Microsoft®
.NET

UNIT 13:
MODELS AND VALIDATION

CONTENTS

- Purpose of Model
- Creating a model
- Using validation annotations and validation

Purpose of Models

- Model represents domain specific data and business logic in MVC architecture. It maintains the data of the application.
- Model objects represent the domain the application focuses on, and the models are the objects you want to display, save, create, update, and delete.
- Model objects retrieve and store model state in the persistance store like a database.

Creating a model

- Model class holds data in public properties. All the Model classes reside in the Model folder in MVC folder structure
- Right click on Model folder -> Add -> click on Class..

Data Annotations

- ASP.NET MVC uses DataAnnotations attributes to implement validations.
- DataAnnotations includes built-in validation attributes for different validation rules, which can be applied to the properties of model class. ASP.NET MVC framework will automatically enforce these validation rules and display validation messages in the view.

Data Annotations

- **The DataAnnotations attributes included in *System.ComponentModel.DataAnnotations* namespace.**

Attribute	Description
Required	Indicates that the property is a required field
StringLength	Defines a maximum length for string field
Range	Defines a maximum and minimum value for a numeric field
RegularExpression	Specifies that the field value must match with specified Regular Expression
CreditCard	Specifies that the specified field is a credit card number
CustomValidation	Specified custom validation method to validate the field
EmailAddress	Validates with email address format
FileExtension	Validates with file extension
MaxLength	Specifies maximum length for a string field
MinLength	Specifies minimum length for a string field
Phone	Specifies that the field is a phone number using regular expression for phone numbers

Data Annotations

- **Required: this attribute raises a validation error if either property value is null or empty**

```
[Required] public string StudentName { get; set; }

[Required(ErrorMessage="Please enter student name.")]
```

- **RegularExpression:**

```
[RegularExpression(@"[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}")]

public string Email { get; set;}
```

- **Compare**

```
[Compare("Email")]

public string EmailConfirm { get; set; }
```

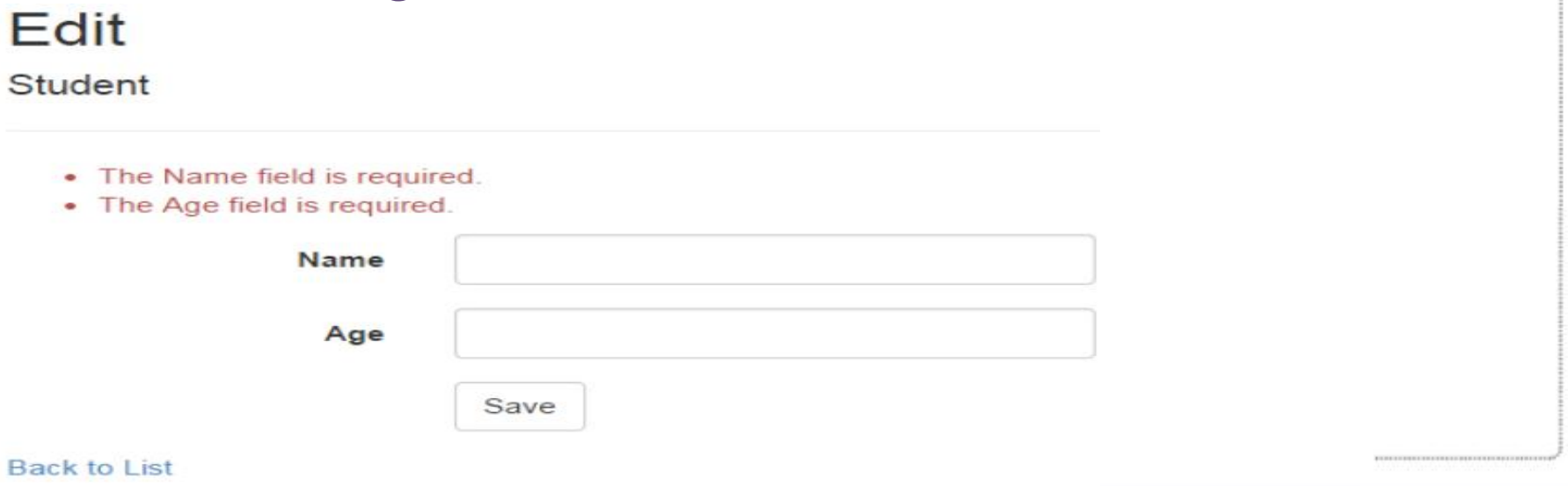
ValidationMessage

- The Html.ValidationMessage() is an extension method, that is a loosely typed method. It displays a validation message if an error exists for the specified field in the ModelStateDictionary object.
- You can specify a message as a second parameter in the ValidationMessage() method:

```
@model Student
@Html.Editor("StudentName") <br />
@Html.ValidationMessage("StudentName", "Please enter student name.", new { @class = "text-danger" })
```

ValidationSummary

- The ValidationSummary helper method generates an unordered list (ul element) of validation messages that are in the ModelStateDictionary object.
- The ValidationSummary can be used to display all the error messages for all the fields.



ValidationSummary

- To display a custom error message, you need to add custom errors into the ModelState in the appropriate action method. Ex:

```
ModelState.AddModelError(string.Empty, "Student Name already exists.");
```