

Bài 4

Tìm hiểu về Models

Mục tiêu

- Giới thiệu Model
- Tạo Model
- Chuyển dữ liệu Model ra View
- Sử dụng Strong Typing
- Binding Data
- Sử dụng tính năng Scaffolding trong Visual Studio

Giới thiệu Model

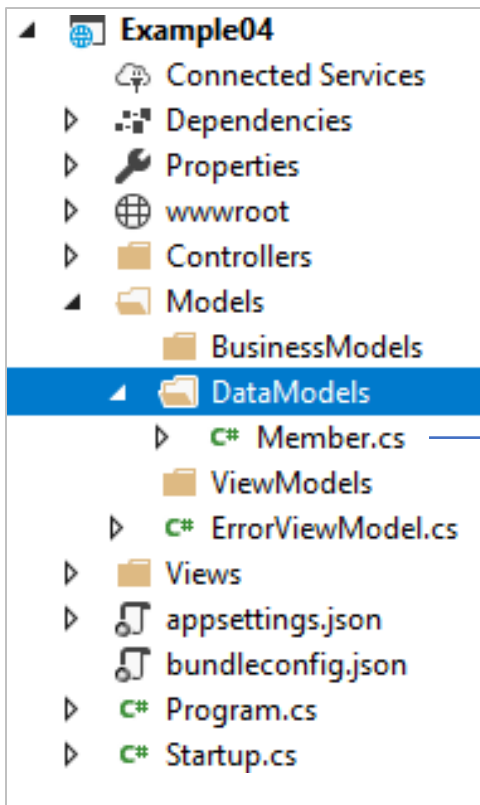
- Model là thành phần biểu diễn dữ liệu của ứng dụng. Nó được chia làm ba loại:

DataModel: là một lớp chứa các thuộc tính biểu diễn dữ liệu của ứng dụng. Các đối tượng của nó có thể nhận dữ liệu hoặc lưu trữ dữ liệu từ các nguồn khác nhau như database...

BusinessModel: là một lớp chứa các chức năng biểu diễn các nghiệp vụ logic của ứng dụng như add, save, delete, show, search, paging,...

ViewModel: là một lớp chứa các thuộc tính biểu diễn dữ liệu được truyền tải giữa controller và View

Tạo Model và đưa dữ liệu dạng Object ra View



```
public class Member
{
    public string MemberId { get; set; }
    public string Username { get; set; }
    public string FullName { get; set; }
    public string Password { get; set; }
    public string Email { get; set; }
}

public class MemberController : Controller
{
    public IActionResult Index()
    {
        //tạo đối tượng sử dụng Model Member
        var member = new Member();
        //gán các thông tin cho đối tượng
        member.MemberId = Guid.NewGuid().ToString();
        member.Username = "chungchanchat";
        member.FullName = "Lại Đức Chung";
        member.Password = "123456";
        member.Email = "chungld@bachkhoa-aptech.edu.vn";
        //truyền ra view qua đối tượng ViewBag
        ViewBag.member = member;
        return View();
    }
}
```

```
<h2>Thông tin Member</h2>
<p>Member Id: @ViewBag.member.MemberId</p>
<p>Username: @ViewBag.member.Username</p>
<p>Password: @ViewBag.member.Password</p>
<p>FullName: @ViewBag.member.FullName</p>
<p>Email: @ViewBag.member.Email</p>
```

Example04 Home About Contact

Thông tin Member

Member Id: 89946aa8-8ff2-49a6-bd10-6900b0f9c14a

Username: chungchanchat

Password: 123456

FullName: Lại Đức Chung

Email: chungld@bachkhoa-aptech.edu.vn

Tạo Model và đưa dữ liệu dạng List ra View

```
public IActionResult GetMembers()
{
    List<Member> members = new List<Member>()
    {
        new Member{MemberId=Guid.NewGuid().ToString(),Username="member1",FullName="Thành viên 1",Email="tv1@gmail.com",Password="123456"},
        new Member{MemberId=Guid.NewGuid().ToString(),Username="member2",FullName="Thành viên 2",Email="tv2@gmail.com",Password="123456"},
        new Member{MemberId=Guid.NewGuid().ToString(),Username="member3",FullName="Thành viên 3",Email="tv3@gmail.com",Password="123456"},
        new Member{MemberId=Guid.NewGuid().ToString(),Username="member4",FullName="Thành viên 4",Email="tv4@gmail.com",Password="123456"},
        new Member{MemberId=Guid.NewGuid().ToString(),Username="member5",FullName="Thành viên 5",Email="tv5@gmail.com",Password="123456"},
    };
    ViewBag.members = members;
    return View();
}
```

Example04 Home About Contact

Danh sách thành viên

Mã TV	Username	Họ và tên	Password	Email
f71e3815-fc1f-4727-a7fc-b402bc3b9ab7	member1	Thành viên 1	123456	tv1@gmail.com
3a259041-98c3-4975-aea5-d1096de7f337	member2	Thành viên 2	123456	tv2@gmail.com
3b954053-aa8c-44be-9f93-e36fdf561f75	member3	Thành viên 3	123456	tv3@gmail.com
6c2b861d-49d7-420d-8211-5d0f2280d69f	member4	Thành viên 4	123456	tv4@gmail.com
909ab4bd-e8ef-4bb5-80b9-64f8ccaddb97	member5	Thành viên 5	123456	tv5@gmail.com

```
<h2>Danh sách thành viên</h2>
<table class="table table-bordered">
  <tr>
    <th>Mã TV</th>
    <th>Username</th>
    <th>Họ và tên</th>
    <th>Password</th>
    <th>Email</th>
  </tr>
  @foreach (var m in ViewBag.members)
  {
    <tr>
      <td>@m.MemberId</td>
      <td>@m.Username</td>
      <td>@m.FullName</td>
      <td>@m.Password</td>
      <td>@m.Email</td>
    </tr>
  }
</table>
```

Đưa dữ liệu ra View qua phương thức View()

- Ngoài **ViewBag**, **ViewData**, **TempData** chúng ta có thể chuyển dữ liệu từ controller ra view thông qua **phương thức View** và nhận dữ liệu trên View qua **biến Model**.

```
public IActionResult Index()
{
    //tạo đối tượng sử dụng Model Member
    var member = new Member();
    //gán các thông tin cho đối tượng
    member.MemberId = Guid.NewGuid().ToString();
    member.Username = "chungchachat";
    member.FullName = "Lại Đức Chung";
    member.Password = "123456";
    member.Email = "chungld@bachkhoa-aptech.edu.vn";
    //truyền ra view qua đối tượng ViewBag
    // ViewBag.member = member;
    //truyền dữ liệu ra view qua phương thức View(data)
    return View(member);
}
```

```
<h2>Thông tin Member</h2>
<p>Member Id: @Model.MemberId</p>
<p>Username: @Model.Username</p>
<p>Password: @Model.Password</p>
<p>FullName: @Model.FullName</p>
<p>Email: @Model.Email</p>
```

Example04 Home About Contact

Thông tin Member

Member Id: 93e3e797-baf5-431b-a4c3-88ec1286a3d2

Username: chungchachat

Password: 123456

FullName: Lại Đức Chung

Email: chungld@bachkhoa-aptech.edu.vn

Sử dụng Strong Typing

- Trong khi đưa dữ liệu Model từ Controller ra View thì View không xác định được chính xác kiểu dữ liệu. Chúng ta có thể sử dụng 1 trong 2 cách sau để định kiểu tường minh cho dữ liệu trên View.

1

Chuyển đổi kiểu dữ liệu

```
var var_name=ViewBag | Model as DataType
```

2

Định kiểu cho biến Model trên View

```
@model DataType
```

Sử dụng Strong Typing

```
public IActionResult Index()
{
    //tạo đối tượng sử dụng Model Member
    var member = new Member();
    //gán các thông tin cho đối tượng
    member.MemberId = Guid.NewGuid().ToString();
    member.Username = "chungchachat";
    member.FullName = "Lại Đức Chung";
    member.Password = "123456";
    member.Email = "chungld@bachkhoa-aptech.edu.vn";
    //truyền ra view qua đối tượng ViewBag
    // ViewBag.member = member;
    //truyền dữ liệu ra view qua phương thức View(data)
    return View(member);
}
```

@{

```
ViewData["Title"] = "Thông tin thành viên";
```

```
//Nếu đưa dữ liệu qua ViewBag thì chuyển kiểu
```

```
// var member = ViewBag.member as Example04.Models.DataModels.Member;
```

```
//Nếu đưa dữ liệu qua Model thì chuyển kiểu
```

```
var member = Model as Example04.Models.DataModels.Member;
```

}

<h2>Thông tin thành viên</h2>

<p>Member Id: @member.MemberId</p>

<p>FullName: @member.F</p>

- Email
- Equals
- FullName
- GetHashCode
- GetType
- MemberId
- Password
- ToString
- Username

Sử dụng Strong Typing

<!--Nếu đưa dữ liệu qua Model chúng ta còn có thể sử dụng cách sau-->

@model Example04.Models.DataModels.Member

@{

ViewData["Title"] = "Thông tin thành viên";

//Nếu đưa dữ liệu qua ViewBag thì chuyển kiểu

// var member = ViewBag.member as Example04.Models.DataModels.Member;

//Nếu đưa dữ liệu qua Model thì chuyển kiểu

var member = Model as Example04.Models.DataModels.Member;

}

<h2>Thông tin thành viên</h2>

<p>Member Id: @member.MemberId</p>

<p>FullName: @Model.</p>

- Email
- Equals
- FullName
- GetHashCode
- GetType
- MemberId
- Password
- ToString
- Username

Khi sử dụng tính năng Scaffolding visual studio sẽ dùng cách này

Binding Data

- Binding data là phương pháp gắn kết phần tử **Input HTML** với thuộc tính của **Model** trong **Strongly Typed View** và ngược lại **Map** các thông tin được **submit** trên **form** tới các thuộc tính trong đối tượng **Model**.
- Quá trình này được diễn ra tự động bởi trình **Model Binder** trong **ASP.NET Core MVC**. Nó tự động xác định **kiểu dữ liệu** và **Annotation** của **thuộc tính** trong **Model** để sinh ra kiểu của **Input HTML** phù hợp.
- Chúng ta có thể dùng cả **HTML Helper** và **Tag Helper** để binding data.

Binding Data

Khai báo danh sách member trong lớp MemberController

```
public static readonly List<Member> members = new List<Member>()
```

```
{
    new Member{MemberId=Guid.NewGuid().ToString(),Username="member1",FullName="Thành viên 1",Email="tv1@gmail.com",Password="123456"},
    new Member{MemberId=Guid.NewGuid().ToString(),Username="member2",FullName="Thành viên 2",Email="tv2@gmail.com",Password="123456"},
    new Member{MemberId=Guid.NewGuid().ToString(),Username="member3",FullName="Thành viên 3",Email="tv3@gmail.com",Password="123456"},
    new Member{MemberId=Guid.NewGuid().ToString(),Username="member4",FullName="Thành viên 4",Email="tv4@gmail.com",Password="123456"},
    new Member{MemberId=Guid.NewGuid().ToString(),Username="member5",FullName="Thành viên 5",Email="tv5@gmail.com",Password="123456"},
};
```

```
public IActionResult GetMembers()
```

```
{
    ViewBag.members = members;
    return View();
}
```

//Default là GET

```
public IActionResult Create()
```

```
{
    return View();
}
```

[HttpPost] //hành động gọi ứng với method là post

```
public IActionResult Create(Member member)
```

```
{
    member.MemberId = Guid.NewGuid().ToString();
    members.Add(member);
    return RedirectToAction("GetMembers");
}
```

```
<h2>Danh sách thành viên</h2>
```

```
<a href="Create" class="btn btn-primary">Thêm mới</a>
```

```
<table class="table table-bordered">
```

```
<tr>
```

```
<th>Mã TV</th>
```

```
<th>Username</th>
```

```
<th>Họ và tên</th>
```

```
<th>Password</th>
```

```
<th>Email</th>
```

```
</tr>
```

```
@foreach (var m in ViewBag.members)
```

```
{
```

```
<tr>
```

```
<td>@m.MemberId</td>
```

```
<td>@m.Username</td>
```

```
<td>@m.FullName</td>
```

```
<td>@m.Password</td>
```

```
<td>@m.Email</td>
```

```
</tr>
```

```
}
```

```
</table>
```

Binding Data

```
public IActionResult GetMembers()
{
    ViewBag.members = members;
    return View();
}
//Default là GET
public IActionResult Create()
{
    return View();
}

[HttpPost] //hành động gọi ứng với method là post
public IActionResult Create(Member member)
{
    member.MemberId = Guid.NewGuid().ToString();
    members.Add(member);
    return RedirectToAction("GetMembers");
}
```

```
@model Example04.Models.DataModels.Member
<h2>Thêm mới thành viên</h2>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Create">
            <div class="form-group">
                <label asp-for="MemberId" class="control-label"></label>
                <input asp-for="MemberId" class="form-control" readonly />
            </div>
            <div class="form-group">
                <label asp-for="Username" class="control-label"></label>
                <input asp-for="Username" class="form-control" />
            </div>
            <div class="form-group">
                <label asp-for="FullName" class="control-label"></label>
                <input asp-for="FullName" class="form-control" />
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-label"></label>
                <input asp-for="Password" class="form-control" />
            </div>
            <div class="form-group">
                <label asp-for="Email" class="control-label"></label>
                <input asp-for="Email" class="form-control" />
            </div>
            <div class="form-group">
                <input type="submit" value=" Lưu " class="btn btn-primary" />
            </div>
        </form>
    </div>
</div>
```


Binding Data

HTML code sinh ra tại browser

```
<div class="col-md-4">
  <form action="/Member/Create" method="post">
    <div class="form-group">
      <label class="control-label" for="MemberId">M&#xE3; th&#xE0;nh vi&#xEA;n</label>
      <input class="form-control" readonly type="text" id="MemberId" name="MemberId" value="" />
    </div>
    <div class="form-group">
      <label class="control-label" for="Username">T&#xEA;n &#x111;&#x103;ng nh&#x1EAD;p</label>
      <input class="form-control" type="text" id="Username" name="Username" value="" />
    </div>
    <div class="form-group">
      <label class="control-label" for="FullName">H&#x1ECD; v&#xE0; t&#xEA;n</label>
      <input class="form-control" type="text" id="FullName" name="FullName" value="" />
    </div>
    <div class="form-group">
      <label class="control-label" for="Password">M&#x1EAD;t kh&#x1EA9;u</label>
      <input class="form-control" type="password" id="Password" name="Password" />
    </div>
    <div class="form-group">
      <label class="control-label" for="Email">Email</label>
      <input class="form-control" type="email" id="Email" name="Email" value="" />
    </div>
    <div class="form-group">
      <input type="submit" value=" Lưu " class="btn btn-primary" />
    </div>
    <input name="__RequestVerificationToken" type="hidden" value="CfDJ8Mh_TEbUMPdFkKxbTt1YxG0i0BFa2esZK
T80eTiwVn15gf0LFW6_eWZ13QfgX4BuLk35tskZdrxWCUpre0mvC2N2QkHnaFoHnnqRf0py6b64ZK22BoN7es" /></form>
  </div>
```

HTML code sinh ra phù hợp với phần khai báo kiểu dữ liệu và Annotation của thuộc tính trong Model

Form tự động sinh input hidden chứa giá trị mã hóa tránh giả mạo request (bảo mật xét sau)

Thêm mới thành viên

Mã thành viên

Tên đăng nhập

Họ và tên

Mật khẩu

Email

Lưu

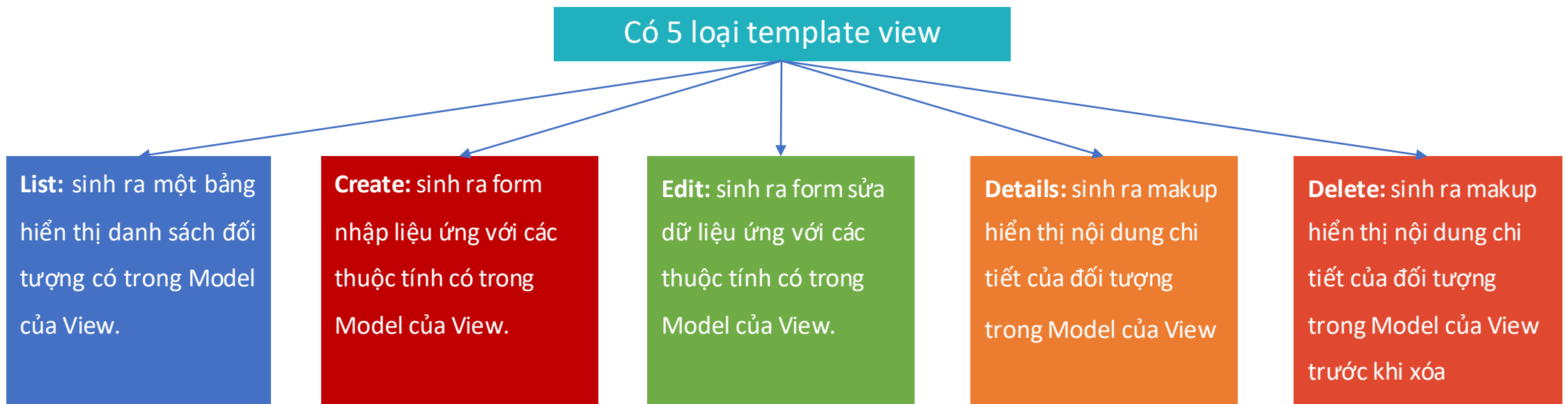
Danh sách thành viên

Thêm mới

Mã TV	Username	Họ và tên	Password	Email
840018fa-7e71-49f5-8ef1-5f178651ef37	member1	Thành viên 1	123456	tv1@gmail.com
f5d0dbec-5baf-42f8-97e6-cdef84720788	member2	Thành viên 2	123456	tv2@gmail.com
c76985b7-0fa1-4382-b303-5f02812044f7	member3	Thành viên 3	123456	tv3@gmail.com
efcabd1f-4332-4717-b813-5dc930741333	member4	Thành viên 4	123456	tv4@gmail.com
1d9d4a55-c05b-47e8-813e-9cc45a3bb3b7	member5	Thành viên 5	123456	tv5@gmail.com
4635b912-d4a5-411a-a5f6-fcee229981c4	chungld	Lại Đức Chung	123456	chungld@bachkhoa-aptech.edu.vn

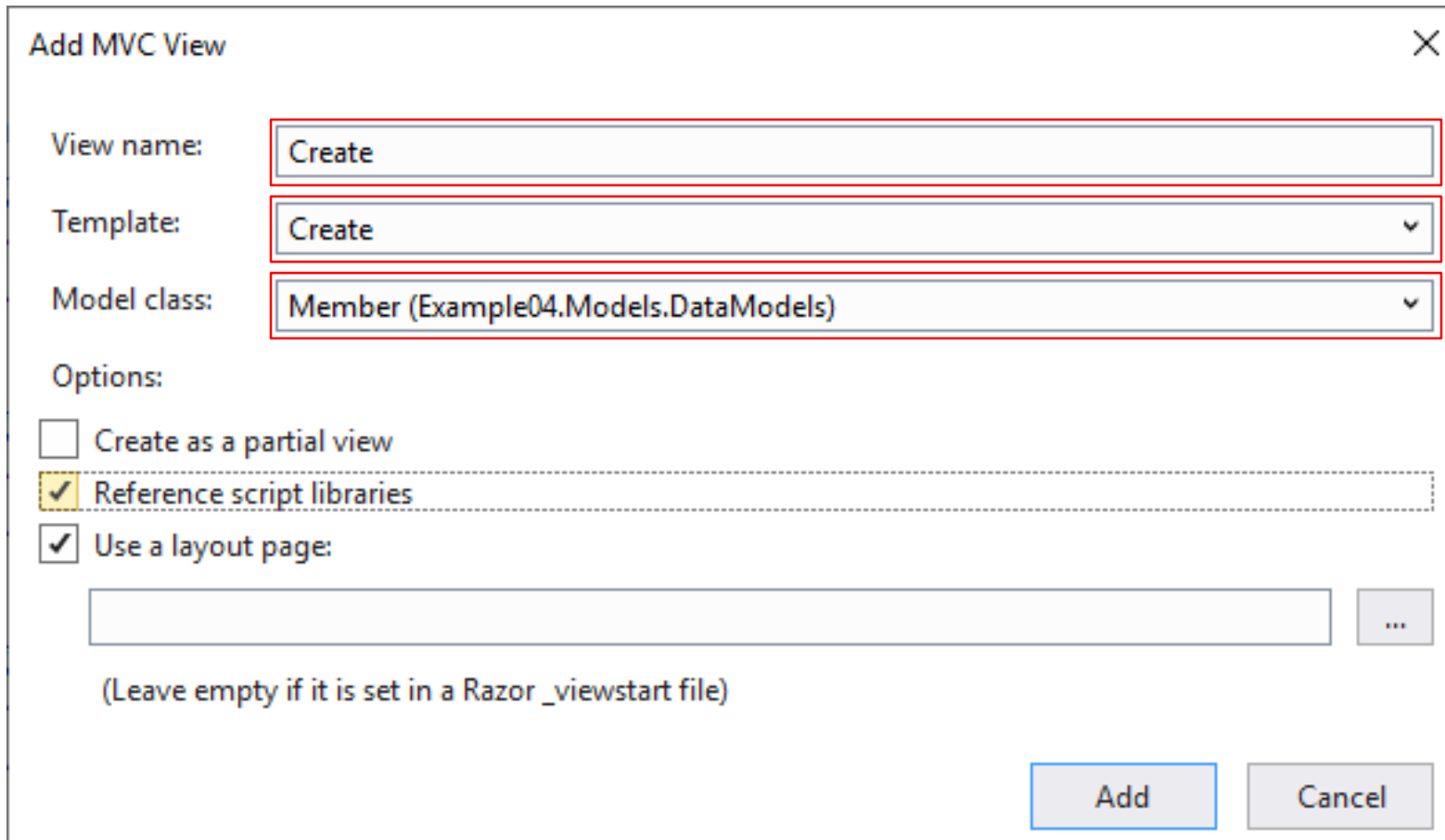
Sử dụng tính năng Scaffolding trong Visual Studio

- ASP.NET Core MVC Framework cung cấp 1 tính năng có tên Scaffolding cho phép bạn có thể sinh ra các View tự động từ một Model class. View sinh ra sẽ được lưu tự động vào vị trí phù hợp.



Sử dụng tính năng Scaffolding trong Visual Studio

Lưu ý khi tạo view bạn nên đặt tên View trùng với tên Template



The screenshot shows the 'Add MVC View' dialog box with the following fields and options:

- View name:** Create
- Template:** Create
- Model class:** Member (Example04.Models.DataModels)
- Options:**
 - ☐ Create as a partial view
 - ☒ Reference script libraries
 - ☒ Use a layout page:

Below the options, there is a text box and a button with three dots (...). Below that, it says: (Leave empty if it is set in a Razor _viewstart file)

At the bottom right, there are two buttons: **Add** and **Cancel**.

HỎI ĐÁP

