Bài 05

Data Validation và Annotation

Mục tiêu

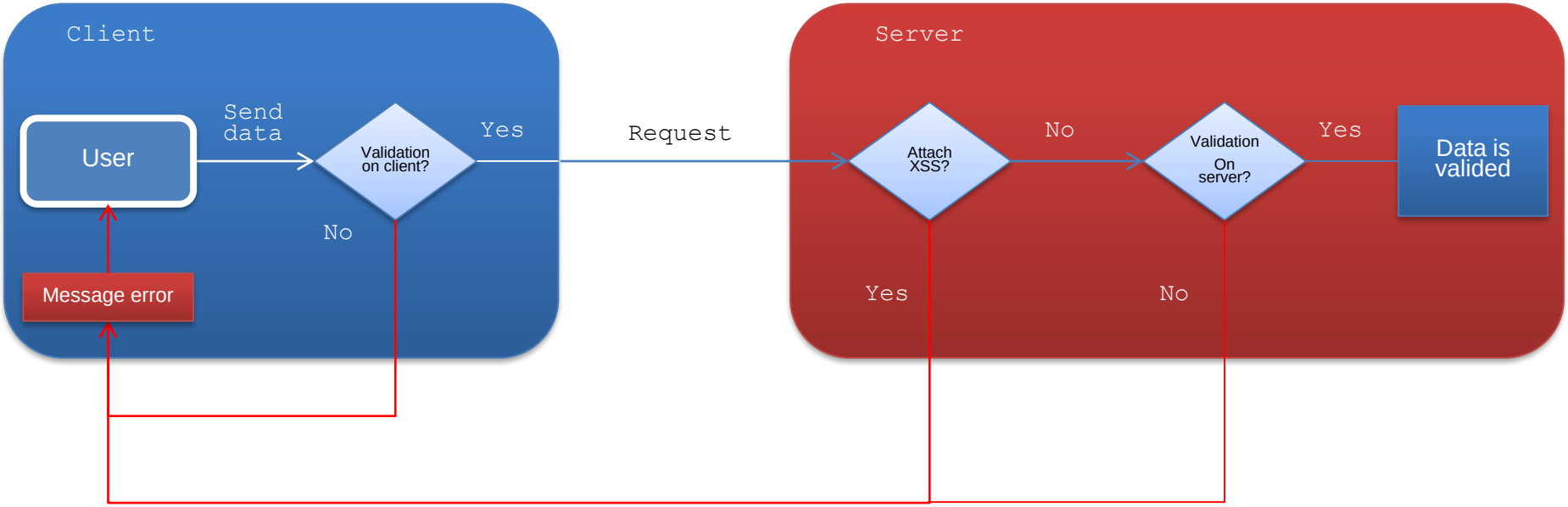
- Giới thiệu về Data Validation
- Quy trình Validation
- Manual Validation
- Sử dụng Annotation
- Thuộc tính ModelState

Data Validation

- Trong ứng dụng ASP.NET MVC người dùng tương tác với ứng dụng theo những cách sau:
  - Nhập URL trên trình duyệt
  - Kích vào link trên ứng dụng
  - Submit thông tin trên form
- Trong tất cả các trường hợp trên thì:
  - Bạn phải đảm bảo dữ liệu truyền đi luôn hợp lệ.
  - Bạn có thể kiểm tra dữ liệu người dùng tại tầng client hoặc server hoặc cả 2 tùy theo ngữ cảnh miễn là dữ liệu trước khi lưu trữ.
  - Tất cả các dữ liệu không hợp lệ phải thông báo cho người dùng ngay để thay đổi lại.

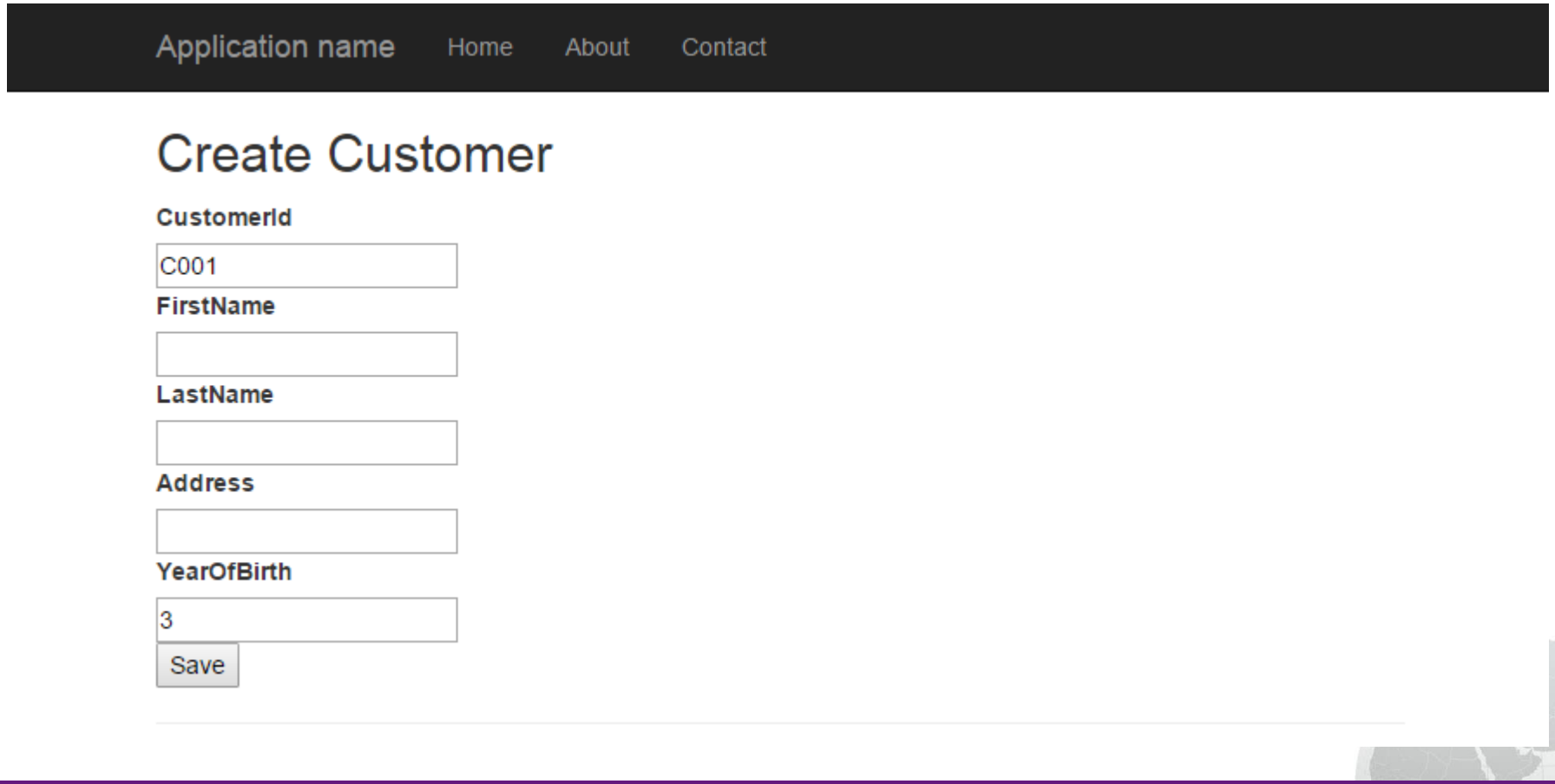
Quy trình Data Validation

- Công việc Validation bắt đầu khi dữ liệu người dùng gửi lên server.



Manual Validation

- Trong ứng dụng ASP.NET MVC bạn có thể validation dữ liệu bằng tay trong action của controller khi nhận dữ liệu từ người dùng.



Sử dụng Data Annotation

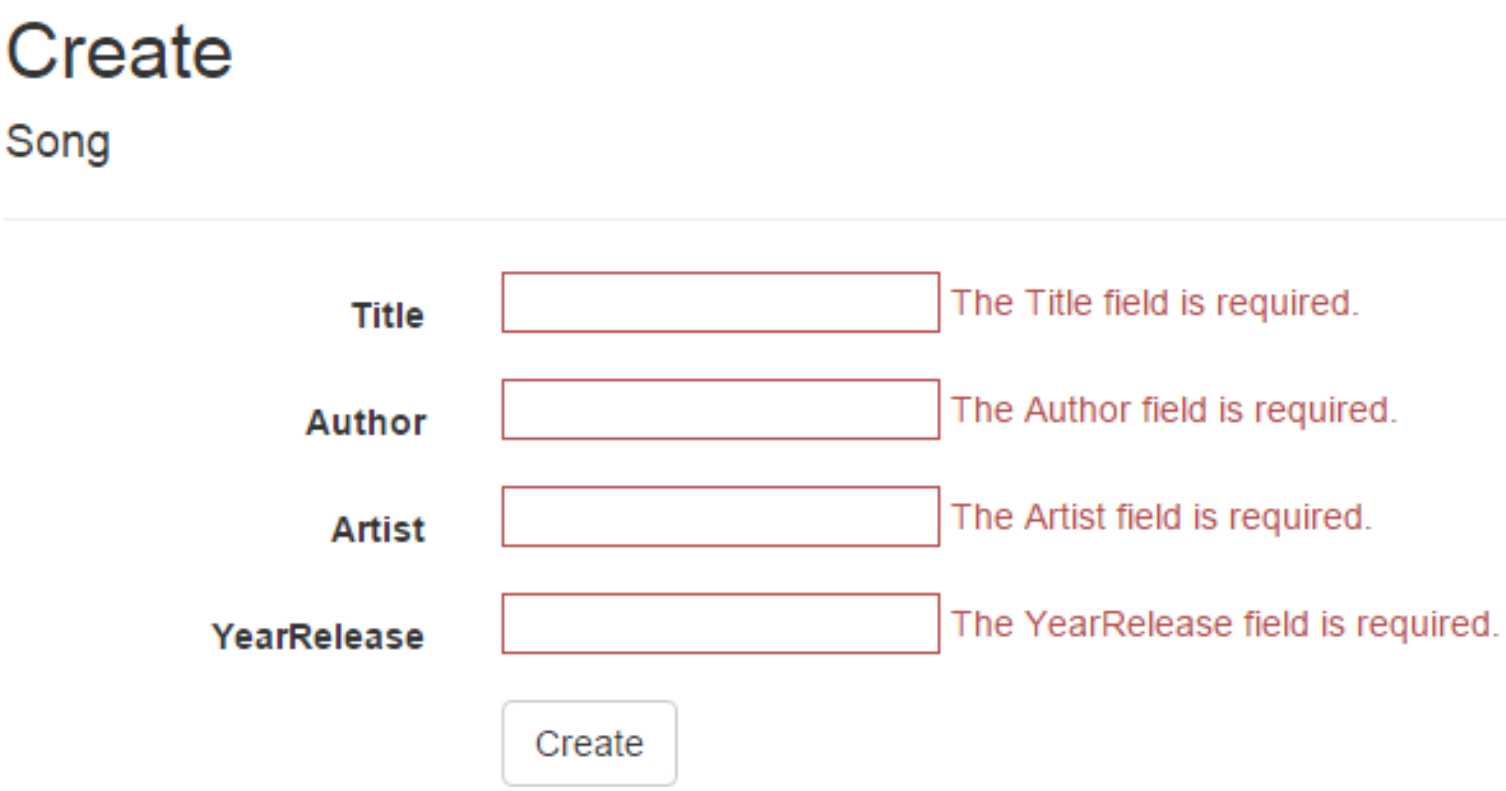
- MVC Framework cung cấp một vài data annotation cho phép bạn áp dụng cho các thuộc tính của model. Data annotations thực thi các nhiệm vụ thường được yêu cầu trong các ứng dụng.
  Một vài Annotation quan trọng
  - StringLength
  - RegularExpression
  - Range
  - Compare
  - DisplayName
  - ReadOnly
  - DataType
  - ScaffoldColumn

Required Annotation

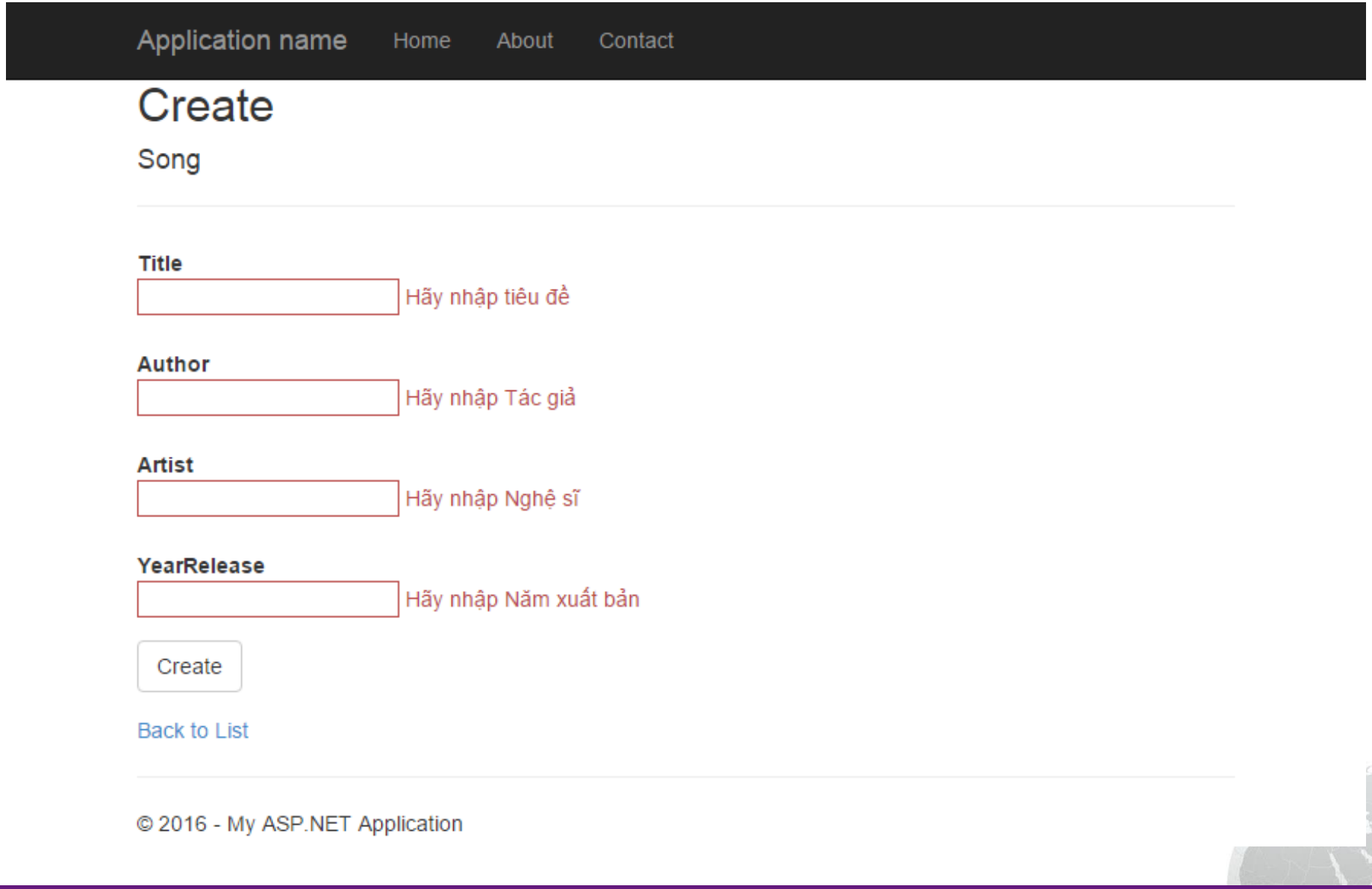
- Required data annotation chỉ ra thuộc tính mà nó liên kết là thuộc tính bắt buộc phải nhập, nếu trống hoặc null thì một lỗi sẽ trả về.
- Cú pháp

```
9 references
public class Song
{
    10 references
    public int Id { get; set; }
    [Required]
    7 references
    public string Title { get; set; }
    [Required]
    7 references
    public string Author { get; set; }
    [Required]
    7 references
    public string Artist { get; set; }
    [Required]
    7 references
    public int YearRelease { get; set; }
}
```

Required Annotation



Chỉ rõ thông báo lỗi

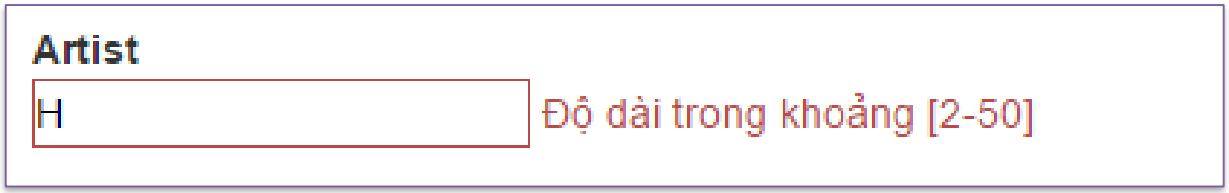


StringLength Annotation

- StringLength Annotation sử dụng để quy định độ dài tối thiểu và tối đa cho một trường
- Cú pháp

  [StringLength(<max_length>, MinimumLength= <min_length>)]
- Ví dụ

```
[Required(ErrorMessage="Hãy nhập Nghệ sĩ")]
[StringLength(50,MinimumLength=2,ErrorMessage="Độ dài trong khoảng [2-50]")]
7 references
public string Artist { get; set; }
```



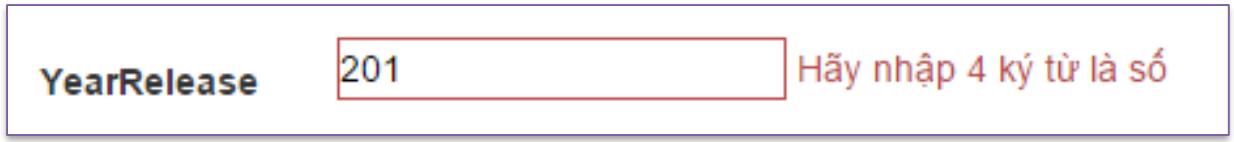
RegularExpression Annotation

- Annotation này cho phép bạn kiểm tra một chuỗi với một mẫu tìm kiếm xem có chưa được một hoặc nhiều ký tự trong chuỗi chỉ ra không?
- Cú pháp

  [RegularExpression(<pattern>)]

- Ví dụ

```
[Required(ErrorMessage="Hãy nhập Năm xuất bản")]
[RegularExpression(@"[0-9]{4}",ErrorMessage="Hãy nhập 4 ký từ là số")]
7 references
public int YearRelease { get; set; }
```



Range Annotation

- Annotation cho phép chỉ ra miền giá trị của trường kiểu số
- Cú pháp

  [Range (<minimum_range>, <maximum_range>
- Ví dụ

```
[Required(ErrorMessage="Hãy nhập Năm xuất bản")]
[RegularExpression(@"[0-9]{4}",ErrorMessage="Hãy nhập 4 ký từ là số")]
[Range(1900,2016,ErrorMessage="Hãy nhập giá trị trong khoảng [1900-2016]")]
7 references
public int YearRelease { get; set; }
```

Compare Annotation

- Annotation này dùng để so sánh giá trị của 2 trường với nhau

```
0 references
public string Password { get; set; }
[Compare("Password",ErrorMessage="Password nhập lại không khớp")]
0 references
public string ConfirmPassword { get; set; }
```

DisplayName Annotation

- Khi bạn sử dụng phương thức @Html.LabelFor() trong View thì nó sẽ hiển thị một Label với tên thuộc tính trong Model, tuy nhiên bạn có thể chỉ ra tên Label tường minh bằng Annotation DisplayName cho thuộc tính muốn hiển thị trong Model.
- Ví dụ

ReadOnly Annotation

- Annotation này cho phép một trường sẽ hiển thị chỉ đọc trên form

```
[ReadOnly(true)]
0 references
public int DiscountAmount { get; set; }
```

DataType Annotation

- Annotation này cung cấp thông tin về kiểu dữ liệu của trường tại thời điểm chạy để View engine sinh ra control tương ứng trên form.

```
[DataType(DataType.Password)]
0 references
public string Password { get; set; }
[DataType(DataType.EmailAddress)]
0 references
public string Email { get; set; }
[DataType(DataType.MultilineText)]
0 references
public string Details { get; set; }
```

ModelState Validation

- ModelState: là một lớp trong namespace System.Web.Mvc mà nó chứa thông tin lỗi và trạng thái của model binding trong.
- Bạn có thể sử dụng thuộc tính ModelState.IsValid để kiểm tra xem Model có valid không?

```
//POST: thực thi khi post dữ liệu lên server từ form tạo
[HttpPost]
0 references
public ActionResult Create(Song song)
{
    //Nếu dữ liệu
    if (!ModelState.IsValid)
        return View();
    //thêm mới vào bộ sưu tập
    catalogs.Add(song);
    //chuyển tới Index action
    return RedirectToAction("Index");
}
```

Hỏi Đáp