### 1.创建新的项目MvcMusicStore

2.修改HomeController控制器

# 3.添加 StoreController控制器

3.1实现在我们的音乐应用商店音乐流派列表页

3.2实现列出所有特定流派音乐 album 的浏览页面

3.3实现显示有关特定音乐唱片集信息的详细信息页

3.4在 StoreController 的实现，并且我们将为 browse （） 和 Details() 添加类方法

3.5修改\_Layout.cshtml视图

## 4.更新样式表

### 4.1在网页下载 CSS和图片，添加到Content

## 使用模型将信息传递到视图

## **5.1在models创建Genre.类和Album类**

## **5.2再修改逻辑，以返回到视图的唱片集对象**

## **5.3添加Details视图**

## **5.4添加/Browse视图以显示类型信息。**

## **5.5添加index视图**

### **5.1.1将循环访问下面的代码已完成的视图中所示的模型中的类型对象。**

## 页之间添加链接

6.1链接文本，将显示类型名称

6.2控制器操作名称 （浏览）

6.3路由参数值，指定的名称 （类型） 和值 （类型名称）

#### 添加艺术家模型类

#### **7.1Artist类**

public class Artist { public int ArtistId { get; set; } public string Name { get; set; } }

#### 更新模型类

### **8.1更新唱片集类**

### **public class Album { public int AlbumId { get; set; } public int GenreId { get; set; } public int ArtistId { get; set; } public string Title { get; set; } public decimal Price { get; set; } public string AlbumArtUrl { get; set; } public Genre Genre { get; set; } public Artist Artist { get; set; } }**

### **8.2更新流派类**

### **public partial class Genre { public int GenreId { get; set; } public string Name { get; set; } public string Description { get; set; } public List<Album> Albums { get; set; } }**

### 在 web.config 文件中创建的连接字符串

### 9.1 <connectionStrings>

### <add name="MusicStoreEntities" connectionString="Data Source=(LocalDb)\MSSQLLocalDB;Initial Catalog=MusicStoreEntities1;Integrated Security=SSPI;" providerName="System.Data.SqlClient" />

### </connectionStrings>

### 添加上下文类

### **10.1添加上下文MusicStoreEntities类**

### **10.1.1MusicStoreEntities类将表示实体框架数据库上下文，并将处理我们创建、 读取、 更新和删除用于我们的操作。**

### **添加SampleData类**

### **11.1在SampleData类设置种子数据，将"种子"数据添加到新创建的数据库实体框架中**

### **11.2 这将预构建存储目录，流派、 艺术家和唱片集的种子数据**

### **11.3把种子数据写入数据库**

### **在Global类添加Database.SetInitializer(new SampleData());**

## 创建 StoreManagerController控制器

## **StoreManager 控制器将（创建、 读取、 更新、 删除）功能**

### **12.2将信息传递给视图使用 ViewBag方法**

### **12.3为流派类增加下拉列表**

### **12.4[HttpPost] 属性，它将告诉 ASP.NET MVC 它只应响应 HTTP POST 请求。**

将实现以下功能:

12.4.1读取窗体值

12.4.2检查窗体值是否传递的任何验证规则

12.4.3如果表单提交有效，将数据保存并显示更新的列表

12.4.4如果提交窗体不是有效的重新显示具有验证错误的窗体

### **13.处理编辑视图**

### **14.处理删除视图**

### **14.1 1HTTP POST 删除控制器操作将实现以下功能：**

14.1.1加载的唱片集的 ID

14.1.2将其删除唱片集并保存更改

14.1.3重定向到显示唱片集已从列表中删除的索引

## **向我们唱片类中添加验证**

|  |
| --- |
| namespace MvcMusicStore.Models{[Bind(Exclude = "AlbumId")]public class Album{[ScaffoldColumn(false)]public int AlbumId { get; set; }[DisplayName("Genre")]public int GenreId { get; set; }[DisplayName("Artist")]public int ArtistId { get; set; }[Required(ErrorMessage = "An Album Title is required")][StringLength(160)]public string Title { get; set; }[Range(0.01, 100.00,ErrorMessage = "Price must be between 0.01 and 100.00")]public decimal Price { get; set; }[DisplayName("Album Art URL")][StringLength(1024)]public string AlbumArtUrl { get; set; }public virtual Genre Genre { get; set; }public virtual Artist Artist { get; set; }}} |

## **添加 AccountController 和视图**

## 17.添加购物车、 顺序和 OrderDetail 模型类

## **17.1管理购物车业务逻辑**

## 增加ViewModels 文件夹

## **17.1在 Viewmodel 文件夹中添加 ShoppingCartViewModel 类**

## **17.1.1它具有两个属性： 车商品和要在购物车中保存的所有项的总价格的十进制值的列表。**

## **17.1.2添加ShoppingCartRemoveViewModel类**

## 添加购物车控制器

### 18.1购物车控制器的主要功能： 将项添加到购物车、 从购物车中，删除项和查看购物车中的项

## 创建 CheckoutController控制器

## 添加 AddressAndPayment 视图

## 21定义验证规则的顺序

|  |
| --- |
| namespace MvcMusicStore.Models { [Bind(Exclude = "OrderId")] public partial class Order { [ScaffoldColumn(false)] public int OrderId { get; set; } [ScaffoldColumn(false)] public System.DateTime OrderDate { get; set; } [ScaffoldColumn(false)] public string Username { get; set; } [Required(ErrorMessage = "First Name is required")] [DisplayName("First Name")] [StringLength(160)] public string FirstName { get; set; } [Required(ErrorMessage = "Last Name is required")] [DisplayName("Last Name")] [StringLength(160)] public string LastName { get; set; } [Required(ErrorMessage = "Address is required")] [StringLength(70)] public string Address { get; set; } [Required(ErrorMessage = "City is required")] [StringLength(40)] public string City { get; set; } [Required(ErrorMessage = "State is required")] [StringLength(40)] public string State { get; set; } [Required(ErrorMessage = "Postal Code is required")] [DisplayName("Postal Code")] [StringLength(10)] public string PostalCode { get; set; } [Required(ErrorMessage = "Country is required")] [StringLength(40)] public string Country { get; set; } [Required(ErrorMessage = "Phone is required")] [StringLength(24)] public string Phone { get; set; } [Required(ErrorMessage = "Email Address is required")] [DisplayName("Email Address")] [RegularExpression(@"[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}", ErrorMessage = "Email is is not valid.")] [DataType(DataType.EmailAddress)] public string Email { get; set; } [ScaffoldColumn(false)] public decimal Total { get; set; } public List<OrderDetail> OrderDetails { get; set; } } } |

## 添加签出完整的视图

## 更新视图时出错

## 创建购物车摘要分部视图

## 创建分部视图的流派菜单

## 更新站点布局，以显示我们的分部视图

## 更新到应用商店浏览页

## 27.更新主页显示顶部销售相册