

**Facultad de Telemática**

**Programación para Web**

**Académico:**

**Damián Reyes Pedro**

**Alumno**

**Valdez Gutierrez Aldo Eduardo**

**Act 15. Registro de Estudiantes**

**Colima, Col; a 14 de marzo de 2024**

**Evidencias de funcionamiento**

|  |  |
| --- | --- |
|  | Evidencia de interfaz de usuario similar al ejemplo |
|  | Evidencia de las validaciones pedidas con sus mensajes de error |
|  | Evidencia de funcionamiento cuando se valida correctamente |
|  | Listado de todos los estudiantes en la lista |
|  | Funcionamiento de eliminado |

Codigo Blazor:

@page "/estudiantes"

<h1>Registro de Estudiantes</h1>

<div class="form-group">

<label for="nombre">Nombre:</label>

<input type="text" class="form-control @nombreBorderStyle" id="nombre" @bind="nombre">

@if (nombreErrorVisible)

{

<small style="color:red;">@nombreError</small>

}

</div>

<br>

<div class="form-group">

<label class="label label-default" for="correo">Correo:</label>

<input type="email" class="form-control @correoBorderStyle" id="correo" @bind="correo">

@if (correoErrorVisible)

{

<small style="color:red;">@correoError</small>

}

</div>

<br>

<div class="form-group">

<label for="calificacion">Calificacion:</label>

<input type="number" class="form-control @calificacionBorderStyle" id="calificacion" @bind="calificacion">

@if (calificacionErrorVisible)

{

<small style="color:red;">@calificacionError</small>

}

</div>

<br>

<button class="btn btn-primary" @onclick="AgregarEstudiante">Agregar Estudiante</button>

<table class="table">

<thead>

<tr>

<th>Nombre</th>

<th>Correo</th>

<th>Calificación</th>

<th>Acciones</th>

</tr>

</thead>

<tbody>

@foreach (var estudiante in estudiantes)

{

<tr>

<td>@estudiante.Nombre</td>

<td>@estudiante.Correo</td>

<td>@estudiante.Calificacion</td>

<td><i class="bi bi-trash3" @onclick="() => EliminarEstudiante(estudiante)"></i></td>

</tr>

}

</tbody>

</table>

@code

{

private string nombre = "";

private string correo = "";

private int calificacion;

private List<Estudiante> estudiantes = new List<Estudiante>();

private string nombreError = "El nombre no debe estar vacio";

private string correoError = "El correo electrónico no es válido";

private string calificacionError = "La calificación debe estar entre 0 y 10.";

private bool nombreErrorVisible = false;

private bool correoErrorVisible = false;

private bool calificacionErrorVisible = false;

private string nombreBorderStyle = "";

private string correoBorderStyle = "";

private string calificacionBorderStyle = "";

private async Task AgregarEstudiante()

{

nombreErrorVisible = false;

correoErrorVisible = false;

calificacionErrorVisible = false;

nombreBorderStyle = "";

correoBorderStyle = "";

calificacionBorderStyle = "";

bool nombreValido = validarNombre();

bool correoValido = validarCorreo();

bool calificacionValida = validarCalificacion();

if (!nombreValido || !correoValido || !calificacionValida)

{

return;

}

estudiantes.Add(new Estudiante(nombre, correo, calificacion));

await CambiarColorYLimpiarCampos();

}

private async Task CambiarColorYLimpiarCampos()

{

nombreBorderStyle = "is-valid";

correoBorderStyle = "is-valid";

calificacionBorderStyle = "is-valid";

StateHasChanged();

await Task.Delay(3000);

nombreBorderStyle = "";

correoBorderStyle = "";

calificacionBorderStyle = "";

nombre = "";

correo = "";

calificacion = 0;

StateHasChanged();

}

private bool validarNombre()

{

if (string.IsNullOrWhiteSpace(nombre))

{

nombreErrorVisible = true;

nombreBorderStyle = "is-invalid";

return false;

}

return true;

}

private bool validarCorreo()

{

if (!correoValido(correo))

{

correoErrorVisible = true;

correoBorderStyle = "is-invalid";

return false;

}

return true;

}

private bool validarCalificacion()

{

if (calificacion < 0 || calificacion > 10)

{

calificacionErrorVisible = true;

calificacionBorderStyle = "is-invalid";

return false;

}

return true;

}

private void EliminarEstudiante(Estudiante estudiante)

{

estudiantes.Remove(estudiante);

}

private bool correoValido(string email)

{

try

{

var addr = new System.Net.Mail.MailAddress(email);

return addr.Address == email;

}

catch

{

return false;

}

}

protected override void OnInitialized()

{

estudiantes.Add(new Estudiante("Juan", "juan@gmail.com", 10));

estudiantes.Add(new Estudiante("Cuarteto de nos", "cuarteto@nos.com", 9));

estudiantes.Add(new Estudiante("aldo", "avaldez0@ucol.mx", 10));

}

}

Estudiante.cs

using System;

public class Estudiante

{

public string Nombre { get; set; } = "";

public string Correo { get; set; } = "";

public int Calificacion { get; set; }

public Estudiante(string nombre, string correo, int calificacion)

{

Nombre = nombre;

Correo = correo;

Calificacion = calificacion;

}

public bool EsNombreValido()

{

return !string.IsNullOrWhiteSpace(Nombre);

}

public bool EsCorreoValido()

{

try

{

var addr = new System.Net.Mail.MailAddress(Correo);

return addr.Address == Correo;

}

catch

{

return false;

}

}

public bool EsCalificacionValida()

{

return Calificacion >= 0 && Calificacion <= 10;

}

public void ActualizarInformacion(string nombre, string correo, int calificacion)

{

if (EsNombreValido())

{

Nombre = nombre;

}

if (EsCorreoValido())

{

Correo = correo;

}

if (EsCalificacionValida())

{

Calificacion = calificacion;

}

}

}