Bahria University,

Karachi Campus

A logo with text on it

Description automatically generated

LAB EXPERIMENT NO.

04

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| **01** | Develop an employee profile within the MVC .NET web application. Design a user interface that allows users to view, create, update, and delete employee records using ADO .Net Entity Framework. |
| **02** | Implement a product catalog in the MVC .NET web application utilizing ADO.NET Entity Framework. Design a user interface that enables users to browse, add, modify, and remove product records. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Submitted On:

14-03-2024

(Date: DD/MM/YYYY)

**Task No. 01**: Develop an employee profile within the MVC .NET web application. Design a user interface that allows users to view, create, update, and delete employee records using ADO .Net Entity Framework.**Solution:**

**EmployeesController**

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using CCLAB4Task1;

namespace CCLAB4Task1.Controllers

{

public class EmployeesController : Controller

{

private EmployeeDBEntities db = new EmployeeDBEntities();

// GET: Employees

public ActionResult Index()

{

return View(db.Employees.ToList());

}

// GET: Employees/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Employee employee = db.Employees.Find(id);

if (employee == null)

{

return HttpNotFound();

}

return View(employee);

}

// GET: Employees/Create

public ActionResult Create()

{

return View();

}

// POST: Employees/Create

// To protect from overposting attacks, enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "Id,Name,Age,Gender,Salary,Designation,City")] Employee employee)

{

if (ModelState.IsValid)

{

db.Employees.Add(employee);

db.SaveChanges();

return RedirectToAction("Index");

}

return View(employee);

}

// GET: Employees/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Employee employee = db.Employees.Find(id);

if (employee == null)

{

return HttpNotFound();

}

return View(employee);

}

// POST: Employees/Edit/5

// To protect from overposting attacks, enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "Id,Name,Age,Gender,Salary,Designation,City")] Employee employee)

{

if (ModelState.IsValid)

{

db.Entry(employee).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

return View(employee);

}

// GET: Employees/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Employee employee = db.Employees.Find(id);

if (employee == null)

{

return HttpNotFound();

}

return View(employee);

}

// POST: Employees/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Employee employee = db.Employees.Find(id);

db.Employees.Remove(employee);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

**Create.cshtml**

@model CCLAB4Task1.Employee

@{

ViewBag.Title = "Create";

}

<h2>Create</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Employee</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Age, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Age, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Age, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Gender, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Gender, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Gender, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Salary, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Salary, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Salary, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Designation, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Designation, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Designation, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.City, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.City, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.City, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

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

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

**A screenshot of a computer

Description automatically generatedOutput:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Task No. 02**: Implement a product catalog in the MVC .NET web application utilizing ADO.NET Entity Framework. Design a user interface that enables users to browse, add, modify, and remove product records.**Solution:**

**EmployeesController**

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using CCLAB4Task1;

namespace CCLAB4Task1.Controllers

{

public class ProductsController : Controller

{

private EmployeeDBEntities db = new EmployeeDBEntities();

// GET: Products

public ActionResult Index()

{

return View(db.Products.ToList());

}

// GET: Products/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Product product = db.Products.Find(id);

if (product == null)

{

return HttpNotFound();

}

return View(product);

}

// GET: Products/Create

public ActionResult Create()

{

return View();

}

// POST: Products/Create

// To protect from overposting attacks, enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "ID,Name,Description,Price")] Product product)

{

if (ModelState.IsValid)

{

db.Products.Add(product);

db.SaveChanges();

return RedirectToAction("Index");

}

return View(product);

}

// GET: Products/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Product product = db.Products.Find(id);

if (product == null)

{

return HttpNotFound();

}

return View(product);

}

// POST: Products/Edit/5

// To protect from overposting attacks, enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "ID,Name,Description,Price")] Product product)

{

if (ModelState.IsValid)

{

db.Entry(product).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

return View(product);

}

// GET: Products/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Product product = db.Products.Find(id);

if (product == null)

{

return HttpNotFound();

}

return View(product);

}

// POST: Products/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Product product = db.Products.Find(id);

db.Products.Remove(product);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

**Create.cshtml**

@model CCLAB4Task1.Product

@{

ViewBag.Title = "Create";

}

<h2>Create</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Product</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Name, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Description, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Description, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Description, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Price, htmlAttributes: new { @class = "control-label col-md-2" })

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

@Html.EditorFor(model => model.Price, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Price, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

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

<input type="submit" value="Create" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "Index")

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

**A screenshot of a computer

Description automatically generated**

**Output:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer screen

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A close up of a product

Description automatically generated**