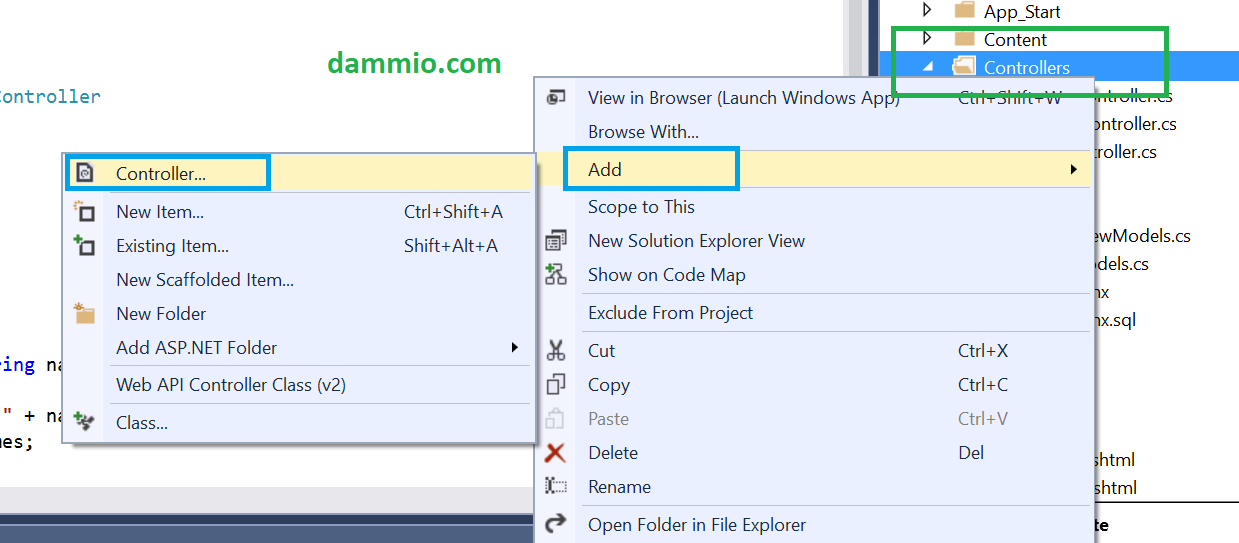
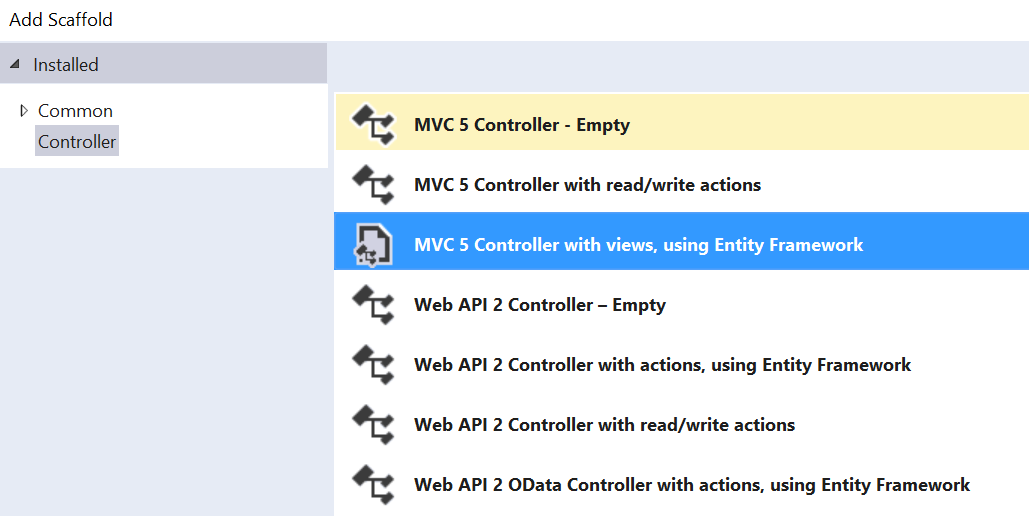
Phần 7: Truy cập mô hình dữ liệu từ Controller

Trong bài này, bạn sẽ tạo một lớp **LinkController** mới và viết code để lấy dữ liệu từ bảng Link trong cơ sở dữ liệu và hiển thị ra trình duyệt sử dụng một bản mẫu xem (view template).

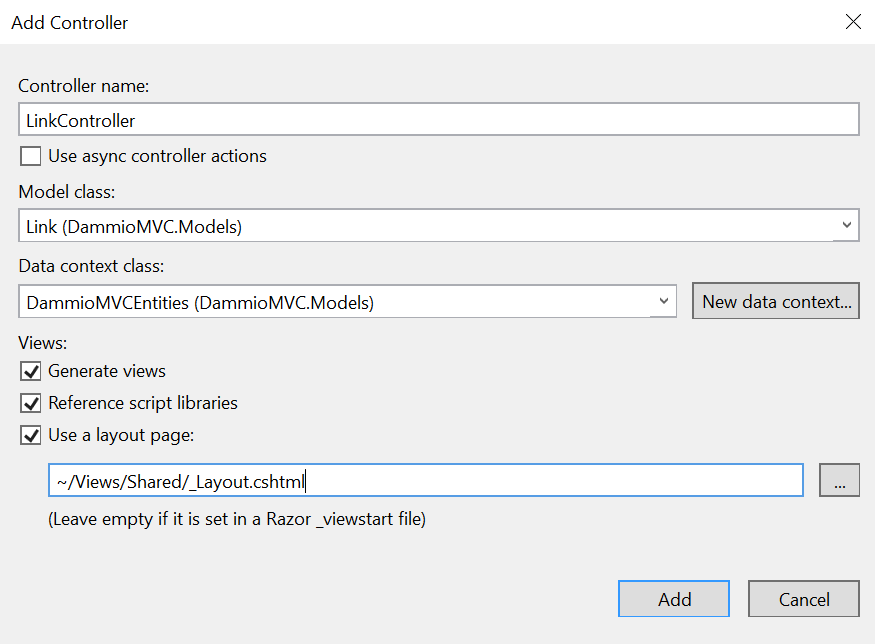
Tiếp theo ứng dụng ở bài trước, để tạo 1 Controller mới, ở Solution Explorer, chuột phải chọn thư mục Controllers, và nhấn Add, sau đó chọn Controller.



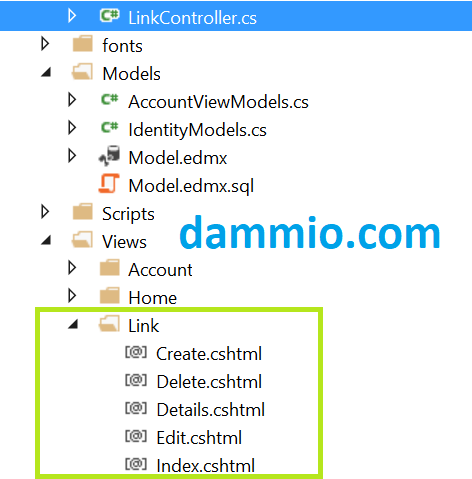
Ở hộp thoại **Add Scaffold**, chọn mục [MVC](https://www.dammio.com/glossary/mvc) 5 Controller with views (điều khiển [MVC](https://www.dammio.com/glossary/mvc) 5 với các view), sử dụng [Entity Framework](https://www.dammio.com/glossary/entity-framework) và nhấn Add.



Sau đó, đối với tên Controller, chọn tên là **LinkController**. Chọn Link (DammioMVC.Models) cho lớp Link và chọn DammioMVCEntities (DammioMVC.Models) cho lớp bối cảnh Data. Tiếp theo, nhấn Add, Visual Studio sẽ tạo các tập tin và thư mục. Bạn có thể dùng Layout mặc định nếu muốn (~/Views/Shared/\_Layout.cshtml).

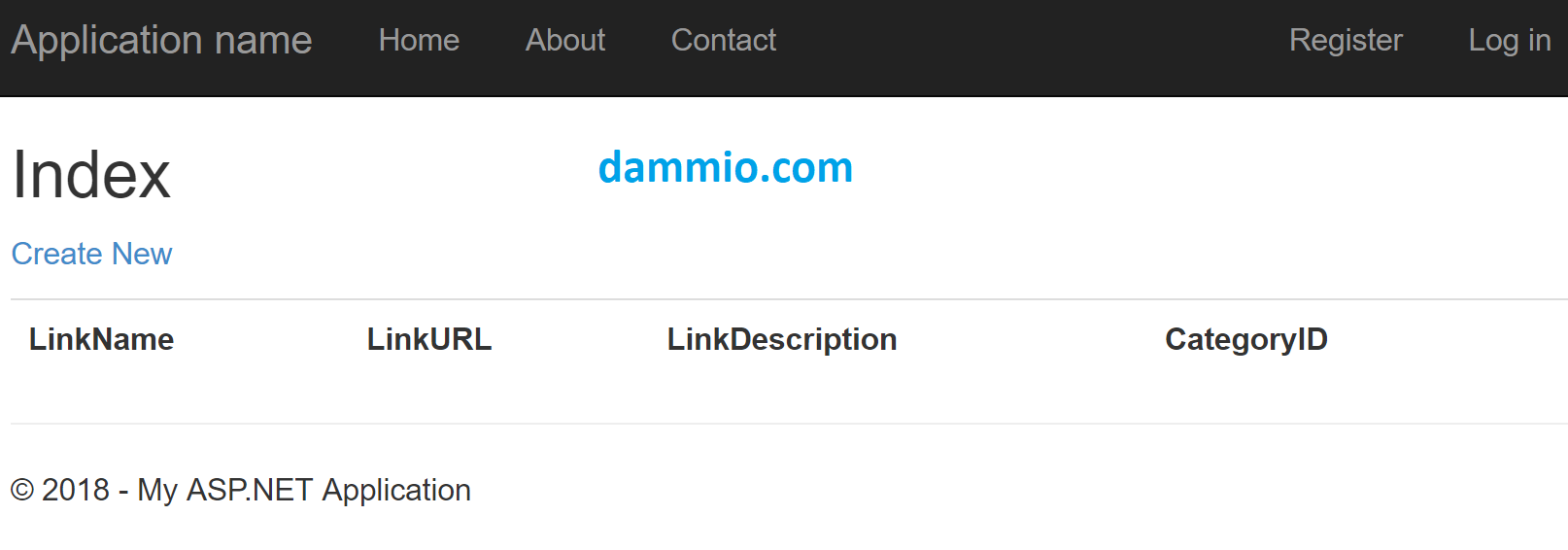


Sau đó, hãy đợi một chút, Visual Studio sẽ tự động tạo mã nguồn tập tin **LinkController.cs** trong thư mục Controllers và tạo thư mục Views/Link chứa các tập tin Create.cshtml, Delete.cshtml, Details.cshtml, Edit.cshtml, và Index.cshtml. Bạn có thể bật cửa sổ Solution Explorer (Ctrl + Alt + L) để xem tổng quan các tập tin mới tạo.



Visual Studio sẽ tự động tạo các phương thức CRUD (create, read, update, và delete) và kèm theo giao diện (views) vì vậy với ASP.NET [MVC](https://www.dammio.com/glossary/mvc), bạn có thể thực hiện làm giao diện trang Admin rất nhanh.

Sau khi tạo xong LinkController, bạn chạy dự án web với đường dẫn **http://localhost:xxxx/Link/** để mở giao diện quản lý bảng Link mặc định. Do chưa có phương thức nào được mô tả, vì vậy server sẽ chạy phương thức Index được mô tả trong App\_Start/RouteConfig.cs, bạn có thể thấy http://localhost:xxxx/Link/ tương đương http://localhost:xxxx/Link/Index.

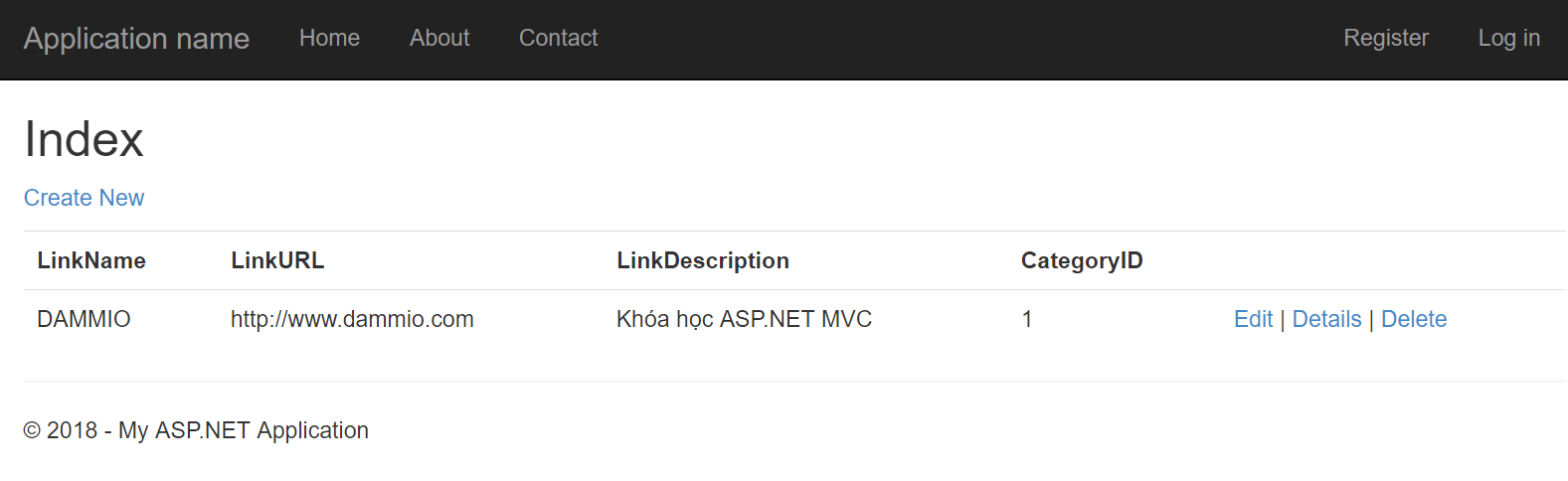


# Tạo một Link mới

Để tạo một Link mới, trong giao diện đường dẫn http://localhost:xxxx/Link/, bạn chọn liên kết Create New, sau đó trang http://localhost:xxxx/Link/Create sẽ hiển thị, bạn điền thông tin và thêm mới 1 Link.



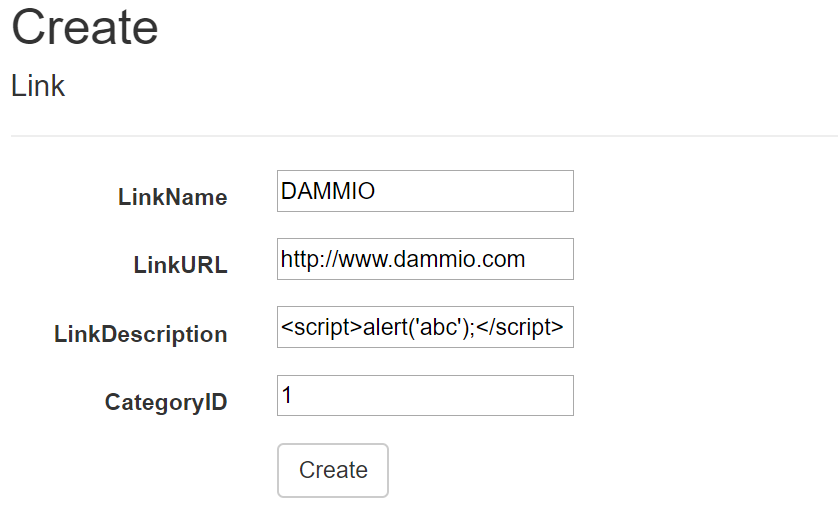
Sau khi điền thông tin, bạn nhấn nút Create để gửi dữ liệu về server, nếu tạo thành công thì trang sẽ Redirect về trang Index với đường dẫn http://localhost:xxxx/Link/ chứa một dòng Link mới tạo.



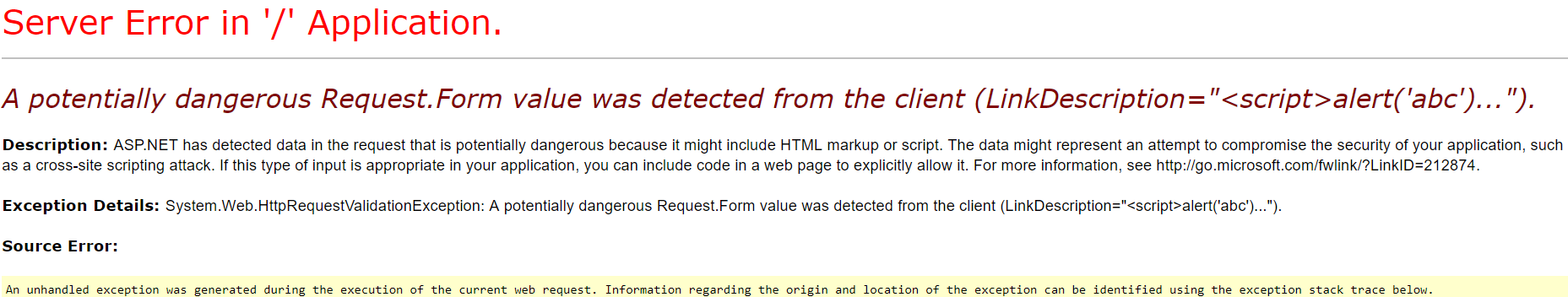
Với Link mới tạo, bạn có thấy các liên kết Edit (chỉnh sửa), Details (xem chi tiết) và Delete (xóa), bạn có thể thử các thao tác này để biết các giao diện chức năng khác mà Visual Studio đã tạo sẵn.

# Thử chèn mã HTML

Bạn thử tạo 1 Link mới ở đường dẫn http://localhost:xxxx/Link/Create, sau đó chúng ta thử chèn đoạn JavaScript nhỏ để xem ASP.NET [MVC](https://www.dammio.com/glossary/mvc) có khả năng bảo mật?



Sau đó nhấn nút Create, bạn sẽ thấy lỗi xuất hiện để chặn mã HTML chèn vào database.



# Kiểm tra mã nguồn

## LinkController.cs

Đến đây, bạn mở xem nội dung tập tin Controllers\LinkController.cs, phương thức Index được gieo ra với đoạn mã như sau.

public class LinkController : Controller

    {

        private DammioMVCEntities db = new DammioMVCEntities();

        // GET: /Link/

        public ActionResult Index()

        {

            return View(db.Links.ToList());

        }

...

}

Trong phương thức Index, bạn có thể thấy phương thức Index có giá trị trả về là **View(db.Links.ToList())**, tức là danh sách các Link trong bảng Link ở cơ sở dữ liệu. Trong đó, db là 1 biến DbContext, db.Links là bảng Link, db.Links.ToList() là trả bảng Link về một danh sách kiểu List.

Bạn có thể thấy thêm đoạn **private DammioMVCEntities db = new DammioMVCEntities();** chính là khai báo biến db kiểu DbContext (như đã giải thích) dùng để tương tác (thêm, xóa, sửa, cập nhật) trong cơ sở dữ liệu.

Trong các bài trước, bạn học được cách một controller đẩy dữ liệu hay các đối tượng về bản mẫu xem (view template) bằng đối tượng ViewBag. Xin giải thích thêm ViewBag là một đối tượng động cung cấp cách ràng buộc trễ thuận tiện để đẩy thông tin tới một view bất kỳ.

Tiếp tục ở tập tin Controllers\LinkController.cs, nhưng bạn hãy xem phương thức Details được gieo sẵn như sau.

// GET: /Link/Details/1

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Link link = db.Links.Find(id);

if (link == null)

{

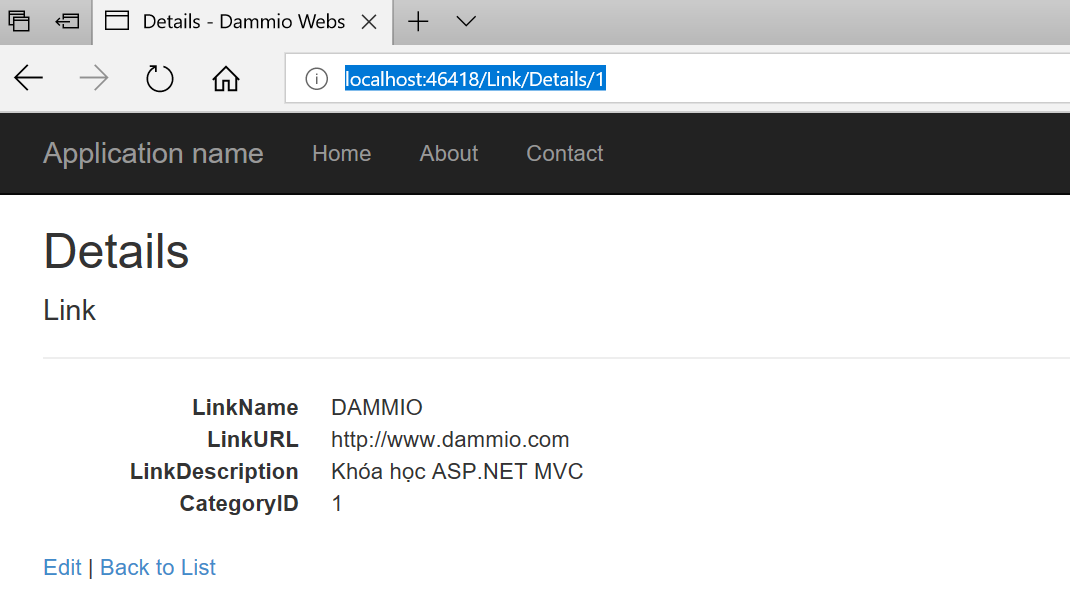
return HttpNotFound();

}

return View(link);

}

Phương thức Details dùng để hiển thị nội dung chi tiết của 1 Link nào đó. Phương thức Details dùng chỉ số id để dò tìm LinkID trong bảng Link. Ví dụ đường dẫn http://localhost:xxxx/link/details/1 có nghĩa là hiển thị thông tin Link có chỉ số LinkID = 1, nếu không tìm thấy thì sẽ hiển thị trống. Trong hình sau, bạn có thể thấy nội dung đường dẫn http://localhost:xxxx/link/details/1.



Ngoài lề 1 tý, kiểu **int?** có nghĩa là biến có thể là int hoặc là null.

int i1 = 1; //OK

int i2 = null; //not OK

int? i3 = 1; //OK

int? i4 = null; //OK, int? cho phép biến bằng null

Nếu một Link được tìm thấy theo chỉ số id, một thể hiện mô hình Link được đẩy sang View bằng dòng return View(link).

## Views\Link\Details.cshtml

Tiếp theo, bạn mở tập tin Views\Link\Details.cshtml để xem nội dung chứa gì.

@model DammioMVC.Models.Link

@{

ViewBag.Title = "Details";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Details</h2>

<div>

<h4>Link</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.LinkName)

</dt>

<dd>

@Html.DisplayFor(model => model.LinkName)

</dd>

<dt>

@Html.DisplayNameFor(model => model.LinkURL)

</dt>

<dd>

@Html.DisplayFor(model => model.LinkURL)

</dd>

<dt>

@Html.DisplayNameFor(model => model.LinkDescription)

</dt>

<dd>

@Html.DisplayFor(model => model.LinkDescription)

</dd>

<dt>

@Html.DisplayNameFor(model => model.CategoryID)

</dt>

<dd>

@Html.DisplayFor(model => model.CategoryID)

</dd>

</dl>

</div>

<p>

@Html.ActionLink("Edit", "Edit", new { id = Model.LinkID }) |

@Html.ActionLink("Back to List", "Index")

</p>

Đầu tiên, bạn thấy mệnh đề @model ở đầu trang mô tả đối tượng mà View cần hiển thị. Trong ví dụ này, đối tượng cần hiển thị là 1 Link với đoạn mã @model DammioMVC.Models.Link phía trên. Một số cú pháp khác như @Html.ActionLink, @Html.DisplayFor,… là **cú pháp Razor** mà bạn cần học ở ASP.NET [MVC](https://www.dammio.com/glossary/mvc). Mục đích của cú pháp Razor giúp bạn rút gọn mã HTML lúc lập trình.

Chỉ thị @model cho phép bạn truy cập một Link mà controller đẩy sang View sử dụng một đối tượng có kiểu mạnh Model. Ví dụ trên, trong bản mẫu Details.cshtml, đoạn mã đẩy mỗi trường Link sang các HTML Helper là DisplayNameFor và DisplayFor với dối tượng Model kiểu mạnh. Các phương thức Create và Edit và các bản mẫu View cũng đẩy một đối tượng mô hình Link tương tự.

## Views/Link/Index.cshtml & phương thức Index trong tập tin LinkController.cs

Tiếp theo bạn mở nội dung tập tin Index.cshtml và phương thức Index trong tập tin LinkController.cs ra xem.

Phương thức Index như sau.

public ActionResult Index()

{

    return View(db.Movies.ToList());

}

Phương thức này trả về danh sách Link trong cơ sở dữ liệu bằng cú pháp View(db.Movies.ToList()).

Tiếp theo, nội dung tập tin Index.cshmtl như sau.

@model IEnumerable<DammioMVC.Models.Link>

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Index</h2>

<p>

@Html.ActionLink("Create New", "Create")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.LinkName)

</th>

<th>

@Html.DisplayNameFor(model => model.LinkURL)

</th>

<th>

@Html.DisplayNameFor(model => model.LinkDescription)

</th>

<th>

@Html.DisplayNameFor(model => model.CategoryID)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.LinkName)

</td>

<td>

@Html.DisplayFor(modelItem => item.LinkURL)

</td>

<td>

@Html.DisplayFor(modelItem => item.LinkDescription)

</td>

<td>

@Html.DisplayFor(modelItem => item.CategoryID)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id=item.LinkID }) |

@Html.ActionLink("Details", "Details", new { id=item.LinkID }) |

@Html.ActionLink("Delete", "Delete", new { id=item.LinkID })

</td>

</tr>

}

</table>

Tương tự như tập tin Details.cshtml, bạn cũng thấy nội dung tập tin Index.cshtml có chứa chỉ thỉ @model mô tả đối tượng DammioMVC.Models.Link. Bởi vì đối tượng Model là một kiểu mạnh, vì vậy mỗi đối tượng trong vòng lặp (@foreach) được đánh kiểu là Link và lần lượt hiển thị các thông tin của các Link.

# Làm việc với SQL Server LocalDB

Nếu bạn dùng [Entity Framework](https://www.dammio.com/glossary/entity-framework) theo kiểu Code First, khi dò tìm kết nối database mà trỏ về một database chưa tồn tại, khi đó Code First sẽ tạo database này tự động. Điều quan trọng là bạn phải tìm được database này nằm đâu. Thông thường, bạn mở thư mục App\_Data xem có tập tin **\*.mdf** nào không, nếu không thì refresh thư mục thử lại. Nếu bạn dùng các phương thức khác để tạo [EF](https://www.dammio.com/glossary/ef) như DatabaseFirst, ModelFirst, thử tìm trong Web.config xem đường dẫn tới server cơ sở dữ liệu.

Ví dụ, dự án DammioMVC có đường dẫn database trong tập tin Web.config như sau:

<add name="DammioMVCEntities" connectionString="metadata=res://\*/Models.Model.csdl|res://\*/Models.Model.ssdl|res://\*/Models.Model.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=(local);initial catalog=DammioMVC;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

Nhìn vào đoạn mã trên, bạn có thể thấy server database là (local), tức là server mặc định, với tên là database là DammioMVC.

1. **Kết luận**

Bài viết đã hướng dẫn bạn cách tạo một Controller và các Views để thực hiện các thao tác xem, xóa, sửa, cập nhật tương ứng với 1 bảng trong cơ sở dữ liệu. Tất cả các bước này đều do Visual Studio gieo mã nguồn cho bạn, rất tiện phải không nào. Việc cần làm là bạn cần tìm hiểu rõ cấu trúc các tập tin này. Do đó, mời bạn theo dõi bài tiếp theo để có thể hiểu rõ hơn về ASP.NET [MVC](https://www.dammio.com/glossary/mvc).

[1]. [DammioMVC\_Phan7.rar](https://www.dammio.com/wp-content/uploads/2018/07/DammioMVC_Phan7.rar)