HoleShot Motor Sports Web site

By

Andy C. Jen

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Engineering Technology

University of Cincinnati
College of Applied Science

June 2008
HoleShot Motor Sports Web-site

By

Andy C. Jen

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements
for
the Degree of Bachelor of Science
in Information Engineering Technology

© Copyright 2008 Andy C. Jen

The author grants to the Information Engineering Technology Program permission
to reproduce and distribute copies of this document in whole or in part.

___________________________________________________ __________________
Andy C. Jen                   Date

___________________________________________________ __________________
Robert Shlemmer, Faculty Advisor     Date

___________________________________________________ __________________
Hazem Said, Ph.D. Department Head     Date
Acknowledgements

I would like to give special thanks to the Faculty Advisor Professor Robert Schlemmer for his help on giving guidance and knowledge for me to complete this project. I would also like to give special thanks to Dr. Hazem Said, for his encouragement to complete my Bachelor of Science degree at College of Applied Science, University of Cincinnati.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>i</td>
</tr>
<tr>
<td>Table of Contents</td>
<td>ii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>iii</td>
</tr>
<tr>
<td>Abstract</td>
<td>iv</td>
</tr>
<tr>
<td>1. Statement of the Problem</td>
<td>1</td>
</tr>
<tr>
<td>2. Description of the Solution</td>
<td>2</td>
</tr>
<tr>
<td>2.1 User Profile</td>
<td>3</td>
</tr>
<tr>
<td>2.2 Design Protocols</td>
<td>3</td>
</tr>
<tr>
<td>2.2.1 Technology</td>
<td>3</td>
</tr>
<tr>
<td>2.2.2 Database Design</td>
<td>4</td>
</tr>
<tr>
<td>3. Deliverables</td>
<td>6</td>
</tr>
<tr>
<td>4. Design and Development</td>
<td>7</td>
</tr>
<tr>
<td>4.1 Timeline</td>
<td>7</td>
</tr>
<tr>
<td>4.2 Budget</td>
<td>9</td>
</tr>
<tr>
<td>4.3 Hardware and Software</td>
<td>10</td>
</tr>
<tr>
<td>5. Proof of Design</td>
<td>10</td>
</tr>
<tr>
<td>5.1 Updated Home Page</td>
<td>10</td>
</tr>
<tr>
<td>5.2 Store Page with different products</td>
<td>12</td>
</tr>
<tr>
<td>5.3 Shopping Cart page</td>
<td>15</td>
</tr>
<tr>
<td>5.4 User Logon Page</td>
<td>16</td>
</tr>
<tr>
<td>5.5 Submit New User and Create a new user</td>
<td>17</td>
</tr>
<tr>
<td>5.6 ViewItem Page</td>
<td>18</td>
</tr>
<tr>
<td>5.7 Confirmation E-mail</td>
<td>19</td>
</tr>
<tr>
<td>6. Testing Plans</td>
<td>20</td>
</tr>
<tr>
<td>7. Conclusions and Recommendations</td>
<td>20</td>
</tr>
<tr>
<td>7.1 Conclusions</td>
<td>21</td>
</tr>
<tr>
<td>7.2 Recommendations</td>
<td>21</td>
</tr>
<tr>
<td>Appendix A</td>
<td>22</td>
</tr>
<tr>
<td>Appendix B</td>
<td>23</td>
</tr>
<tr>
<td>Bibliography</td>
<td>33</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1: Current Web site at www.holeshotms.com 2
Figure 2: Database Relationship Diagram 6
Figure 3: Original Project Timeline 7
Figure 4: Updated Timeline for Senior Design III - Spring 2008 9
Figure 5: Updated Home Page 11
Figure 6: Contact Us Page 12
Figure 7: Store Page with All Products 13
Figure 8: Store Page with Leather Jackets 14
Figure 9: Store Page with Gloves 15
Figure 10: Shopping Cart Page 16
Figure 11: User Login Page 17
Figure 12: New User Register Form 18
Figure 13: ViewItem Page with Grand Total 19
Figure 14: Confirmation E-mail Notification 20

List of Tables

Table 1: Budget Table 9
Abstract

HoleShot Motor Sports Web is a Web site that utilizes advanced technologies, such as Microsoft C#.NET, Microsoft ASP.NET, and Microsoft Access. This Web site was developed with Visual Studio .Net as the Integrated Development Environment. It is designed to improve the business process currently deployed by a local motorcycle parts shop located in Wilder, Kentucky, USA. The owner of this local motorcycle shop located in Kentucky had a need to facilitate the business process by using a Web site to make the business process easier by eliminating the need for buyers to be in the store physically.
Holeshot Motor Sports Web site

1. Statement of the Problem

Holeshot Motor Sports is a motorcycle parts shop located in Kentucky, south of Cincinnati, Ohio. In order to promote popularity and accessibility, the shop owner, Mr. Dale Cruze, wants to have a customer-oriented eCommerce Web site. After interviewing Mr. Cruze, the need he presented was to create a database-driven eCommerce Web site that is to conduct online transactions and also to develop a Web site similar in look and feel to the following Web sites: such as Chaparral Racing and Iron Pony (2 and 5). The existing Web site for the shop (see figure 1) does not meet the demands coming from the daily users because it does not have the capability to interact with the back-end database engine if customers want to purchase online. Also, it lacks a user-friendly menu for easy navigation for customers. The Web site currently serves the purpose of informing local customers or friends of past events and upcoming events. The Web site displays photos from past activities and the address and phone number of the store. Users will not come back to the same Web site if they cannot find what they need the most, and they will connect to other Web sites and look for what they want. Thus, this proposed Web site will organize information in such a way that allows users to find pertinent information easily. The menu items are gloves, helmets, leather jackets, and protective wears. The items are similar to those displayed on the current Web site for this motorcycle parts shop.
Figure 1: Current Web site at www.holeshotms.com

2. Description of the Solution

The use of this eCommerce Web application is to allow customers to purchase motorcycle parts and apparel from Holeshot motor sports store. Use of this application can eliminate the need for customers to purchase items in the store physically. This application is technology-driven, meaning that the heart of this application uses a three-tier structure. The three-tier structure is the following: (1) several front-end graphical user interfaces; (2) ASP.Net technology works in the middle tier; and (3) a back-end database engine behind the application.
With this eCommerce Web application, all customers are able to view descriptions for apparel, place orders, and then view a report on total payment in detail.

2.1 User Profile

Three groups of users who will interact with this application. They are administrator, customer, and employee. Each of them is described as the following:

- **Administrator**: Users in this group will have full rights to modify the database and generate reports as necessary.

- **Customer**: Users in this group will be given the least rights for working with this application. They are able to place orders with the user interface pages, view the order detail, and print the order detail.

- **Employee**: Users in this group will have more rights than those granted to customers, but fewer than those of the administrator. They can view the report and place orders when working with customers on the phone.

2.2 Design Protocols

The following paragraph describes the design protocol utilized to complete this senior project.

2.2.1 Technology

The main technology and tool for developing this senior project are the Microsoft products and technologies. Microsoft .Net Framework and C#.Net are the two technologies chosen for developing this senior project, for they are widely used and well supported. Also, both of the technologies are easily integrated with other Microsoft-based applications, such as Microsoft XP operating system and Microsoft Access 2003. This Web site is developed in the Microsoft .NET Framework environment. The .Net Framework allows different programming
languages and their libraries to work together seamlessly to manage the processes with other networked systems (4).

The .NET Framework consists of:

- Common Language Runtime (CLR) – it provides an environment where regardless of the programming languages it will manage the execution constantly
- Framework Class Library (FCL) – it provides a prepackaged object-oriented library to be used

The advantages of using the .Net Framework are as follows:

- Supports different network protocols
- Supports different programming languages
- Supports different libraries developed by different programming languages
- Supports the application to be run on different platforms

Therefore, using .Net Framework would also allow use of the ASP.net because ASP.net is part of the .Net Framework. ASP.net is necessary for building powerful web application in that it provides web controls and web forms. In order to build a more robust web application, C#.net programming language is chosen to interact with the database backend, and C#.net is part of Microsoft Visual Studio.Net software.

2.2.2 Database Design

Database is very important for a Web application to be successful. As Dr. Thuraisingham states in her book, *Web Data Management and Electronic Commerce*, "database systems play a key role in web data management. Having good data is key to effective web data management” (6, p. 21).
Dr. Thuraisinngham also states that "designing a database is a complex process" (6, p. 25). She continues to suggest following three steps for designing a good database. The first step is to "capture the entities of the application and the relationships between the entities" to utilize the concept of object-oriented approach for this” (6, p. 26). The second step is to find the relationships among these entities” (6, p. 26). Thirdly, to achieve a good database design, one needs to design a database to capture relationships, called the normalization process (6, p. 27). Therefore, the process of designing databases in general is to follow (1) conceptual, (2) logical, and (3) physical steps.

In the beginning of developing this Web project, the Microsoft SQL Server was selected for handling database-related problems because it combines both server and client advantages. Server applications provide advantages such as "security, fault tolerance, performance, concurrency, and reliable backups, whereas client application provide user interface and can contain empty reports"(7, p. 18). However, later in the process, Microsoft Access is used to replace the Microsoft SQL Server because the owner of the shop is very knowledgeable on how to use the Access database tables. There are five database tables. Figure 2 demonstrates the Database Relationship Diagram.
3. Deliverables

The following items are deliverables for this project:

- Create a consistent and dynamic navigational menu.
- Create a database relational diagram.
- Utilize MS C#.NET programming code and MS ADO.Net to interact with MS Access Database.
- Create user-friendly interfaces written by using ASP.NET.
- Implement secure user authentication.
- Create a new account for a new user.
- Implement an on-line shopping cart that shows total prices on customer’s order.
4. Design and Development

4.1 Timeline

Steps listed in the book, Developing eCommerce Systems, were followed to create this senior project: (1) Problem Identification, (2) Requirements Analysis, (3) Design, (4) Construction, (5) Testing, and (6) Implementation (1, pp. 23-24).

Originally, the first prototype was planned to be completed by June 27th, 2005. The processes to complete the first prototype consisted of the following stages: (1) Research, (2) Write proposal, (3) Finish proposal, (4) Present proposal, (5) Database development/test, (6) GUI development/test, (7) Design prototype presentation, (8) Finish and testing programming phase, (9) Testing and implementing the final product phase 1, (10) Testing final product phase 2, and lastly (11) The final presentation. Listing each milestone makes this senior project more realistic to finish.

Figure 3: Original Project Timeline
Senior Design I – Winter 2005

• Researched feasibility of potential projects
• Researched and explored important Web based technologies

Senior Design II – Summer 2005

• Database Design Completion
• User Interface Design Completion
• Made documentation
• Demonstrated prototype successfully
• Performed preliminary testing on functionalities
• Created design freeze documentation

Senior Design III – Winter 2006

• To complete all project functionalities
• To complete testing and troubleshooting, including Alpha & Beta testing.
• To submit final product and final product description and usage.

For senior design III class in winter 2006, the development of this project did not go well.

However, after applying new skills, this senior project is completed in spring of 2008. The following section is the updated timeline and the Microsoft project file.

Senior Design III – Spring 2008

• Completed all project functionalities
• Completed testing and troubleshooting, including Alpha & Beta testing.
• Submitted final report.
<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Duration</th>
<th>Start</th>
<th>Finish</th>
<th>Predecessor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Discussion with advisor</td>
<td>5 days</td>
<td>Thu 1/24/08</td>
<td>Wed 1/30/08</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Design and Complete physical</td>
<td>12 days</td>
<td>Mon 1/28/08</td>
<td>Tue 2/12/08</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Forms Implementation &amp; Shaw</td>
<td>4 days</td>
<td>Tue 2/19/08</td>
<td>Fri 2/22/08</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Coding</td>
<td>35 days?</td>
<td>Fri 2/22/08</td>
<td>Thu 4/10/08</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Bug Corrections</td>
<td>32 days?</td>
<td>Fri 4/11/08</td>
<td>Fri 5/23/08</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>Final Testing</td>
<td>4 days?</td>
<td>Mon 5/26/08</td>
<td>Thu 5/29/08</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>End of Project</td>
<td>1 day?</td>
<td>Thu 5/29/08</td>
<td>Thu 5/29/08</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 4: Updated Timeline for Senior Design III - Spring 2008**

### 4.2 Budget

The following table shows the total price for developing this Web application in the real-world environment. Real-world is regarded as the business environment.

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Development Budget</strong></td>
<td></td>
</tr>
<tr>
<td>Microsoft Office Professional 2003</td>
<td>$ 499</td>
</tr>
<tr>
<td>Macromedia Studio MX 2004</td>
<td>$ 899</td>
</tr>
<tr>
<td><strong>Development Total</strong></td>
<td><strong>$ 2,197</strong></td>
</tr>
<tr>
<td><strong>Production Budget</strong></td>
<td></td>
</tr>
<tr>
<td>ISP Hosting Service (include a Web Server)</td>
<td>$ 100 (yearly)</td>
</tr>
<tr>
<td><strong>Production Total</strong></td>
<td><strong>$ 100</strong></td>
</tr>
<tr>
<td><strong>Development Total</strong></td>
<td><strong>$ 2,197</strong></td>
</tr>
<tr>
<td><strong>Overall Total</strong></td>
<td><strong>$ 2,297</strong></td>
</tr>
</tbody>
</table>

**Table 1: Budget Table**

The actual total development cost for this Web application is free because all of the software is provided by IT Laboratory at the College of Applied Science, University of Cincinnati. Therefore, the actual overall total is $100, which is the yearly fee for hosting on a server.
4.3 Hardware and Software

Software requirements

- MS Access 2003
- MS Visual Studio.Net 2003
- Macromedia Firework MX 2004

Hardware requirements

- A web server provided by an Internet Service Provider (ISP)
- Client computer with Internet access service

5. Proof of Design

5.1 Updated Home Page

Figure 5 shows the updated home page for Holeshot Motor Sports Web site. In the middle section of this Web page, customers can easily get the contact information of the shop. There are two buttons: (1) Shop and (2) Contact Us on the home page.
Figure 5: Updated Home Page

After clicking on the Shop button, the user will be directed to the store Web page where s/he can see the product catalog. If the user clicks on Contact Us button, s/he will be directed to the Contact Us Web page. On the Contact Us Web page, there is a map that gives direction for how to get the shop.
Figure 6: Contact Us Page

5.2 Store Page with Different Products
The Store.aspx Web page initially shows all products. Figure 7 shows the page.

![Store Page with All Products](image)

**Figure 7: Store Page with All Products**

In addition, there is a menu dropdown list on top of this page. The content of the store page will be changed based on the user’s selection. The menu has the following menu items:

- **Dropdown List Menu Items** -
  - ALL
  - Protective Wears
  - Leather Jackets
  - Gloves
  - Z1R Helmets
  - Thor Helmets
Figure 8 and Figure 9 each display the store page with Leather Jackets and Gloves, respectively.

Figure 8: Store Page with Leather Jackets
5.3 Shopping Cart page

The following figure shows the shopping cart web-page. The user can click on the Edit link button to the left of each item, and s/he is able to modify the quantity for that item. Or, s/he could click on delete link to remove the item from shopping cart Web page. In addition, on the bottom right hand corner, there is a line total that shows the total price for the items the customer has selected so far.
5.4 User Logon Page

The following figure shows the User Logon Web page. If a returning customer provides wrong credentials, then the error message in red displays and then alerting the customer to enter the correct information again. Also, if a new customer wants to register to this Web site, s/he needs to click on the new customer button. Then, the new customer will be redirected to the new user registration form Web page.
5.5 Submit New User and Create a New User

The following figure shows the new user register Web page. The red dot indicates that the new user needs to provide all the information necessary for the new user to be registered to the site successfully. This Web page also requires the new customer to provide a credit card number and a credit card type. In order to make sure the new customer enters the 16 digits credit card number, a simple validation method for the credit card number was developed by using the maximum length property for the textbox. If the customer enters fewer than 16 digits, an error message shows and s/he is not registered to the Web site successfully. There are some algorithms to ensure the credit card number provided is correct.
5.6 ViewItem Page

The following figure shows the ViewItem Web page upon a successful logged in. The information shows ItemID, ItemName, quantity for the item the user wants to buy, and the item total. In addition to that, if the user decides to remove an item at the very last minute, s/he can do so by clicking on the delete link button on the left hand side. On the bottom of the page, the customer is able to see the sub total, taxes, shipping cost, and the grand total effectively.

Figure 12: New User Register Form
5.7 Confirmation E-mail

Once the customer clicks the buy-it-now button, s/he will get a confirmation e-mail notification. Figure 14 shows the content of the e-mail.
6. Testing Plan

There are two testing phases for testing this Web application. First, it is the Alpha testing plan, in which the author and some friends will pretend to be the users of this application and use it. Secondly, after passing the Alpha test stage, then the clients will have a chance to view the Web site and allow them to test and give valuable feedback after testing this Web application.
7. Conclusions and Recommendations

7.1 Conclusions

In conclusion, using this Web application eliminates the need for customers to shop for motorcycle parts in the store physically, so the customers are able to rely on using this application to purchase parts securely at home. In addition, this Web application is technology-driven, meaning that the heart of this application relies on a three-tier architecture. The architecture layers are: (1) front-end user-friendly interfaces written with ASP.Net, (2) C#.Net and ADO.Net programming code working at the middle layer: and (3) a database sitting behind the application. This application will meet the following deliverables: (1) a consistent and dynamic navigational menu; (2) a well-sketched relational database diagram; (3) secure user authentication; (4) C#.Net and ADO.Net programming code to support the functionalities of this application; (5) several user-friendly front-end interfaces; (6) a new account for every new user; and (7) an on-line shopping cart that shows total prices on customers’ orders.

7.2 Recommendations

While developing this project, many technical difficulties were presented. Fortunately, many professors and colleagues provided suggestions to help the development of this senior project. The use of session variable and other features that come with ASP.Net made it a great lesson. The next phase is to install a Secure Socket Layer (SSL) certificate on the Internet Information Services (IIS) server, so that information such as credit card numbers will be encrypted when passed through Internet thus making on-line transactions secure.
Appendix A

The flow chart shown below illustrates the business logic from the customers’ perspective. It provides the process flow from seeing the Home page, making orders, and to finally logging out of the application.
Appendix B

The following section is the code sample for the store.aspx Web page that contains two panels. It also has the functionality of a shopping cart. With the code, a user can update the quantity, delete a product s/he selected, or cancel the operation of updating the quantity.

Code Sample

```csharp
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace WebApplication1
{
    /// <summary>
    /// Summary description for jenweb.
    /// </summary>
    public class jenweb : System.Web.UI.Page
    {
        protected System.Web.UI.WebControls.Panel pnlShowItems;
        protected System.Web.UI.WebControls.DataGrid dgCart;
        protected System.Web.UI.WebControls.Label lblTotal;
        protected System.Web.UI.WebControls.LinkButton Linkbutton1;
        protected System.Web.UI.WebControls.LinkButton lnkCheckOut;
        protected System.Web.UI.WebControls.Panel pnlShowCart;
        protected System.Web.UI.WebControls.DropDownList dplCategory;
        //DataRow objDR;
        public DataTable objCartDT;

        //Declare a DataTable
        public DataTable objCartDT;

        //protected System.Web.UI.WebControls.Label Label1;
        public void Page_Load(object sender, System.EventArgs e)
        {
            if (Page.Session["Status"] == null)
            {
```
Page.Session.RemoveAll();
Page.Response.Redirect("Login.aspx");
}*/
try {
    // if this page is loaded for the first time
    if (!IsPostBack) {
        // grab the items and store in Adapter at first
dgDataBind("ALL");
pnlShowCart.Visible = false;
    }
} catch (Exception test) {
    Console.WriteLine("Exception: {0}", test.ToString());
}

public void dgDataBind(string Category) {
    string myQuery="";
    if (Category == "ALL") {
        myQuery = "SELECT ItemID,ItemName,[Size],Cost FROM tblItem";
    } else {
        myQuery = @"SELECT ItemID,ItemName,[Size],Cost FROM tblItem WHERE CategoryID="+Category;
    }
    OleDbConnection objConn = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\SD2.mdb;");
    OleDbDataAdapter objDA = new OleDbDataAdapter(myQuery, objConn);
    DataSet objDS = new DataSet();
    // fill DataSet using DataAdapter
    objDA.Fill(objDS);
    // Then, do databinding to the user control
dgItems.DataSource = objDS;
dgItems.DataBind();

//When user clicks on the AddToCart Button, the AddToCart Function is invoked
public void AddToCart(Object s,
{

    //hide the panel with all of the items
    pnlShowItems.Visible = false;
    //Show the panel with each item the user selects
    pnlShowCart.Visible = true;

    //we need to get the ItemID
    //long ItemID = Convert.ToInt32(dgItems.DataKeys[e.Item.ItemIndex]);
    long ItemID = Convert.ToInt32(e.CommandArgument);

    DataTable objCart;
    if (Session["Cart"] == null)
    {
        objCart = CreateCart(ItemID);
    }
    else
    {
        objCart = UpdateCart(ItemID);
    }

    //then, perform a databinding by using the new objCartDT
    //dgCart.DataSource = objCartDT;
    dgCart.DataSource = objCart;
    dgCart.DataBind();

    //Invokes GetItemTotal function
    lblTotal.Text = "$" + GetItemTotal();

}

public DataTable CreateCart(long ItemID)
{

}
DataTable objCart1 = new DataTable("Cart");
objCart1.Columns.Add("CartID", typeof(Int32));
objCart1.Columns["CartID"].AutoIncrement = true;
objCart1.Columns["CartID"].AutoIncrementSeed = 1;
objCart1.Columns.Add("ItemID", typeof(Int32));
objCart1.Columns.Add("Quantity", typeof(Int32));
objCart1.Columns.Add("Item", typeof(string));
objCart1.Columns.Add("Cost", typeof(Decimal));

DataSet objItemInfo = FindItem(ItemID);
DataRow objDR1;
objDR1 = objCart1.NewRow();
//objDR = new DataRow();

objDR1["ItemID"] = ItemID;
objDR1["Item"] = objItemInfo.Tables["ItemInfo"].Rows[0]["ItemName"];
objDR1["Quantity"] = 1;
objDR1["Cost"] =
    objItemInfo.Tables["ItemInfo"].Rows[0]["Cost"];
objCart1.Rows.Add(objDR1);
Session["Cart"] = objCart1;
return objCart1;

public DataTable UpdateCart(long ItemID)
{
    DataTable objCart1;
    //set a flag called blnMatch to false
    bool blnMatch = false;

    //dump the current data in Session to a datatable called objCart1
    objCart1 = (DataTable)Session["Cart"];  

    //make copy of cart1
    DataTable objCart2 = new DataTable("Cart");
    objCart2.Columns.Add("CartID", typeof(Int32));
    objCart2.Columns["CartID"].AutoIncrement = true;
    objCart2.Columns["CartID"].AutoIncrementSeed = 1;
    objCart2.Columns.Add("ItemID", typeof(Int32));
    objCart2.Columns.Add("Quantity", typeof(Int32));
    objCart2.Columns.Add("Item", typeof(string));
    objCart2.Columns.Add("Cost", typeof(Decimal));
for (int x=0;x<objCart1.Rows.Count;x++)
{
    DataRow DR = objCart2.NewRow();
    objCart2.Rows.Add(DR);
    for (int y=0;y<objCart1.Columns.Count;y++)
    {
        objCart2.Rows[x][y] = objCart1.Rows[x][y];
    }
}

foreach (DataRow objDR2 in objCart1.Rows)
{
    //if condition
    //if there is already one item in the basket
    //then, increment the quantity by one
    //and then set the quantity number to the objDR's quantity property
    if (Convert.ToInt32(objDR2["ItemID"]) == ItemID)
    {
        int Quantity = Convert.ToInt32(objDR2["Quantity"]);
        Quantity += 1;
        objDR2["Quantity"] = Quantity;
        DataSet objItemInfo = FindItem(ItemID);
        objDR2["Cost"] = Quantity * Convert.ToDecimal(objItemInfo.Tables[0].Rows[0]["Cost"]);
        blnMatch = true;
        Session["Cart"] = objCart1;
        return objCart1;
    }
}

//if the item selected by the user isn't already inside the basket
//then, create a new row and then added to the objCartDT
//DataTable object

if (blnMatch == false)
{
    DataRow objDR;
    objDR = objCart2.NewRow();
    //objDR = new DataRow();
    DataSet objItemInfo = FindItem(ItemID);
    objDR["ItemID"] = ItemID;
    objDR["Item"] = objItemInfo.Tables["ItemInfo"].Rows[0]["ItemName"];  
    objDR["Quantity"] = 1;
    objDR["Cost"] =
objItemInfo.Tables["ItemInfo"].Rows[0]["Cost"];
objCart2.Rows.Add(objDR);
}
Session["Cart"] = objCart2;
return objCart2;

public DataSet FindItem(long ItemID)
{
    DataSet objItemInfo = new DataSet();
    OleDbConnection objConn = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\SD2.mdb;");
    OleDbDataAdapter objDA = new OleDbDataAdapter("SELECT * FROM tblItem WHERE ItemID=" + ItemID, objConn);
    objDA.Fill(objItemInfo, "ItemInfo");
    return objItemInfo;
}

public DataSet FindItemByName(string ItemName)
{
    DataSet objItemInfo = new DataSet();
    OleDbConnection objConn = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\SD2.mdb;");
    OleDbDataAdapter objDA = new OleDbDataAdapter("SELECT * FROM tblItem WHERE ItemName='" + ItemName + "'", objConn);
    objDA.Fill(objItemInfo, "ItemInfo");
    return objItemInfo;
}

public Decimal GetItemTotal()
{
    objCartDT = (DataTable)Session["Cart"];
}
decimal decRunningTotal = 0;

foreach (DataRow objDR in objCartDT.Rows)
{
    //    decRunningTotal +=
    //    Decimal.Round(Convert.ToDecimal(objDR["Cost"])) *
    //    Convert.ToInt32(objDR["Quantity"], 2);
    decRunningTotal +=
        Decimal.Round(Convert.ToDecimal(objDR["Cost"]));
}

return decRunningTotal;

//Edit command for datagrid object dgCart
public void dgCart_Edit(Object s, DataGridCommandEventArgs e)
{
    dgCart.EditItemIndex = e.Item.ItemIndex;
    dgCart.DataSource = (DataTable)Session["Cart"];  
    dgCart.DataBind();
}

//Cancel command for datagrid object dgCart
//Set EditItemIndex property of dgCart to -1
public void dgCart_Cancel(Object s, DataGridCommandEventArgs e)
{
    dgCart.EditItemIndex = -1;

    dgCart.DataSource = (DataTable)Session["Cart"];
    dgCart.DataBind();
}

//Update command for datagrid object dgCart

public void dgCart_Update(Object s, DataGridCommandEventArgs e)
{
    DataTable objCartDT=(DataTable)Session["Cart"];  

    //set txtQuantity to be a type of TextBox
    //set integer type var called intCartID
    TextBox txtQuantity;
    txtQuantity = (TextBox)e.Item.FindControl("txtQuantity");
    string ItemName=e.Item.Cells[2].Text;
foreach (DataRow objDR in objCartDT.Rows)
{
    if (itemName == objDR["Item"].ToString())
    {
        objDR["Quantity"] = Convert.ToInt32(txtQuantity.Text);
        DataSet objItemInfo = FindItemByName(itemName);
        objDR["Cost"] = Convert.ToInt32(txtQuantity.Text) * Convert.ToDecimal(objItemInfo.Tables[0].Rows[0]["Cost"]);
        //if found, break out the foreach loop
        break;
    }
}
Session["Cart"] = objCartDT;

//call GetItemTotal
lblTotal.Text = "$" + GetItemTotal();
//Set the EditItem mode to cancel
dgCart.EditItemIndex = -1;

//perform databinding to the datagrid by using the updated
//objCartDT DataTable
dgCart.DataSource = objCartDT;
dgCart.DataBind();

//Delete the item selected by the user
public void dgCart_Delete(Object s, DataGridCommandEventArgs e)
{
    objCartDT = (DataTable)Session["Cart"];  
    //using delete property to deleted the item selected by the
    //user
    objCartDT.Rows[e.Item.ItemIndex].Delete();
    Session["Cart"] = objCartDT;
    //perform databinding to the datagrid object by using the
    //DataTable object called objCartDT
dgCart.DataSource = objCartDT;
dgCart.DataBind();

    //call GetItemTotal function and return the calculated
    //value and set the text property
    //of the label lblTotal to the value
lblTotal.Text = "$" + GetItemTotal();
}

public void ContinueShopping(Object s, EventArgs e)
{
    pnlShowCart.Visible = false;
    pnlShowItems.Visible = true;
}

public void CheckOut(Object s, EventArgs e)
{
    //Page.Response.Redirect("FormRegister.aspx",false);
}

#region Web Form Designer generated code
override protected void OnInit(EventArgs e)
{
    //
    // CODEGEN: This call is required by the ASP.NET Web Form
    //Designer.
    //
    InitializeComponent();
    base.OnInit(e);
}

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.dplCategory.SelectedIndexChanged += new System.EventHandler(this.dplCategory_SelectedIndexChanged);
    this.lnkCheckOut.Click += new System.EventHandler(this.lnkCheckOut_Click);
    this.Load += new System.EventHandler(this.Page_Load);

}
#endregion

private void lnkCheckOut_Click(object sender, System.EventArgs e)
{
    Page.Response.Redirect("Logon.aspx",false);
}

private void lnkLogout_Click(object sender, System.EventArgs e)
{ Page.Session.RemoveAll();
    Page.Response.Redirect("Logon.aspx",false);
}

private void lnkViewItem_Click(object sender, System.EventArgs e)
{
    Page.Response.Redirect("ViewItem.aspx",false);
}

public void PageIndexChanged_Click(object sender, DataGridPageChangedEventArgs e)
{
    dgItems.CurrentPageIndex=e.NewPageIndex;
    dgDataBind(dplCategory.SelectedItem.Value);
}

private void dplCategory_SelectedIndexChanged(object sender, System.EventArgs e)
{
    dgItems.CurrentPageIndex=0;
    dgDataBind(dplCategory.SelectedItem.Value);
}
Bibliography


