VALIDATING DATA:

Validation: Test the Form input Fields.

Data Annotation:

Validate the Data easily and efficiently in ASP.NET MVC.

Note:

Only use in model class.

Header file to implement Data Annotation:

Using System.CompentModel.DataAnnotations(use in Model)

Types of data annotations are:

Required – This attribute raises a validation error if the property value is null or empty.

StringLength - min & max string lengths validation.

Regualar expression – accept user input in a specific text pattern (@ symbol is must).

Range – min & max numerical value validation.

Compare - Two fields Compare (ex.password, retype password)

Display – Display name for a model property

ReadOnly – Read only (don't write)

DataType – To add Password char, Currency, Date, Time, MultilineText.

ScaffoldColumn – you do not want to render a particular field on a view.

Implementing Validation: (use in Views)

@Html.ValidationMessageFor() → this helper method is used to display default error message.

@Html.ValidationSummary() → show all error at one place.

Introduction to Model State (Controller)

If the input is correct format ModelState.isValid property returns true. Otherwise False.

Ex:

If(ModelState.isValid) // only use [HttpPost]

{} else {}

Adding Errors to the ModelState using server side scripting:

You can add errormessage in **ModelState** using AddModelError() method.

Syntax:

[HttpPost]

Public ActoinResult Create(ModelClassname obj)

{

      If(usernameExists(obj.userName)) {

            ModelState.AddModelError("username","your message not available");

      }

      If (ModelState.isValid)

            return View(obj);

      else

            retutn View();

}

Here : usernameExists( ) – user defined function.

Example:

```
[HttpPost]
    public ActionResult Create(Emp obj)
    {
        if (function1(obj.ename))
            ModelState.AddModelError("ename", "already exist");

        if (ModelState.IsValid==true)
        {
            try
            {
                var s = MvcApplication.alldata;
                s.Add(obj);

                return RedirectToAction("Index");
            }
            catch
            {
                return View();
            }
        }
        else
            return View();
    }
```

This ename is model class variable name

```csharp
public bool function1(String name)
{
    var k = MvcApplication.alldata;
    bool ch = false;
    foreach(Emp e in k)
    {
        if (e.ename == name)
        {
            ch = true;
            break;
        }
    }
    if (ch == true)
        return true;
    else
        return false;
}
```

Create New

eno	ename	eadd	esal	password	This is Compare password			
0	gowthaman	karur 80	sdkj	kljds		Edit	Details	Delete
0	gowthamank	9	kjh	kj		Edit	Details	Delete
0	naveen	karur 88	88	88		Edit	Details	Delete
0	gowtham	89	98	89		Edit	Details	Delete

Emp

ename
gowthaman   already exist
eadd

esal
23
password
fgdsf
This is Compare password
dsfda

Create

Back to List

https://www.codeproject.com/Articles/1037026/Validations-in-MVC

https://www.c-sharpcorner.com/UploadFile/13048b/model-validation-in-Asp-Net-mvc909/

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;


namespace MyValidation.Models
{
    public class Emp
    {
         public int eno { set; get; }        // default integer is required
        // [Required]
        // [StringLength(10,MinimumLength=3,ErrorMessage="Only accept 3 to 10 chars")]
        // [StringLength(2)] // maximum length is 2
        // [RegularExpression(@"[a-zA-Z]{5,10}",ErrorMessage="Enter only a-z or A-Z min 5 max 10 char's")]
         public String ename { get; set; }   // default string is not required
         // [Required (ErrorMessage="Please fill Employee salary")]
         //public int esal { get; set; }
         // [Range (0,1000)]
         // [Range(typeof(int),"10000","15000")]
         //[Range(typeof(int),"10000","15000")]
         //[Range(typeof(decimal), "0.00", "49.99")]
[Range(typeof(int),"10000","20000",ErrorMessage="salary only 10000 to 20000")]
         public decimal esal { get; set; }

        public String password { get; set; }

         [Compare("password")]
         [DataType(DataType.Password)]
        //[DataType(DataType.MultilineText)]  // address type purpose
         public String comparePassword { get; set; }


        [Display(Name="This is Compare password")]   // to change label name at list views
         public String comparePassword { get; set; }


        [Required]
         [ScaffoldColumn(false)]
         public int eno { get; set; } // do not show eno field in views
                                      // before add view in controller page , first
scaffoldColumn(false) annotations is must.

    }
}
```

Controller File:

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
```

```csharp
using MyValidation.Models;

namespace MyValidation.Controllers
{
    public class HomeController : Controller
    {
        //
        // GET: /Home/

        public ActionResult Index()
        {
            var k = MvcApplication.alldata;
            return View(k);
        }

        //
        // POST: /Home/Create

        [HttpPost]
        public ActionResult Create(Emp obj)
        {
            if (ModelState.IsValid)
            {
                try
                {   // TODO: Add insert logic here
                    var k = MvcApplication.alldata;
                    k.Add(obj);
                    return RedirectToAction("Index");
                }
                catch
                {
                    return View();
                }
            }
            else
                return View();
        }

    }
```