**HƯỚNG DẪN LẬP TRÌNH WEB VỚI ASP.NET MVC**

**( Phần 3 - Tạo trang chi tiết sản phẩm và giỏ hàng)**

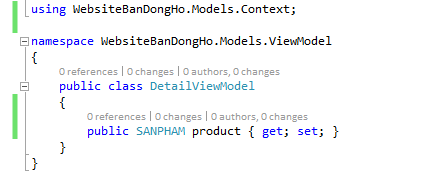
1. Nhiệm vụ

Trong phần trước chúng ta đã làm quen với ASP.NET MVC và tạo ra trang chủ có kết nối với CSDL. Trong chương này chúng ta sẽ đi tiếp vào việc xây dựng một trang chi tiết sản phẩm với một layout dựng sẵn, đỗ dữ liệu từ Model ra View. Đồng thời chúng ta củng sẽ làm việc với Session và Ajax để xử lý phần giỏ hàng.

Về phần masterpage các bạn xem lại phần trước nhé.

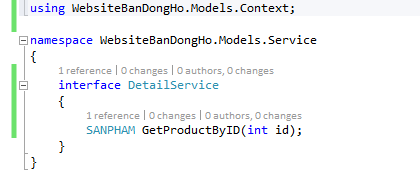
1. Phần chi tiết sản phẩm
   1. Xây dựng Model:

Việc đầu tiên là cần xác định trang chi tiết sẽ chứa những thông tin gì. Tạm thời đơn giản tôi muốn trang chi tiết chứa hình lớn của sản phẩm và các hình nhỏ kèm theo. Bên cạnh đó còn có các thông tin như: Tên sản phẩm, giá bán, mô tả, đánh giá và một button để thêm sản phẩm vào giỏ hàng. Vì vậy trước hết cần tạo DetailViewModel với product (là đối tượng của SANPHAM chứa các thông tin của sản phẩm)

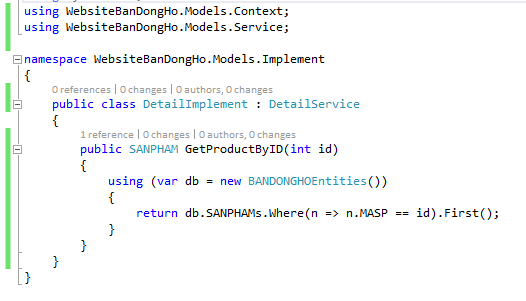


Và chúng ta nhớ using WebsiteBanDongHo.Models.Context; thì mới sử dụng được đối tượng SANPHAM.

Tiếp theo chúng ta tạo một Service để lấy dữ liệu về:



Cuối cùng tạo một class Implement DetailService



Mọi thứ ở Model đã chuẩn bị đầy đủ giờ ta tiếp tục việc gọi service trên Detail Controller.

* 1. Xây dựng DetailController:

Nhiệm vụ quan trọng của DetailController là lấy dữ liệu từ Model gửi lên View thông qua ViewModel

|  |
| --- |
| C# |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebsiteBanDongHo.Models.ViewModel;  using WebsiteBanDongHo.Models.Service;  using WebsiteBanDongHo.Models.Implement;  namespace WebsiteBanDongHo.Controllers  {  public class DetailController : Controller  {  //khai báo và khởi tạo ViewModel  DetailViewModel detailViewModel = new DetailViewModel();  DetailService detailService;  // GET: Detail  public ActionResult Index(int id)  {  //khởi tạo service  detailService = new DetailImplement();  //gán giá trị cho ViewModel  detailViewModel.product = detailService.GetProductByID(id);  //gửi ViewModel ra View  return View(detailViewModel);  }  }  } |

* 1. Xây dựng trang Detail View:

Dữ liệu đã được chuyển sang View rồi giờ làm cách nào để View đón nhận được và hiển thị chúng ra màn hình. ASP.Net sử dụng một đối tượng là Model để lấy ra các dữ liệu được gửi từ Controller. Cách dùng như sau:

|  |
| --- |
| CSHTML |
| @model WebsiteBanDongHo.Models.ViewModel.DetailViewModel  @{  ViewBag.Title = "Detail";  Layout = "~/Views/Shares/masterpage.cshtml";  }  <link href="~/Contents/detail.css" rel="stylesheet" />  <h1>Trang chi tiết sản phẩm</h1>  <div class="container">  <div class="content-left">  <div class="imgContainer">  <ul class="imgBig">  <img src="@Url.Content("~/Contents/images/HINHLON/"+Model.product.HINHLON)" class="img-big" alt="" />  </ul>  <ul class="imgSmall">  <li>  <img src="@Url.Content("~/Contents/images/HINHNHO/"+Model.product.HINHNHO+"/1.jpg")" class="img-responsive" alt="" />  </li>  <li>  <img src="@Url.Content("~/Contents/images/HINHNHO/"+Model.product.HINHNHO+"/2.jpg")" class="img-responsive" alt="" />  </li>  <li>  <img src="@Url.Content("~/Contents/images/HINHNHO/"+Model.product.HINHNHO+"/3.jpg")" class="img-responsive" alt="" />  </li>  </ul>  </div>  </div>  <div class="content-right">  <h2>@Model.product.TENSP</h2>  <h5 class="item\_price">@String.Format("{0:0,0}", (Model.product.DONGIA)) vnđ</h5>  <p><span>Mô tả:</span> @Model.product.MOTA</p>  <p><span>Đánh giá:</span> @Model.product.DANHGIA</p>  <a href="~/Cart/AddProduct/@Model.product.MASP" class="add-cart item\_add">ADD TO CART</a>  </div>  <div class="clearfix"> </div>  </div> |

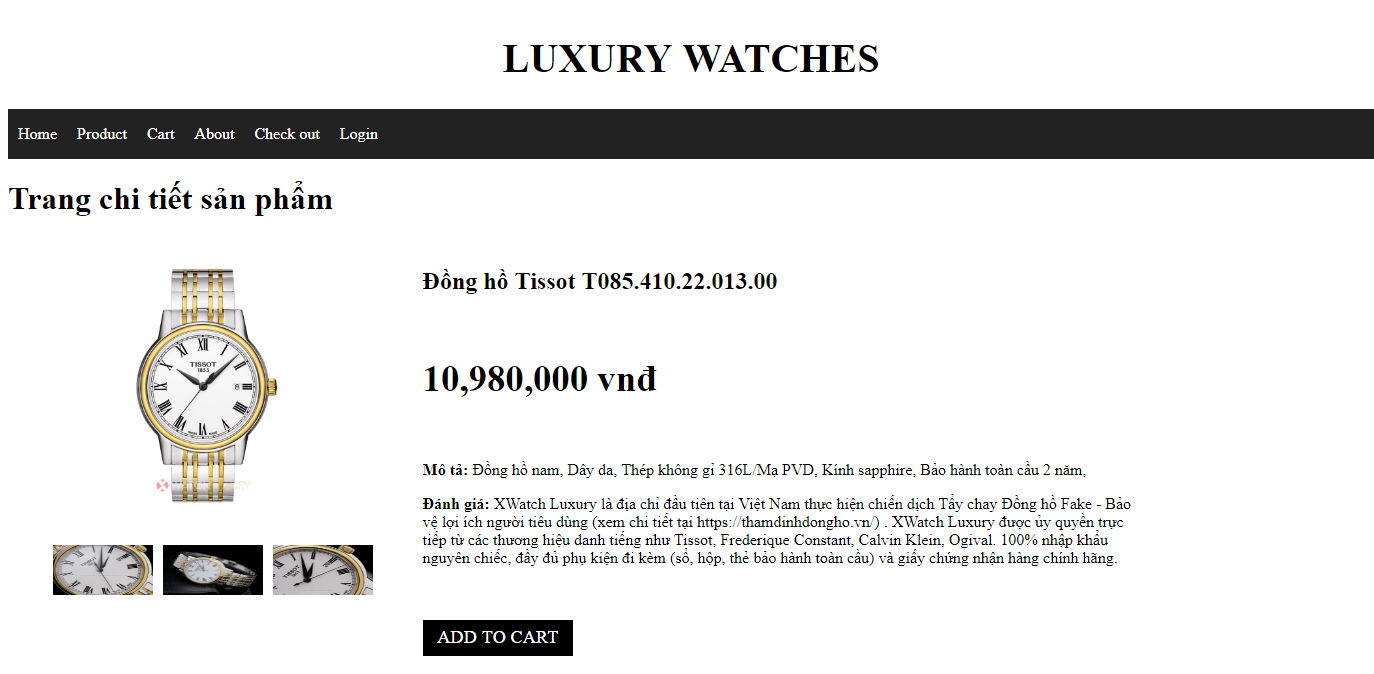
|  |
| --- |
| CSS |
| body {  }  .container {  width: 1140px;  max-width: 100%;  display: inline-flex;  }  .imgContainer {  width: 350px;  }  .img-big {  width: 100%;  }  .imgSmall {  display: inline-flex;  }  .imgSmall li{  margin-left: 5px;  margin-right: 5px;  }  .item\_price {  color: #000;  font-size: 2.3em;  }  .add-cart {  text-decoration: none;  color: #fff;  background: #000;  padding: 0.4em 0.8em;  font-size: 1.15em;  text-transform: uppercase;  margin-top: 2em;  display: inline-block;  }  ul {  list-style: none;  }  .content-left {  width: 40%;  }  .content-right {  width: 70%;  }  .img-responsive {  width: 100px;  height: 50px;  }  .content-right span{  font-weight: bold;  } |

Sửa lại “href” ở thẻ “a” trong Home/Index.cshtml, tìm đến 2 cái foreach trong đó và sửa lại href thẻ a như sau:

href="~/Detail/Index?id=@item.MASP"

Tức là khi ta bấm vào hình ở trang home thì nó sẽ dẫn ta tới trang chi tiết sản phẩm.

Kết quả:



* 1. Tổng kết:

Trong phần này chúng ta đã ôn lại cách tạo ra một layout cho một trang web, lấy dữ liệu lên từ Model, gửi dữ liệu ra View. Qua phần tiếp theo chúng ta sẽ học cách sử dụng session và Ajax để xử lý giỏ hàng.

1. Phần giỏ hàng:
   1. Xây dựng Model:

Việc đầu tiên là cần xác định trang giỏ hàng sẽ chứa những thông tin gì. Tạm thời đơn giản tôi muốn trang giỏ hàng chứa Mã sp, tên sp, ảnh sp, Số lượng muốn mua, đơn giá và tổng tiền. Vì vậy trước hết cần tạo CartItemViewModel với product (là đối tượng của SANPHAM chứa các thông tin của sản phẩm) và Quantity là số lượng sản phẩm muốn mua.

|  |
| --- |
| C# |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using WebsiteBanDongHo.Models.Context;  namespace WebsiteBanDongHo.Models.ViewModel  {  [Serializable]  public class CartItemViewModel  {  public SANPHAM product { get; set; }  public int Quantity { get; set; }  }  } |

Tiếp theo chúng ta tạo một Service để lấy dữ liệu về:

|  |
| --- |
| C# |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using WebsiteBanDongHo.Models.Context;  namespace WebsiteBanDongHo.Models.Service  {  interface CartService  {  SANPHAM GetProductByID(int id);  }  } |

Cuối cùng tạo một class Implement CartService

|  |
| --- |
| C# |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using WebsiteBanDongHo.Models.Context;  using WebsiteBanDongHo.Models.Service;  namespace WebsiteBanDongHo.Models.Implement  {  public class CartImplement : CartService  {  public SANPHAM GetProductByID(int id)  {  using (var db = new BANDONGHOEntities())  {  return db.SANPHAMs.Find(id);  }  }  }  } |

Mọi thứ ở Model đã chuẩn bị đầy đủ giờ ta tiếp tục việc gọi service trên Cart Controller.

* 1. Xây dựng CartController:

Sơ lược về session:

Một session được hiểu như một phiên làm việc giữa client và server, bắt đầu khi có request từ client, kết thúc khi người dùng đóng ứng dụng hoặc khi session hết thời gian timeout. Cũng giống như Cookie, Session được dùng để lưu trữ thông tin liên quan đến người dùng, nhưng thay vì được lưu trên máy client, các biến session được lưu trên máy server.

|  |
| --- |
| C# |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebsiteBanDongHo.Models.Implement;  using WebsiteBanDongHo.Models.Service;  using WebsiteBanDongHo.Models.ViewModel;  namespace WebsiteBanDongHo.Controllers  {  public class CartController : Controller  {  private const string CartSession = "CartSession"; //chuỗi string này là tên hay còn gọi là key của SESSION  CartItemViewModel cartItemViewModel = new CartItemViewModel();  CartService cartService;  // GET: Cart  public ActionResult Index()  {  var cart = Session[CartSession]; //Lấy đối tượng session, mỗi session có key riêng do ta định trước nên ta phải truyền vào chính xác.  var list = new List<CartItemViewModel>();  if (cart != null)  {  list = (List<CartItemViewModel>)cart;  }  return View(list);  }  //Action thêm sản phẩm và số lượng vào Session[CartSession]  public ActionResult AddItem(int productId, int quantity)  {  cartService = new CartImplement();  var product = cartService.GetProductByID(productId);  var cart = Session[CartSession];  if (cart != null)  {  var list = (List<CartItemViewModel>)cart;  if (list.Exists(x => x.product.MASP == productId))  {  foreach (var item in list)  {  if (item.product.MASP == productId)  {  item.Quantity += quantity;  }  }  }  else  {  //tạo mới đối tượng cart item  var item = new CartItemViewModel();  item.product = product;  item.Quantity = quantity;  list.Add(item);  }  //Gán vào session  Session[CartSession] = list;  }  else  {  //tạo mới đối tượng cart item  var item = new CartItemViewModel();  item.product = product;  item.Quantity = quantity;  var list = new List<CartItemViewModel>();  list.Add(item);  //Gán vào session  Session[CartSession] = list;  }  return RedirectToAction("Index");  }  //Action cập nhập số lượng sản phẩm trong Session[CartSession]  public string Update(int id, int quantity)  {  var sessionCart = (List<CartItemViewModel>)Session[CartSession];  foreach (var item in sessionCart)  {  if(item.product.MASP == id)  {  item.Quantity = quantity;  }  }  Session[CartSession] = sessionCart;  return "Ok";  }  //Action xóa sản phẩm trong Session[CartSession]  public string Delete(int id)  {  var sessionCart = (List<CartItemViewModel>)Session[CartSession];  sessionCart.RemoveAll(n => n.product.MASP == id);  return "Ok";  }  }  } |

* 1. Xây dựng Cart View:

Trước tiên chúng ta cần sửa lại ở phần View của DetailController, tìm đến thẻ a của `ADD TO CART` và thêm vào href như bên dưới. Mục đích là khi ta bấm vào `ADD TO CART` ở trang detail thì sản phẩm đó sẽ dược thêm vào giỏ hàng và lưu vào biến session.

|  |
| --- |
| CSHTML |
| <div class="content-right">  <h2>@Model.product.TENSP</h2>  <h5 class="item\_price">@String.Format("{0:0,0}", (Model.product.DONGIA)) vnđ</h5>  <p><span>Mô tả:</span> @Model.product.MOTA</p>  <p><span>Đánh giá:</span> @Model.product.DANHGIA</p>  <a href="~/Cart/AddItem?productID=@Model.product.MASP&quantity=1" class="add-cart item\_add">ADD TO CART</a>  </div> |

Ở phần này chúng ta sẽ sử dụng Jquery và Ajax để xử lý phần xóa và cập nhập số lượng giỏ hàng. Để sử dụng được Jquery ta mở masterpage.cshtml lên và thêm vào trong thẻ head:

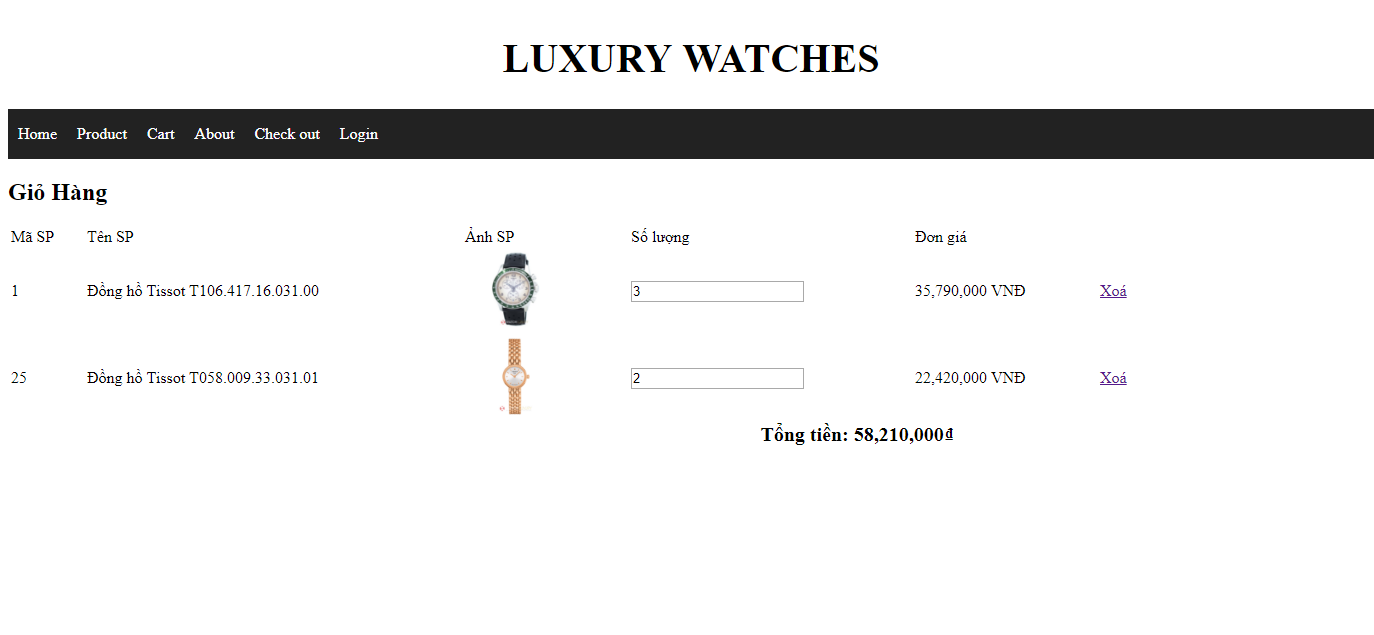
<script src="~/scripts/jquery-1.11.0.min.js"></script>

Phần view của giỏ hàng:

|  |
| --- |
| CSHTML |
| @model List<WebsiteBanDongHo.Models.ViewModel.CartItemViewModel>  @{  ViewBag.Title = "Index";  Layout = "~/Views/Shares/masterpage.cshtml";  }  <link href="~/Contents/cart.css" rel="stylesheet" />  <h2>Giỏ Hàng</h2>  <div class="content">  <div class="section group">  @if (Model.Count > 0)  {  <table class="table">  <thead>  <tr>  <td>Mã SP</td>  <td>Tên SP</td>  <td>Ảnh SP</td>  <td>Số lượng</td>  <td>Đơn giá</td>  <td></td>  </tr>  </thead>  <tbody>  @foreach (var item in Model)  {  <tr>  <td class="txtID" data-id="@item.product.MASP">@item.product.MASP</td>  <td>@item.product.TENSP</td>  <td><img src="~/Contents/images/HINHLON/@item.product.HINHLON" width="100" /></td>  <td><input class="txtQuantity" type="number" min="1" value="@item.Quantity" /></td>  <td class="txtPrice" data-price="@item.product.DONGIA">@String.Format("{0:0,0}", (item.product.DONGIA)) vnđ</td>  <td><a href="#" class="btn-delete">Xoá</a></td>  </tr>  }  </tbody>  </table>  <div class="totalCart">  <span class="txtVND">Tổng tiền: </span>  <span class="txtTotalPrice" data-checkout="">  0₫  </span>  </div>  }  else  {  <span>Chưa có sản phẩm nào trong giỏ hàng</span>  }  </div>  </div>  @\*Phần này để chạy jquery và ajax, các bạn có thể tách ra file riêng củng được (tự tìm hiểu nhé)\*@  <script>  $(document).ready(function () {  setTotalCart();  setPriceCart();  //mỗi khi số lượng item cart thay đổi thì nó chạy hàm này  $(".txtQuantity").change(function () {  var quantity = $(this).val();  var price = $(this).closest("tr").find(".txtPrice").attr("data-price");  var id = $(this).closest("tr").find(".txtID").attr("data-id");  //tính lại tiền khi thay đổi số lượng  var totalMoney = price \* quantity;  var format = addPeriod(totalMoney);  $(this).closest("tr").find(".txtPrice").html('<span class="cost">' + format + ' VNĐ' + '</span>');  setTotalCart();  //ajax gọi tới controller Cart và action Update  $.ajax({  url: "/Cart/Update", //đường dẫn ajax gọi tới  type: "POST",  data: {  //những tham số truyền qua controller để xử lý  id: id,  quantity: quantity  },  success: function (data) {  //data nhận được từ controller trả ra  if (data == "Ok") {  alert("Cập nhập thành công");  }  }  })  });  //sự kiện được gọi khi nút xóa được bấm  $(".btn-delete").click(function () {  var self = $(this);  var id = $(this).closest("tr").find(".txtID").attr("data-id");  //ajax gọi tới controller Cart và action Delete  $.ajax({  url: "/Cart/Delete", //đường dẫn ajax gọi tới  type: "POST",  data: {  //tham số truyền qua controller để xử lý  id: id  },  success: function (data) {  //data nhận được từ controller trả ra  if (data == "Ok") {  alert("Xóa thành công");  self.closest("tr").remove();  setTotalCart();  }  }  })  });  });  //hàm xử lý tính tổng tất cả tiền trong giỏ hàng  function setTotalCart() {  var totalcart = 0;  $(".txtPrice").each(function () {  var price = $(this).attr("data-price");  var quantity = $(this).closest("tr").find(".txtQuantity").val();  totalcart = totalcart + price \* quantity;  var temp = addPeriod(totalcart);  $(this).closest(".section").find(".txtTotalPrice").text(temp + '₫');  })  }  //hàm xử lý tính lại tổng tiền item cart  function setPriceCart() {  var totalMoney = 0;  $(".txtPrice").each(function () {  var price = $(this).attr("data-price");  var quantity = $(this).closest("tr").find(".txtQuantity").val();  totalMoney = price \* quantity;  var tempMoney = addPeriod(totalMoney);  $(this).closest("tr").find(".txtPrice").html('<span class="cost">' + tempMoney + ' VNĐ' + '</span>');  })  }  //hàm format tiền sang định dạng việt nam (ví dụ: 56800000 -> 56,800,000)  function addPeriod(nStr) {  nStr += '';  x = nStr.split('.');  x1 = x[0];  x2 = x.length > 1 ? '.' + x[1] : '';  var rgx = /(\d+)(\d{3})/;  while (rgx.test(x1)) {  x1 = x1.replace(rgx, '$1' + ',' + '$2');  }  return x1 + x2;  }  </script> |
| CSS |
| body {  }  .table {  width: 1140px;  max-width: 100%;  }  .totalCart {  float: right;  margin-right: 420px;  }  .txtVND, .txtTotalPrice{  font-weight: bold;  font-size: 20px;  } |

Cái lợi của Ajax là nó vẫn gọi tới Controller và Action theo url đã định mà không cần load lại trang web.

Kết quả:



* 1. Tổng kết:

Như vậy qua bài này chúng ta đã biết được làm thế nào để lưu đối tượng và biến session, củng như thao tác với Ajax để gọi tới Controller và Action tương ứng mà không cần load lại trang. Vì ở mức căn bản nên tôi không nhắc sâu về nó, thực ra để nói hết thì còn rất nhiều cái, qua quá trình làm và thực hiện đồ án của mình thì các bạn cố gắng tìm hiểu thêm nhé. Ngoài ra còn phần thanh toán các bạn thử bắt tay vào làm coi như bài tập của phần này.