**ASP .NET MVC N-E CRUD OPERATIONS**

1. Models 🡪 Product.cs

namespace CRUD.web.Models

{

public class Product

{

public int Id { get; set; }

public string? Name { get; set; }

public decimal Price { get; set; }

}

}

2. Controller 🡪 ProductController.cs

using CRUD.web.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Data.SqlClient;

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

namespace CRUD.web.Controllers

{

public class ProductController : Controller

{

string connectionString = "Data Source=Server name;Initial Catalog=Database name;Integrated Security=True;Trust Server Certificate=True";

// GET: ProductController

public ActionResult Index()

{

List<Product> products = new List<Product>();

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "SELECT Id, Name, Price FROM Product";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

con.Open();

using (SqlDataReader rdr = cmd.ExecuteReader())

{

while (rdr.Read())

{

Product product = new Product

{

Id = Convert.ToInt32(rdr["Id"]),

Name = rdr["Name"].ToString(),

Price = Convert.ToDecimal(rdr["Price"])

};

products.Add(product);

}

}

}

}

return View(products);

}

// GET: ProductController/Details/5

public ActionResult Details(int id)

{

Product product = new Product();

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "SELECT Id, Name, Price FROM Product WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

cmd.Parameters.AddWithValue("@Id", id);

con.Open();

using (SqlDataReader rdr = cmd.ExecuteReader())

{

if (rdr.Read())

{

product.Id = Convert.ToInt32(rdr["Id"]);

product.Name = rdr["Name"].ToString();

product.Price = Convert.ToDecimal(rdr["Price"]);

}

}

}

}

if (product == null)

{

return HttpNotFound();

}

return View(product);

}

// GET: ProductController/Create

public ActionResult Create()

{

return View();

}

// POST: ProductController/Create

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create(Product product)

{

try

{

if (ModelState.IsValid)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "INSERT INTO Product (Name, Price) VALUES (@Name, @Price)";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

cmd.Parameters.AddWithValue("@Name", product.Name);

cmd.Parameters.AddWithValue("@Price", product.Price);

con.Open();

cmd.ExecuteNonQuery();

}

}

return RedirectToAction("Index");

}

return View(product);

}

catch

{

return View("Index");

}

}

// GET: ProductController/Edit/5

public ActionResult Edit(int id)

{

Product product = new Product();

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "SELECT Id, Name, Price FROM Product WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

cmd.Parameters.AddWithValue("@Id", id);

con.Open();

using (SqlDataReader rdr = cmd.ExecuteReader())

{

if (rdr.Read())

{

product.Id = Convert.ToInt32(rdr["Id"]);

product.Name = rdr["Name"].ToString();

product.Price = Convert.ToDecimal(rdr["Price"]);

}

}

}

}

if (product == null)

{

return HttpNotFound();

}

return View(product);

}

private ActionResult HttpNotFound()

{

throw new NotImplementedException();

}

// POST: ProductController/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit(int id, Product product)

{

try

{

if (ModelState.IsValid)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "UPDATE Product SET Name = @Name, Price = @Price WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

cmd.Parameters.AddWithValue("@Name", product.Name);

cmd.Parameters.AddWithValue("@Price", product.Price);

cmd.Parameters.AddWithValue("@Id", product.Id);

con.Open();

cmd.ExecuteNonQuery();

}

}

return RedirectToAction("Index");

}

return View(product);

}

catch

{

return View();

}

}

// GET: ProductController/Delete/5

public ActionResult Delete(int id)

{

Product product = new Product();

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "SELECT Id, Name, Price FROM Product WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

cmd.Parameters.AddWithValue("@Id", id);

con.Open();

using (SqlDataReader rdr = cmd.ExecuteReader())

{

if (rdr.Read())

{

product.Id = Convert.ToInt32(rdr["Id"]);

product.Name = rdr["Name"].ToString();

product.Price = Convert.ToDecimal(rdr["Price"]);

}

}

}

}

if (product == null)

{

return HttpNotFound();

}

return View(product);

}

// POST: ProductController/Delete/5

[HttpPost, ActionName("DeleteConfirmed")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

try

{

using (SqlConnection con = new SqlConnection(connectionString))

{

string sqlQuery = "DELETE FROM Product WHERE Id = @Id";

using (SqlCommand cmd = new SqlCommand(sqlQuery, con))

{

cmd.Parameters.AddWithValue("@Id", id);

con.Open();

cmd.ExecuteNonQuery();

}

}

return RedirectToAction("Index");

}

catch

{

return View();

}

}

}

}

3. Views

1. Index.chtml

@{

ViewBag.Title = "Product List";

}

<h2>Product List</h2>

<p>

@Html.ActionLink("Create New", "Create", null, new { @class = "btn btn-primary" })

</p>

<table class="table">

<thead>

<tr>

<th>Id</th>

<th>Name</th>

<th>Price</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

@foreach (var product in Model)

{

<tr>

<td>@product.Id</td>

<td>@product.Name</td>

<td>@product.Price.ToString("c")</td>

<td>

@Html.ActionLink("Details", "Details", new { id = product.Id }, new { @class = "btn btn-info btn-sm" })

@Html.ActionLink("Edit", "Edit", new { id = product.Id }, new { @class = "btn btn-primary btn-sm" })

@Html.ActionLink("Delete", "Delete", new { id = product.Id }, new { @class = "btn btn-danger btn-sm" })

</td>

</tr>

}

</tbody>

</table>

2.Create.chtml

@model Product // Replace 'Product' with your actual model class

@{

ViewData["Title"] = "Create Product";

}

<h2>Create Product</h2>

<div class="row">

<div class="col-md-6">

<**form** **asp-action**="Create">

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

<div class="form-group">

<**label** **asp-for**="Name" class="control-label"></**label**>

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

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

</div>

<div class="form-group">

<**label** **asp-for**="Price" class="control-label"></**label**>

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

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

</div>

<div class="form-group">

<button type="submit" class="btn btn-primary">Create</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Back to List</**a**>

</div>

</**form**>

</div>

</div>

@section Scripts {

@{

await Html.RenderPartialAsync("\_ValidationScriptsPartial");

}

}

3.Details.chtml

@model CRUD.web.Models.Product

<div>

<h4>Product</h4>

<hr />

<dl class="row">

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.Id)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.Id)

</dd>

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.Name)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.Name)

</dd>

<dt class = "col-sm-2">

@Html.DisplayNameFor(model => model.Price)

</dt>

<dd class = "col-sm-10">

@Html.DisplayFor(model => model.Price)

</dd>

</dl>

</div>

<div>

@Html.ActionLink("Edit", "Edit", new { id = Model.Id }) |

<**a** **asp-action**="Index">Back to List</**a**>

</div>

4. Edit.chtml

@model Product // Replace 'Product' with your actual model class

@{

ViewData["Title"] = "Edit Product";

}

<h2>Edit Product</h2>

<div class="row">

<div class="col-md-6">

<**form** **asp-action**="Edit">

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

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

<div class="form-group">

<**label** **asp-for**="Name" class="control-label"></**label**>

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

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

</div>

<div class="form-group">

<**label** **asp-for**="Price" class="control-label"></**label**>

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

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

</div>

<div class="form-group">

<button type="submit" class="btn btn-primary">Save</button>

<**a** **asp-action**="Index" class="btn btn-secondary">Back to List</**a**>

</div>

</**form**>

</div>

</div>

@section Scripts {

@{

await Html.RenderPartialAsync("\_ValidationScriptsPartial");

}

}

5. Delete.chtml

@model Product // Replace 'Product' with your actual model class

@{

ViewData["Title"] = "Delete Product";

}

<h2>Delete Product</h2>

<div>

<h4>Are you sure you want to delete this product?</h4>

<div>

<h4>Product Details:</h4>

<hr />

<dl class="row">

<dt class="col-sm-2">

@Html.DisplayNameFor(model => model.Name)

</dt>

<dd class="col-sm-10">

@Html.DisplayFor(model => model.Name)

</dd>

<dt class="col-sm-2">

@Html.DisplayNameFor(model => model.Price)

</dt>

<dd class="col-sm-10">

@Html.DisplayFor(model => model.Price)

</dd>

</dl>

<**form** **asp-action**="DeleteConfirmed" method="post" class="form-horizontal">

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

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Delete" class="btn btn-danger" />

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

</div>

</div>

</**form**>

</div>

</div>

Program.cs

app.MapControllerRoute(

name: "default",

pattern: "{controller=Product}/{action=Index}/{id?}");