# 迁移、种子数据与模型改变

## 迁移

在设计迁移之前，若数据库已经存在，可在连接字符串中变更一次数据库名称

### 启动迁移

|  |
| --- |
| enable-migrations |
| enable-migrations命令在项目文件夹中创建迁移文件夹，在该文件夹中放入Configuration.cs可编辑以配置 Migrations 的文件。 |

### 新增变迁

**Add-Migration** 将基于自上次迁移创建以来对模型所做的更改来构建下一次迁移

如果启动迁移前数据库已经存在，启动迁移后系统将自动查找现有数据库并自动自行一次Add-Migration命令。

|  |
| --- |
| add-migration InitialCreate |
| add-migration 迁移名称（概述迁移原由等） |

### 更新数据库

**Update-Database** 将对数据库应用任意挂起的迁移

## 种子文件

可以将种子函数的内容移动到迁移配置类文件Configuration.cs中

多方实体实行新增获更新

联系实体依靠查询两个多方获得ID，使用更新

|  |
| --- |
|  |

# 表单授权

## 创建一个用户模型

|  |
| --- |
| namespace ContosoUniversity.Models  {  public class User  {  public int UserID { get; set; }  public string UserName { get; set; }  public string Password { get; set; }  }  } |

在种子函数中创建一个种子实例

在上下文里添加一个数据集

变更数据模型应迁移数据库

运行一次

add-migration 名称

update-database

## 创建一个Authentication控制器，并创建一个Login行为及视图

|  |
| --- |
| public class AuthenticationController : Controller  {  // GET: Authentication  public ActionResult Login()  {  return View();  }  } |

|  |
| --- |
| @model ContosoUniversity.Models.User  @{  ViewBag.Title = "Login";  }  <h2>Login</h2>  <div>  @Html.ValidationMessage("CredentialError", new { style = "color:red;" })  @using (Html.BeginForm("Login", "Authentication", FormMethod.Post))  {  @Html.LabelFor(c => c.UserName)  @Html.TextBoxFor(x => x.UserName)  <br />  @Html.LabelFor(c => c.Password)  @Html.PasswordFor(x => x.Password)  <br />  <input type="submit" name="BtnSubmit" value="Login" />  }  </div> |

## 在student控制器Create行为处置入授权特性

|  |
| --- |
| [Authorize]  // GET: Student/Create  public ActionResult Create()  {  return View();  } |

## 配置文件设置表单授权处理路由

|  |
| --- |
| <system.web>  <compilation debug="true" targetFramework="4.5.2" />  <httpRuntime targetFramework="4.5.2" />  <authentication mode="Forms">  <forms loginUrl="~/Authentication/Login"></forms>  </authentication>  </system.web> |

## 判断用户输入是否合法函数

此函数可以访问数据库获得用户名和密码，此处用文本替代

该判断可以置入业务层，此处作为Authentication控制器的一个非行为方法

|  |
| --- |
| [NonAction]  public bool IsValidUser(User u)  {  if (u.UserName == "Admin" && u.Password == "Admin")  {  return true;  }  else  {  return false;  }  } |

## 接收用户输入决定下一步操作

|  |
| --- |
| public ActionResult Login(User u)  {  if (IsValidUser(u))  {  FormsAuthentication.SetAuthCookie(u.UserName, false);  return RedirectToAction("Index", "Home");  }  else  {  ModelState.AddModelError("CredentialError", "用户名密码无效");  return View("Login");  }  } |

## 测试验证登录