

Bài 5:

Tạo trang Login trong ASP.NET MVC với Entity Framework Code First

- Ghép giao diện trang login vào hệ thống
- Tạo mới Login Model
- Tạo tầng Models và cài đặt Entity Framework
- Tạo tầng Code first bằng Entity Framework tool for Visual
- Tạo view Login bằng HTML Razor
- Login và thông báo lỗi bằng Razor

Chuẩn bị: Project của bài 04:

Database:

```
CREATE DATABASE WebShopHocTap
go
USE [WebShopHocTap]
GO
/***** Object: Table [dbo].[Account]    Script Date: 03/06/2018 09:25:42 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[Account](
    [UserName] [varchar](20) NOT NULL,
    [PassWord] [varchar](20) NULL,
    CONSTRAINT [PK_Account] PRIMARY KEY CLUSTERED
(
    [UserName] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET ANSI_PADDING OFF
GO
INSERT [dbo].[Account] ([UserName], [PassWord]) VALUES (N'admin', N'admin')
/***** Object: StoredProcedure [dbo].[SP_Account_Login]    Script Date: 03/06/2018
09:25:44 *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROC [dbo].[SP_Account_Login]
    @UserName VARCHAR(20) ,
    @PassWord VARCHAR(20)
AS
```

```
BEGIN
    DECLARE @count INT;
    DECLARE @res BIT;
    SELECT @count = COUNT(*)
    FROM    dbo.Account
    WHERE   UserName = @UserName
           AND Password = @PassWord;
    IF @count > 0
        SET @res = 1;
    ELSE
        SET @res = 0;
    SELECT @res;
END;
```

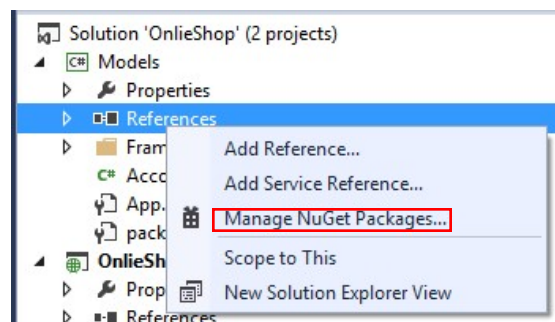
GO

Thực hiện:

Thực hiện tạo Project Models để sử dụng entity framework

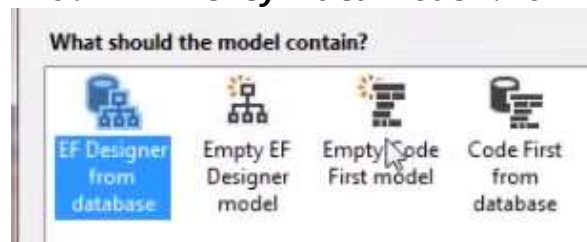
Bước 1: Tạo 1 project classLibrary tên là Models

Bước 2: Cài EntityFramework cho project: sử dụng Manager NuGetPackage để tìm kiếm EntityFramework






Bước 3: Cài đặt [EntityFramework tool for visual studio](#) (nếu máy chưa có).

Bước 4: Tạo thư mục Framework trong project, Sau đó tạo 1 Item ADO.NET Entity Data Model: OnlineShopDbContext



- Tạo 1 chuỗi kết nối bằng winzad để tạo chuỗi kết nối.
- Sau khi tạo xong, thì entity code first sẽ tự động tạo ra mỗi bảng sẽ là 1 Entity, và các class của database.

 Framework
 Account.cs
 OnlineShopDbContext.cs

Bước 5: Tạo lớp AccountModel để thực hiện lấy kết quả từ thủ tục kiểm tra đăng nhập

```

public class AccountModel : IDisposable
{
    private OnlineShopDbContext context = null;
    public void Dispose()
    {
        if (context != null)
        {
            context.Dispose();
            context = null;
        }
    }
    public AccountModel()
    {
        context = new OnlineShopDbContext();
    }
    public bool Login(string UserName, string Password)
    {
        object[] Sqlparams = new SqlParameter[] {
            new SqlParameter("@UserName", UserName),
            new SqlParameter("@Password", Password)
        };
        var res = context.Database.SqlQuery<bool>("SP_Account_Login
@UserName,@PassWord", Sqlparams).SingleOrDefault();
        return res;
    }
}
  
```

Tạo view Login cho Admin

```

@{
    Layout = null;
}
@model OnlieShop.Areas.Admin.Models.LoginModel
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <meta name="description" content="">
    <meta name="author" content="">
    <title>SB Admin 2 - Bootstrap Admin Theme</title>
  
```

```

<!-- Bootstrap Core CSS -->
<link
href="/Assets/Admin/vendor/bootstrap/css/bootstrap.min.css"
rel="stylesheet">
<!-- MetisMenu CSS -->
<link href="/Assets/Admin/vendor/metisMenu/metisMenu.min.css"
rel="stylesheet">
<!-- Custom CSS -->
<link href="/Assets/Admin/dist/css/sb-admin-2.css"
rel="stylesheet">
<!-- Custom Fonts -->
<link href="/Assets/Admin/vendor/font-awesome/css/font-
awesome.min.css" rel="stylesheet" type="text/css">
</head>
<body>
<div class="container">
<div class="row">
<div class="col-md-4 col-md-offset-4">
<div class="login-panel panel panel-default">
<div class="panel-heading">
<h3 class="panel-title">Login</h3>
</div>
<div class="panel-body">
@using(Html.BeginForm())
{
    @Html.AntiForgeryToken()
    @Html.ValidationSummary(true,null,new
{@class="alert alert-danger"})
    <fieldset>
        <div class="form-group">
            @Html.TextBoxFor(model =>
model.UserName, new { @class="form-control",
@placeholder="UserName", @name="UserName",
@autofocus="autofocus"});@* ,@type="email"*@
        </div>
        <div class="form-group">
            @Html.TextBoxFor(model =>
model.Password, new { @class = "form-control", @placeholder =
"Password", @name = "password", @type = "password", value = "" });
        </div>
        <div class="checkbox">
            <label>

```

```

                                @Html.CheckBoxFor(model =>
model.RememberMe, new { name = "remember", type = "checkbox",
value = "Remember Me" });
                                Remember Me
                                </label>
                                </div>
                                <!-- Change this to a button or
input when using this as a form -->
                                <button type="submit" class="btn
btn-lg btn-success btn-block"> Login</button>
                                </fieldset>
                                }
                                </div>
                                </div>
                                </div>
                                </div>
                                <!-- jQuery -->
                                <script
src="/Assets/Admin/vendor/jquery/jquery.min.js"></script>
                                <!-- Bootstrap Core JavaScript -->
                                <script
src="/Assets/Admin/vendor/bootstrap/js/bootstrap.min.js"></script>
                                <!-- Metis Menu Plugin JavaScript -->
                                <script
src="/Assets/Admin/vendor/metisMenu/metisMenu.min.js"></script>
                                <!-- Custom Theme JavaScript -->
                                <script src="/Assets/Admin/dist/js/sb-admin-2.js"></script>
                                </body>
                                </html>

```

Ghi chú:

- Chép code của file app.config trong project modeels vào file web.config để khi chạy web sẽ đọc file web.config này.
- Phải cài entityFramework cho project startup..

Bài 6:

Cách đăng nhập với Custom Membership Provider trong ASP.NET MVC

Trong bài học này chúng ta sẽ thay thế cách login thông thường với Session truyền thống bằng Custom Membership Provider có sẵn trong .NET.

- Lợi điểm của phương pháp này là sử dụng được cơ chế bảo mật có sẵn của .NET.

- Giảm thiểu mã nguồn mỗi khi kiểm tra login

- Đơn giản để triển khai Bao gồm có 3 bước:

1. Cấu hình trong webconfig.
2. Tạo lớp CustomMembershipProvider dẫn xuất từ lớp MembershipProvider có sẵn
3. Thực hiện validate và set cookie bằng FormAuthentication
4. Logout cũng bằng FormAuthentication.Logout()
5. Check đăng nhập hay chưa sử dụng thuộc tính Authorize

Chuẩn bị

Project của bài 5.

Thực hiện.

Cấu hình web.config

Trong tab system.web

Thêm cấu hình sau

```
<authentication mode="Forms">
  <forms loginUrl="~/Admin/Login" timeout="2000"/>
</authentication>
<membership defaultprovider="CustomMembershipProvider">
  <providers>
    <clear>
      <add applicationname="OnlineShop"
connectionstringname="OnlineShopDbContext"
enablepasswordreset="true" maxinvalidpasswordattempts="5"
minrequirednonalphanumericcharacters="0"
minrequiredpasswordlength="1" name="CustomMembershipProvider"
passwordattemptwindow="10" passwordformat="Hashed"

```

```
requiresquestionandanswer="false" requiresuniqueemail="false"
type="OnlineShop.Areas.Admin.Codes.CustomMembershipProvider">
    </add>
</clear>
</providers>
</membership>
```

Bước 2: tạo 1 class có tên là

```
OnlineShop.Areas.Admin.Codes.CustomMembershipProvider.cs
public class CustomMembershipProvider:MembershipProvider
{
    public override bool ValidateUser(string username, string
password)
    {
        return new AccountModel().Login(username, password);
    }
}
```

Thực thi nạp chồng lên phương chức ValidateUser

Bước 3: Thêm đoạn code vào trong Action Login vị trí đăng nhập thành công. `FormsAuthentication.SetAuthCookie(model.UserName, model.RememberMe);`

```
return RedirectToAction("Index", "Home");
```

Thêm thuộc tính `[Authorize]` vào Controller Home của Admin để chỉ định bắt buộc phải có quyền mới cho vào.

Bước 4: Thực hiện chức năng logout ra khỏi Admin

- Thêm phương thức action

```
public ActionResult Logout()
{
    FormsAuthentication.SignOut();
    return RedirectToAction("Index", "Home");
}
```

- Chỉnh link của mục logout trong `_Layout.html`

```
@Html.ActionLink("Đăng xuất", "Logout", "Login");
```