IT2030 Assignment 11 Matt Brown

**Temp.cs**

using System;

using System.ComponentModel.DataAnnotations;

using Microsoft.AspNetCore.Mvc;

namespace TempManager.Models

{

public class Temp

{

public int Id { get; set; }

[Required(ErrorMessage = "Please enter a date.")]

[Remote("CheckDate", "Validation")]

public DateTime? Date { get; set; }

[Required(ErrorMessage = "Please enter a low temperature.")]

[Range(-200, 200, ErrorMessage = "Low temperature must be between -200 and 200.")]

public double? Low { get; set; }

[Required(ErrorMessage = "Please enter a high temparature.")]

[Range(-200, 200, ErrorMessage = "High temperature must be between -200 and 200.")]

public double? High { get; set; }

}

}

**HomeController.cs**

using System.Linq;

using Microsoft.AspNetCore.Mvc;

using TempManager.Models;

namespace Ch11Ex1TempManager.Controllers

{

public class HomeController : Controller

{

private TempManagerContext data { get; set; }

public HomeController(TempManagerContext ctx) => data = ctx;

public ViewResult Index()

{

var temps = data.Temps.OrderBy(t => t.Date).ToList();

return View(temps);

}

[HttpGet]

public ViewResult Add() => View(new Temp());

[HttpPost]

public IActionResult Add(Temp temp)

{

Temp check = data.Temps.FirstOrDefault(t => t.Date == temp.Date);

if (check != null)

{

ModelState.AddModelError("Date",

$"The date {temp.Date?.ToShortDateString()} is already in the database.");

}

if (ModelState.IsValid) {

data.Temps.Add(temp);

data.SaveChanges();

return RedirectToAction("Index");

}

else {

ModelState.AddModelError("", "Please correct all errors.");

return View(temp);

}

}

[HttpGet]

public ViewResult Delete(int id)

{

var temp = data.Temps.Find(id);

return View(temp);

}

[HttpPost]

public RedirectToActionResult Delete(Temp temp)

{

data.Remove(temp);

data.SaveChanges();

return RedirectToAction("Index");

}

}

}

**Add.cshtml**

@model Temp

@{

ViewData["Title"] = "Add";

}

@section scripts {

<script src="~/lib/jquery-validation/dist/jquery.validate.min.js"></script>

<script src="~/lib/jquery-validation-unobtrusive/jquery.validate.unobtrusive.min.js"></script>

}

<h2 class="p-2">Add temp</h2>

<div asp-validation-summary="ModelOnly" class="text-danger"></div>

<form asp-action="Add" method="post">

<div class="form-group">

<label asp-for="Date"></label>

<input asp-for="Date" type="text" class="form-control" />

<span asp-validation-for="Date" class="text-danger"></span>

</div>

<div class="form-group">

<label asp-for="Low"></label>

<input asp-for="Low" class="form-control" />

<span asp-validation-for="Low" class="text-danger"></span>

</div>

<div class="form-group">

<label asp-for="High"></label>

<input asp-for="High" class="form-control" />

<span asp-validation-for="High" class="text-danger"></span>

</div>

<input type="hidden" asp-for="Id" />

<button type="submit" class="btn btn-dark mr-2">Add</button>

<a asp-action="Index" class="btn btn-dark">Cancel</a>

</form>

**ValidationController.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using TempManager.Models;

using Microsoft.AspNetCore.Mvc;

namespace TempManager.Controllers

{

public class ValidationController : Controller

{

private TempManagerContext data { get; set; }

public ValidationController(TempManagerContext ctx) => data = ctx;

public JsonResult CheckDate(string date)

{

DateTime dt = DateTime.Parse(date);

Temp temp = data.Temps.FirstOrDefault(t => t.Date == dt);

if (temp == null)

return Json(true);

else

return Json($"The date {date} is already in the database.");

}

}

}