Teaching a Comprehensive Web Based Application within an IS Curriculum

David C. Wallace
Applied Computer Science Department
Illinois State University
Normal, Illinois 61790-5150
dcwalla@ilstu.edu

Abstract

As Internet technologies have exploded over the past few years, corporations struggled to develop standards for managing their Internet resources. Since the passing of "Y2K," organizations have increasingly focused on developing an effective Internet presence. Consumer shopping on the Internet is expected to grow to around $1 trillion by the year 2003, and Electronic Data Interchange (EDI) is expected to grow to around $300 billion by the same year. With this escalating demand for Internet-based commerce, organizations are attempting to recruit IS personnel who can develop Internet applications, and they expect academic institutions to provide IS professionals with the skills necessary for this rapidly changing technological environment as well as traditional skills needed for the mainframe environment. With limited resources, academic institutions have been revamping their curriculum to accommodate the growth of the Internet and to maintain the more traditional IS core curriculum. By utilizing readily available computer hardware and software, academic institutions can provide an effective and active Internet learning environment.

1. Introduction

Preparing IS professionals is a challenge for academic institutions factoring in a rapidly changing technological environment. With the explosion of Internet technologies over the past few years, organizations have struggled to develop standards for effectively managing their Internet resources. Consumer shopping on the Internet is expected to grow to around $1 trillion by the year 2003, and Electronic Data Interchange (EDI) is expected to grow to around $300 billion by the same year (Grover, 2001). With this escalating demand for Internet-based commerce, organizations are trying to recruit IS personnel who can develop Internet applications, and they expect academic institutions to provide IS professionals with the skills that are needed for this rapidly changing technological environment as well as traditional skills needed for the mainframe environment. With limited resources, academic institutions have been revamping their curriculum to accommodate the growth of the Internet and to maintain the more traditional IS core curriculum. The demand for both Internet and mainframe skills remains strong as indicated by the high salaries and vacant positions in the job market for IS professionals.

Eventually, the mainframe and the Internet (client/server) environments will have to be integrated into one complete, comprehensive system. Skills developed in the more traditional mainframe environment will have to be incorporated into a complex networking system that can seamlessly transfer information from one environment to another without giving up security or efficiency.
The IS professional has to possess the skills for both environments; he/she will play the role of an integrator.

With limited resources academic institutions must provide these skills for their IS professionals. Much has been written concerning the implementation of mainframe base technology. Yet, how can academic institutions incorporate the client-server technologies of the Internet with traditional mainframe technologies? One of the most popular personal computer (PC)-mainframe configurations has been to provide a mainframe connection called a gateway to a local area network (LAN) where the PC becomes a terminal for the mainframe. Under this configuration academic institutions can access both PC and mainframe software. By utilizing computer hardware and software products already available on the PC, academic institutions can provide an effective and active Internet learning environment.

2. Tools

The software tools needed for the client-server environment can be readily found within the operating system of the PC and the common software products that academic institutions provide for their introductory classes to the PC environment. Software like Web browsers, text editors, and database management software are common components of a PC environment. Since students often use their own computers to work on projects at home or in their dormitory rooms, it is also important that resources be readily available for this use.

The client-server environment requires the operating system to be both a client computer as well as a server computer. This dual role demands software that can be

used to develop the client-side applications and software that can be used to emulate the server-side operations. Figure 1 illustrates this relationship.

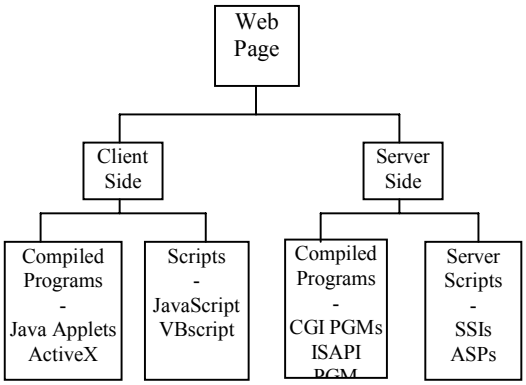

Figure 1. Client-Side and Server-Side Relationship

Web site development requires client-side components like Java applets, ActiveX, and scripting languages (JavaScript or VB Script). The server-side components are compiled programs called CGI (Common Gateway Interface) programs written in languages like Visual Basic, JAVA, PERL, ISAPI, or COBOL; and scripting languages like Active Server Pages (ASP) and Server-Side Includes (SSIs). Choosing components is a matter of surveying the market to find the most popular components or testing the components in typical Web applications to determine which performs the most effectively. Using the most common tools is a reasonable approach to this decision; Java Applets, JavaScripting, and Active Server Pages are some common Internet components.

Windows 98 provides a web server manager software called "Personal Web Server" that can support server-side applications. Either Netscape or Internet Explorer can support client-side applications. The database management software called ACCESS is a common database management tool that can be used to illustrate database applications on the Internet. ACCESS is a relational database that can be accessed and updated using the Structured Query Language (SQL). Microsoft's SQL Server is a more industrial strength database management system that can be upsized from within ACCESS. Both relational databases and SQL are the most common database components on the Internet today. Students can also acquire ACCESS at a considerable discount for their home use.

2.1 Setup

Personal Web Server (**PWS**) can be used to develop server-side applications very easily. Unfortunately, it can only handle 10 Web connections from client computers. Each connection forms a communication socket between the server and the client computer. Yet, for a development environment, it gives us the type of control and performance that can be expected from a more enterprise-driven server like Microsoft's Internet Information Server (IIS) or the Apache Web Server. Once the Web site is developed using PWS, a developer simply moves the Web pages, images, databases, and other components to a production server like IIS or Apache that can be accessed by anyone on the Internet.

PWS allows the developer to establish root directories and virtual directories needed to house a Web site. Figure 2 illustrates the main window that points to the root directory "C:\DAVE." The root directory is the actual starting position on the Web server for the Web site. It can be changed to any directory on the server using the PWS manager's "Advance Menu." The advance menu can also be used to establish virtual directories that can hold executable files, databases, and other components that are often linked to Web pages.

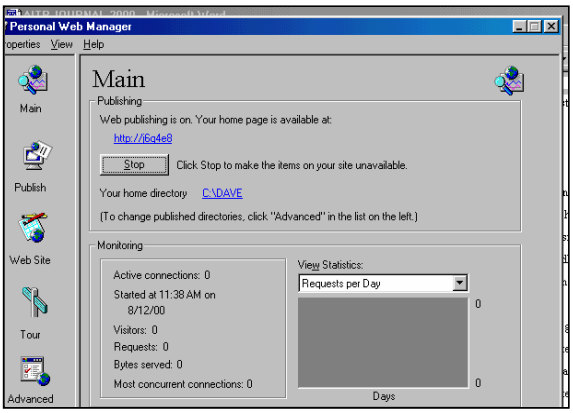

Figure 2. Personal Web Server – Manager

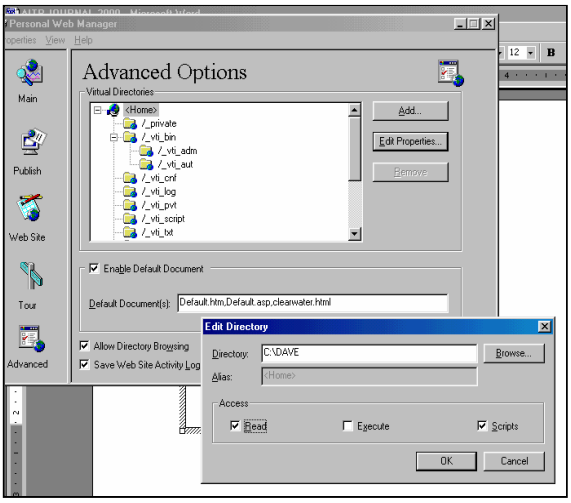

Figure 3. PWS Advanced Options

Figure 3 demonstrates the use of the advance menu to add ("ADD") virtual directories or change ("EDIT") the root directory. The PWS manager default document can point to the index page or the main page within the root directory; these are the pages first seen by the user when they type the URL address within their browsers. Using the Snap-In tool within IIS, the user can also designate root directories, virtual directories, and document settings.

The PWS manager allows the developer to control access to directories connected to the Web site in terms of "READ," "EXECUTE," or "SCRIPT." The "READ" option permits the user to download Web pages, images, or any other components on the Web pages that reside in the current directory. The "EXECUTE" option permits the user to execute any CGI programs within this directory. Finally, the "SCRIPT" option allows the user to execute server-side scripts within this directory. These privileges are important because they allow the Web developer to control the directories where Web pages, images, databases, and programs are stored. This setup establishes an environment that can be used to train IS professionals.

3. Database Design

Database design is essential for developing any Web site that requires tables of information to update and retrieve data. This topic is often presented in earlier courses in an IS curriculum; but could be presented as part of a comprehensive Web site development course. The discussion should focus on relational database design, since over 95% of database access on the Internet is relational databases. Relational design involves the normalization process, which can help IS students understand possible relationships between tables of information. The normalization process is beyond the scope of this paper and should be reviewed if necessary, but it will produce easily maintainable and accessible data tables. The relationship diagram is an important tool generated by this process; Figure 4 provides an example of a relationship diagram.

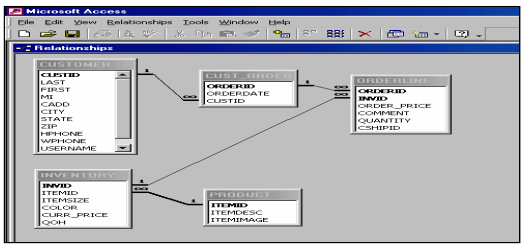

Figure 4. Relationship Diagram

Once the relationship diagram is generated, ACCESS or Oracle (ACCESS is more readily available on student's computers) can be used to design each table: to identify the data fields, the characteristics of each field (text, number, size, validation rules, etc.), primary keys, and foreign keys. ACCESS is a personal database which can be used in a development environment to gain the knowledge and experience of relational databases. Yet, Oracle would be a more appropriate relational database for the client-server environment of the Internet. SQL commands will work with either database allowing ACCESS to be a viable alternative to Oracle in the development process.

The next step is to load the tables with sample information. To expedite the loading process, the instructor can provide the data for copying into the tables.

3.1 Structured Query Language (SQL)

The database will reside on the server-side computer. The Web pages that will access the database and process the resulting information will also be stored on the server-side computer. When a user requests a Web page that has some server-side processing, the server-side scripts or CGI program is executed before the page is sent to the client's computer. Structured Query Language (SQL) is used within the server-side scripts or CGI program to access the database. Specifically, embedded SQL commands access and update the tables within the database. Internet developers should have a good working knowledge of SQL statements.

By using the Query option within ACCESS, the IS developer can easily generate from the most basic to the most complex SQL statements. Figure 6 is an example of a standard JOIN statement between the "Customer" and the "Cust_Order" tables. The resulting table is a combination of information from two tables for customer number 5.

```
SELECT FIRST, MI, LAST, CADD, CITY, ZIP, HPHONE,
     ORDERID, ORDERDATE
FROM CUSTOMER, CUST_ORDER
WHERE
  CUSTOMER.CUSTID = CUST_ORDER.CUSTID
AND
 CUSTOMER.CUSTID =5;
```

FIRST	MI	LAST	CADD	CITY	ZIP	HPHONE	ORDERID	ORDERDATE
Sally	T	Flag	891Willis	Normal	61790	(309) 555-2331	5	11/1/00
Sally	T	Flag	891Willis	Normal	61790	(309) 555-2331	6	11/1/00

Figure 6. Join SQL Statement

The "JOIN" statement reflects the power of relational databases. SQL can take information from more than one table from one query. The relationship diagram helps the developer trace the connections between tables using the "WHERE" portion of the SQL statement.

Notice the connection between the customer table and customer order table is customer ID. The "WHERE" portion of the SQL statement can shape this relationship with an "equal to " condition (CUSTOMER.CUSTID= UST_ORDER.CUSTID). ACCESS thus provides a mechanism to build a solid insight into relational design and database processing.

4. Web Site Development

Generating a sound Web Site Plan will prevent confusion and add continuity to the development process. The focus of the Web Site Plan should reflect the logical information flow between the client and the server computers. The logical flow should be developed into a Web Site Flowchart. Figure 7 provides an example of a Web Site Flowchart.

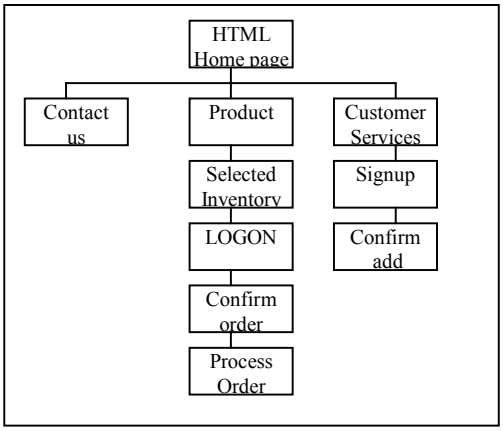

Figure 7. Web Site Flowchart

Once the Web Site Flowchart is developed, the developer can start to visualize the information needed for each page on the diagram. This information might be needed to access databases, to pass to subsequent pages, to be placed in cookie files, or simply to add aesthetic value to the web page. For example, the "Selected Inventory" page in Figure 8 requires the image of the selected product along with its sizes, colors, and availability. This information is taken from two tables: (1) product and (2) inventory. The user provides the quantity and the selection of an item. According to the Web Site Plan, the "Selected Inventory" page will to pass inventory item and quantity to the "LOGON" page so that the information can be passed along to the "Order Confirmation" page. The latter page will calculate shipping, taxes, and totals. Figure 8 shows the portion of the Web Site Plan involving this process. The complete Web Site Plan will entail the complete development process.

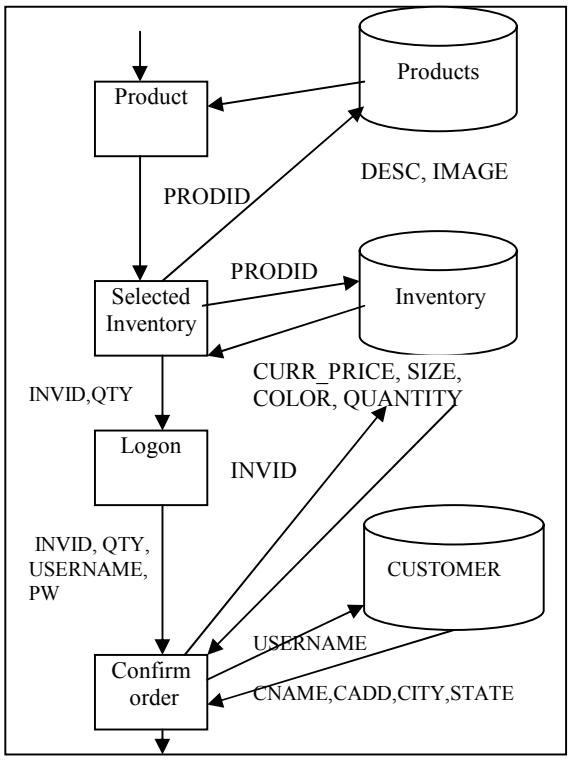

Figure 8. Partial Web Site Plan

4.1 Coding

The Web Site Plan provides the more detail information needed to process data and display this information on a Web page. Yet, the Web pages will also include the images, logos, background colors, style sheets, animation, and other design components that will bring the Web pages to life for the user. A **layout sheet** is a good tool for displaying how colors, images, animation, frames, tables, and scripts should interact and coordinate with each other to present a logical, consistent, well-constructed Web page.

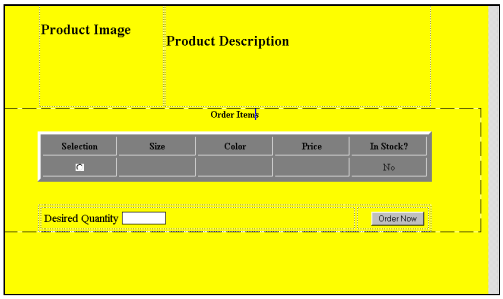

Figure 9. Web Page Layout Sheet

Figure 9 illustrates an abbreviated sample of a layout sheet with its various components. By subdividing the various components of the Web page into the heading that is read first by a browser and the body where the action takes place or originates, the developer can place server-side scripts in strategic locations where the results can be displayed on the Web page. For example, product's image and description from the product table must be retrieved before the individual sizes and colors from the inventory table. Both tools serve to formulate the Web page and guide the coding process.

4.2 Active Server Pages (ASP)

The title bar on the Web page will carry the description of the product. The developer can access the product table and retrieve this description using Active Server Scripts (ASP) that relies on an Open Database Connectivity (ODBC). ASP uses a server object that can be opened and executed using embedded SQL statements.

Setup Server Object:

```
<% Set ConDW  =  Server.CreateObject("ADODB.Connection")%>
```

Open Server Object:

```
<% Dim DBPath, DrvType %>
<% DBPath = Server.MapPath("customer.mdb") %>
<% DrvType = "Driver={Microsoft Access Driver (*.mdb)}; " &
                          "DBQ=" & DBPath %>
<% Set connCW = Server.CreateObject("ADODB.Connection") %>
<% connCW.Open DrvType %>
```

Execute Server Object with query string:

```
<% queryString = "SELECT proddesc, prodimage " & _
                 "FROM product " & _
                 "WHERE prodid = " & intProdID %>
<% Set rsProduct = conDW.Execute(queryString) %>
```

Figure 10. Server Object Sequence

```
<!--create a query string to retrieve item and image -->
<% queryString = "SELECT * " & _
                 "FROM inventory " & _
                 "WHERE prodid = " & intProdID %>
<% Set rsItem = conDW.Execute(queryString) %>
<% intloopCount = 0 %>
<% Do While Not rsItem.EOF %>
 <!-- set first button checked -->
   <TR><TD ALIGN=CENTER width="112"><INPUT
TYPE=RADIO
          NAME=invid  VALUE=<% =rsItem("INVID")
%>></TD>
    <TD ALIGN=CENTER width="117"><% =rsItem("ITEMSIZE")
    %></TD>
   <TD ALIGN=CENTER width="114"><% =rsItem("COLOR")
    %></TD>
   <TD ALIGN=CENTER width="113"><% =rsItem("CURR_PRICE")
```

Figure 11. Displaying Retrieved Items

Figure 10 illustrates the sequence of commands that retrieves records from a table within a database. Notice that the embedded SQL statements are placed in a string variable called "queryString" and that ASP scripts are surrounded by the "<%" and "%>" delimiters. The Server Object executes this query string. This pattern can be replicated for each Web page that accesses a database using ASP scripts.

Figure 11 demonstrates a SQL execution where the result is a table of records. These records are placed in a record set called rsProduct. The developer can display these records with ASP scripts (<% =rsProduct(PRODDESC) %>) using a looping process. Figure 11 demonstrates the looping process that can be coded to display records coming from the record set.

Passing information between Web pages is essential when subsequent Web pages require the information or simply needs to pass it to the next Web page. As indicated by the Web Site Plan, some Web pages will have to use or pass information from a previous Web page. Figure 12 shows how product ID (prodid) is sent to the Order Page using the URL method. The Order Web Page receives the variable using the "Request.Querystring" method. Once the product ID is received, the embedded SQL statement (queryString) can use it to retrieve a record from the product table (See Figure 12). Notice that the name of Web pages with ASP scripts has the ".asp" extension.

Send Product Page:

```
  <A href="order.asp?prodid=<% =rsProd("PRODID") %>">
     <% =rsProd("PRODDESC") %></A></td>
```

Receive Order Page

```
<% intProdID = Request.Querystring("prodid") %>
```

Figure 12. Passing URL Variables

Often variables must be sent to another Web page using the "Form" method. The sending page will have a form with variables. This form can be passed to another Web page along with its variables. The receiving page will use the "Request.Form" method to receive the variables.

Form Data Passed:

```
<FORM NAME=FrmLogin METHOD="POST"
ACTION="ordersconfirm.asp">
<INPUT TYPE=HIDDEN NAME=invid
          VALUE=<% =Request.Querystring("invid") %>>
<INPUT TYPE=HIDDEN NAME=quantity
          VALUE=<% =Request.Querystring("quantity") %>>
<TABLE WIDTH=60%>
<TR><TD WIDTH=50% ALIGN=RIGHT>User ID:</TD>
   <TD WIDTH=50% ALIGN=LEFT><INPUT TYPE=TEXT
     SIZE=15 NAME=username></TD></TR>
<TR><TD ALIGN=RIGHT>Password: </TD>
   <TD ALIGN=LEFT><INPUT TYPE=PASSWORD SIZE=15
     NAME=password></TD></TR>
```

Form Data Received:

```
<% passInvID = Request.Form("invid") %>
<% passQuantity = Request.Form("quantity") %>
<% passUsername = Request.Form("username") %>
<% passPassword = Request.Form("password") %>
```

Figure 13. Form Data Pass

Figure 13 shows how form data between the Logon Web page and the Confirm Order Web page is sent and received. The Web Site Plan along with the Layout Sheets should show the variables included with forms and the variables to be passed as URL data. Figure 13 shows that the Logon Web page receives inventory ID (invid) and quantity from the Order Web page. The sending page places these values as hidden variables within the form called "frmlogin." This form also receives the username and password from the user. The Logon Web page will pass to the Confirm Order Web page four form variables as indicated by the Web Site Plan.

4.3 Testing/Debugging

Testing and debugging generally occurs at four levels: (1) the code level, (2) the page level, (3) the unit level, and (4) the system level. The code level focuses on segments of code that accomplish a specific task like accessing specific records from a database, displaying images, or linking to other pages. At this level display statements can be used to evaluate the status of variables at a specific point in the execution of the Web page or to identify the extent of execution of the Web page. Figure 14 identifies some typical display statements for a Web page with ASP script variables and some alert statements for JavaScript variables. The display statements will help the developer track the execution of the program and the value of the variables during execution. This technique is very beneficial when debugging a Web page.

```
ASP Display Method:
Quantity = <% =passquantity %>
Password = <% =passpassword %>
Javascripting ALERT method:
<SCRIPT LANGUAGE="JavaScript">
  function checkit(frmOrderItems)
   { alert("within checkit")
    var strUserName, strInvID, strQuantity,
        strURLParams
     alert("before loop")
```

Figure 14. Testing/Debugging Statements

The page level testing looks at how all of the components on the Web page work together to achieve an overall purpose. Figure 15 illustrates a simple Test Plan that can determine the accuracy and success of all the actions involved on one Web page. Notice that the expected and actual results generate remarks that help the developer solve any differences between them.

Unit level testing involves identifying a complete process to accomplish a given task. Customer ordering involves more than just one Web page, but the ordering process is one unit or transaction. The Test Plan should identify all possible logical outcomes associated with the ordering process. Each of these outcomes should be tested in order to verify that the Web site handles these situations correctly. Any discrepancies should be noted in the remark column and addressed with the appropriate actions. Other possible transactions or units can include customer signup, customer services, and customer feedback. Of course, the number of units depends on the size of the organization and the Internet services that it provides.

Item	Expected Result	Actual Result	Remarks
1. Initial load	Home Page	Home Page	
2. Image Load	5 Images	4 of 5	Logo failed
3. Receive Pass ProdID product	Selected	Selected	
4. Link to Program	Pgm Page	Pgm Page	
5. Retrieved Product Data	Description	No Values	SQL failed
6. Date Retrieved	Current Date	Current Date	
7. Calculate total	Total Sales	Wrong Total	Calc. not correct
8. Text Spelling	Correct Sp	Correct Sp	

Figure 15. Test Plan

Finally, testing and debugging at the system level focuses on navigating amongst all the various units on the Web site including ordering, customer services, and administration. A Test Plan at this level will list all the possible units on the Web site, which are often listed on the index or main page as links to other Web pages. The expected and actual results consist of the appearance of the correct Web page for each hyperlink to include both forward and backward links. Any differences between expected and actual results should be noted and placed as an appropriate remark on the Test Plan. These remarks help the developer rectify any user navigation problems.

4.4 Documentation

The final stage in the development process is documentation. Documentation is used to provide a basis for review and to update the Web site. Web sites are dynamic because they change with technology and market conditions. The organizations must be able to this challenge of a rapidly changing environment. Establishing a documentation standard for the organization will provide continuity and consistency in the development process. The documentation becomes the "road map" during the analysis process that will expedite future enhancements and improvements of the Web site. Figure 16 shows an abbreviated set of documentation for a Web Site. The

documentation can include test plans to verify the Web site's performance.

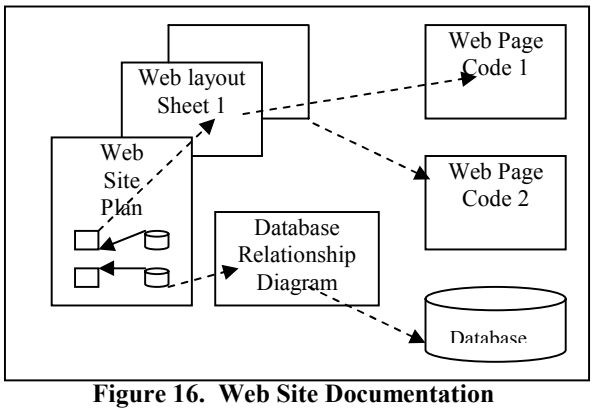

Figure 16. Web Site Documentation

Internet will dramatically increase over the next few years, and with this escalating demand for Internet-based commerce, organizations are expecting to recruit IS personnel who can develop applications on the Internet. With limited resources, academic institutions have been revamping their curriculum to accommodate the growth of the Internet as well as to maintain the more traditional IS core curriculum. By utilizing readily available computer hardware and software, academic institutions can provide an effective and active Internet learning environment.

References

[1] Cashman, Shelly, and Quasney, Dorin. JavaScript: Complete Concepts and Techniques. Cambridge, MA. Course Technology, 2000.

[2] V. Grover and J. Teng, "E-Commerce and the Information Market: Breeding the New Infomediaries," Communications of the ACM, Vol.44, No.4, 2001, pp.79-87

[3] Morneau, Keith, and Batistick. Active Server Pages. Cambridge, MA. Course Technology, 2001.

[4] Morrison, Mike. and Morrison, Joline. Database-Driven . Cambridge, MA. Course Technology, 2001.

[5] Schneider, Gary P. and Perry, James T. Electronic Commerce. Cambridge, MA. Course Technology, 2001.