Personal Document Management System

By

Caryle Patrick Clear Jr.

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

March 2001

Personal Document Management System

By

Caryle Patrick Clear Jr.

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

Author: Caryle Patrick Clear Jr.	Date: 3-20-01
Faculty Project Advisor	Date: 3/13/01
Department Head	Date: 3/19/01

Acknowledgements/Dedication

I would like to thank all of the faculty members that have helped me expand, improve and test my abilities and skills. I would also like to thank Prof. Soleda Leung for showing me that there is nothing that I cannot do.

I would also like to thank all of the friends that I have met during this time: Ashika Wijesekera, Candise Wise, Pam Naber, Jason Benson, Hao Doan, Jeff McCord, Jeremy Louis, Jason Kumpf and Robert Brown.

Finally, to my wife, Anneliese, who has put up with all of the late nights, the disasters, and the victories. You have kept me going more time that I could have counted.

Table of Contents

List of Figures

Abstract

The Personal Document Management System allows users to store their documents on a web server. Files are uploaded and downloaded using a web browser. This is a secure environment that is not directly assessable to users.

The Personal Document Management System has two different types of accounts: Account holders and User accounts. Account holders are users that can upload and download files along with the ability to create user accounts. User accounts let a users have permissions to files that belong to the Account holder. The Account holder must give these permissions to the user.

The Personal Document Management System also has a document revision system that stores revisions of files. This allows users to revert back to a previous document if necessary or view the creation process of a document.

The Personal Document Management System insures that users will have access to their personal documents regardless of time or location. This prevents users from losing or forgetting their disks or forgetting where the document has been placed on their hard drive. It also ensures that a user will have access to the most current version of their document. The Personal Document Management System solves the problem of managing documents that a person needs on a daily basis.

Personal Document Management System

1. Statement of Problem

Many businesses have document management systems that are used by their employees. They are set up for companies and/or departments and are used to keep track of Enterprise Data. Metaphase, a product of Structural Dynamics Research Corporation, is one of these systems. Metaphase is capable of revision tracking, document locking, and archiving. It provides a business the ability to expand their development process allowing employees around the world to continue the development cycle on a twenty-four hour basis. The revision tracking and locking capabilities of the software "*eliminates expensive rework caused by isolated decision-making during the early stages of product design* (SDRC Website, collaborate.html)." They are not designed for personal use but are used to help keep track of revisions, document locking and archiving for business projects. They do a very good job but lack the functionality that many people use on a daily basis.

This country uses an average of 749 pounds of paper products per individual per year or 187 billion pounds of paper. The Mead Corporation (one of the top paper manufactures) sales of paper and office products accounted for over $2293.6 million dollars of their $3800.5 million dollar sales. People are always printing documents or have to turn in some sort of paper document. There are many different systems to keep track of important papers ranging from file cabinets, the pile system and folders. Organization is the most important component for a person to be effective at paper tracking.

A Personal Document System has the flexibility to improve efficiency and organization, allowing for easier collaboration and functionality on a daily basis. The user interface should be configurable by the users, promoting a very intuitive environment. An Internet based design is necessary with communication and business moving to an increasingly technical platform.

1.1 Solution

A Personal Document Management System (PDMS) would solve many of the problems that a person faces on a daily basis. The first major advantage of a PDMS would be that it is Internet based and would be accessible using the World Wide Web. This way all documents would be available over the World Wide Web anywhere in the world at any time. A person would not have to worry about forgetting or losing a document because they would be able to retrieve it using a web browser. A person would also be able to travel and still have all the documents that they would normally have that the office or at home.

Collaboration would be another advantage of such a system. A person would be able to give others permission to read, write, or upload documents. This way a person would not have to get a document from someone else. This functionality would be handled using accounts that only could be set up by the owner of the PDMS. All a person would have to do would be to login to the system and then retrieve the documents that they would need. They would also be able to make changes to a document if needed or upload a document. A user would also be able to e-mail documents to a person of a list of people. The system would use meeting dates and e-mail lists to send documents to people in time for the meeting or other important dates.

The functionality of the system would be the last feature. The PDMS user interface would be customizable. That way the user would have a choice in the way their screens would look and could style it based on their preferences. Finally the system would be searchable using a key word so documents could be easily found. Data would be in one location allowing users to easily back up documents and information.

2. Description of Solution

This project will allow the user to manage, retrieve and store documents online. This will be done through a Web site that allows authorized users to access the information stored in a database. Account holder accounts would be created by the owner/administrator. These accounts would be able to add guest users and give them permission for each document that they have stored in the system. Account holders would create user names and passwords for the guest users. Account holders would have full control over any documents that they place in the system. Account holder status does not automatically mean that they have access to all documents in the system. If a user does not own the document they must have rights given to them by the owner of the file. This document management system will also have a method of tracking revisions of a document. This will allow a user to upload a newer version of a document and the old version will be archived in case the older document is needed.

2.1 User profile

The target users for the system will be intermediate to experienced users. These users should be familiar with the Microsoft Windows file system and be familiar with the user of a Web browser, preferably Microsoft's Internet Explorer 4.0 and above. The user

would be able to easily navigate a Web site that has good site design and intuitive site navigation. Account holders should understand file permissions and how they can be used to grant and denied access to documents that have been stored using the Web site.

2.2 Design protocols

2.2.1 Welcome/Signin page

This page will let users login and be validated it to the system. The database will contain the user name and password that will be used to compare what was entered in the page. The page will have two text boxes for the user to enter their user name and password. Once they have entered this information and selected the Submit button the authentication process will start. A reset button will allow the user to clear the form in case of mistakes.

2.2.2 Create/Manage users page

This page will be used by Account holders to create new users and change permissions that the users currently have. It will display the list of users, the documents they have permission to use and the type of permissions that they have. There will be text boxes for entering the new user's name and password. In the list of current users there will be a list of users and the documents (this will be done with a table). A drop-down list will display the current permissions and let the account holder change permissions to a new setting. Pressing the update permissions button will update the permissions to the value selected.

2.2.3 Manage Files page

This will allow the account holder to delete documents and view different revisions of the document. This will be done using a table for the structure of the page and contain the following:

A text listing of all documents
A command button to delete files
A link to the document history and revisions page (this will update the page and display the revision information)

2.2.4 Upload Files page

This page will let the any user with the proper permissions to upload a file. It will have a text box that will let them specify the local path of a file they want to upload and a command button that will display a dialog box to traverse the file system. A text box and a memo field will be used to type the name and any notes about the file. When the user clicks the upload page the upload process will begin. The page will also have a drop-down list that contains file names that will be used for the document history.

2.2.5 Delete Page

The Delete Page will contain a page of documents that are on the system and allow for the deletion of pages. There will be a list of documents and a command button next to each one. When the command button is selected the document will be deleted. There will be an option on the page to delete the document and all previous revision of that document or just the document, keeping the revisions.

2.2.6 Download (View) Page

This page will display all the documents that a user has permission to view and display a list of them with all relevant information. The user will be able to select the

document that they wish to download. Once this has been done, the download process will start.

2.2.7 Search

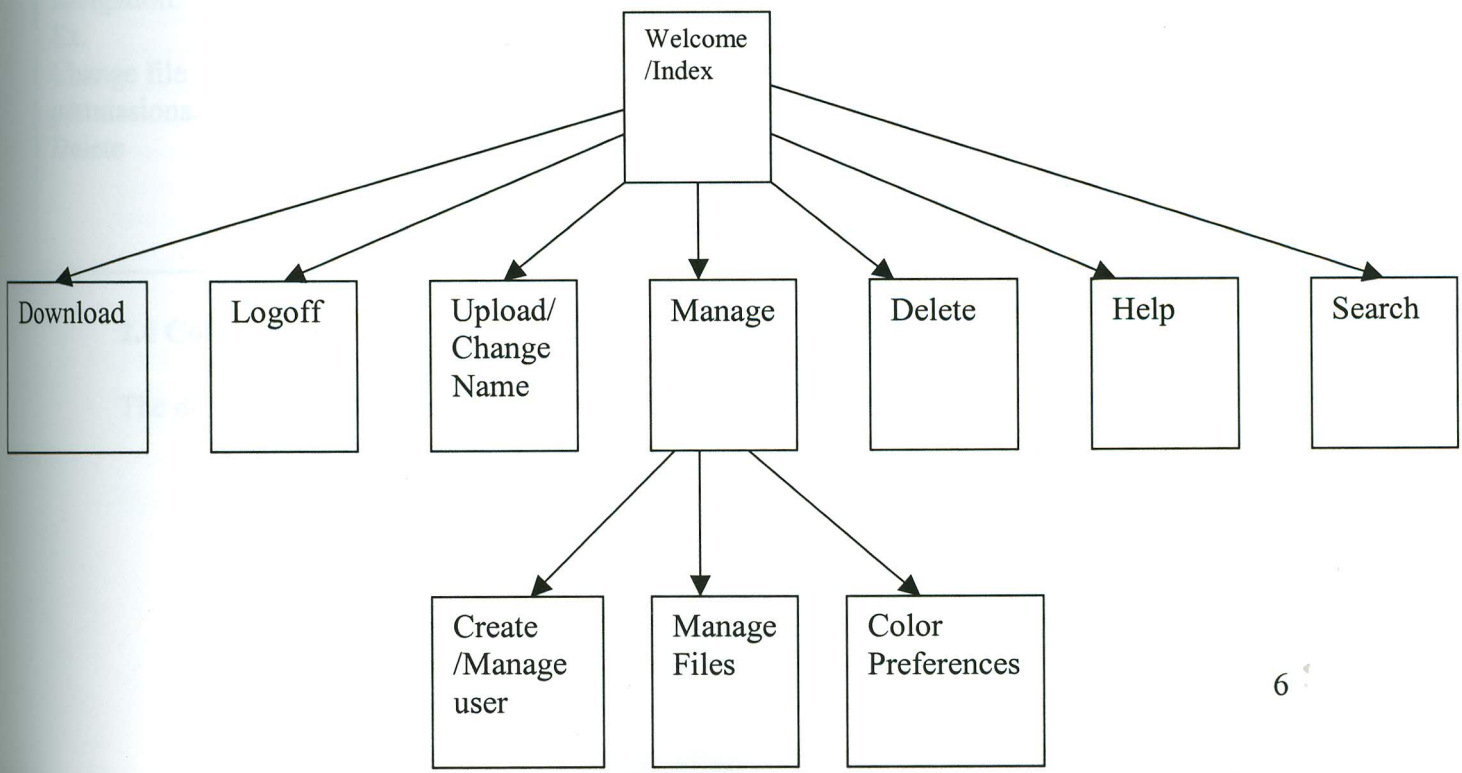
The Search page will let a user search for documents and display information about those files. They will be able to perform any operations that they have permissions to on this page with out having to go to another page. There will be a drop-down list that will have all of the available options. When they select an option and press the command, button that operation will proceed.

2.2.8 Logoff

The Logoff pages will logoff the user and return the user to the welcome page. This will be done when they click on the logoff top-level navigation.

2.2.9 FAQ/Help

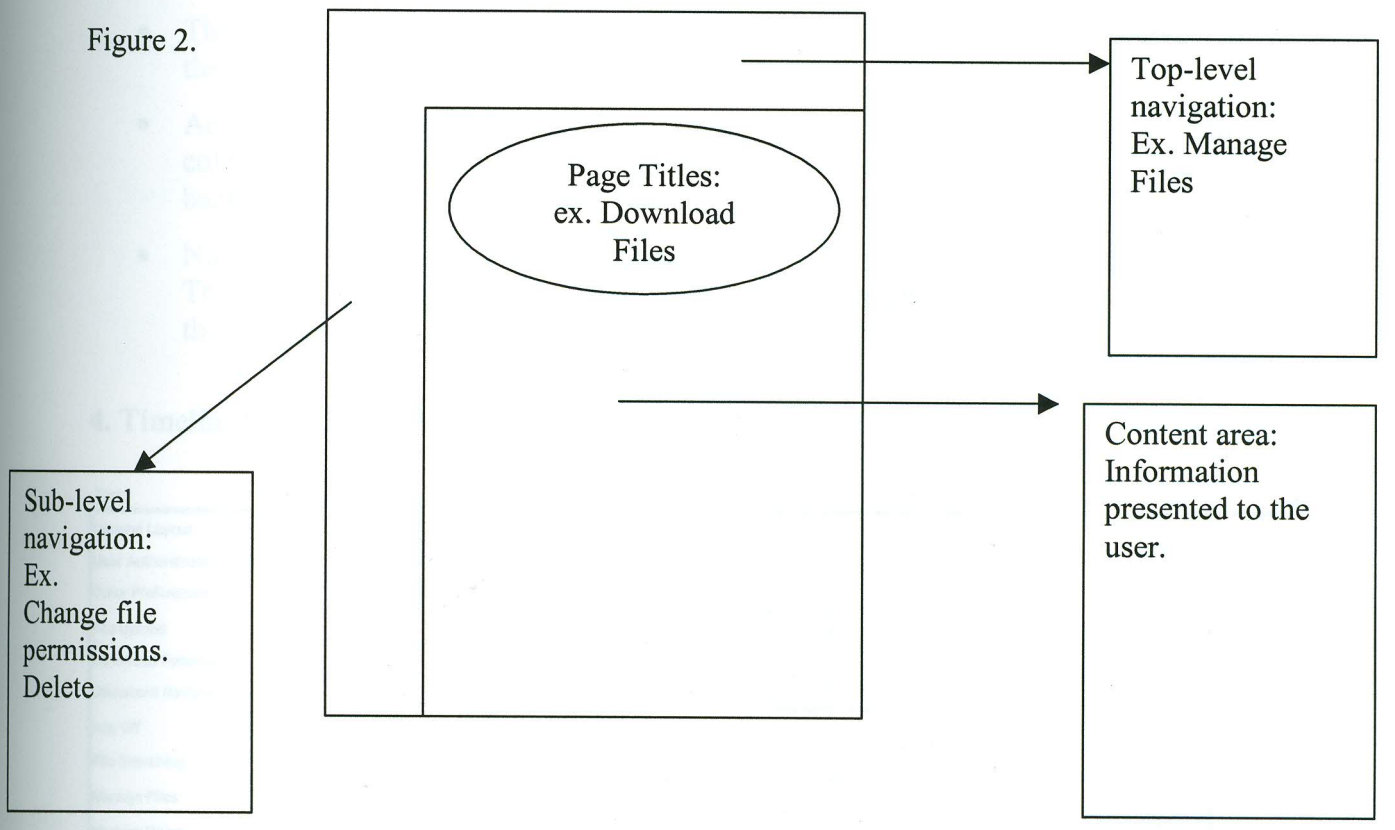
The FAQ will have a description the major functions and how they work.



2.3 Interface Design/Navigation

This site will use the standard top-level navigation for main categories and left-side navigation for sub categories. These levels will be hyperlinks that will take the user to other pages that will allow for further operations. The main content area will be the center and right of the page. This will be used to present information to the user and should be considered the focal point of the web page. The page titles will be placed at the top of the content area.

Figure 2.



2.4 Color scheme

The default color scheme includes:

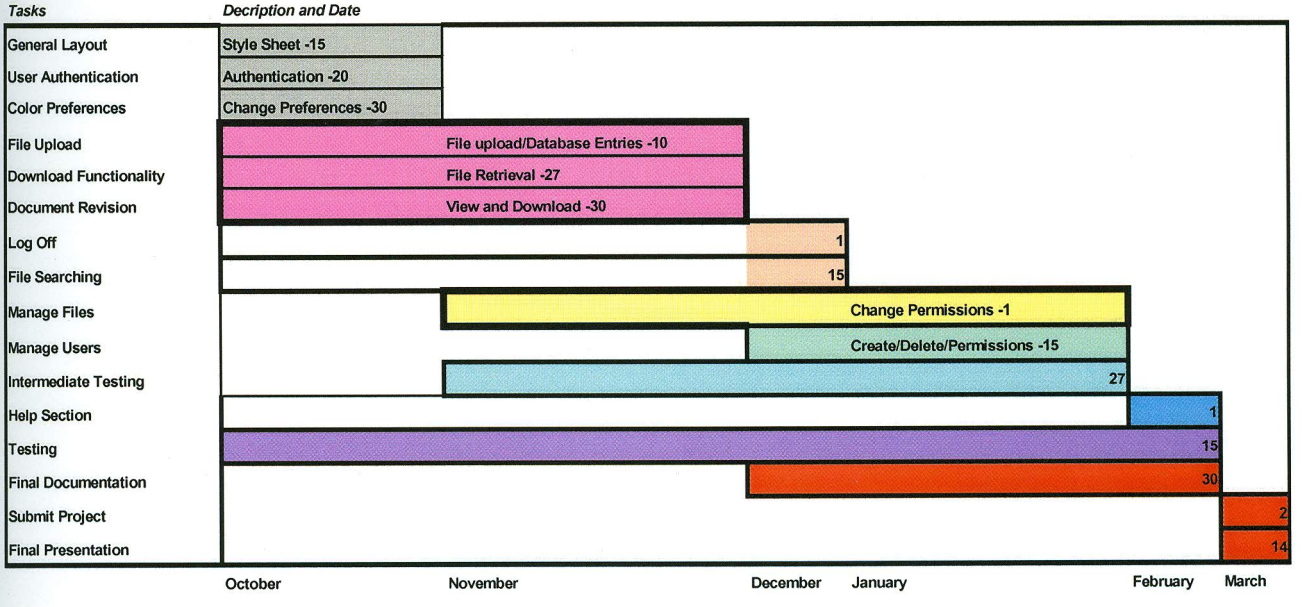
Background: White
Text color: Black

The user will be able to change the background and text color to a color of there liking. This information will be stored in the database and the user will be able to change the color at any time.

3. Deliverables

- The site will verify all account holders and guest users with information in the database.
- The site will let authenticated users to upload, download and delete files in the system.
- Users will be able to keep track of the revisions to a document via revision tracking (this will happen when a file is re-uploaded to the system).
- The site will allow account holders to create users, delete their users and give them permissions to files on the system.
- Account holders will also be able to customize their environment by changing the color scheme of the web pages. The default color scheme will consist of a black background, silver text coloring, white heading and white hyperlinks.
- Navigation will be consistent on all pages with a format similar to Figure 1.2. Top-level navigation will be at the top, Sub-level navigation will be to the left of the page and the content will be in the center of the page.

4. Timeline* and Budget



*Dates represent completion of task/functionality

4.1 Budget

4.1.1 Hardware

PII –450 or equivalent	$150	
128 Megs of RAM	$220	(Server RAM)

4.1.2 Software

BinFile Component	Free	Free download on www.15seconds.com
ASP Smart Component	Free	Free download on www.15seconds.com
IIS	Free	Included with NT/Downloadable from Internet
NT 4.0	$299	
SQL Server	$899	
Photoshop	$299	
Total	$1867	

5. Proof of Design:

5.1 Welcome/Signin page

Figure 3.

This page allows users to sign in to the system. This is accomplished by letting users enter their username and password. They then select what kind of account they have, a User Account or an Account Holder. The user of the system then presses the Login button. This page then goes to a login page that checks to see if the account exists in the system. A check is also done to see if data has been entered. If there is no user information submitted then the browser is redirected to the index page to re-login. This is accomplished by looking in the database to see if the entries are in the database.

```
'Check to see if there any data. If not then redirect to index.asp page
if strUserName = "" then
```

```
Response.Redirect ("index.asp")
elseif strPassword = "" then
Response.Redirect ("index.asp")
end if
If strAccountType = "AccountHolder" then
'Build sql statement to look up user in database.
strSQL = "select AccountHolderId from AccountHolders where UserName='" &
strusername & "'" & " and Password='" & strPassword & "'"
        else
         strSQL = "select UserId from Users where UserName='" & strusername
& "'" & " and Password='" & strpassword & "'"
end if
```

A cookie is created and placed on the users machine. This cookie contains the type of account and there account Id. This is used on other pages in the system to check what type of account they have and the files that are associated with that Id number. If the user exists in the system then they are taken to a general page that will display all of the options that they have to select form.

```
'Set the cookie
If strAccountType="AccountHolder" then
Response.Cookies("uid") = objRst.Fields ("AccountHolderId")
 Response.Cookies ("acc") = strAccountType
Else
Response.Cookies("uid") = objRst.Fields ("UserID")
Response.Cookies ("acc") = strAccountType
End If
 'take them to the right page
Response.Redirect ("choice.asp")
```

5.2 Create/Manage users page

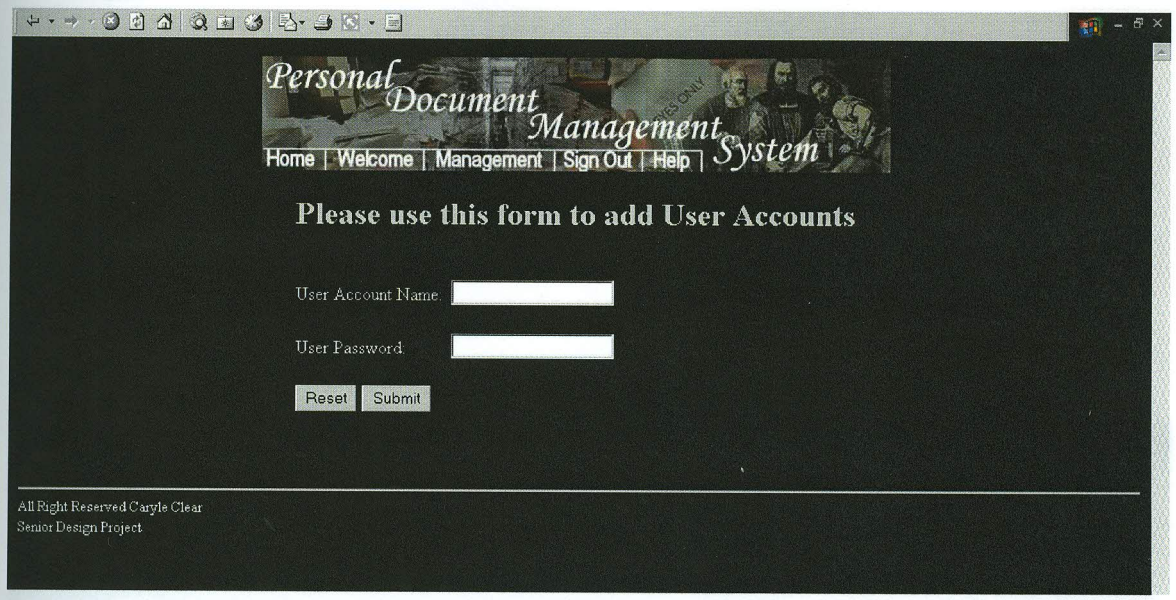


Figure 4.

The user enters the new user name and the new password for the user. A check is done to see if the user is permitted to create new user accounts. Only Account Holders are permitted to create new accounts. The new user information is then sent to a stored procedure that then adds entry's into the database. At this time the users color preferences are also entered in to the database. These entries are default entries that can be changes later by the new user.

```
strNewUser=Request.Form ("UserName")
strPassword=Request.Form ("Password")
if Request.Cookies ("Acc")="AccountHolder" then
strId=Request.Cookies ("uid")
Response.Write ("Processing your request....")

strSQL="exec AddUser " & strNewUser & "," & strPassword & ","  & strId
objRst.Open strSQL, objConn

Response.Write ("<br> A User Account has been created <br> User Name:" &
strNewUser )
Response.Write ("<br> Password: " & strPassword)
```

```
else
Response.Write ("Sorry, you are not authorised to add users to the
system.<br>Please contact your account holder for further information.")
end if
```

Account Holders are also able to delete user accounts that they have created. The Delete user page allows the Account Holder to do this. The Account holder clicks on a users name and the account is then deleted.

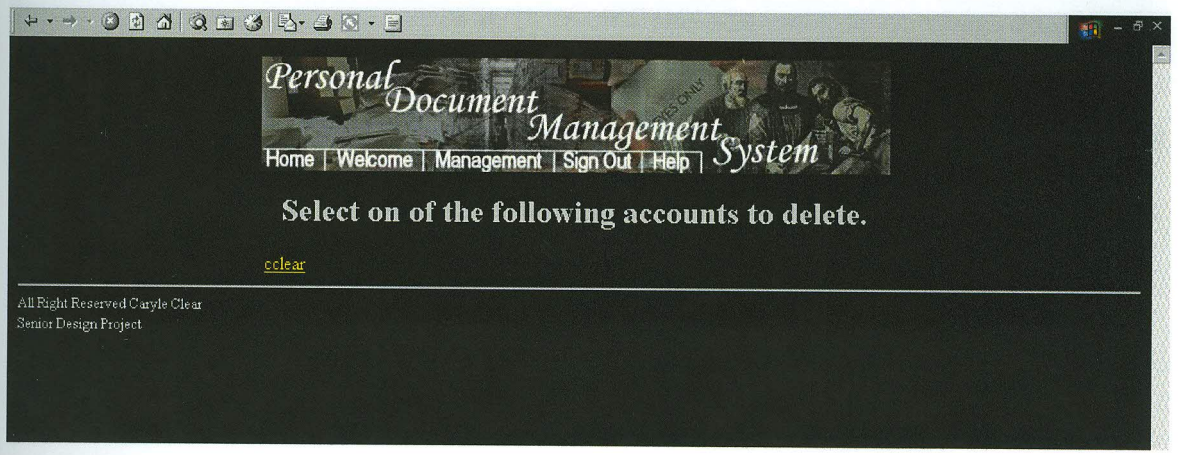


Figure 5.

When the Account Holder clicks the user name a page executes the following code. This code first checks to see if the user is an Account Holder using the cookie. When this check has passed the user account information, it is sent to a stored procedure that finds the users Id and then deletes the entries associated with that user. It also deletes all references in the Permissions table. This table keeps track of all the files that a user has permissions to.

```
if Request.Cookies ("Acc")="AccountHolder" then
        strID=Request.Cookies ("uid")
        strUserID=Request.QueryString ("ID")
        strSQL="exec DeleteUserAccount " & strUserID
        objRst.Open strSQL, objConn
        strSQL="exec DeleteUserAccountData " & strUserId
        objRst.Open strSQL, objConn
```

5.3 Manage File Permissions

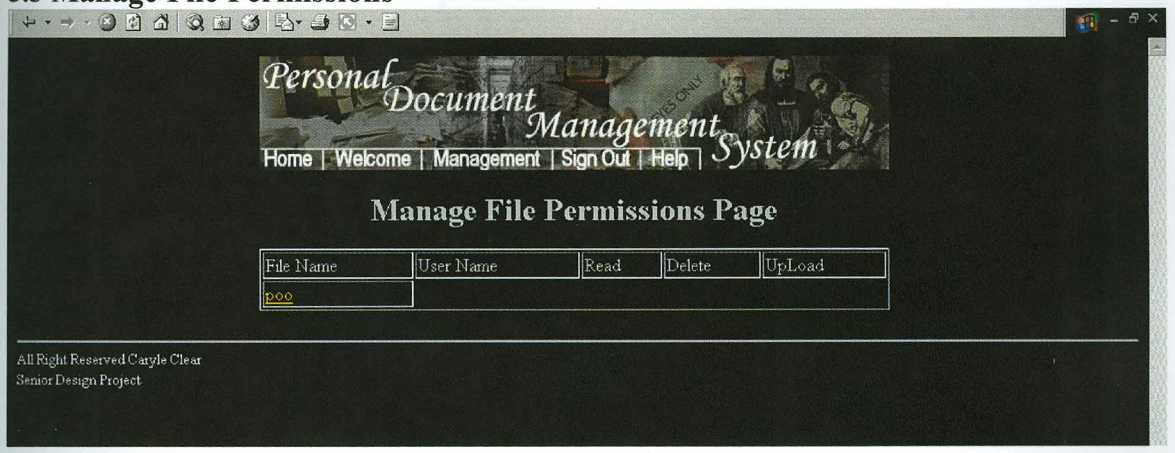


Figure 6.

This page lists all the files that an Account Holder has uploaded to the system and all of the users that have permissions to the files. It also displays the types of permissions that the users have permission to. When an Account Holder wants to add a user to a file they only have to click on the name of the file. This causes a new window to be displayed with all of the users that the Account Holder has. From this window, they can select the user they would like to have permissions to along with the permissions.

The following code creates the table that is shown above.

```
'Build the sql statement to get all of the files information that the Account Holder has
cmd.CommandText ="Select * from DataTable where AccountHolderId=Id"
set rst2=cmd.Execute
'strat a loop to display the files
Do while not rst2.EOF
'select all of the users that have permission to the current file
ShapeCommand="Select * from Permissions where FileId=" & rst2("FileId")
cmd.CommandText =ShapeCommand
'use the onclick to display a new window to add users to the file in question
Response.Write "<tr><td><a href=""permissions.asp"" title=""test""
onclick=window.open('filepermissions.asp?FileId=" & trim(rst2("FileId")) &
"&FileName=" & trim(rst2("FileName")) &
"','Manage_Permissions','toolbar=0,location=1,directories=0,status=0,menubar=0,
scrollbars=1,resizable=1,width=620,height=453')>" & rst2("FileName") &
"</a></td></tr>" '<td> " & rst2("FileId") & "</td></tr>"
```

```
set rst1= cmd.Execute
'an inner loop is used to print all the users of the file and the permissions that they have
Do while Not rst1.EOF
        ShapeCommand="Select * from Users where UserId=" & rst1("UserId")
        cmd.CommandText=ShapeCommand
        set rst3=cmd.Execute
'check the information in the database and depending if the information is TRUE/FALSE then display a 1 or 0
        if rst1("Readfile") then
        strper="&Read=1"
        else
        strper="&Read=0"
        end if
        if rst1("Deletefile")then
        strper=strper & "&Delete=1"
        else
        strper=strper & "&Delete=0"
        end if
        if rst1("Upload") then
        strper=strper & "&Upload=1"
        else
        strper=strper & "&Upload=0"
        end if
'use the onclick properity to display a new window that allows changing of user permissions
        Response.Write "<tr><td></td><td><a href=""permissions.asp""
onclick=window.open('userpermissions.asp?UId=" & trim(rst1("UserId")) &
"&FId=" & trim(rst2("FileId")) & "&Name=" & trim(rst3("UserName")) &
"&File=" & trim(rst2("FileName")) & strper &
"','Manage_Permissions','toolbar=0,location=1,directories=0,status=0,menubar=0,
scrollbars=1,resizable=1,width=620,height=453')>" & rst3("UserName") &
"</a></td>"
        if rst1("Readfile") then
        Response.Write "<td>1</td>"
        else Response.Write "<td> </td>"
        end if
        if rst1("Deletefile") then
        Response.Write "<td>1</td>"
        else Response.Write "<td> </td>"
        end if
        if rst1("Upload") then
        Response.Write "<td>1</td></tr>"
        else Response.Write "<td> </td></tr>"
        end if
```

```
        'Response.Write "Permissions--->" & rst1("Readfile") & " " &
rst1("Deletefile") & " " & rst1("Upload") & "<br>"
        rst1.MoveNext

    loop 'loop for the inner loop (Users that have permissions to the files)

    rst2.MoveNext
    loop 'loop for the outer loop (Files that the Account Holder owns)
```

When an Account Holder clicks on a file to add users to a new browser window appears. This window displays all of the users that can be added to the file and the permissions that can be given to those users.

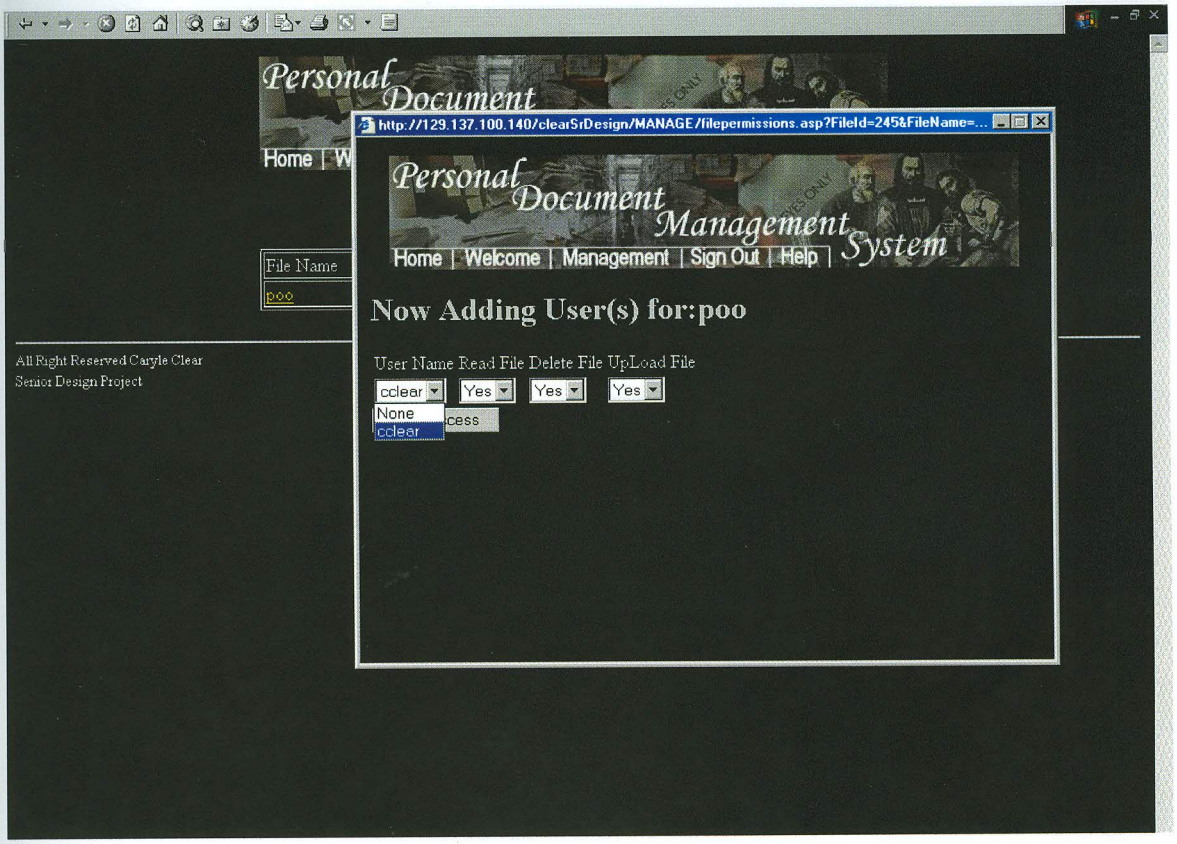


Figure 6.1.

When an Account Holder clicks on a user name a new widow appears and the Account Holder is able to change the permissions for current users.

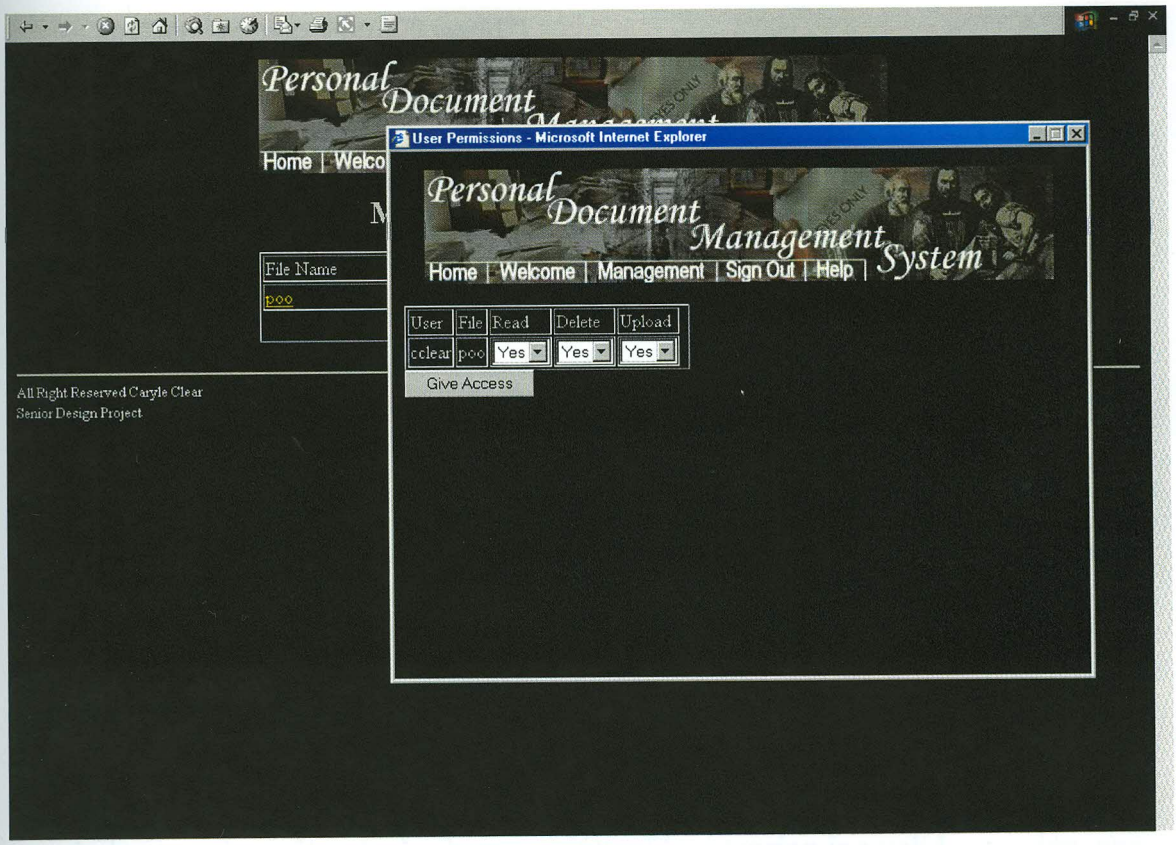


Figure 6.2.

```
'Check to see which option has been selected for each choice
strRead=Request.Form("Read")
if strRead="Yes" then
strRead=1
else
strRead=0
end if

strDel=Request.Form("Delete")
if strDel="Yes" then
strDel=1
else
strDel=0
end if

strUp=Request.Form("UpLoad")
if strUp="Yes" then
strUp=1
```

```
else
strUp=0
end if

strUser=Request.Form("UserId")
strFileId=Request.Form("FileId")
'send information to a stored procedure that updates the users information
based on there Userid
strSQL="exec UpDateUserPermissions " & strUser & "," & strFileId & ","  &
strRead  & "," & strDel & "," & strUp
objRst.Open strSQL, objConn
```

5.4 Upload Files page

When a user wants to upload a file to the system they only have to enter a valid file location and a title for the file. The browse button will display a dialog box that will allow them to choose a file and then place the full path in the text box. The title field is used for display purposes. This title name will be display for the user to refer to. Users will then click on the submit button.

The information is then sent to another page that first checks to see if the user has permission to upload files. It checks to see if the file extension is a valid extension, one that would not open the server up to any security holes. This prevents users from uploading files that the system can be tricked in to executing.

```
Set MyUpload = Server.CreateObject("AspSmartUpLoad.SmartUpLoad")
MyUpLoad.DeniedFilesList = "bat,exe,com,asp,vbs,shs,reg"
```

It then checks to see if a file with the same file name has been uploaded to the system. This is done by searching the database for the same file name. If a file has not been uploaded, it then is uploaded to the system. If it has been uploaded to the system a new file name will be created. The file will have a "1" appended to the end of the file.

This new name will be used to conduct a new search in the database. This process will continue until a file name is found that is not in the database.

```
Do While NOT objRst.EOF and NOT objRst.BOF
objRst.Close
ChangeFile = Left(ChangeFile,Len(ChangeFile)-4) & "1" & Right(ChangeFile,4)
strSQL = "Select * from DataTable where Location = '" & ChangeFile & "'"
Response.Write (".")
objRst.Open strSQL, objConn
Loop
```

The old file will be renamed to the new file name (ex. Test11.txt). Then the new file will be added to the system and the database will be updated. The "Grandfather Table" is also updated to reflect the new and old file names. This table is used to keep track of the revisions of the documents.

```
set objFS= Server.CreateObject ("Scripting.FileSystemObject")
ObjFS.CopyFile OldFileSource,NewFileSource
objFs.DeleteFile OldFileSource,1
MyUpload.Save ("C:\cclearData")
objRst.Close
strSQL="Update DataTable set Location='" & ChangeFile & "' where Location='"
& SystemFileName & "'"

objRst.Open strSQL, objConn
objRst.Close
strSQL="exec UpLoadDataTable '" & DisplayName & "','" &
SystemFileName & "','" & strId & "'"
objRst.Open strSQL, objConn
Response.Write ("<br>The Old File name:" & ChangeFile)
else
objRst.Close
```

The files are placed in a directory that is not directly accessible to users via the Internet.

```
SystemFileName = MyUpLoad.Files.Item("file1").FileName
DisplayName = MyUpLoad.Form.Item("DisplayName").Values
MyUpload.Save ("C:\cclearData") 'save files to this location
```

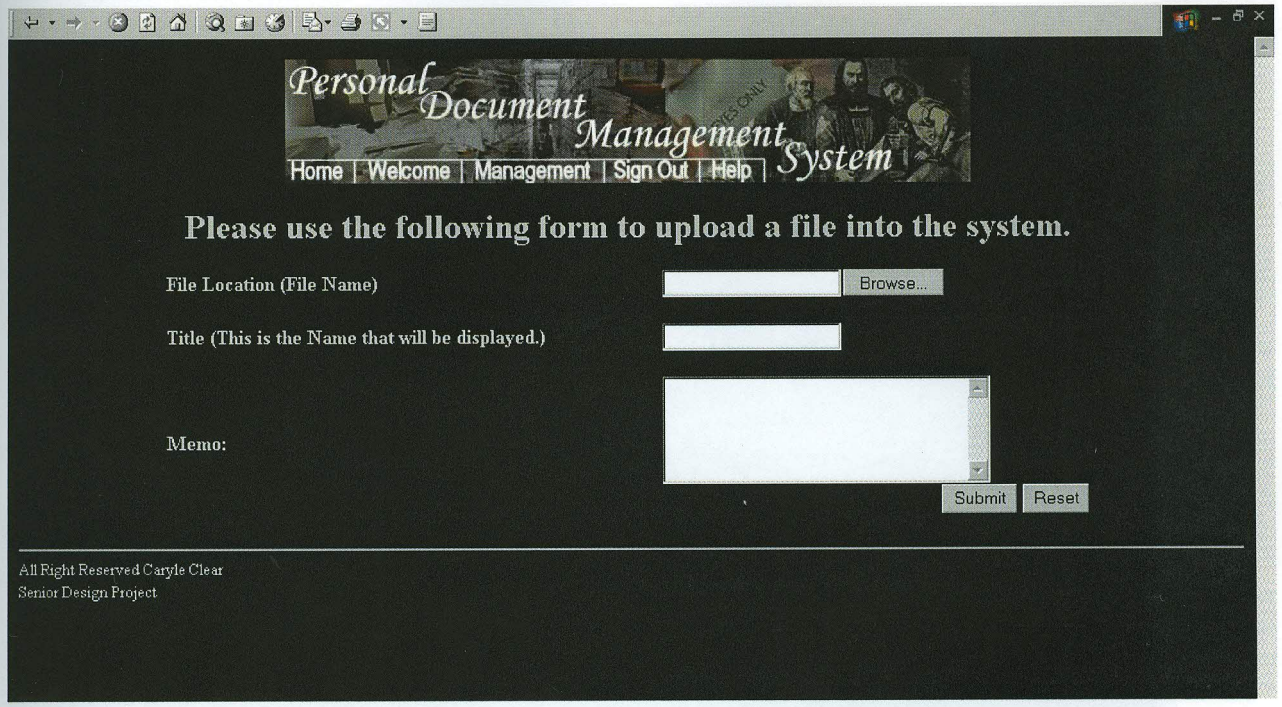


Figure 7.

5.5 Delete Page

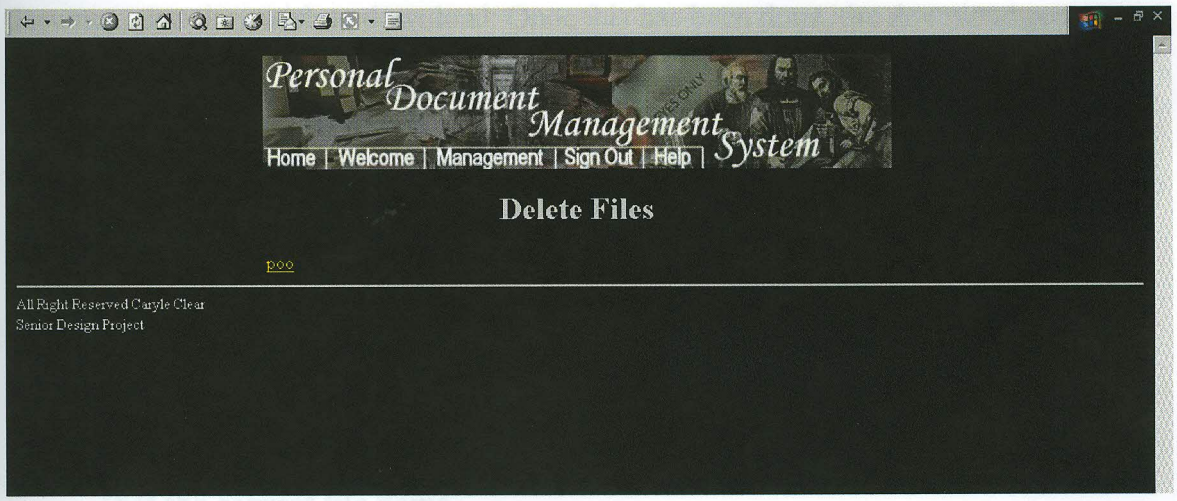


Figure 8.

The deletion page works by clicking on a users name and then the user is deleted.

Clicking on a file name causes another page to load. This page looks up the file

information in the database and then deletes the references in the database. This does not delete any revisions that have been placed in the "Grandfather" table.

5.6 Download (View) Page

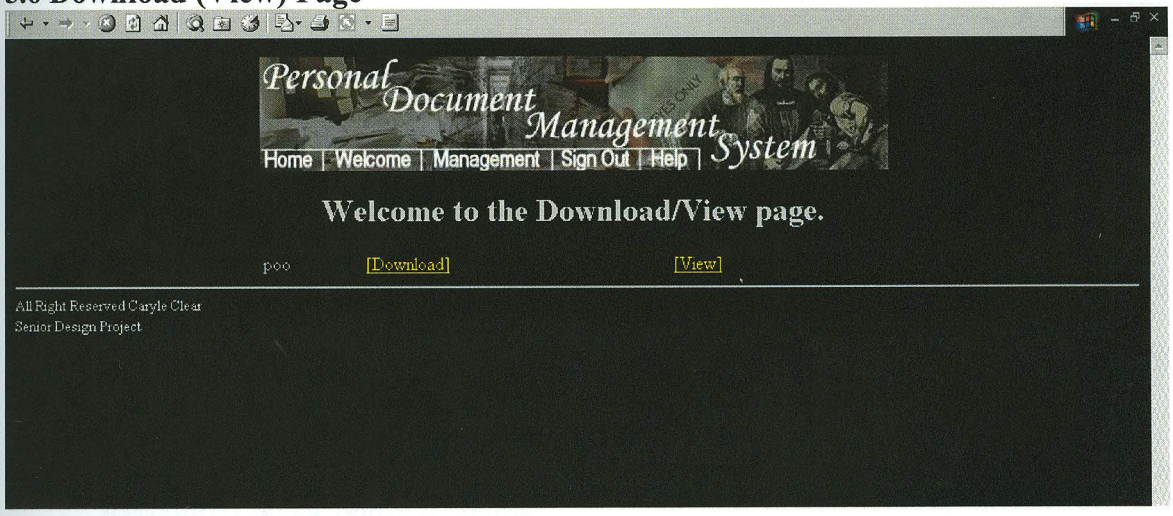


Figure 9.

This page will display all the documents that a user has permission to view and display a list of them with all relevant information. The user will be able to select the document that they wish to download. Once this has been done, the download process will start. There is also an option to display the file in the browser. If the file can be opened by an application instead of the web browser the program will launch and load the file.

Code for viewing inside the browser

```
Set objBinFile = Server.CreateObject("ASPBinFile.clsASPBinFile")

'Set the correct file name
strFileReq="C:\cclearData\" & Request.QueryString("ID")
strfilename=Request.QueryString("ID")
mFile =  strFileReq
mStream = objBinFile.BinFileRead(mFile)
'Set the content type to display the file inside the browser
```

```
Response.AddHeader "Content-Disposition", "inline;Filename=" &
Request.QueryString("ID")
Response.BinaryWrite mStream
Set objBinFile = Nothing
Response.End
```

Code for downloading

```
Set objBinFile = Server.CreateObject("ASPBinFile.clsASPBinFile")

strFileReq="C:\cclearData\" & Request.QueryString("ID")
strfilename=Request.QueryString("ID")
mFile =  strFileReq
mStream = objBinFile.BinFileRead(mFile)


Response.Addheader "Content-Disposition", "attachment; filename=" &
"serverop.doc" & Request.QueryString("ID")
Response.AddHeader "Document", Request.QueryString("ID")
Response.AddHeader "Script_Name", Request.QueryString("ID")
Response.CacheControl = "public"

Response.BinaryWrite mStream
Set objBinFile = Nothing
Response.End
```

5.7 Search

The search page contains an input box that allows a user to enter a word to search for. When the Account Holder enters a word and clicks on the search button all of the results are displayed. All documents, including revisions, are searched. The results that are displayed are click able links that will prompt the user to download the document. This feature is only available for Account Holders.

5.8 Logoff

When the logoff page is display the cookie will have been destroyed. A message will be displayed letting the user know that they have been logged out of the system.

5.9 FAQ/Help

This page has a description of the basic functionality of all the pages in the system. A user can view this page once they have login.

6. Conclusion/ Recommendations

This product does what the original description of the solution stated. There were, however several things that were changed during the course of the project. The user interface has changes slightly on several pages. These changes were enhancements to the navigation and usability of the system. Many of the pages were very crowded and needed to be separated to allow for easier use. The system does allow users to upload and download pages to and from there computers.

During the course of the project functionality was scaled back in order to meet the time scale. This functionality included an email program. This would allow Account Holders to send documents using email to users. It was also to allow the Account Holders to pick a day in the future in which the program would send the email. Account Holders would have been able to have email lists and distribution groups that could be created. This is a function that should be added to the system to allow users to manage there documents better.

Currently, there is an account deletion field in the user table but there is no functionality that uses this information. In addition to the E-mail program it should also delete user accounts that expire on the given day. This executable was to be designed so that it was executed on the server once a day. The time in which it was to run was left the

discretions of the user. When the program was run, it would send all e-mails out and then delete accounts that expired.

The system also needs an administration section to allow for the administrating of the system. This account would allow for the creation of the Account Holders. Currently there is no functionality to do this instead this must be carried out by adding information directly to the database. Having an administrative section would allow for easer administration and eliminate the need to have direct access to the database.

Finally, the system should have a build in system for letting the Account Holder e-mail documents using a web biased front end. This would let the Account Holder select a file that is on the system and then either type in an e-mail address or choose from a group of users. This is mainly to provide more robust functionality to the system itself.

Appendix A.
Selected Active Server Pages

Permissions.asp

```
<!-- #INCLUDE FILE="../loged_in2.asp"-->
<html>
<head>
<TITLE>Manage File(s)</TITLE>
</head>
<body><!-- #INCLUDE FILE="../HeadderSub.asp"-->
<form action="permissionsprocess.asp" id="FORM1" method="post"
name="FORM1">
<h1 align=center>Manage File Permissions Page</h1>
<%
'To do on this page
'1.Display all files belonging to the Account Holder
'2.Display all users that belong to the Account Holder
'3.Display all users that have access to a file and the type of access
'4.Allow giving permissions to a user(s)
'5.Allow looking at file detals

'if Request.Cookies ("acc")="AccountHolder" then
```

```
'strSQL= "exec A_GetData " & Request.Cookies ("uid")
'else
'strSQL= "exec U_DeleteData " & Request.Cookies ("uid")
'end if
'objRst.execute strSQL, objConn,2,2

'   strSQL="exec FileUsers " & objRst.Fields.Item("FileId").Value
'         objRst2.execute strSQL, objConn , 2,2
if Request.Cookies ("Acc")="AccountHolder" then
set conn = server.CreateObject ("ADODB.Connection")
set cmd = server.CreateObject ("ADODB.Command")
set rst1 = server.CreateObject ("ADODB.Recordset")
set rst2= server.CreateObject ("ADODB.Recordset")
Response.Buffer= True

conn.Provider = "MSDataShape"
'dsn= "Driver={SQL
Server};Server=129.137.100.140;Database=ClearSrDesign;UID=cclear;PWD=wi
nterishere;"
'conn.Open "Driver={SQL
Server};Server=129.137.100.140;Database=ClearSrDesign;UID=cclear;PWD=wi
nterishere;"
conn.Open strDSN
cmd.ActiveConnection =conn

Response.Write ShapeCommand
Response.Flush

cmd.CommandText ="Select * from DataTable where AccountHolderId=1"
set rst2=cmd.Execute
if rst2.EOF and rst2.BOF then
         Response.Write ("You have No files in the system.")
end if
Response.Write "<table border=1 align=center width=550>"
Response.Write "<tr>"
Response.Write "<td>File Name</td>"
Response.Write "<td>User Name</td>"
Response.Write "<td>Read</td>"
Response.Write "<td>Delete</td>"
Response.Write "<td>UpLoad</td>"
Response.Write "</tr>"

Do while not rst2.EOF
'ShapeCommand= "SHAPE {select FileId,UserId from Permissions} APPEND
({SELECT FileId from DataTable where FileName='" & rst2("FileName") & "'}
as rstest RELATE FileId TO FileId)"
```

```
ShapeCommand="Select * from Permissions where FileId=" & rst2("FileId")
cmd.CommandText =ShapeCommand
'Response.Write ShapeCommand

Response.Flush
'Response.Write "<BR>FileName:" & rst2("FileName")& "----" & "FileID:" &
rst2("FileId")
Response.Write "<tr><td><a href=""permissions.asp"" title=""" & rst2("Notes")
& """ onclick=window.open(""filepermissions.asp?FileId=" &
trim(rst2("FileId")) & "&FileName=" & replace (trim(rst2("FileName"))," ","+")
&
""","Manage_Permissions"",toolbar=0,location=1,directories=0,status=0,menuba
r=0,scrollbars=1,resizable=1,width=620,height=453)>" & trim(rst2("FileName"))
& "</a></td></tr>" '<td> " & rst2("FileId") & "</td></tr>"

set rst1= cmd.Execute

Do while Not rst1.EOF
        ShapeCommand="Select * from Users where UserId=" & rst1("UserId")
        cmd.CommandText=ShapeCommand
        set rst3=cmd.Execute
        'Response.Write "<BR>--------->" & rst3("UserName")& "<---UserId  "
        if rst1("Readfile") then
        strper="&Read=1"
        else
        strper="&Read=0"
        end if
        if rst1("Deletefile")then
        strper=strper & "&Delete=1"
        else
        strper=strper & "&Delete=0"
        end if
        if rst1("Upload") then
        strper=strper & "&Upload=1"
        else
        strper=strper & "&Upload=0"
        end if
        Response.Write "<tr><td></td><td><a href=""permissions.asp""
onclick=window.open(""userpermissions.asp?UId=" & trim(rst1("UserId")) &
"&FId=" & trim(rst2("FileId")) & "&Name=" & trim(rst3("UserName")) &
"&File=" & replace (trim(rst2("FileName"))," ","+") & strper &
""","Manage_Permissions"",toolbar=0,location=1,directories=0,status=0,menuba
r=0,scrollbars=1,resizable=1,width=620,height=453)>" & rst3("UserName") &
"</a></td>"
        if rst1("Readfile") then
        Response.Write "<td>1</td>"
```

```
            else Response.Write "<td> </td>"
            end if
            if rst1("Deletefile") then
            Response.Write "<td>1</td>"
            else Response.Write "<td> </td>"
            end if
            if rst1("Upload") then
            Response.Write "<td>1</td></tr>"
            else Response.Write "<td> </td></tr>"
            end if
            'Response.Write "Permissions--->" & rst1("Readfile") & " " &
      rst1("Deletefile") & " " & rst1("Upload") & "<br>"
            rst1.MoveNext


      loop

      rst2.MoveNext
      loop

      Response.Write "</table>"
      else

            Response.Write ("<center>You are not permitted to perform this
      opperation.</center>")


      end if
      %>
      </form><!-- #INCLUDE FILE="../Footer.asp"-->
      </body>
      </html>

ViewRevisions.asp
      <%@ Language=VBScript %>
      <!-- #INCLUDE FILE="../loged_in2.asp"-->
      <HTML>
      <HEAD>
      <TITLE>View Revision Information</TITLE>
      </HEAD>
      <BODY>
      <!-- #Include File="../HeadderSub.asp" -->
      <!-- #Include File="../DatabaseInfo.asp"-->
      <h1 align=center>Revision Infromation</h1>
      <TABLE BORDER=0 WIDTH=550 ALIGN=center>
      <TR><TD>
```

```
<%
'ToDo:
'1.Display all of the files in the Grandfather table that belong to
'  the current Account Holder
'2.Order them by the File name where the oldest is the first one.
'  ex. file.txt
'           file1.txt
'           file11.txt

if Request.Cookies ("acc")="AccountHolder" then
Id=Request.Cookies ("uid")
strSQL="select distinct OriginalFile from GrandFather where AccountHolderId="
& Id & " order by OriginalFile"
set rst1 = objConn.Execute (strSQL)
i=0
if rst1.eof and rst1.bof then
        Response.Write ("You have no files in the system.")
end if
do while not rst1.eof
        i=i+1'counter used for sub menus
strSQL= "exec GrandFatherInfo " & Id & ",'" & rst1("OriginalFile") & "'"

        set rest1 = objconn.Execute (strSQL)
        Response.Flush
        Response.Write ("<A CLASS=""Level1"" ID=""OUT" & i & "t""><IMG
SRC=""/vstudio/art/plus.gif"" BORDER=""0"" CLASS=""LEVEL1""
ID=""OUT" & i & "i"">" & trim(rst1("OriginalFile")) & "    <a title=""This will
delete all revisions of this document. Please Use with Care!""
href=""versionsdelete.asp?File=" & replace(trim(rst1("OriginalFile"))," ","+") &
""">Delete</a><br>")
        Response.Write ("<DIV STYLE=""display:none"" ID=""OUT" & i &
"s"">")
        do while not rest1.eof
                Response.Write ("<A
HREF=""../download/downloadprocess.asp?Id=" & trim(rest1("ChildFile")) &
""" CLASS=""Level2"">" & trim(rest1("ChildFile")) & "  ")
                Response.Write ("    " & trim(rest1("TimeStamp")) & "</a><br>")
                rest1.movenext
        Loop
        Response.Write ("</DIV>")
        rst1.MoveNext
Loop
else
'deny them code here
Response.Write ("You dont have permissions to perform this opperation.")
end if
```

```
%>
</TD></TR>
</TABLE>

<script LANGUAGE="JavaScript">
var img1 = new Image();
img1.src = "/vstudio/art/plus.gif";
var img2 = new Image();
img2.src = "/vstudio/art/minus.gif";

function doOutline() {
  var srcId, srcElement, targetElement;
  srcElement = window.event.srcElement;
  if (srcElement.className.toUpperCase() == "LEVEL1" ||
srcElement.className.toUpperCase() == "FAQ") {
                    srcID = srcElement.id.substr(0, srcElement.id.length-1);
                    targetElement = document.all(srcID + "s");
                    srcElement = document.all(srcID + "i");

          if (targetElement.style.display == "none") {
                                    targetElement.style.display = "";
                                    if (srcElement.className == "LEVEL1")
srcElement.src = img2.src;
          } else {
                                    targetElement.style.display = "none";
                                    if (srcElement.className == "LEVEL1")
srcElement.src = img1.src;
     }
  }
}

document.onclick = doOutline;

</script>


<!-- #Include File="../Footer.asp" -->
</BODY>
</HTML>
```

Upload.asp

```
<%@ Language=VBScript %>
```

```
<!-- #INCLUDE FILE="../loged_in2.asp"-->
<!-- #Include File="../DatabaseInfo.asp"-->
<%
Server.ScriptTimeout =300
On Error Resume Next
Dim MyUpLoad
Dim lngNBFile
Dim FileName
Dim DisplayName
Set MyUpload = Server.CreateObject("AspSmartUpLoad.SmartUpLoad")
MyUpLoad.DeniedFilesList = "bat,exe,com,asp,vbs,shs,reg"

MyUpload.Upload
strId=Request.Cookies("uid")
SystemFileName = MyUpLoad.Files.Item("file1").FileName
DisplayName = MyUpLoad.Form.Item("DisplayName").Values
if Request.Cookies ("acc")="AccountHolder" then

  If Err Then
        'Write an Error Message if Needed
                message= "You have Tried to <B>Download</B> a File Type that
is NOT allowed."
                message= message & "<br>Please rename the file and try again."

                MyUpload=NULL
        Else

                SystemFileName = MyUpLoad.Files.Item("file1").FileName
                DisplayName = MyUpLoad.Form.Item("DisplayName").Values
                if (MyUpLoad.Form.Item("Memo").Values = "") then
                        MemoField="No Memo Provided"
                else
                        MemoField=MyUpLoad.Form.Item("Memo").Values
                end if
                strSQL = "Select * from DataTable where Location = '" &
SystemFileName & "'"
                objRst.Open strSQL, objConn
                If NOT objRst.EOF and NOT objRst.BOF Then
                        message= message & "This File is a Duplicate
File<BR>The ""OLD:"" " & SystemFileName & " file will be renamed and the
new fill will be uploaded.<BR>"

                        objRst.Close
                        'Use Do loop to check if the new file name for the revision
is in the
                        'database. If it is change that name and redo loop
```

```
                    '-------------------------------------------------------
                    ChangeFile=SystemFileName
                    'ChangeFile=Left(SystemFileName,Len(SystemFileName)-
4) & "1" & Right(SystemFileName,4)
                    strSQL = "Select * from GrandFather where ChildFile = '"
& ChangeFile & "'"
                    objRst.Open strSQL, objConn
                    message= message & "Now Checking For Revisions."

                    Do While NOT objRst.EOF and NOT objRst.BOF
                    objRst.Close
                    ChangeFile = Left(ChangeFile,Len(ChangeFile)-4) & "1"
& Right(ChangeFile,4)
                    strSQL = "Select * from GrandFather where ChildFile = '"
& ChangeFile & "'"
                    message= message & "."
                    objRst.Open strSQL, objConn
                    Loop

                    '-------------------------------------------------------

                    message= message & "<br>Revision check complete."
                    message= message & "<br>The old file has been
renamed:" & ChangeFile & " and the new file"
                    message= message & "will be saved."



                    OldFileSource="c:\cclearData\" & SystemFileName
                    NewFileSource="c:\cclearData\" & ChangeFile
                    set objFS= Server.CreateObject
("Scripting.FileSystemObject")
                    ObjFS.CopyFile OldFileSource,NewFileSource
                    objFs.DeleteFile OldFileSource,1
                    MyUpload.Save ("C:\cclearData")
                    objRst.Close

                    'grandfather processing
                    '--------------------------------
                    ChildFile=ChangeFile
                    GrandFather=SystemFileName
                    TimeStamp=Now()
                    strSQL="exec GetOriginal '" & strId & "','" &
GrandFather & "'"
                    objRst.Open strSQL, objConn
                    OrginalFile= objRst.Fields ("OriginalFile")
```

```
                    Response.Flush
                    AccountHolderId=strId
                    objRst.close
                    Response.Flush

                    set oNewConn =
Server.CreateObject("ADODB.Connection")
                    oNewConn.Open "Driver={SQL Server};
Server=129.137.100.140; User ID=cclear; PWD=winterishere;
Database=ClearSrDesign"

                    strSQL="Insert Into GrandFather
(ChildFile,GrandFather,TimeStamp,AccountHolderId,OriginalFile) values ('" &
ChildFile & "','" & GrandFather & "','" & TimeStamp & "','" & AccountHolderId
& "','" & OrginalFile & "'" & ")"
                    'strSQL="exec GrandFatherFile '" & ChildFile & "','" &
GrandFather & "','" & TimeStamp & "','" & AccountHolderId & "','" &
OrginalFile & "'"
                    oNewConn.execute strsql
                    oNewConn.Close
                    Response.Flush


                    'get the old fileid for later update of permissions table
                    Response.Flush
                    strSQL="Select * from DataTable where Location='" &
SystemFileName & "'"
                    objRst.Open strSQL, objConn
                    Response.Flush

                    If NOT objRST.Eof Then
                            OldFileId=objRst.Fields("FileId")
                    Else
                            Response.Flush
                    End If
                    objRst.Close
                    Response.Flush
                    strSQL="Delete from DataTable where Location='" &
SystemFileName & "'"
                    objRst.Open strSQL, objConn
                    If objRst.State = 1 Then        'If the recoredset is open

                            objRst.Close
                    End If
```

```
                    '-----------------------------------
                    'end grandfather
                    Response.Flush
                    strSQL="exec UpLoadDataTable """ & DisplayName &
""","""" & SystemFileName & """,""" & strId & """,'" & MemoField & "'"
                objRst.Open strSQL, objConn
                If objRst.State = 1 Then
                        objRst.Close
                    Else
                        'Do nothing at all
                        Response.Flush

                    End If

                    'Get the new FileId
                    Response.Flush
                    strSQL="Select FileId from DataTable where Location='"
& SystemFileName & "'"
                    objRst.Open strSQL, objConn
                    NewFileId=objRst.Fields("FileId")
                    objRst.Close

                    'update the permissions table with the new file ID using
Old Id
                    Response.Flush
                    strSQL="UpDate Permissions set FileId=" & NewFileId &
" where FileId=" & OldFileId
                objRst.Open strSQL, objConn
                if objRst.state = 1 Then
                        objRst.Close
                    Else
                        'Do nothing
                        Response.Flush
                    End if
                    Response.Flush
            else
            objRst.Close
            message= message & "Your file has been uploaded."
            Response.Flush
            SystemFileName = MyUpLoad.Files.Item("file1").FileName
            DisplayName = MyUpLoad.Form.Item("DisplayName").Values

            MyUpload.Save ("C:\cclearData")

            'gf
            ChildFile=SystemFileName
```

```
                GrandFather=SystemFileName
                TimeStamp=Now()
                OrginalFile=SystemFileName
                AccountHolderId=strId
                strSQL="exec GrandFatherFile """ & ChildFile & """,""" &
GrandFather & """,""" & TimeStamp & """,""" & AccountHolderId & """,""" &
OrginalFile & """"


                objRst.Open strSQL, objConn
                'gf
                Response.Flush
                strSQL="exec UpLoadDataTable """ & DisplayName & """,""" &
SystemFileName & """,""" & strId & """," & MemoField & """
                Response.Flush
                objRst.Open strSQL, objConn
                '  Display the number of files uploaded
                ' ************************************
                End IF
  End If

else
'User portion of the upload page.
'Get the files that have been give access to up load.
'-----------------------------
strSQL="Select FileId from DataTable where Location='" &  SystemFileName &
"'"
Response.Flush
set FileId = ObjConn.Execute (strSQL)
strSQL=""
If Not FileId.eof and Not FileId.BOF then
strSQL="Select * from Permissions where FileId=" & FileId("FileId") & "and
UserId=" & Request.Cookies ("uid")
Response.Flush
Set FilePermission= objConn.Execute(strSQL)
strSQL=""
FilePresent=True
else
FilePresent=False
end if

If not FilePresent = False then
        'upload the file
        message= message & "File is bing uplaoded"
        'Upload the file and up date the tables
        '-----------------------------
```

```
AccountHolderId=FilePermission("AccountHolderId")
strId=FilePermission("AccountHolderId")
if (MyUpLoad.Form.Item("Memo").Values = "") then
                MemoField="No Memo Provided"
else
                MemoField=MyUpLoad.Form.Item("Memo").Values
end if

                        ChangeFile=SystemFileName
                'ChangeFile=Left(SystemFileName,Len(SystemFileName)-
4) & "1" & Right(SystemFileName,4)
                strSQL = "Select * from GrandFather where ChildFile = '"
& ChangeFile & "'"
                objRst.Open strSQL, objConn
                message= message & "Checking For and Revisions."

                Do While NOT objRst.EOF and NOT objRst.BOF
                objRst.Close
                ChangeFile = Left(ChangeFile,Len(ChangeFile)-4) & "1"
& Right(ChangeFile,4)
                strSQL = "Select * from GrandFather where ChildFile = '"
& ChangeFile & "'"
                message= message & "."
                objRst.Open strSQL, objConn
                Loop

                '------------------------------------------------------------

                message= message & "<br>Revision check complete."
                message= message & "<br>The old file has been
renamed:" & ChangeFile & " and the new file"
                message= message & "will be saved."



                OldFileSource="c:\cclearData\" & SystemFileName
                NewFileSource="c:\cclearData\" & ChangeFile
                set objFS= Server.CreateObject
("Scripting.FileSystemObject")
                ObjFS.CopyFile OldFileSource,NewFileSource
                objFs.DeleteFile OldFileSource,1
                MyUpload.Save ("C:\cclearData")
                objRst.Close

                'grandfather processing
                '--------------------------------
```

```
                    ChildFile=ChangeFile
                    GrandFather=SystemFileName
                    TimeStamp=Now()
                    strSQL="exec GetOriginal """ & strId & """,""" &
GrandFather & """"
                    objRst.Open strSQL, objConn

                    OrginalFile= objRst.Fields ("OriginalFile")



                    Response.Flush
                    AccountHolderId=strId
                    objRst.close
                    Response.Flush

                    set oNewConn =
Server.CreateObject("ADODB.Connection")
                    oNewConn.Open "Driver={SQL Server};
Server=129.137.100.140; User ID=cclear; PWD=winterishere;
Database=ClearSrDesign"

                    strSQL="Insert Into GrandFather
(ChildFile,GrandFather,TimeStamp,AccountHolderId,OriginalFile) values ('" &
ChildFile & "','" & GrandFather & "','" & TimeStamp & "','" & AccountHolderId
& "','" & OrginalFile & """ & ")"
                    'strSQL="exec GrandFatherFile '" & ChildFile & "','" &
GrandFather & "','" & TimeStamp & "','" & AccountHolderId & "','" &
OrginalFile & """
                    oNewConn.execute strsql
                    oNewConn.Close
                    Response.Flush

                    'get the old fileid for later update of permissions table
                    Response.Flush
                    strSQL="Select * from DataTable where Location='" &
SystemFileName & """
                    objRst.Open strSQL, objConn
                    Response.Flush

                    If NOT objRST.Eof Then
                            OldFileId=objRst.Fields("FileId")
                    Else
                            Response.Flush
                    End If
                    objRst.Close
```

```
                    Response.Flush
                    strSQL="Delete from DataTable where Location='" &
SystemFileName & "'"
                    objRst.Open strSQL, objConn
                    If objRst.State = 1 Then        'If the recoredset is open

                            objRst.Close
                    End If
                    '----------------------------------
                    'end grandfather
                    Response.Flush
                    strSQL="exec UpLoadDataTable '" & DisplayName &
"','" & SystemFileName & "','" & strId & "'," & MemoField & "'"
                objRst.Open strSQL, objConn
                If objRst.State = 1 Then
                            objRst.Close
                    Else
                            'Do nothing at all
                            Response.Flush

                    End If

                    'Get the new FileId
                    Response.Flush
                    strSQL="Select FileId from DataTable where Location='"
& SystemFileName & "'"
                    objRst.Open strSQL, objConn
                    NewFileId=objRst.Fields("FileId")
                    objRst.Close

                    'update the permissions table with the new file ID using
Old Id
                    Response.Flush
                    strSQL="UpDate Permissions set FileId=" & NewFileId &
" where FileId=" & OldFileId
                objRst.Open strSQL, objConn
                if objRst.state = 1 Then
                            objRst.Close
                    Else
                            'Do nothing
                            Response.Flush
                    End if


                    message= message & "<br>Your file has been uploaded."
                    Response.Flush
```

```
else
        message= message & "You do not have the permissions to perform this
operation."
end if

end if
%>
<%
                Response.Write("<HTML>")
                Response.Write("<HEAD>")
                Response.Write("<TITLE>Upload Results</TITLE>")
                Response.Write("</HEAD>")

%>
                <!-- #Include File="../StyleSheet.asp" -->
<%
                Response.Write("<BODY>")
%>
                <!-- #Include File="../HeadderSub.asp" -->
<%
                Response.Write("<br><br><TABLE border=0 align=center
cellPadding=1 cellSpacing=1 width=""75%"">")
                Response.Write("<TR>")
                Response.Write("<TD>")
                Response.Write (message)
                Response.Write ("</TD></TR></table><br><br>")
%>
                <!-- #Include File="../Footer.asp" -->
<%
                Response.Write("</BODY>")
                Response.Write("</HTML>")
%>
```

StyleSheet.asp

```
<!-- #INCLUDE FILE="DatabaseInfo.asp"-->
<%
'Database Connection Objects
Set ObjConn = Server.CreateObject ("ADODB.Connection")
Set ObjCmd = Server.CreateObject ("ADODB.Command")
Set objParam = Server.CreateObject ("ADODB.Parameter")
Set objRst = Server.CreateObject ("ADODB.Recordset")
'open the connection
objConn.Open strDSN
strAccountType=Request.Cookies ("Acc")
```

```
If strAccountType= "AccountHolder" then
strSQL = "select bgcolor,textcolor,linkcolor,Vlink,headding from
AccountHolders where AccountHolderId='" & Request.Cookies ("uid") & "'"
else
strSQL = "select bgcolor,textcolor,linkcolor,Vlink,headding from Users where
UserID='" & Request.Cookies ("uid") & "'"
End IF

objRst.Open strSQL, objConn

%>
<STYLE>
BODY
{
  BACKGROUND-COLOR: <% Response.Write (objRst.Fields ("bgcolor"))%>;
  COLOR: <% Response.Write (objRst.Fields ("textcolor"))%>;
}
H1
{
  COLOR: <% Response.Write (objRst.Fields ("headding"))%>;
  Font-Size:20pt;
}
A:hover
{
  COLOR: <% Response.Write (objRst.Fields ("Vlink")) %>;
}
A:link
{
  COLOR: <% Response.Write (objRst.Fields ("linkcolor")) %>;
}
A:visited
{
  COLOR: <% Response.Write (objRst.Fields ("Vlink")) %>;
}
.LEVEL1 {
        color: <% Response.Write (objRst.Fields ("linkcolor")) %>;
        cursor:hand;
        text-decoration:none;
        font-weight:bold; }

A.LEVEL1:hover, A.LEVEL1a:hover, A.LEVEL2:hover {
        color:<% Response.Write (objRst.Fields ("Vlink")) %>;
        text-decoration:underline; }

A.LEVEL1:visited, A.LEVEL1a:visited, A.LEVEL2:visited {
```

```
        color: <% Response.Write (objRst.Fields ("Vlink"))  %>; }

<%
objRst.Close
%>
</STYLE>
```

References

1. "15 Seconds" . http://15seconds.com ASP and COM

2. "CompUSA" . http://shop.compusa.com/cgi-bin/live/lib/navigation/leftnav.jsp?BV_SessionID=@@@@@0350781281.0959780123@@@@@&BV_EngineID=jalhljhgkdlbgfbmhcgecfeg.0&category=-8158 .

3. "Facts on Paper" . http://www.mead.com/mediallibary/fact.html .

4. Graham, Ian. HTML Sourcebook Third Edition.
Wiley Computer Publishing, 1997

5. Harrington, Spenik, Brumbaugh, and Diamond. Visual basic 5 Interactive Course.
Waite Group Press, 1997

6. "Structural Dynamics Corporation" . http://www.sdrc.com/nav/software-services/metaphase/ Metaphase

7. "The Mead Corporation" . http://www.mead.com pdf fact book

8. Wynkoop, Stephen. Special edition Using Microsoft SQL Server 7.0.
QUE, 1999