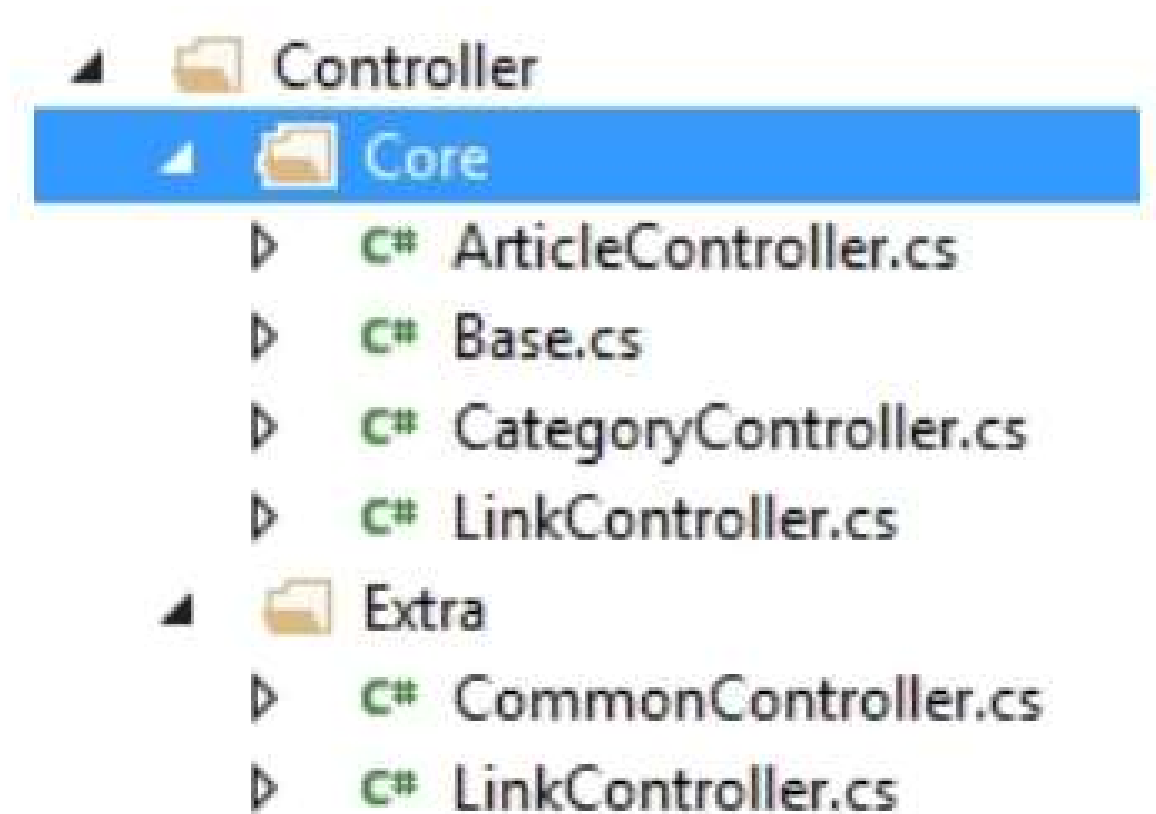
Ở thư viện **Model**, bảng Link được gieo thành lớp **Link.cs**

```
namespace CodeProject.Model
{
    using System;
    using System.Collections.Generic;

    public partial class Link
    {
        public int LinkID { get; set; }
        public string LinkName { get; set; }
        public string LinkDescription { get; set; }
        public string LinkURL { get; set; }
        public Nullable<int> LinkParentID { get; set; }
    }
}
```

# Integrate EF to Web Project

Ở dự án **Controller**, tổ chức kiến trúc thư mục như sau:



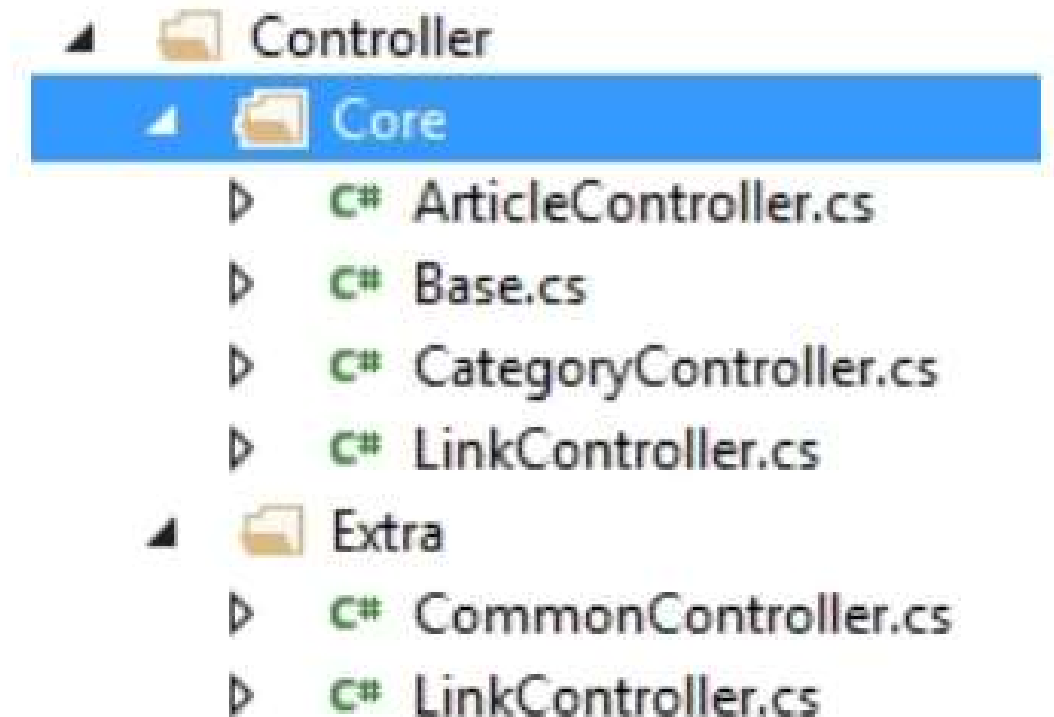
# Integrate EF to Web Project

Trong đó:

- **Thư mục Core** là nơi chứa các Controller, ví dụ có lớp Link.cs thì có LinkController.cs chứa trong thư mục này. LinkController.cs lưu trữ các phương thức thêm, xóa, sửa, ... cơ bản. Trong đó, **Base.cs** là lớp cha chứa Context để thực hiện thao tác tương tác với database.
- **Thư mục Extra** cũng là nơi chứa Controller, nhưng chứa các phương thức cấp cao hơn tùy theo yêu cầu dự án.

# Integrate EF to Web Project

LinkController.cs ở Core và LinkController.cs Extra có phải là 2 file khác nhau?



# Integrate EF to Web Project

Nội dung tập tin **LinkController.cs** ở **Core** chứa các phương thức thêm, xóa, sửa, ...

```
1 /// -----  
2 /// Author: Ta Hoang Thang  
3 /// IT Department, Dalat University, Vietnam  
4 /// Email: thangth@dlu.edu.vn, tahoangthang@gmail.com  
5 /// -----
```

# Integrate EF to Web Project

## LinkController.cs

```
6 using System;
7 using System.Collections.Generic;
8 using System.Data;
9 using System.Linq;
10 using System.Linq.Dynamic;
11 using System.Text;
12 using System.Threading.Tasks;
13 using Model;
14 using System.Data.Entity;
15 using System.Data.Entity.Infrastructure;
16 using Entity = Model.Link;
```



# Integrate EF to Web Project

## LinkController.cs

```
18 namespace Controller
19 {
    1 reference
20 public partial class LinkController : Base
21 {
22     public DbSet<Entity> Execute;
    0 references
23 public LinkController()
24 {
25     Execute = db.Links;
26 }
27
28 + Common execution
40
41 + Insert
68
69 + Delete queries
115
116 + Select queries
179
180 + Update
210 }
211 }
```

# Integrate EF to Web Project

## Base.cs

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Data.Entity;
6 using System.Collections;
7 using Model;
8 namespace Controller
9 {
10     45 references
    public abstract class Base
11     {
12         public EF db;
13         0 references
        public Base() { db = new EF(); }
14     }
15 }
16
```

# Integrate EF to Web Project

## Truy vấn sql ở LinkController.cs

```
#region Common execution
/// <summary>
/// Execute query string (if select always use * - select *) for this object only
/// </summary>
/// <param name="query">query</param>
/// <returns>bool</r
/// eturns>
0 references
public List<Entity> ExecuteQuery(string query)
{
    return Execute.SqlQuery(query).ToList();
}
#endregion
```

# Integrate EF to Web Project

## Insert ở LinkController.cs

```
/// <summary>
/// Insert an entity
/// </summary>
/// <param name="e">Entity</param>
/// <returns>bool</returns>
0 references
public bool Insert(Entity e)
{
    Execute.Add(e);
    return (db.SaveChanges() > 0);
}
```

```
/// <summary>
/// Insert entity list
/// </summary>
/// <param name="list">Entity list</param>
/// <returns>bool</returns>
0 references
public bool Insert(List<Entity> list)
{
    foreach (Entity e in list)
    {
        Execute.Add(e);
    }
    try { db.SaveChanges(); }
    catch { return false; }
    return true;
}
```

# Integrate EF to Web Project

## Delete ở LinkController.cs

```
/// <summary>
/// Delete by conditions, not use for Like operator
/// </summary>
/// <param name="conditions"></param>
/// <returns></returns>
0 references
public bool DeleteWhere(string conditions)
{
    Execute.RemoveRange(Execute.AsQueryable().Where(conditions).ToList());
    try { db.SaveChanges(); }
    catch { return false; }
    return true;
}
```

# Integrate EF to Web Project

## Delete ở LinkController.cs

```
/// <summary>  
/// Delete class object  
/// </summary>  
/// <param name="1">Class object</param>  
/// <returns>bool</returns>  
0 references  
public bool Delete(Entity e)  
{  
    Execute.Attach(e);  
    Execute.Remove(e);  
    try { db.SaveChanges(); }  
    catch { return false; }  
    return true;  
}
```

# Integrate EF to Web Project

## Delete ở LinkController.cs

```
/// <summary>
/// Delete object list
/// </summary>
/// <param name="list"></param>
/// <returns></returns>
0 references
public bool Delete(List<Entity> list)
{
    foreach (Entity e in list)
    {
        Execute.Attach(e);
        Execute.Remove(e);
    }
    try { db.SaveChanges(); }
    catch { return false; }
    return true;
}
#endregion
```

# Integrate EF to Web Project

## Update ở LinkController.cs

```
/// <summary>
/// Update an entity
/// </summary>
/// <param name="e">Entity</param>
/// <returns>bool</returns>
0 references
public bool Update(Entity e)
{
    Execute.Attach(e);
    db.Entry(e).State = EntityState.Modified;
    return (db.SaveChanges() > 0);
}
```



# Integrate EF to Web Project

## Update ở LinkController.cs

```
/// <summary>
/// Update entity list
/// </summary>
/// <param name="list">Entity list</param>
/// <returns>bool</returns>
0 references
public bool Update(List<Entity> list)
{
    foreach (Entity e in list)
    {
        Execute.Attach(e);
        db.Entry(e).State = EntityState.Modified;
    }
    try { db.SaveChanges(); }
    catch { return false; }
    return true;
}
```

# Integrate EF to Web Project

## Select ở LinkController.cs

```
/// <summary>
/// Select by conditions, not use for Like operator
/// </summary>
/// <param name="conditions"></param>
/// <returns></returns>
0 references
public List<Entity> SelectWhere(string conditions)
{
    return Execute.AsQueryable().Where(conditions).ToList();
}
```

# Integrate EF to Web Project

## Select ở LinkController.cs

```
/// <summary>  
/// Select by conditions and sort by orders, not use for Like operator  
/// </summary>  
/// <param name="conditions">conditions</param>  
/// <param name="orders">orders</param>  
/// <returns></returns>  
0 references  
public List<Entity> SelectOrderWhere(string conditions, string orders)  
{  
    return Execute.AsQueryable().Where(conditions).OrderBy(orders).ToList();  
}
```

# Integrate EF to Web Project

## Select ở LinkController.cs

```
/// <summary>  
/// Select all records  
/// </summary>  
/// <returns>List</returns>  
0 references  
public List<Entity> SelectAll()  
{  
    return Execute.ToList();  
}
```

# Integrate EF to Web Project

## Select ở LinkController.cs

```
/// <summary>
/// Select all by orders (Order = "LinkID ASC, LinkURL DESC")
/// Use Linq.Dynamic library
/// </summary>
/// <param name="Order">Orders string</param>
/// <returns>List</returns>
0 references
public List<Entity> SelectAll(string orders)
{
    return Execute.AsQueryable().OrderBy(orders).ToList();
}
```

# Integrate EF to Web Project

## Select ở LinkController.cs

```
/// <summary>
/// Select top
/// </summary>
/// <param name="number">the number of records</param>
/// <returns>List</returns>
0 references
public List<Entity> SelectTop(int number)
{
    return Execute.Take(number).ToList();
}
```

# Integrate EF to Web Project

## Select ở LinkController.cs

```
/// <summary>
/// Select top by orders
/// </summary>
/// <param name="number">the number of records</param>
/// <param name="orders">orders' string - example: LinkID ASC, LinkName DESC</param>
/// <returns>List</returns>
0 references
public List<Entity> SelectTop(int number, string orders)
{
    return Execute.AsQueryable().OrderBy(orders).Take(number).ToList();
}
#endregion
```

# Integrate EF to Web Project

Ngoài ra còn nhiều phương thức lấy dữ liệu khác tùy vào nhu cầu của dự án. Các phương thức đó có thể viết thêm ở **LinkController.cs** ở thư mục **Extra**.



# Integrate EF to Web Project

## 3. Sử dụng code

Lập trình có thể sử dụng dự án Model, Controller để tương tác với database, trong trường hợp “lười” thì có thể viết phương thức trực tiếp ở giao diện thông qua Model đã được EF gieo sẵn mà không cần phải cập nhật bên dự án Controller.