Home Controllers:

﻿using Microsoft.AspNetCore.Mvc;

using System.Diagnostics;

using WebPizzaApp.Models;

namespace WebPizzaApp.Controllers

{

public class HomeController : Controller

{

private readonly ILogger<HomeController> \_logger;

public HomeController(ILogger<HomeController> logger)

{

\_logger = logger;

}

public IActionResult Index()

{

return View();

}

public IActionResult Privacy()

{

return View();

}

[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]

public IActionResult Error()

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

}

}

Pizza Controller:

﻿using Microsoft.AspNetCore.Mvc;

using WebPizzaApp.Models;

namespace WebPizzaApp.Controllers

{

public class PizzaController : Controller

{

private static readonly List<Pizza> PizzaTypes = new List<Pizza>

{

new Pizza { PizzaId = 1, PizzaType = " Mushroom Pizza ", Price = 450.45 },

new Pizza { PizzaId = 2, PizzaType = "Pepperoni Pizza", Price = 350 },

new Pizza { PizzaId = 3, PizzaType = " Chicken Pizza ", Price = 550.50 },

new Pizza { PizzaId = 4, PizzaType = "Plain Pizza", Price = 99.50 },

new Pizza { PizzaId = 5, PizzaType = " Panner Pizza ", Price = 450.50 },

new Pizza { PizzaId = 6, PizzaType = " Margherita ", Price = 660.50 }

// Add more pizza types as needed

};

[HttpGet]

public IActionResult PizzaSelection()

{

// Pass pizza types to the PizzaSelection view

return View(PizzaTypes);

}

[HttpGet]

public IActionResult OrderCheckout(string pizzaType)

{

// Get the selected pizza based on the pizza type

var selectedPizza = PizzaTypes.FirstOrDefault(pizza => pizza.PizzaType == pizzaType);

if (selectedPizza == null)

{

// Handle invalid pizza type

return RedirectToAction("PizzaSelection");

}

// Pass data to the OrderCheckout view

var model = new Order

{

Pizza = pizzaType

};

return View(model);

}

[HttpPost]

public IActionResult OrderConfirmation(string pizzaType, int quantity)

{

// Your logic to process the order (save to database, etc.)

// Retrieve pizza details based on the selected pizza in the order

var selectedPizza = PizzaTypes.FirstOrDefault(pizza => pizza.PizzaType == pizzaType);

if (selectedPizza == null)

{

// Handle invalid pizza type

return RedirectToAction("PizzaSelection");

}

// Calculate the total order amount

var orderAmount = CalculateOrderAmount(selectedPizza.Price, quantity);

// For simplicity, assuming you save the order and retrieve the order details

var confirmedOrder = new ConfirmOrder

{

OrderId = GenerateOrderId(),

Pizza = selectedPizza.PizzaType,

Quantity = quantity,

Amount = orderAmount

};

// Redirect to OrderConfirmation view with the confirmed order details

return View("OrderConfirmation", confirmedOrder);

}

public string GenerateOrderId()

{

// Replace this with your actual logic to generate a unique order ID

// For simplicity, returning a dummy order ID

return Guid.NewGuid().ToString();

}

public double CalculateOrderAmount(double pizzaPrice, int quantity)

{

// Replace this with your actual logic to calculate the order amount

// For simplicity, returning a fixed amount for each pizza type

return pizzaPrice \* quantity;

}

}

}

ConfirmOrder:

﻿namespace WebPizzaApp.Models

{

public class ConfirmOrder

{

public string? OrderId { get; set; }

public string? Pizza { get; set; }

public int Quantity { get; set; }

public double Amount { get; set; }

}

}

ErrorViewModel:

namespace WebPizzaApp.Models

{

public class ErrorViewModel

{

public string? RequestId { get; set; }

public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);

}

}

Order.cs:

﻿namespace WebPizzaApp.Models

{

public class Order

{

public string? Pizza { get; set; }

public int Quantity { get; set; }

public double Amount { get; set; }

}

}

Pizza.cs:

﻿namespace WebPizzaApp.Models

{

public class Pizza

{

public int PizzaId { get; set; }

public string? PizzaType { get; set; }

public double Price { get; set; }

}

}

Index.cshtml:

﻿@{

ViewData["Title"] = "Home Page";

}

<div class="text-center">

<h1 class="display-4">Welcome</h1>

<p>Learn about <a href="https://docs.microsoft.com/aspnet/core">building Web apps with ASP.NET Core</a>.</p>

</div>

Privacy.cshtml:

﻿@{

ViewData["Title"] = "Privacy Policy";

}

<h1>@ViewData["Title"]</h1>

<p>Use this page to detail your site's privacy policy.</p>

OrderCheckout.cshtml:

﻿@model WebPizzaApp.Models.Order

@{

ViewData["Title"] = "OrderCheckout";

}

<h1>Order Checkout</h1>

<p>Selected Pizza: @Model.Pizza</p>

<form asp-action="OrderConfirmation" asp-controller="Pizza" method="post">

<input type="hidden" name="pizzaType" value="@Model.Pizza" />

<label for="quantity">Quantity:</label>

<input type="number" id="quantity" name="quantity" min="1" required />

<button type="submit">Place Order</button>

</form>

OrderConfirmation.cshtml:

﻿@model WebPizzaApp.Models.ConfirmOrder

@{

ViewData["Title"] = "OrderConfirmation";

}

<h1>Order Confirmation</h1>

<p>Order ID: @Model.OrderId</p>

<p>Pizza: @Model.Pizza</p>

<p>Quantity: @Model.Quantity</p>

<p>Amount: @Model.Amount</p>

<form asp-action="MakePayment" asp-controller="Pizza">

<button type="submit">Make Payment</button>

</form>

<form asp-action="CancelOrder" asp-controller="Pizza">

<button type="submit">Cancel Order</button>

</form>

PizzaSElection:

﻿@model IEnumerable<WebPizzaApp.Models.Pizza>

@{

ViewData["Title"] = "PizzaSelection";

}

<h1>PizzaSelection</h1>

<table class="table">

<thead>

<tr>

<th>

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

</th>

<th>

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

</th>

<th>

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

</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.PizzaId)

</td>

<td>

@Html.DisplayFor(modelItem => item.PizzaType)

</td>

<td>

@Html.DisplayFor(modelItem => item.Price)

</td>

<td>

<a asp-action="OrderCheckout" asp-controller="Pizza" asp-route-pizzaType="@item.PizzaType">Order</a>

</td>

</tr>

}

</tbody>

</table>