


第八組 組員：資訊四甲 10527144 周謐嬰 資訊三乙 10620115 王俊儒

主頁面：

[購物網站](#) [首頁](#) [購物車](#) [註冊](#) [登入](#)




product1

商品一描述

售價：100

庫存：9

[購買](#) [詳細資料](#)




product2

一隻貓

售價：200

庫存：200

[購買](#) [詳細資料](#)



product3

紙箱的貓

售價：300

庫存：300

[購買](#) [詳細資料](#)

© 2020 - 進階網路資訊系統第八組

登入註冊頁面：

[購物網站](#) [首頁](#) [購物車](#)

登入.

使用本機帳戶進行登入。

電子郵件

密碼

☐ 記住我?

[登入](#)

[以新使用者身分註冊](#)
[忘記密碼?](#)

[購物網站](#) [首頁](#) [購物車](#)

註冊.

建立新帳戶。

電子郵件

密碼


確認密碼

[註冊](#)

© 2020 - 進階網路資訊系統第八組

一般使用者帳號顯示頁面：

[購物網站](#) [首頁](#) [購物車](#) 您好 a@a! [登出](#)




product1

商品一描述

售價：100

庫存：9

[購買](#) [詳細資料](#)




product2

一隻貓

售價：200

庫存：200

[購買](#) [詳細資料](#)



product3

紙箱的貓

售價：300

庫存：300




[購買](#) [詳細資料](#)

© 2020 - 進階網路資訊系統第八組

管理者帳號登入畫面：

[購物網站](#) [首頁](#) [新增商品](#) 您好 Admin@admin! [登出](#)

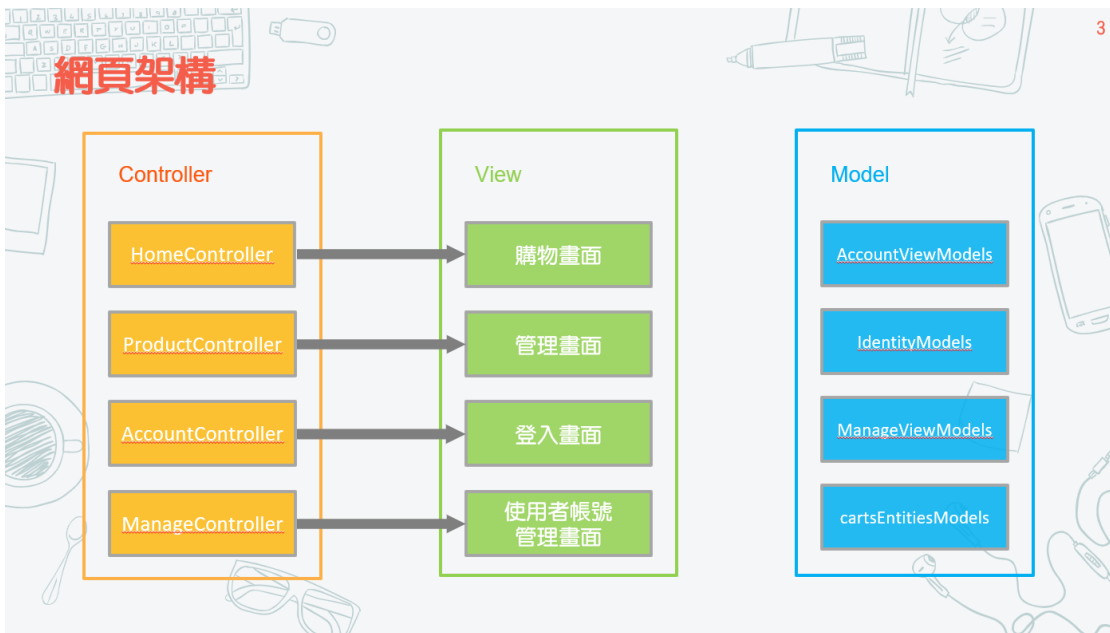
商品清單

序號	商品名稱	fimage	價錢	描述	數量	
1	product1		100	商品一描述	9	編輯 詳細資料 刪除
2	product2		200	一隻貓	200	編輯 詳細資料 刪除
3	product3		300	紙箱的貓	300	編輯 詳細資料 刪除

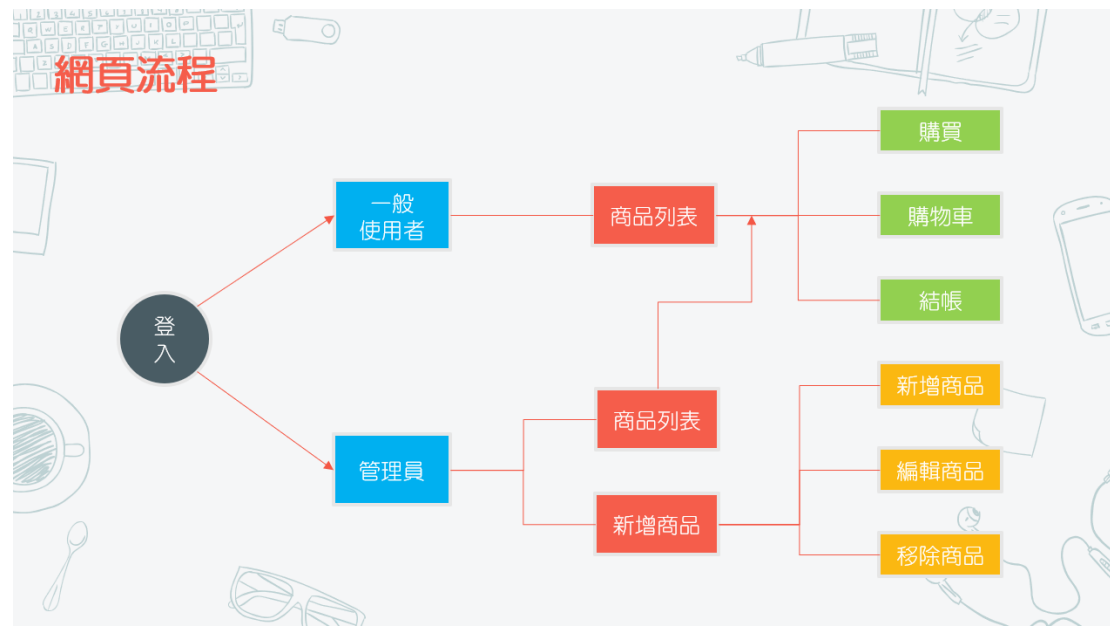
[新增商品](#)

© 2020 - 進階網路資訊系統第八組

網頁架構：



網頁流程



程式碼語法解釋：

```
[Required]
[Display(Name = "價錢")]
[Range(1, 9999, ErrorMessage = "請輸入大於1的數字")]
[RegularExpression(@"^[0-9]*$", ErrorMessage = "請輸入數字")]
3 個參考
public int fPrice { get; set; }
```

Required: 需填寫此欄位

Range: 表示資料範圍

RegularExpression: 輸入資料須符合此表示式

此驗證機制，需要在回傳整個表單的時候使用，如果只回傳其中一項資料驗證會出錯。

Login: 透過 PasswordSignInAsync 進行登入驗證，通過 RESULT 把登入結果後該執行的程式碼進行區分。

shouldLockout: 是否在登入失敗紀錄次數，預設為 true

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
0 個參考
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    if (!ModelState.IsValid)
    {
        return View(model);
    }

    // 這不會計算為帳戶鎖定的登入失敗
    // 若要啟用密碼失敗來觸發帳戶鎖定，請變更為 shouldLockout: true
    var result = await SignInManager.PasswordSignInAsync(model.Email, model.Password, model.RememberMe, shouldLockout: false);
    switch (result)
    {
        case SignInStatus.Success:
            if (model.Email == "Admin@admin")
                return RedirectToRoute(new { controller = "Product", action = "Index" });
            else
                return RedirectToLocal(returnUrl);
        case SignInStatus.LockedOut:
            return View("Lockout");
        case SignInStatus.RequiresVerification:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = model.RememberMe });
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "登入嘗試失誤。");
            return View(model);
    }
}
```

如果為管理員帳號導向可以新增商品的頁面。

如果不是管理員的畫，導向購物畫面。

Register: 用 ApplicationUser 新增一筆使用者資料

SignInAsync 加上使用者密碼加密後放入資料庫。

使用 Await 確保後續物件已經建立，且除錯完成才會被執行。

```
// POST: /Account/Register
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
0 個參考
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser { UserName = model.Email, Email = model.Email };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            await SignInManager.SignInAsync(user, isPersistent:false, rememberBrowser:false);
            if (model.Email == "Admin@admin")
            {
                return RedirectToRoute(new { controller = "Product", action = "Index" });
            }
            else
            {
                return RedirectToAction("Homepage", "Home");
            }
        }
        AddErrors(result);
    }

    // 如果執行到這裡，發生某項失敗，則重新顯示表單
    return View(model);
}
```

模型驗證:

[Required] :使此欄位為必填。

[Display(Name = "價錢")] :使此欄位的標題顯示為 name 的值。

[Range(1, 9999, ErrorMessage = "請輸入大於 1 的數字")]:使此欄位的數字範圍為 1-9999，如果不符合範圍，顯示錯誤訊息。

[RegularExpression(@"^[0-9]*\$")]:使此欄位必須符合正規表示式的東西。

```
public partial class tProducts
{
    [Display(Name = "序號")]
    9 個參考
    public int fId { get; set; }

    [Required]
    [Display(Name = "商品名稱")]
    4 個參考
    public string fName { get; set; }

    8 個參考
    public string fImage { get; set; }

    [Required]
    [Display(Name = "價錢")]
    [Range(1, 9999, ErrorMessage = "請輸入大於1的數字")]
    [RegularExpression(@"^[0-9]*$", ErrorMessage = "請輸入數字")]
    3 個參考
    public int fPrice { get; set; }

    [Required]
    [Display(Name = "描述")]
    3 個參考
    public string fDescription { get; set; }
```

View 的頁面的 Layout:

```
@{
    ViewBag.Title = "登入";
    Layout = "~/Views/Shared/_HomeLayoutPage.cshtml";
}
```

根據管理員與一般使用者設定不同的 layout。

新增商品：

圖片上傳之後，根據輸入的商品名稱，建立檔名，儲存在 Images。

```
[HttpPost]
[ValidateAntiForgeryToken]
0 個參考
public ActionResult Create(string fName, string fImage, int fPrice, string fDescription, int fQuantity, HttpPostedFileBase file)
{
    tProducts product = new tProducts();
    //product.fId = fId;
    product.fName = fName;
    product.fImage = fImage;
    product.fPrice = fPrice;
    product.fDescription = fDescription;
    product.fQuantity = fQuantity;

    if (file != null && file.ContentLength > 0)
    {
        string fileName = Request.Form["fName"] + Path.GetExtension(file.FileName);
        product.fImage = fileName;
        var path = Path.Combine(Server.MapPath("~/Images"), fileName);
        file.SaveAs(path);
    }

    db.tProducts.Add(product);
    db.SaveChanges();
    return RedirectToAction("Index");
}
```

編輯商品：

讓圖片的資料會隨著商品名稱變動改變。

```
public ActionResult Edit(int id, string fName, string fImage, int fPrice, string fDescription, int fQuantity, HttpPostedFileBase file)
{
    var product = db.tProducts.Where(m => m.fId == id).FirstOrDefault();
    string path = "";
    if (product.fImage != null)
        path = Path.Combine(Server.MapPath("~/Images"), product.fImage.ToString());

    product.fName = fName;
    product.fPrice = fPrice;
    product.fDescription = fDescription;
    product.fQuantity = fQuantity;

    /*如果有新的檔案上傳 就要把之前的刪掉再新增*/
    if (file != null && file.ContentLength > 0)
    {
        if (System.IO.File.Exists(path))
            System.IO.File.Delete(path);

        string fileName = fName + Path.GetExtension(file.FileName);
        product.fImage = fileName;
        path = Path.Combine(Server.MapPath("~/Images"), fileName);
        file.SaveAs(path);
    }

    db.SaveChanges();

    return RedirectToAction("Index");
}
```

購買頁面：

HomeController: Homepage

從 cartEntities 資料庫內的 tProducts 資料表取出所有資料，並轉換成 List，傳到 View 檢視中。

```
30 public ActionResult Homepage()  
31 {  
32     using (Models.cartEntities db = new Models.cartEntities())  
33     {  
34         var result = (from pd in db.tProducts select pd).ToList();  
35         return View(result);  
36     }  
37 }  
38
```

(圖一)Homepage 的 Action

Homepage View:

使用資料模型將資料庫之資料傳到前端。

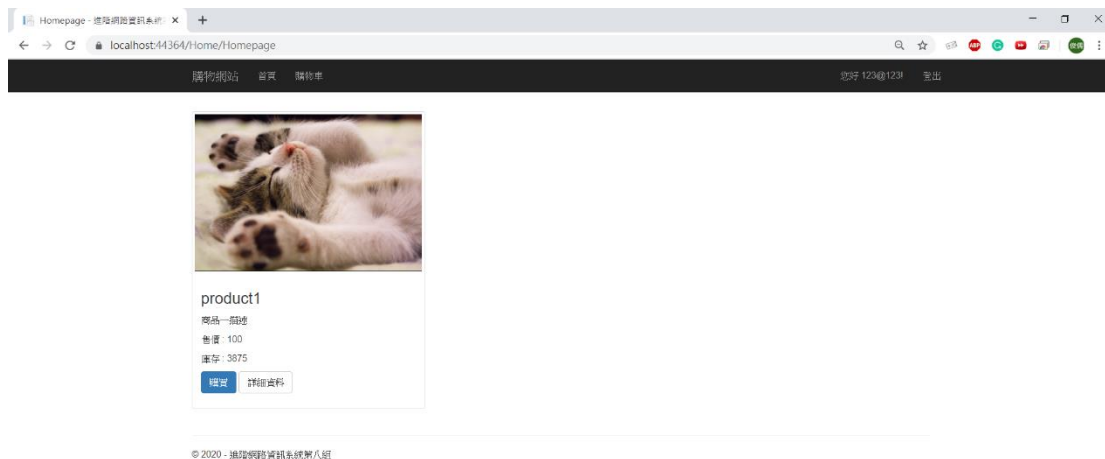
```
1 @model IEnumerable<finalProject_V1.Models.tProducts>  
2  
3 @{  
4     ViewBag.Title = "Homepage";  
5 }  
6
```

(圖二) Homepage.cshtml(View 檢視)使用資料模型

使用 Razor 語法，將模型內資料用 foreach 顯示在網頁內，而每個商品都有自己的購買按鈕，當按下購買時，會連結到 HomeController 內的 AddintoCart 這個 Action，並且傳遞商品 id 過去。

```
11 <div class="row">  
12     @foreach (var product in Model)  
13     {  
14         <div class="col-sm-6 col-md-4">  
15             <div class="thumbnail">  
16                   
17                 <div class="caption">  
18                     <h3>@product.fName</h3>  
19                     <p>@product.fDescription</p>  
20                     <p>售價 : @product.fPrice</p>  
21                     <p>庫存 : @product.fQuantity</p>  
22                 </div>  
23                 @if (Request.IsAuthenticated)  
24                 {  
25                     <a href="@Url.Action("AddintoCart", "Home", new RouteValueDictionary(new { id = product.fId })))" class="btn btn-primary">購買</a>  
26                 }  
27                 else  
28                 {  
29                     <a href="@Url.Action("Login", "Account", routeValues: null)" class="btn btn-primary" role="button">購買</a>  
30                 }  
31                 <a href="@Url.Action("Details", "Product", new RouteValueDictionary(new { id = product.fId })))" class="btn btn-default">詳情</a>  
32             </div>  
33         </div>  
34     }  
35 </div>  
36 </div>  
37 </div>  
38
```

(圖三) 使用 Razor 顯示所有商品資料



(圖四) 網頁呈現

HomeController: AddintoCart

當 AddintoCart 這個 Action 接收到 id 這個參數(我們按下購買的那樣商品的 id)時，會利用此 id 到商品清單的資料表(tProduct)內，利用 id 的唯一性，將我們所購買的商品資料取出，回傳到 View 檢視。

```

56 public ActionResult AddintoCart(int id)
57 {
58     var product = db.tProducts.Where(m => m.fId == id).FirstOrDefault();
59     return View(product);
60 }

```

(圖五) AddintoCart Action

AddintoCart View:

我們一樣在 View 檢視中使用資料表模型，並用 Razor 語法將商品資訊重新顯示在網頁中，讓購買人可以重新確認商品的資訊。

```

1 @model finalProject_V1.Models.tProducts
2 @{
3     ViewBag.Title = "AddintoCart";
4     Layout = "~/Views/Shared/_HomeLayoutPage.cshtml";
5 }
6
7 <h2>Buy</h2>
8 <hr />
9 @Html.ValidationSummary(true, "", new { @class = "text-danger" })
10 <div class="col-md-6">
11     <div class="thumbnail">
12         
13         <div class="caption">
14             <h3>@Model.fName</h3>
15             <p>@Model.fDescription</p>
16             <p>售價 : @Model.fPrice</p>
17             <p>庫存 : @Model.fQuantity</p>
18         </div>
19     </div>
20 </div>
21 </div>

```

(圖六) AddintoCart.cshtml 顯示商品資訊

(圖七) AddintoCart 網頁呈現

而在此頁面的右邊會有讓購買人填寫想購滿的數量的欄位，而這個欄位添加了 required 屬性，所以如果購買人忘記填寫，就會跳出 Message 提醒購買人。

```

23 using (Html.BeginForm(Html.BeginForm("AddintoCart", "Home", FormMethod.Post, new { enctype = "multipart/form-data", id = Model.fId })))
24 {
25     @Html.AntiForgeryToken()
26     <div class="form-group">
27         <h4>數量</h4>
28         <div>
29             <input type="text" name="fQuantity" class="form-control" required="required" />
30         </div>
31     </div>
32     <div class="form-group">
33         <div class="col-md-offset-2">
34             <input type="submit" value="加入購物車" class="btn btn-default" />
35         </div>
36     </div>
37 </div>
38
39
40
41 <div>
42     <a href="@Url.Action("Homepage", "Home")" class="btn btn-primary" role="button">返回</a>
43 </div>

```

(圖八) AddintoCart.cshtml 填寫商品數量表單

數量

加入購物車

返回

數量

加入購物車

返回

! 請填寫這個欄位。

© 2020 - 進階網路資訊系統第八組


(圖九) 填寫商品數量頁面

(圖十) 資料未填產生 message 提醒購買者

購物網站 首頁 購物車

您有 123 件商品 空出

Buy



product1
商品一瓶
售價: 100
庫存: 3875

數量

加入購物車

返回

© 2020 - 進階網路資訊系統第八組

(圖十一) 網頁內容呈現

當我們成功填寫欄位按下加入購物車時，會有另一個 AddintoCart 接收我們所填寫的購買數量，並將商品資料和購買數量加入 tCart 資料表(購物車)，並重新導向至 ShortCarts Action。

```
62 [HttpPost]
63 0 個參考
64 public ActionResult AddintoCart(int id, string fQuantity)
65 {
66     tCart cart = new tCart();
67     var product = db.tProducts.Where(m => m.fId == id).FirstOrDefault();
68     cart.fUserId = User.Identity.GetUserId();
69     cart.fPdImage = product.fImage;
70     cart.fPdName = product.fName;
71     cart.fPdDescription = product.fDescription;
72     cart.fPdPrice = product.fPrice * Convert.ToInt32(fQuantity);
73     cart.fBuyQuantity = Convert.ToInt32(fQuantity);
74     product.fQuantity -= Convert.ToInt32(fQuantity);
75     db.tCart.Add(cart);
76     db.SaveChanges();
77     return RedirectToAction("ShowCarts");
78 }
```

(圖十二) AddintoCart Action

HomeController: ShowCarts

ShowCarts 這個 Action 會從 tCarts 資料表內取出所有購物車內容，並轉換成 List 傳到 View 檢視中呈現。

```
93 public ActionResult ShowCarts()
94 {
95     var cart = db.tCart.OrderBy(m => m.fId).ToList();
96     return View(cart);
97 }
```

(圖十三) ShortCarts Action

ShowCarts View

這個 View 檢視中，套用 List 範本，並套用使用者的 Layout(為了區別使用者與管理員)

```
1 @model IEnumerable<finalProject_V1.Models.tCart>
2 @using Microsoft.AspNet.Identity
3 @f
4 ViewBag.Title = "ShowCarts";
5 Layout = "~/Views/Shared/_HomeLayoutPage.cshtml";
6 }
```

(圖十三) ShowCarts View

並且使用 Razor 語法的 foreach 將傳進來的資料內容呈現在網頁中，這裡使用到了 Microsoft.AspNet.Identity 模組內 User.Identity.GetUserId() 方式，取地目前登入的用戶的 UserID，這個 UserID 其實是隨機生成的亂碼，所以可以利用 UserID 的唯一性，顯示出這個 User 的購物車內容，不用擔心目前用戶改到

別的用戶的購物車內容。

```
8 <h2>ShowCarts</h2>
9 <table class="table">
10 <tr>
11 <th>商品名稱 </th><th>商品圖片</th><th>商品描述 </th><th>數量</th><th>金額</th><th></th></tr>
12 </th></tr>
13
14 <foreach (var item in Model)>
15 {
16 if (User.Identity.GetUserId() == item.fUserId)
17 {
18 <tr>
19 <td>
20 <@Html.DisplayFor(modelItem => item.fPdName)>
21 </td>
22 <td>
23 
24 </td>
25 <td>
26 <@Html.DisplayFor(modelItem => item.fPdDescription)>
27 </td>
28 <td>
29 <@Html.DisplayFor(modelItem => item.fBuyQuantity)>
30 </td>
31 <td>
32 <@Html.DisplayFor(modelItem => item.fPdPrice)>
33 </td>
34 <td>
35 <a href="@Url.Action("DeleteCartItem", "Home", new RouteValueDictionary(new { id = item.fId })))" class="btn btn-danger" role="button">
36 </td>
37 </tr>
38 }
39 }
40 }
```

(圖十四) ShowCarts View 使用 Razor 顯示網頁內容

並且使用 Razor 將所有商品價錢總合，形成總金額呈現在網頁中，讓購買人知道如果要結帳需要多少錢。

```
41
42 var totalAmount = 0;
43 foreach (var item in Model)
44 {
45 if (User.Identity.GetUserId() == item.fUserId)
46 {
47 totalAmount += item.fPdPrice;
48 }
49 }
50 }
```

(圖十五) ShowCarts View 使用 Razor 計算購物車總金額

在網頁最下方會讓購買人填寫收件人姓名、電話、地址、付款方式，這些欄位也有加入屬性 required，所以當漏填時也會跳出 message 提醒使用者。

```
using (Html.BeginForm(Html.BeginForm("PayMoney", "Home", FormMethod.Post, new { enctype = "multipart/form-data" })))
{
    <@Html.AntiForgeryToken()>
    <div class="form-group">
    <div>收件人姓名</div>
    <input type="text" name="Name" class="form-control" required="required" />
    </div>
    <div class="form-group">
    <div>收件人電話</div>
    <input type="text" name="Phone" class="form-control" required="required" />
    </div>
    <div class="form-group">
    <div>收件人地址</div>
    <input type="text" name="Address" class="form-control" required="required" />
    </div>
    <div class="form-group">
    <div>付款方式</div>
    <select name="Paymety" class="form-control" required="required">
    <option value="取貨付款">取貨付款</option>
    </select>
    </div>
}
```

(圖十六) 表單皆有 required 屬性

收件人姓名
小名

收件人電話
0912345678

收件人地址
|

付款方式
請填寫這個欄位。
取貨付款

繼續選購 結帳

(圖十七) 若沒填就會顯示 message

底下還有結帳按鈕和繼續選購按鈕，當我們按下繼續選購，此時會連結到 Homepage 頁面，當我們按下結帳時，會將購買人填寫的收件資訊和購物車的商品內容一併傳送到 HomeController 的 PayMoney Action。

```

107 <div class="form-group">
108 <div class="col-md-offset-2">
109 <a href="@Url.Action("Homepage", "Home")" class="btn btn-primary" role="button">繼續選購</a>
110 <input type="submit" value="結帳" class="btn btn-danger" />
111 </div>
112 </div>
113
114

```

(圖十八) 繼續選購及結帳按鈕

The screenshot shows a web browser displaying the 'ShowCarts' application. At the top, there's a navigation bar with '購物網站', '首頁', and '購物車'. The main content area has a title 'ShowCarts' and a table representing the shopping cart. The table has columns for '商品名稱', '商品圖片', '商品描述', '數量', and '金額'. It contains one item: 'product1' with a quantity of 100 and an amount of 10000. Below the table, there's a '結帳' (Checkout) button. Underneath the table, there's a form for checkout details, including fields for '收件人姓名' (Receiver Name), '收件人電話' (Receiver Phone), '收件人地址' (Receiver Address), and a '付款方式' (Payment Method) dropdown menu. The form also has '繼續選購' (Continue Shopping) and '結帳' (Checkout) buttons. At the bottom, there's a copyright notice: '© 2020 - 遠東網際網路系統第八期'.

(圖十九) 網頁呈現

HomeController: PayMoney

PayMoney 這個 Action 接收購物車網頁傳來的表單內容，並將商品資訊、UserID、收件資訊加到 tOrder 資料表(訂單)內，並重新導向到 BuyList Action。

```

99 [HttpPost]
100 0 個參考
101 public ActionResult PayMoney(string Name, string Phone, string Address, string Payway)
102 {
103     tOrder order = new tOrder();
104     var carts = db.tCart.ToList();
105     foreach (var item in carts)
106     {
107         if (User.Identity.GetUserId() == item.fUserId)
108         {
109             order.fBuyQuantity = item.fBuyQuantity;
110             order.fPdDescription = item.fPdDescription;
111             order.fPdImage = item.fPdImage;
112             order.fPdName = item.fPdName;
113             order.fPdPrice = item.fPdPrice;
114             order.fUserId = User.Identity.GetUserId();
115             order.fUserName = Name;
116             order.fUserPhone = Phone;
117             order.fUserAddress = Address;
118             order.fUserPayway = Payway;
119             db.tOrder.Add(order);
120             db.tCart.Remove(item);
121             db.SaveChanges();
122         }
123     }
124     return RedirectToAction("BuyList");
125 }
126

```

(圖二十) PayMoney Action

HomeController: BuyList

BuyList 這個 Action 會將 tOrder 這個資料表的所有內容轉換成 List 並 return 到 View 檢視中。

```
128 public ActionResult BuyList()  
129 {  
130     var order = db.tOrder.OrderBy(m => m.fId).ToList();  
131     return View(order);  
132 }
```

(圖二十一) BuyList Action

BuyList View

這個檢視中會使用 tOrder 資料模型，並套用使用者的 Layout。

```
1 @model IEnumerable<finalProject_V1.Models.tOrder>  
2 using Microsoft.AspNet.Identity  
3  
4 @{  
5     ViewBag.Title = "BuyList";  
6     Layout = "~/Views/Shared/_HomeLayoutPage.cshtml";  
7 }  
8
```

(圖二十二) BuyList View 使用模型

然後透過 Razor 的 foreach 將 tOrder 資料表(訂單資訊)內容，經 Microsoft.AspNet.Identity 的 User.Identity.GetUserId()，將目前登入的使用者的訂單全部呈現在網頁中。

```
51 @foreach (var item in Model)  
52 {  
53     if (User.Identity.GetUserId() == item.fUserId)  
54     {  
55         <tr>  
56             <td>  
57                 @Html.DisplayFor(modelItem => item.fPdName)  
58             </td>  
59             <td>  
60                   
61             </td>  
62             <td>  
63                 @Html.DisplayFor(modelItem => item.fPdDescription)  
64             </td>  
65             <td>  
66                 @Html.DisplayFor(modelItem => item.fBuyQuantity)  
67             </td>  
68             <td>  
69                 @Html.DisplayFor(modelItem => item.fPdPrice)  
70             </td>  
71             <td></td>  
72         </tr>  
73     }  
74 }  
75
```

(圖二十三) BuyList View 使用 Razor 顯示所有訂單

BuyList - 進階網路資訊系統第八組








localhost:44364/Home/BuyList

購物網站 首頁 購物車

您好 123 退出

BuyList

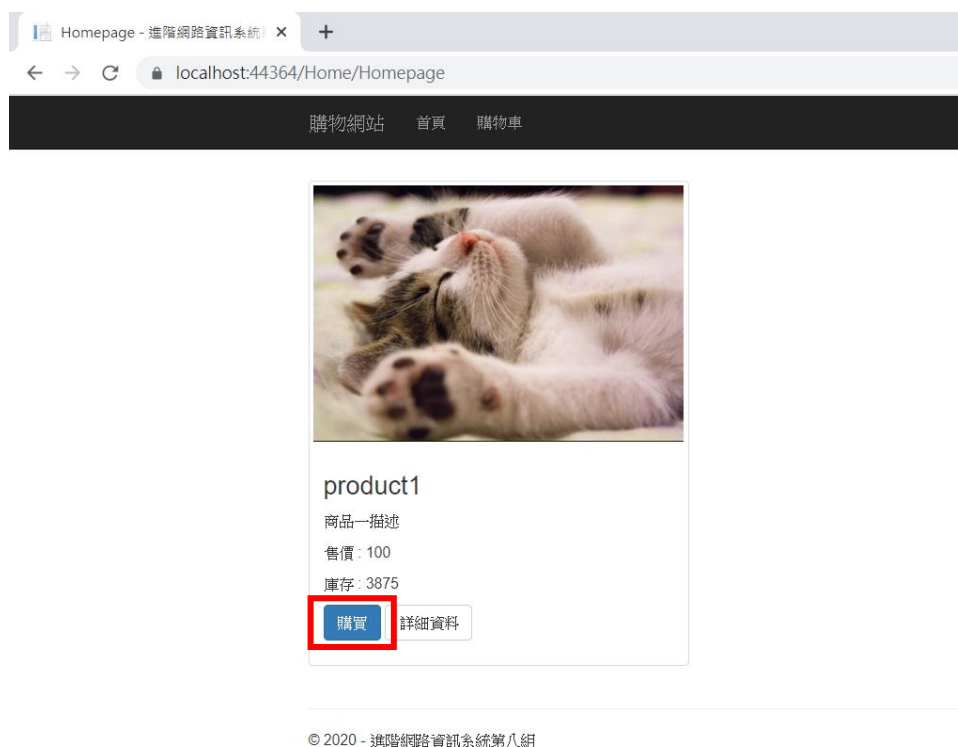
新增

fPdName	fPdImage	fPdDescription	fBuyQuantity	fPdPrice	
product1		商品一描述	2000	200000	
product1		商品一描述	2000	200000	
product1		商品一描述	123	12300	
product1		商品一描述	2000	200000	
product1		商品一描述	1	100	
product1		商品一描述	10	1000	
product1		商品一描述	100	10000	
		小名	123	中版大場	取貨付款

© 2020 - 進階網路資訊系統第八組

(圖二十四) BuyList 網頁呈現

購買功能網頁展示：



(圖二十五)按下購買按鈕

Buy



product1

商品一描述

售價 : 100

庫存 : 3765

數量

1000

加入購物車

返回

© 2020 - 進階網路資訊系統第八組


(圖二十六)填寫購買數量按下加入購物車

資訊系統 x +

localhost:44364/Home/ShowCarts

購物網站 首頁 購物車 您好 123@123! 登出

ShowCarts

商品名稱	商品圖片	商品描述	數量	金額	
product1		商品一描述	1000	100000	刪除
總金額				100000	

收件人姓名

小名

收件人電話

123

收件人地址

中原大學

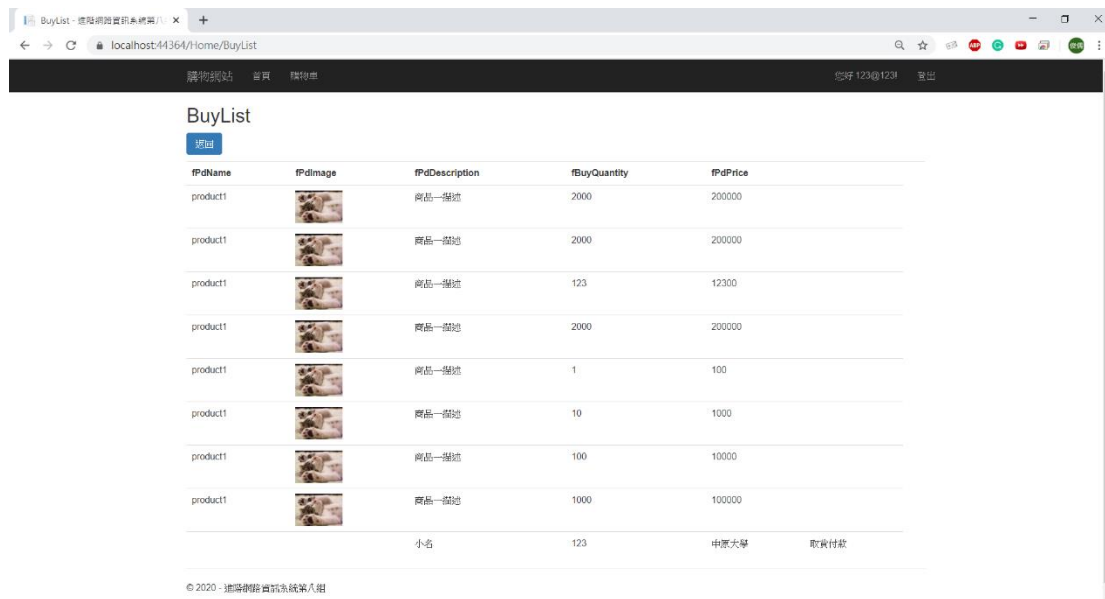
付款方式

取貨付款

繼續選購 結帳

© 2020 - 進階網路資訊系統第八組

(圖二十七)填寫收件資料按下結帳



(圖二十八)檢查購買清單內容，完成購買