using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Threading.Tasks;

namespace Joes\_Pizza\_Shop.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

public partial class CustomerOrderTable

{

[Key]

public int CustomerOrderID { get; set; }

[ForeignKey("PizzaItem")]

public Nullable<int> PID { get; set; }

public int Quantity { get; set; }

public int Purchased { get; set; }

[ForeignKey("Order")]

public Nullable<int> OrderID { get; set; }

public Nullable<double> TotalPricePerItem { get { return (double)Quantity \* PizzaItem.Price; } }

public virtual Order Order { get; set; }

public virtual PizzaItem PizzaItem { get; set; }

}

}

using Microsoft.EntityFrameworkCore;

namespace Joes\_Pizza\_Shop.Models

{

public class DataContext : DbContext

{

public DataContext()

{

}

public DataContext(DbContextOptions<DataContext> options)

: base(options)

{

}

public virtual DbSet<PizzaItem> PizzaItems { get; set; }

public virtual DbSet<CustomerOrderTable> CustomerOrderTables { get; set; }

public virtual DbSet<Order> Orders { get; set; }

protected override void OnModelCreating(ModelBuilder builder)

{

builder.Entity<CustomerOrderTable>().ToTable("CustomerOrderTable");

builder.Entity<Order>().ToTable("Order");

}

}

}

using System;

namespace Joes\_Pizza\_Shop.Models

{

public class ErrorViewModel

{

public string RequestId { get; set; }

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

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Threading.Tasks;

namespace Joes\_Pizza\_Shop.Models

{

public partial class Order

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Order()

{

this.CustomerOrderTables = new HashSet<CustomerOrderTable>();

}

[Key]

public int OrderID { get; set; }

public Nullable<System.DateTime> OrderDate { get; set; }

public Nullable<double> TotalAmount { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<CustomerOrderTable> CustomerOrderTables { get; set; }

}

}

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

namespace Joes\_Pizza\_Shop.Models

{

public partial class PizzaItem

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public PizzaItem()

{

this.CustomerOrderTables = new HashSet<CustomerOrderTable>();

}

[Key]

public int PID { get; set; }

public string PizzaName { get; set; }

public double Price { get; set; }

public string image { get; set; }

public string Description { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<CustomerOrderTable> CustomerOrderTables { get; set; }

}

}

using Joes\_Pizza\_Shop.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

namespace Joes\_Pizza\_Shop.Controllers

{

public class HomeController : Controller

{

private readonly DataContext db;

public HomeController(DataContext context)

{

//DataContext object

db = context;

}

public IActionResult Index()

{

TheCart\_OnLoad();

//Show the products' contents for the customers

var productsList = (from d in db.PizzaItems

select d).ToList();

return View(productsList);

}

// Menu page

public IActionResult Menu()

{

TheCart\_OnLoad();

//Show the products' contents for the customers

var productsList = (from d in db.PizzaItems

select d).ToList();

return View(productsList);

}

public IActionResult AddToCart(int id)

{

PizzaItem PID = db.PizzaItems.Find(id);

if (ModelState.IsValid)

{

using (db)

{

//check if the item already added to the cart by chicking the existence of PID

var obj = db.CustomerOrderTables.Where(model => model.PID == PID.PID && model.Purchased == 0).FirstOrDefault();

if (obj != null)

{

TempData["AlertMsg"] = "<script> divModal.style.display = 'block';</script>";

}

else

{

//insert the values to the table

CustomerOrderTable CCT = new CustomerOrderTable();

// fields to be inserted

CCT.PID = PID.PID;

CCT.Quantity = 1;

CCT.Purchased = 0;

db.CustomerOrderTables.Add(CCT);

db.SaveChanges();

}

}

}

return RedirectToAction("Menu");

}

//Fuction to count the cart content

public int TheCart\_OnLoad()

{

var CartNumbers = (from d in db.CustomerOrderTables

where d.Purchased == 0

select d).Count();

TempData["CartNum"] = CartNumbers;

return (CartNumbers);

}

public IActionResult OrderCheckout()

{

// Call the function to display the cart content' count.

TheCart\_OnLoad();

//Calculate the total price of all products that are added to the cart.

var TotalAmount = (from cot in db.CustomerOrderTables

join p in db.PizzaItems on cot.PID equals p.PID

where cot.Purchased == 0

select cot.Quantity \* p.Price).Sum();

if (TotalAmount == 0)

{

TempData["msg"] = "<script> divEmptyCart.style.display = 'block'</script>";

}

else

{

//Save the result in the Session and TempData

HttpContext.Session.SetString("TotalAmountSession", Convert.ToString(String.Format("{0:0.00}", TotalAmount)));

TempData["TotalAmount"] = Convert.ToString(String.Format("{0:0.00}", TotalAmount));

}

//Use the CustomerOrderTables to displays the cart contents

return View(db.CustomerOrderTables.Where(x=> x.Purchased == 0).ToList());

}

//Function to increase the selected product quantity

public IActionResult IncreaseQuantity(int id)

{

//Update the Quantity in CustomerOrderTable Table by using CustomerOrderID and Purchased

CustomerOrderTable COT = db.CustomerOrderTables.Where(u => u.CustomerOrderID == id && u.Purchased == 0).FirstOrDefault();

if (ModelState.IsValid)

{

using (db)

{

//Update the Quantity by Increasing the value in every click on the plus icon

COT.Quantity += 1;

db.SaveChanges();

}

}

return RedirectToAction("OrderCheckout");

}

//Function to decreasee the selected product quantity

public IActionResult DecreaseQuantity(int id)

{

CustomerOrderTable COT = db.CustomerOrderTables.Where(u => u.CustomerOrderID == id && u.Purchased == 0).FirstOrDefault();

if (ModelState.IsValid)

{

using (db)

{

//Check if the value is less than 1 before decreasing

if (COT.Quantity > 1)

{

//Update the Quantity by decreasing the value in every click on the minus icon.

COT.Quantity -= 1;

db.SaveChanges();

}

}

}

return RedirectToAction("OrderCheckout");

}

//Delete specific item from the cart

public ActionResult DeleteItem(int id)

{

CustomerOrderTable COT = db.CustomerOrderTables.Find(id);

db.CustomerOrderTables.Remove(COT);

db.SaveChanges();

return RedirectToAction("OrderCheckout");

}

public IActionResult Checkout()

{

var TotalAmount = HttpContext.Session.GetString("TotalAmountSession");

List<CustomerOrderTable> COT = db.CustomerOrderTables.Where(u => u.Purchased == 0).ToList();

Order ord = new Order();

if (ModelState.IsValid)

{

using (db)

{

//Insert the values to the Order table

ord.TotalAmount = Convert.ToDouble(TotalAmount);

ord.OrderDate = DateTime.Now;

db.Orders.Add(ord);

db.SaveChanges();

//Update the Purchased and OrderID columns in CustomerOrderTable Table

foreach (var item in COT)

{

item.OrderID = ord.OrderID;

item.Purchased = 1;

db.SaveChanges();

//Save the item.OrderID value to a session

HttpContext.Session.SetString("OrderIDSession", Convert.ToString(item.OrderID));

}

}

}

return RedirectToAction("OrderConfirmation");

}

//Show the the final order's details when the customer finished of paying by using OrderID Session.

public IActionResult OrderConfirmation()

{

int OrderID = Convert.ToInt32(HttpContext.Session.GetString("OrderIDSession"));

ViewData["Orders Details"] = db.CustomerOrderTables.Where(model => model.OrderID == OrderID && model.Purchased == 1).ToList();

//Use FirstOrDefault to show only one order's details

ViewData["Order"] = db.Orders.Where(model => model.OrderID == OrderID).FirstOrDefault();

ViewBag.OrderID = OrderID;

return View();

}

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

public IActionResult Error()

{

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

}

}

}

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<!DOCTYPE html>

<html lang="en">

<head>

<title>Joe's Pizza</title>

</head>

<body data-spy="scroll" data-target="#navbar" class="static-layout">

<div class="boxed-page">

<div class="hero">

<div class="container">

<div class="row d-flex align-items-center">

<div class="col-lg-6 hero-left">

<h1 class="display-4 mb-5">We Love <br>Delicious Foods!</h1>

<div class="mb-2">

<a class="btn btn-primary btn-shadow btn-lg" href="/Home/Menu" role="button">Explore Menu</a>

</div>

<ul class="hero-info list-unstyled d-flex text-center mb-0">

<li class="border-right">

<span class="lnr lnr-rocket"></span>

<h5>

Fast Delivery

</h5>

</li>

<li class="border-right">

<span class="lnr lnr-leaf"></span>

<h5>

Fresh Food

</h5>

</li>

<li class="">

<span class="lnr lnr-bubble"></span>

<h5>

24/7 Support

</h5>

</li>

</ul>

</div>

<div class="col-lg-6 hero-right">

<div class="owl-carousel owl-theme hero-carousel">

<div class="item">

<img class="img-fluid" src="/img/pizza-1.jpg" alt="">

</div>

<div class="item">

<img class="img-fluid" src="/img/pizza-2.jpg" alt="">

</div>

<div class="item">

<img class="img-fluid" src="/img/pizza-3.jpg" alt="">

</div>

</div>

</div>

</div>

</div>

</div> <!-- About Section -->

<section id="gtco-about" class="bg-white section-padding mt-5 mb-5">

<div class="container">

<div class="section-content">

<div class="row">

<div class="col-sm-5 img-bg d-flex shadow align-items-center justify-content-center justify-content-md-end img-2" style="background-image: url(/img/pizza-2.jpg);">

</div>

<div class="col-sm-7 py-5 pl-md-0 pl-4">

<div class="heading-section pl-lg-5 ml-md-5">

<span class="subheading">

About

</span>

<h2>

Welcome to Joe's Pizza

</h2>

</div>

<div class="pl-lg-5 ml-md-5">

<p>

It attracts with its original interior but above all the quality of the dishes served. Carefully prepared dishes are characterized by freshness and a lot of ingredients. Visit us and discover the world of unparalleled flavors. The atmospheric interior will allow you to relax and fully reflect the delicious taste of our dishes.

</p>

</div>

</div>

</div>

</div>

</div>

</section>

<!-- End of About Section --> <!-- Special Dishes Section -->

<section id="gtco-special-dishes" class="bg-grey section-padding">

<div class="container">

<div class="section-content">

<div class="heading-section text-center">

<span class="subheading">

Specialties

</span>

<h2>

Special Dishes

</h2>

</div>

<div class="row mt-5" id="first-meal">

<div class="col-lg-5 col-md-6 align-self-center py-5">

<h2 class="special-number">01.</h2>

<div class="dishes-text">

<h3><span>Chicken</span><br> Mushrooms</h3>

<p class="pt-3">Baked Chicken breast, Mushrooms, Mozzarella Cheese, American Cheese & Dynamite Sauce.</p>

<h3 class="special-dishes-price">$ 15.99</h3>

<a href="#" class="btn-primary mt-3">Order Now</a>

</div>

</div>

<div class="col-lg-5 offset-lg-2 col-md-6 align-self-center mt-4 mt-md-0">

<img src="/PizzaImages/Chicken Mushrooms.jpg" alt="" class="img-fluid shadow w-100">

</div>

</div>

<div class="row mt-5" id="second-meal">

<div class="col-lg-5 col-md-6 align-self-center order-2 order-md-1 mt-4 mt-md-0">

<img src="/PizzaImages/Veggie.jpg" alt="" class="img-fluid shadow w-100">

</div>

<div class="col-lg-5 offset-lg-2 col-md-6 align-self-center order-1 order-md-2 py-5">

<h2 class="special-number">02.</h2>

<div class="dishes-text">

<h3><span>Veggie</span><br> Pizza</h3>

<p class="pt-3">Red Peppers, Onions, Mushrooms, Black Olives, basil, Mozzarella & Signature Tomato Sauce.</p>

<h3 class="special-dishes-price">$ 14.99</h3>

<a href="#" class="btn-primary mt-3">Order Now <span><i class="fa fa-long-arrow-right"></i></span></a>

</div>

</div>

</div>

</div>

</div>

</section>

<!-- End of Special Dishes Section -->

</div>

</body>

</html>

@model IEnumerable<Joes\_Pizza\_Shop.Models.PizzaItem>

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<!DOCTYPE html>

<html lang="en">

<head>

<title>The Menu</title>

<!-- JS -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

<script src="/vendor/bootstrap/retain-position.js"></script>

</head>

<body data-spy="scroll" data-target="#navbar">

<div class="boxed-page">

<!-- Menu Section -->

<section id="gtco-menu" class="section-padding">

<div class="container">

<div class="section-content">

<div class="row mb-5">

<div class="col-md-12">

<div class="heading-section text-center">

<span class="subheading">

Specialties

</span>

<h2>

Our Menu

</h2>

</div>

</div>

</div>

<div class="row">

@foreach (var item in Model)

{

<div class="col-lg-6 menus d-flex">

<div class="menu-img">

<**img** class=".img-square" **src**="/PizzaImages/@item.image" **asp-append-version**="true" alt="">

</div>

<div class="text-wrap">

<div class="row align-items-start">

<div class="col-8">

<h4>@Html.DisplayFor(modelItem => item.PizzaName)</h4>

<p>@Html.DisplayFor(modelItem => item.Description)</p>

</div>

<div class="col-4">

<h4 class="text-muted menu-price">$@Html.DisplayFor(modelItem => item.Price)</h4>

<div class="text-right">

<a href="/Home/AddToCart/@item.PID" class="btn btn-primary" style="padding: 3px 12px !important;"><img class="img-style" style="width:20px; height:20px" src="/img/add.png" /></a>

</div>

</div>

</div>

</div>

</div>

//pop-up

<div id="divModal" class="modal">

@Html.Raw(TempData["AlertMsg"])

<!-- modal content -->

<div class="modal-content">

<div class="modal-body justify-content-center text-center">

<**img** class="pb-2" style="width: 60px;" src="~/img/check-mark.png" />

<h6 class="pb-2">It is Already in The Cart.</h6>

<span class="btn-primary close ok">OK</span>

</div>

</div>

</div>

}

</div>

</div>

</div>

</section>

<!-- End of menu Section -->

</div>

</body>

</html>

@model List<Joes\_Pizza\_Shop.Models.CustomerOrderTable>

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<!DOCTYPE html>

<html lang="en">

<head>

<title>Order Checkout</title>

<!-- JS -->

<script src="/vendor/checkout/checkout.js"></script>

</head>

<body data-spy="scroll" data-target="#navbar" onload="disabledBtnProceed()">

<div class="boxed-page">

<!-- Order Checkout Section -->

<section id="gtco-OrderCheckout" class="bg-fixed bg-white section-padding justify-content-center">

<div class="container text-center">

<div class="row justify-content-center pb-5">

<span class="col-4 subheading-red"><strong>&#9312; Shopping Cart</strong></span>

<span class="col-4" id="spanCheckoutColor"><strong>&#9313; Billing & Checkout</strong></span>

<span class="col-4"><strong>&#9314; Order Confirmation</strong></span>

</div>

<!-- Shopping Cart Div -->

<div class="row justify-content-center" id="DivShoppingCart">

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

<div class="section-content bg-white">

<div class="heading-section">

<h2>

The Cart

</h2>

</div>

<div class="row">

<div class="col-md-12 col-lg-12">

<div class="row mb-0 text-center">

<div class="col-1 themed-grid-col text-muted"></div>

<div class="col-3 themed-grid-col text-muted"><h6>Item Name</h6></div>

<div class="col-2 themed-grid-col text-muted"><h6>Price</h6></div>

<div class="col-3 themed-grid-col text-muted"><h6>Quantity</h6></div>

<div class="col-2 themed-grid-col text-muted"><h6>Total</h6></div>

</div>

<hr class="my-4">

<div class="Customeritems">

@foreach (var item in Model)

{

<div class="rowClass row mb-1 text-center">

<div class="col-1 themed-grid-col text-muted">

<a href="/Home/DeleteItem/@item.CustomerOrderID" class="btn btn-primary" style="padding: 0px 3px !important;"> <img class="img-style" style="width:15px; height:15px" src="/img/x.png" /></a>

</div>

<div class="col-3 themed-grid-col text-muted">

<h6 class="my-0">@item.PizzaItem.PizzaName</h6>

<br />

</div>

<div class="col-2 themed-grid-col text-muted"><span class="text-muted price">$@item.PizzaItem.Price</span></div>

<div class="col-3 themed-grid-col text-muted">

<a href="/Home/DecreaseQuantity/@item.CustomerOrderID" class="btn btn-primary" style="padding: 0px 3px !important;"> <img class="img-style" style="width:15px; height:15px" src="/img/minus.png" /></a>

<span class="text-muted Quantity">@item.Quantity</span>

<a href="/Home/IncreaseQuantity/@item.CustomerOrderID" class="btn btn-primary" style="padding: 0px 3px !important;"> <img class="img-style" style="width:15px; height:15px" src="/img/add.png" /></a>

</div>

<div class="col-2 themed-grid-col text-muted"><span class="text-muted Total">$@item.TotalPricePerItem</span></div>

</div>

}

<div id="divEmptyCart" class="heading-section text-center" style="display:none">

<span class="subheading" style="font-size:20px">@Html.Raw(TempData["msg"])</span>

<img style="width: 200px;" src="/img/empty-cart.png" />

</div>

<hr class="my-4">

<div class="row">

<div class="col-8 themed-grid-col text-muted">

<h5 class="my-0 text-left">Total Price. $@TempData["TotalAmount"]</h5>

</div>

<div class="col-4 themed-grid-col text-muted">

<button class="btn btn-lg btn-primary text-right" id="btnProceed" onclick="hideCart()">Proceed</button>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- Payment Div -->

<div class="row justify-content-center" id="DivPayment" style="display:none">

<div class="col-lg-9">

<div class="section-content bg-white p-3 shadow">

<div class="heading-section">

<span class="subheading">

Order

</span>

<h2>

Checkout

</h2>

</div>

@using (Html.BeginForm("Checkout", "Home", FormMethod.Post, new { @class = "needs-validation", @novalidate = "" }))

{

@Html.AntiForgeryToken()

<div class="row">

<div class="col-6">

<div class="col-10">

<label for="fullName" class="form-label">Full Name</label>

<input type="text" class="form-control" id="fullName" placeholder="" value="" required>

<div class="invalid-feedback">

Valid first name is required.

</div>

</div>

<div class="col-10">

<label for="email" class="form-label">Email</label>

<div class="input-group has-validation">

<input type="email" class="form-control" id="email" placeholder="you@example.com" required>

<div class="invalid-feedback">

Please enter a valid email address.

</div>

</div>

</div>

<div class="col-10">

<label for="address" class="form-label">Address</label>

<input type="text" class="form-control" id="address" placeholder="1234 Main St" required>

<div class="invalid-feedback">

Please enter your shipping address.

</div>

</div>

</div>

<div class="col-6">

<div class="text-left">

<h5 class="mb-3">Payment</h5>

<div class="my-3">

<div class="form-check">

<input id="credit" name="paymentMethod" type="radio" class="form-check-input" checked required>

<label class="form-check-label" for="credit">Credit card</label>

</div>

<div class="form-check">

<input id="debit" name="paymentMethod" type="radio" class="form-check-input" required>

<label class="form-check-label" for="debit">Debit card</label>

</div>

<div class="form-check">

<input id="paypal" name="paymentMethod" type="radio" class="form-check-input" required>

<label class="form-check-label" for="paypal">PayPal</label>

</div>

</div>

</div>

<div class="row justify-content-start">

<div class="col-6">

<label for="cc-name" class="form-label">Name on card</label>

<input type="text" class="form-control" id="cc-name" placeholder="" required>

<small class="text-muted">Full name as displayed on card</small>

<div class="invalid-feedback">

Name on card is required

</div>

</div>

<div class="col-6">

<label for="cc-number" class="form-label">Credit card number</label>

<input type="text" class="form-control" id="cc-number" placeholder="" required>

<div class="invalid-feedback">

Credit card number is required

</div>

</div>

</div>

<div class="row">

<div class="col-3">

<label for="cc-expiration" class="form-label">Exp</label>

<input type="text" class="form-control" id="cc-expiration" placeholder="" required>

<div class="invalid-feedback">

Expiration date required

</div>

</div>

<div class="col-3">

<label for="cc-cvv" class="form-label">CVV</label>

<input type="text" class="form-control" id="cc-cvv" placeholder="" maxlength="3" required>

<div class="invalid-feedback">

Security code required

</div>

</div>

</div>

</div>

</div>

<hr class="my-4">

<div class="col-md-12">

<div class="row">

<div class="col-5 text-left">

<button id="btnBack" type="button" class="btn" onclick="hideCheckout()"><img style="width: 20px;" src="/img/left-arrow.png" /></button>

</div>

<div class="col-7 text-left">

<button id="btnPay" class="btn btn-primary btn-shadow btn-lg " type="submit" onclick="payment()">Pay</button>

</div>

</div>

</div>

}

</div>

</div>

</div>

</div>

</section>

<!-- End of Order Checkout Section -->

</div>

</body>

</html>

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

@{

List<CustomerOrderTable> COT = ViewData["Orders Details"] as List<CustomerOrderTable>;

Order ord = ViewData["Order"] as Order;

}

<!DOCTYPE html>

<html lang="en">

<head>

<title>Order Confirmation</title>

</head>

<body data-spy="scroll" data-target="#navbar">

<div class="boxed-page">

<section id="gtco-OrderConfirmation" class="bg-fixed bg-white section-padding justify-content-center">

<div class="container text-center">

<div class="row justify-content-center pb-5">

<span class="col-4 subheading-red"><strong>&#9312; Shopping Cart</strong></span>

<span class="col-4 subheading-red"><strong>&#9313; Billing & Checkout</strong></span>

<span class="col-4 subheading-red"><strong>&#9314; Order Confirmation</strong></span>

</div>

<div class="row justify-content-center" id="DivOrderConfirm">

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

<div class="section-content bg-white p-3 shadow">

<div class="heading-section">

<span class="subheading">

Confirmation

</span>

<h2>

The Order

</h2>

<img style="width: 85px;" src="/img/checked.png" />

<div class="py-3">

<span class="subheading" style="font-size:18px">Order No. @ord.OrderID</span>

</div>

</div>

<div class="row">

<div class="col-md-12 col-lg-12">

<div class="row mb-0 text-center">

<div class="col-1 themed-grid-col text-muted"></div>

<div class="col-3 themed-grid-col text-muted"><h6>Item Name</h6></div>

<div class="col-2 themed-grid-col text-muted"><h6>Price</h6></div>

<div class="col-3 themed-grid-col text-muted"><h6>Quantity</h6></div>

<div class="col-2 themed-grid-col text-muted"><h6>Total</h6></div>

</div>

<hr class="my-4">

<div class="Customeritems">

@foreach (var item in COT)

{

<div class="rowClass row mb-1 text-center">

<div class="col-1 themed-grid-col text-muted"></div>

<div class="col-3 themed-grid-col text-muted">

<h6 class="my-0">@item.PizzaItem.PizzaName</h6>

<br />

</div>

<div class="col-2 themed-grid-col text-muted"><span class="text-muted price">$@item.PizzaItem.Price</span></div>

<div class="col-3 themed-grid-col text-muted">

<span class="text-muted Quantity">@item.Quantity</span>

</div>

<div class="col-2 themed-grid-col text-muted"><span class="text-muted Total">$@item.TotalPricePerItem</span></div>

</div>

}

<hr class="my-4">

<div class="row mb-2">

<div class="col-12 themed-grid-col text-muted">

<h5 class="my-0 text-center">Total Price. $@ord.TotalAmount</h5>

</div>

</div>

<div class="row mb-2 justify-content-center">

<div class="col-6 themed-grid-col ">

<span class="subheading">Order Date. @ord.OrderDate </span>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</section>

<!-- End of Order Checkout Section -->

</div>

</body>

</html>

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"AllowedHosts": "\*",

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\MSSQLLocalDB;Database=Mohan;Trusted\_Connection=True;"

}

}

Feature: PizzaPurchasing

*A simple website is for purchasing a pizza.*

@PizzaPurchasing

Scenario: Select pizza and purchase it.

Given launch joe's pizza website

Then Click on Navbar Menu Link

Then Menu page should be displayed

And select pizza

And Click on Cart Icon

Then Order Checkout page should be displayed

Then change the quantity in Checkout page

And Click on Proceed button

Then Payment section should be displayed

Then fill the input fields

And Click on Pay button

Then Order Confirmation page should be displayed

using Joes\_Pizza\_Test.pages;

using NUnit.Framework;

using OpenQA.Selenium;

using OpenQA.Selenium.Firefox;

using TechTalk.SpecFlow;

namespace Joes\_Pizza\_Test

{

[Binding]

public class PizzaPurchasingSteps

{

//Create objects

Pages\_ElementsDefinition elm = new Pages\_ElementsDefinition();

IWebDriver webDriver = new FirefoxDriver();

[Given(@"launch joe's pizza website")]

public void GivenLaunchJoeSPizzaWebsite()

{

webDriver.Navigate().GoToUrl("https://localhost:44379/");

elm = new Pages\_ElementsDefinition(webDriver);

}

[Then(@"Click on Navbar Menu Link")]

public void ThenClickOnNavbarMenuLink()

{

elm.ClickInkMenuNavbar();

}

[Then(@"Menu page should be displayed")]

public void ThenMenuPageShouldBeDisplayed()

{

//Using Assert to verify the existence of the Menu page

Assert.That(elm.IslnkMenuPageExist, Is.True);

}

[Then(@"select pizza")]

public void GivenSelectPizza()

{

elm.ClickbtnAddToCart1();

elm.ClickbtnAddToCart2();

elm.ClickbtnAddToCart3();

elm.ClickbtnAddToCart1();

//if the customer clicks on the same product, an popup alert message will be shown.

elm.ClickbtnPopupAlert();

}

[Then(@"Click on Cart Icon")]

public void GivenClickOnCartIcon()

{

//Click on the cart icon to display the Order Checkout page

elm.ClickCart();

}

[Then(@"Order Checkout page should be displayed")]

public void ThenOrderCheckoutPageShouldBeDisplayed()

{

//Using Assert to verify the existence of the Order Checkout page

Assert.That(elm.IslnkCheckoutPageExist, Is.True);

}

[Then(@"change the quantity in Checkout page")]

public void ThenChangeTheQuantityInCheckoutPage()

{

//The customer can increase or decrease the quantity or even delete the item from the cart.

//NOTE:These IWebElement numbers are constantly changing automatically.

//elm.ClickbtnIncrease1();

//elm.ClickbtnIncrease1();

//elm.ClickbtnIncrease2();

//elm.ClickbtnDecrease();

//elm.ClickbtnDeleteItem();

}

[Then(@"Click on Proceed button")]

public void ThenClickOnProceedButton()

{

elm.ClickbtnProceed();

}

[Then(@"Payment section should be displayed")]

public void ThenPaymentSectionShouldBeDisplayed()

{

//Using Assert to verify the existence of the Payment section

Assert.That(elm.IsPaymentSectionExist, Is.True);

}

[Then(@"fill the input fields")]

public void ThenFillTheInputFields()

{

elm.FilltxtFullName();

elm.FilltxtEmail();

elm.FilltxtAddress();

elm.FilltxtCCName();

elm.FilltxtCCNumber();

elm.FilltxtCCExpiration();

elm.FilltxtCCcvv();

}

[Then(@"Click on Pay button")]

public void ThenClickOnCheckoutButton()

{

//Click on the pay button to display the Order Confirmation page

elm.ClickbtnPay();

}

[Then(@"Order Confirmation page should be displayed")]

public void ThenOrderConfirmationPageShouldBeDisplayed()

{

//Using Assert to verify the existence of the Order Confirmation page

Assert.That(elm.IslnkOrderConfirmationPageExist, Is.True);

}

}

}