Work Order Request Form (WORF)

By

Edward J. Baker

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 2003

Work Order Request Form (WORF)

by

Edward J. Baker

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

Edward J. Baker	Date
Tamisra Sanyal, Faculty Advisor	Date
James F. Sullivan, Department Head	Date

Acknowledgements

I would like to thank Gene Koehl, the manager of Energy Management Systems at Cinergy for giving me the opportunity to help enhance the way work orders and projects are handled. Next, I would like to thank my co-workers at Cinergy for their constructive criticism and help with the testing of the application. Also, I would like to thank the IET faculty and staff for all of their help and input throughout the last year with my senior design project. Finally, I would like to give a special thanks to my wife, Beth Anne for all of her patience and understanding during this project.

Table of Contents

List of Figures

Abstract

The purpose of WORF (Work Order Request Form) is to provide the users of the Energy Management Systems (EMS) Division with an easy way to address and document a problem that needs to be resolved. WORF also provides the administrators with a set of project management tools for tracking and completing the work order. WORF is a Web-based application using ASP.NET for the front end and Microsoft SQL Server 2000 for the database. This application enables the users and the administrators to create their own user accounts, create their own work orders, and to see the status of the work orders. Additionally, the administrators have the ability to adjust any of the tables in the database and generate a special report form for tracking the work orders.

Work Order Request Form (WORF)

1. Statement of the Problem

Electrical utility companies of all sizes tend to lose track of who is doing what in the company. To take an assigned project from start to finish, a department head usually assigns teams of employees with checklists to keep track of the milestones and where they should be at after a certain time has passed (3). If an employee misses a milestone, then the project gets pushed back and the sponsor of the project has to wait longer for the project to finish. The slide of a project can create loss of efficiency, production, and savings.

There are products out on the market like Microsoft Project that keep track of project's progress (9), but what keeps track of all of the work orders and other assignments that tend to cause the project to miss it's completion date? Some employees create tasks or to do lists, others use e-mail, and others try to memorize what they have to do. From my experience, these can work well on a small scale, but on a larger scale, it can produce mass chaos. What happens if the employees fail to write things down or they delete their e-mail or they forget what they have to do? There needs to be a better way to keep track of what is assigned to employees. The potential market for a better way to keep track of things is enormous. This can increase efficiency, production and savings, which in turn makes the overall bottom line look good.

2. Description of the Solution

The solution was to develop a work order request-tracking program for Gene Koehl, Manager of Energy Management Systems (EMS) that was reliable, user-friendly, compatible, and cost effective. This application was called the Work Order Request Form

or “WORF”. The WORF application provides the ability to create work orders and to track them via the Web. This application also provides different report queries for managing the work orders. WORF is a Web based application that uses SQL Server 2000 to host the database because of its security features and VS.Net to link the application.

There are two main components to this application: the Web interface and the SQL database. The Web interface is the user’s link to the SQL database. When a user has a request, an enhancement or a project that needs the assistance by the EMS department hosting the WORF application, the user will fill out the work order request form and submit it. It can then be prioritized and tracked so the solution to the request can be expedited as efficiently as possible.

The WORF application has to be reliable. It should run 24/7 with little technical help and be user-friendly so the users won’t be misled about what to do next. Some of the specifications that made this application reliable and user-friendly are:

- Security will be built into the application to prevent from possible intrusions and possible crashes.
- Multiple users will test the application to resolve programming issues as well as navigation issues throughout the application.
- The multiple users will also assist in making the application easy to use and understand.

In today’s market, compatibility in a product is necessary as well as the overall cost and efficiency. Windows has over 86% of the sales for client side machines and of the other 14%, 5% purchased the MAC (11). By using the Windows platform, the application has the ability to expand and take advantage of the platform’s new features. The factors involved in running the application are:

- Windows NT/2000 server

- Windows 95/98/NT/2000 for the clients with Outlook
- VS.NET (comes with Microsoft SQL Server Desktop Engine)

2.1. User Profiles

The intended users for the Work Order Request Form will be grouped into two levels. The two levels are the Primary Users (Users) and the Secondary Users (Administrators).

The employees who interface with EMS applications are the primary users of WORF. These users are people who are responsible for accounting, energy generation, energy transmission and energy trading. Whether the work order is a request, an enhancement or a project, the users have to submit a request so that the request can be viewed, prioritized and resolved.

The majority of the users have some elementary computer skills. These employees tend to be full time with the exceptions of some co-ops. The users should know the basics of how to use a Web browser and how to navigate a Web site using the navigation buttons.

Next, the administrators are second level users of WORF. These are employees of the EMS group who will be working on the work orders. Administrators are responsible for maintaining, updating, inserting, and deleting information in the WORF database. This mainly includes the information in the Employee and Work Order tables. Some minor maintenance would include the Company, Department, and Assigned To tables. The administrators are full time employees and are considered to be advanced computer users. The administrators will have had some training in database management.

2.2. Design Protocols

Techniques from four areas of IT were used in completing this project. The main emphasis was on creating a Web interface and a SQL database. These were linked together with VB.Net and HTML. The multimedia and networking techniques were used to support the Web interface design and the networking of the clients and server. Below I have listed the different techniques and how they were used in the project.

2.2.1. Web Interface / Programming

A user-friendly Web interface was created using ASP.NET with HTML. The programming executes the processes as well as the different queries between the Web pages and the database.

2.2.2. Microsoft SQL 2000 Database

The database was created with one-to-many relationships and normalized. The database can be accessed by both the Intranet and the Internet, which could pose a possible security issue.

2.2.3. Multimedia / Networking

The multimedia technique guided the design of the interface and navigation of the pages. As far as networking, this application has been installed on an already set up PC with Windows Server 2000, SQL Server 2000, and Internet Information Server (IIS).

3. Deliverables

In order to provide a well-organized project, certain items were considered necessary in the development of this project. During the design phase of this project, the following deliverables were defined:

1. A Web Based Work Order Request application that manages and tracks work orders.

2. The user interface is written in VS.NET and HTML, which allows simple user navigation.

3. The WORF application uses ASPX Pages with embedded VB.NET to communicate between the client and database.

4. Users of the WORF application will be able to complete the following tasks:
   - Create a new account
   - Secure login authenticated by the Microsoft SQL2000 database
   - Create work orders

5. Administrators of the WORF application will be able to complete the following tasks:
   - Secure login authenticated by the Microsoft SQL2000 database
   - Insert, update, or delete a company record
   - Insert, update, or delete a department record
   - Insert, update, or delete an employee record
   - Insert, update, or delete a work order record
   - Run reports on the work orders that group them by work order, employee id, date requested, requested completion date, and priority.

4. Design and Development

The next section describes the project's budget, the timeline, and the hardware and software costs.

4.1. Budget

One of the main concerns of any company is how much is the project going to cost. Figure 1. provides a breakdown of the hardware and software needed to run the application. From the table, the cost of just the hardware and software can be enough of an expense to terminate the project, but since the company standard is Microsoft, there is a minimal expense to set up the application. Therefore, the only purchase or license needed for the project is the VS.NET, which costs around $1000 dollars for the professional version [A].

Hardware	
Item	**Cost**
Client PC with Windows Windows 2000 Server PC	$300 - $3000 $890 - $1800
Software	
Item	**Cost**
VS.NET SQL Server Microsoft Internet Information Server Internet Explorer 6.0	$1000 - $2500 $1300 - $10000 Free Free
Project Total	$3490 - $17300

Figure 1. Cost of the Hardware and Software

4.2. Timeline

The project involved several challenges, learning mistakes, opportunities, and accomplishments. Below in Figure 2. is a breakdown of the timeline for the WORF project.

Task	Start Date	End Date	Complete
Senior Design I			
Research	7/1/02	10/31/02	Yes
Write Proposal	11/1/02	11/14/02	Yes
Present Proposal	12/6/02	12/6/02	Yes
Senior Design II			
Develop / Test Database	12/7/02	1/13/03	Yes
Develop / Test GUI	1/14/03	4/18/03	Yes
Present Prototype	3/13/03	3/13/03	Yes
Senior Design III			
Testing (Group)	3/14/03	4/5/03	Yes
Complete Software	3/14/03	5/18/03	Yes
Compile final Project	5/19/03	5/19/03	Yes
Readme file	5/20/03	5/31/03	Yes
Present Completed Project	6/6/03	6/6/03	Yes

Figure 2. The Timeline for WORF

5. Proof of Design

This section shows in detail how the deliverables were fulfilled and what obstacles were encountered.

5.1. Home (default) Page

Default.htm is the main page of the WORF program. It contains the home page, the user login, create new user, request progress, open a new work order, and the logout link. There are three other minor links which include the Cinergy link, the e-mail Admin, and the about WORF page. Any user has full access to these links (see Figure 3.).

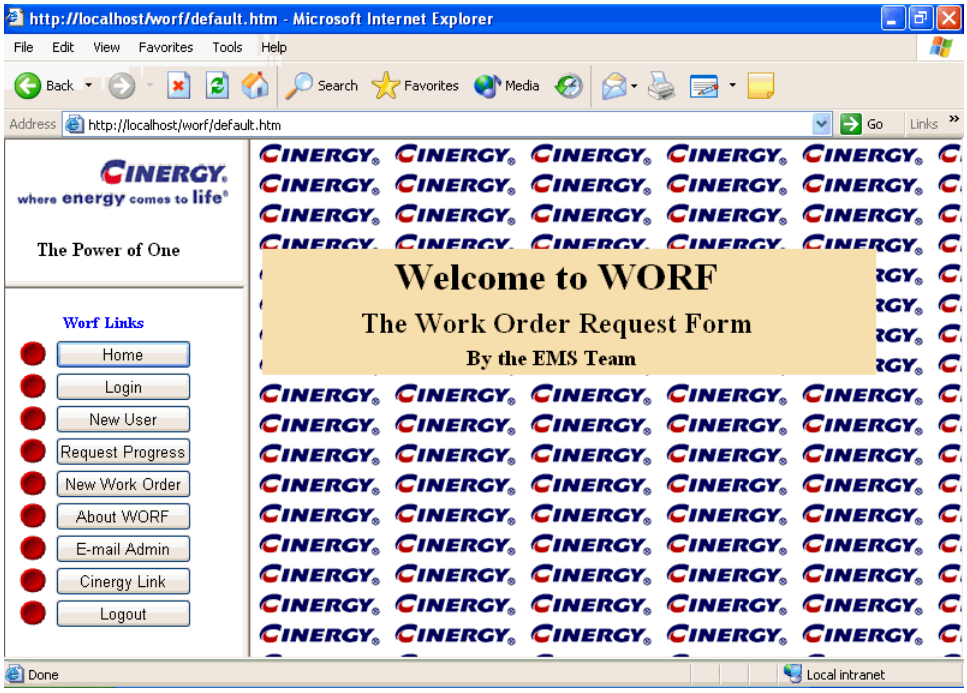

Figure 3. Home Page

5.2. Create New User

If a user has never issued a work order from this application before, the user needs to create a new user account. To create a new user account, the user must fill in the following information:

Employee ID
Mail Drop

First Name
Last Name
Department
E-Mail
Phone
Password

The User ID is issued when the user verifies his Employee ID. The User ID is the same as the Employee ID. The user is then given the options to change the Employee ID after it has been verified or to submit the new user (see Figure 4.).

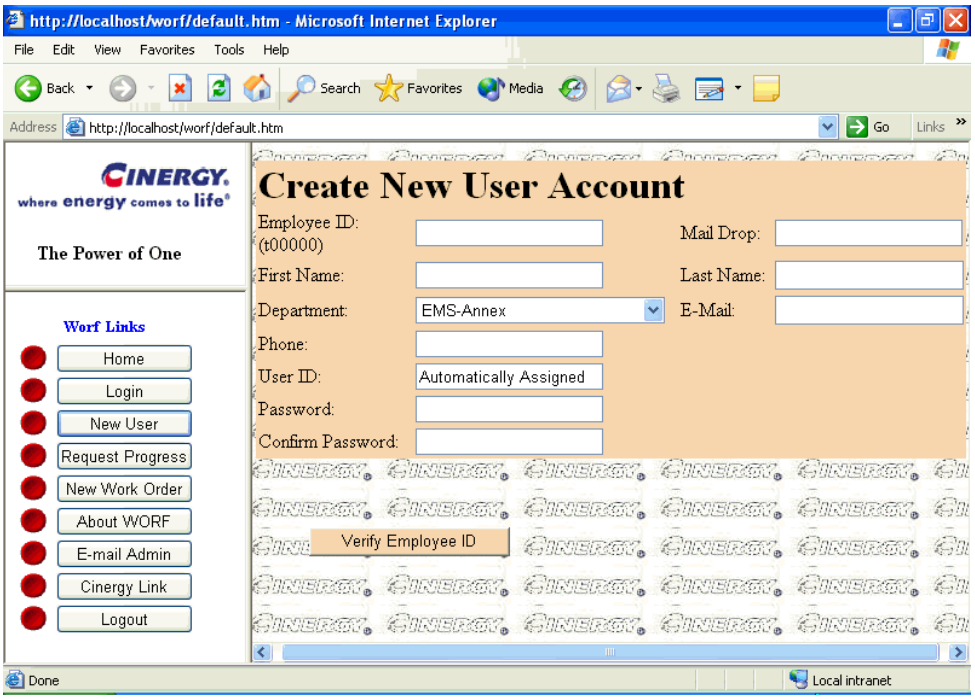

Figure 4. Create New User

5.3. Login

After the user creates an account, the user is then directed to the login page. The Login page asks for the User ID and Password. Once this is submitted, the information is checked against the database to see if the user can proceed. If the User ID or Password is wrong, a message will be displayed on the screen (see Figure 5.).

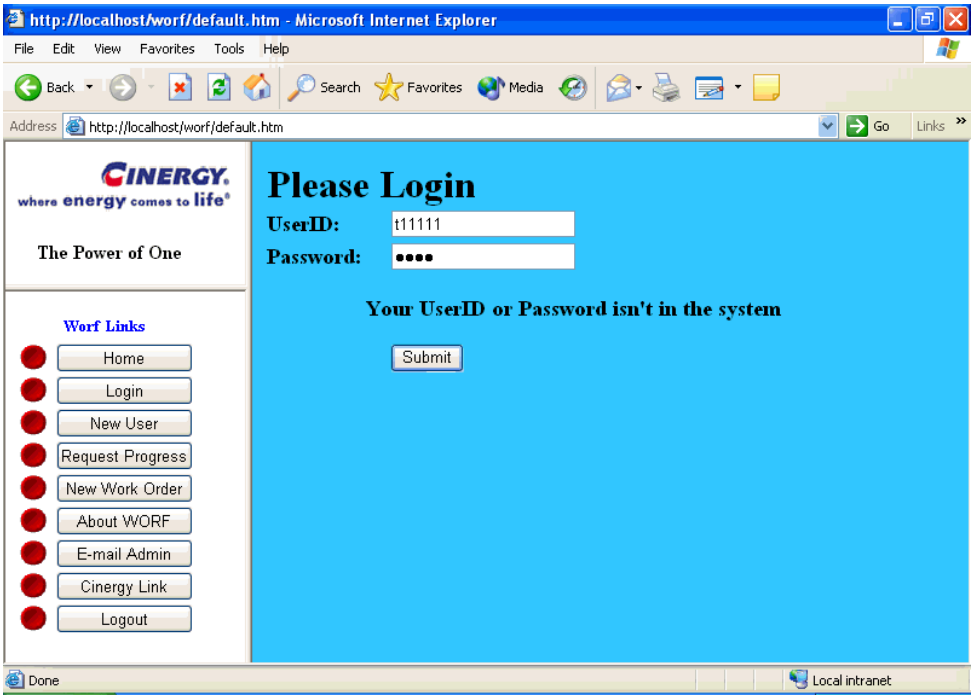

Figure 5. Login

5.4. Work Order Request Form

If the User ID and Password are valid, the user can be directed to the Work Order Request Form or the Admin page. This is dependent on an administrator issuing the Admin privilege to the user (the default is User). When the work order request form comes up, it is populated with the Work Order ID, the Date Requested, and the user's Employee ID. The user fills out the Requested Completion Date, the Brief Description of the problem, and the Description of the problem. Once completed, the work order is submitted and stored in the database (see Figure 6.).

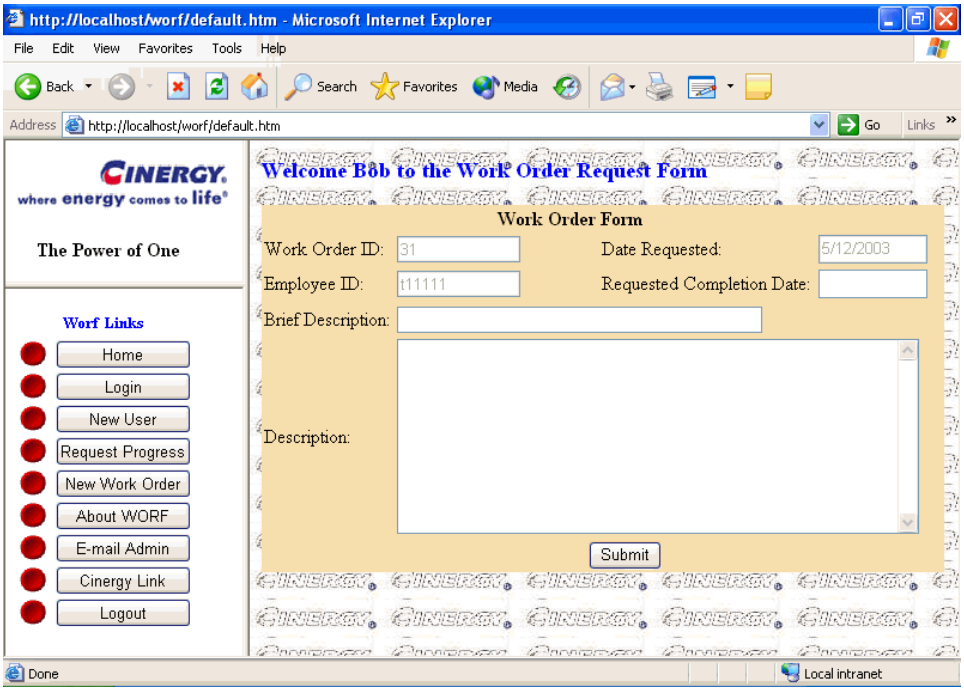

Figure 6. Work Order Form

5.5. Admin Menu

The Admin has the same privileges as the user, plus access to the Admin page. This page provides the administrators with five separate links to perform inserts, updates, and deletes on any of the tables in the database. These tables are: the Employee table, the Work Order table, the Department table, the Company table and the Assigned To table. This gives a lot of freedom to the administrators because they are able to access the database from any PC with the proper connection and security.

For tracking purposes, there is additional information to be filled out by the administrators on the work order form. The additional information includes the priority, the completion status, and whom the work order is assigned to. The completion status is automatically assigned one week from the requested date. See Appendix A for a detailed layout of the Admin Tables.

5.6. Reports

When the administrator selects the reports page from the Admin page, the administrator can choose to see a report of the work orders by Work Order ID, Employee ID, Priority, Date Requested, and Requested Completion Date (see Figure 7.).

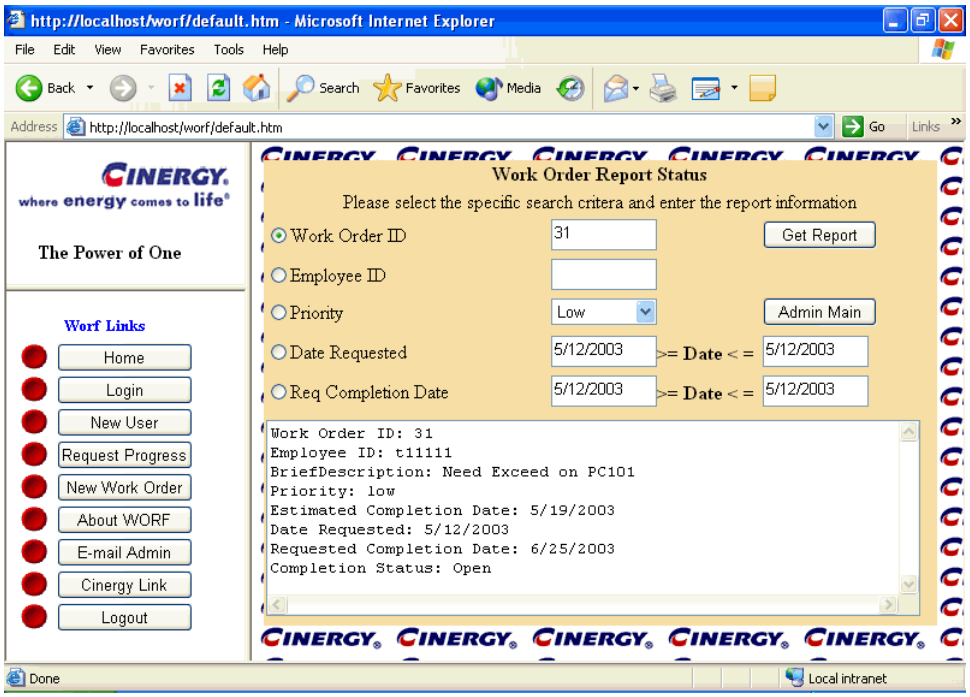

Figure 7. Reports

5.7. Database Design

At a glance, the database design is one-to-many relationships that have been normalized. The one-to-many relationships give this database some added built in security. In order to delete an item besides a single work order, this requires the administrator to delete all of the records associated with the keyed relationships to the record you want to remove. See Appendix B for a detailed layout of the database design.

6. Testing

Modular testing was an integral part of the development process. Each piece of code was tested after every update or modification by the original developer. It's difficult to find every bug or break your own code, so a group of users were used to test the software and comment on what they liked and disliked. Did it have a good look and feel and was it user friendly. This approach allows the application to build character and take on a new look and feel. From this approach, the two main user concerns were the backgrounds and the screen layouts. By talking with the users and getting their feedback, we were able to come to an agreement on how things should be displayed. Finally by integrating all of these ideas together, WORF was transformed.

7. Conclusions and Recommendations

7.1. Conclusions

This project was created in response to Cinergy's EMS department's need for a Web based Work Order Request Form / Project Management tool. I created a reliable, user-friendly site for the users of the EMS department. There are four main sections for the user and they are: the login page, create new user page, request progress page, and the work order request form. These sections are graphically detailed with easy to follow navigation. To prepare the project, I used VB.NET, ASP.NET, Microsoft SQL Server 2000, and Adobe PhotoShop 6.0. The project was completed over the three quarter Senior Design sequence. The budget of approximately $1000 would be the real-world estimate for the completion of this project before labor costs since Cinergy's standard is Microsoft. Finally, the project fulfilled all of the Design Freeze Deliverables and testing was performed to ensure the product's stability and reliability.

7.2. Recommendations

While working on the project, many obstacles were encountered. The main obstacle in the beginning was honing my skill set to the different software and hardware that was chosen. This could take a few weeks to many weeks depending on your familiarity of the subject. One of the biggest hurdles was syntax. Syntax can be tricky when you use FORTRAN, C, VB, VB.NET, and ASP.NET on a frequent basis. Another hurdle was the database. I used Microsoft SQL Server 2000, but at work I use Oracle RDB. These are two totally different databases. RDB is basically edited by a command line where SQL 2000 has a nice user interface. Next, always ask for help or get a second opinion. You can only benefit from this help or opinion. The final obstacles I encountered were the screen layouts and the backgrounds. These were a bit tricky because I started out with a light background, but then I wanted to change it. I had to recreate my screens with a table that could hold the current information. This enabled me to change the table background and the screen background. Now a style sheet can be used to control the backgrounds and fonts.

Appendix A.

Admin Tables

Figure 8. Admin Menu

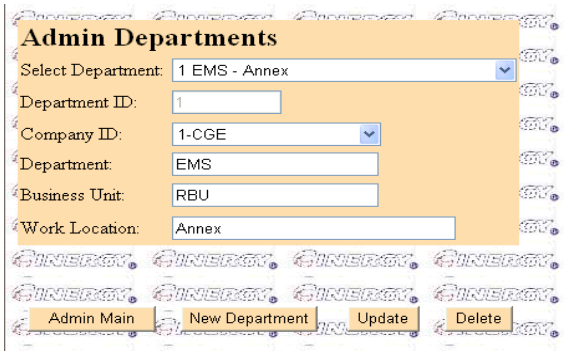

Figure 9. Department Table

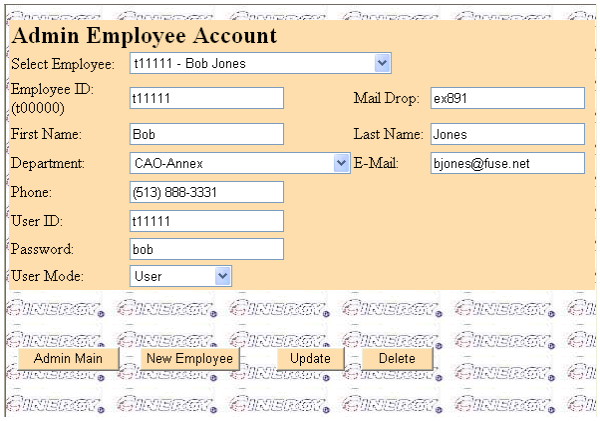

Figure 10. Employee Table

Figure 11. Company Table

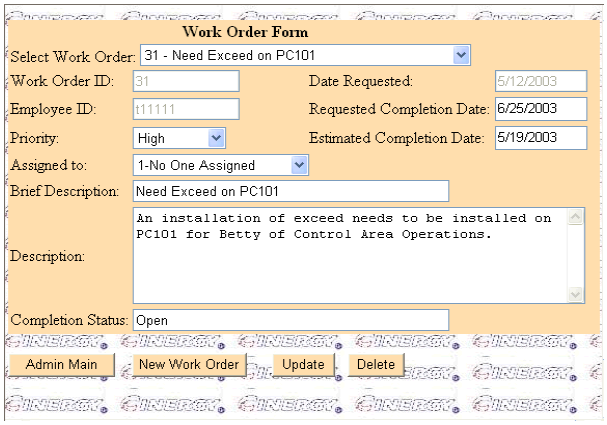

Figure **12. Work Order Table**

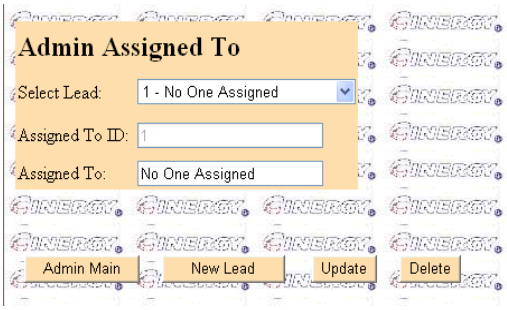

Figure 13. Assigned To Table

Appendix B.

Database Design

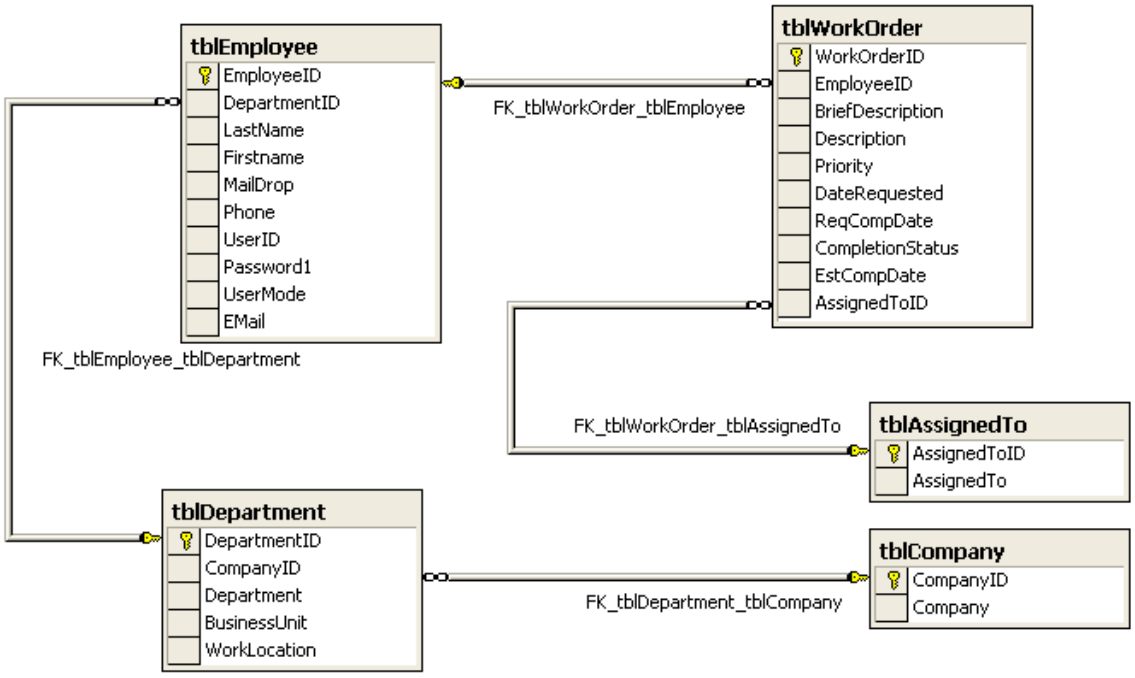

Figure 14. Table Relationships

tblCompany		
Column Name	Data Type	Mandatory
CompanyID	Int (4) (PK)	Yes
Company	Varchar (20)	Yes

Purpose:

This table stores information about companies (divisions) inside the company.

tblAssignedTo		
Column Name	Data Type	Mandatory
AssignedToID	Int (4) (PK)	Yes
AssignedTo	Varchar (50)	No

Purpose:

This table stores the department's employee names so that work orders can be assigned to them.

tblDepartment		
Column Name	Data Type	Mandatory
DepartmentID	Int (4) (PK)	Yes
CompanyID	Int (4) (FK)	Yes
Department	Varchar (30)	No
BusinessUnit	Varchar (30)	No
WorkLocation	Varchar (30)	No

Purpose:

This table stores information about the different departments in the company.

tblEmployee		
Column Name	Data Type	Mandatory
EmployeeID	Varchar (10) (PK)	Yes
DepartmentID	Int (4) (FK)	Yes
LastName	Varchar (25)	No
FirstName	Varchar (25)	No
MailDrop	Varchar (15)	No
Phone	Varchar (14)	No
UserID	Varchar (10)	Yes
Password1	Varchar (15)	Yes
UserMode	Varchar (10)	Yes
EMail	Varchar (50)	No

Purpose:

This table stores information about the employees and the login access.

tblWorkOrder		
Column Name	Data Type	Mandatory
WorkOrderID	Int (4)(PK)	Yes
EmployeeID	Varchar (10)(FK)	Yes
BriefDescription	Varchar (50)	No
Description	Text (16)	No
Priority	Varchar (6)	No
DateRequested	Smalldatetime (4)	No
ReqCompDate	Smalldatetime (4)	No
CompletionStatus	Varchar (100)	No
EstCompDate	Smalldatetime (4)	No
AssignToID	Int (4)	Yes

Purpose:

This table stores the work order information and the different tracking criteria.

Appendix C.

Code Snippets

C 1. Login Page Code Snippet

The Login page is an ASP.NET code snippet. It shows how to connect to the SQL Server 2000 database and check for authentication. Depending on whether you a User or Admin, you are redirected to the proper page by what value the session variable contains.

```
Dim Constring As String = "data source=127.0.0.1;initial
catalog=WORF;integrated security=SSPI;persist security
info=False;workstation id=WATASH;packet size=4096"
    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'Put user code to initialize the page here
    End Sub

    Private Sub cmdSubmit_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles cmdSubmit.Click
        Dim strUserID As String
        Dim strPassword1 As String
        Dim UserMode As String
        Dim strSQL As String

        strUserID = txtUserID.Text
        strPassword1 = txtPassword1.Text

                strSQL = "Select * from tblEmployee where UserID='" &
txtUserID.Text & "'"

        Try
            Dim DBConn As New SqlClient.SqlConnection(Constring)
            Dim myCommand As New SqlDataAdapter(strSQL, DBConn)
            Dim ds As DataSet = New DataSet()
            myCommand.Fill(ds, "tblEmployee")

            'check for no user id or password
            If strUserID = "" Or strPassword1 = "" Then
                lblMessage.Text = ("You didn't enter a User ID or
Password")

                'check to see if row count is somthing other than one
            ElseIf (ds.Tables("tblEmployee").Rows.Count <> 1) Then
                                lblMessage.Text = ("Your UserID or
Password isn't in the system")


            ElseIf
Convert.ToString(ds.Tables("tblEmployee").Rows(0)("Password1")) =
strPassword1 Then
```

```
                UserMode = ds.Tables("tblEmployee").Rows(0)("UserMode")
                Session("First") =
ds.Tables("tblEmployee").Rows(0)("FirstName")
                Session("Last") =
ds.Tables("tblEmployee").Rows(0)("LastName")

                If UserMode = "Admin" Then
                    'Response.Write("Admin")
                    Session("UserMode") = "Admin"
                    Session("UserID") = strUserID
                    Response.Redirect("../Admin/Admin.aspx")

                Else
                    'Response.Write("User")
                    Session("UserMode") = "User"
                    Session("UserID") = strUserID
                    Response.Redirect("../User/User.aspx")

                End If

            Else

                lblMessage.Text = ("Your UserID or Password isn't in
the system")
            End If

        Catch obj As Exception
            'Response.Write(obj.ToString)
            'Response.Write(obj.Message)
            lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
        End Try

    End Sub
```

C 2. User Work Order Request Form

The User Work Order Request Form is an ASP.NET code snippet. It shows how to connect to the database. It also shows how to grab the user session id, how to grab the date and add 7 days to it, how to create a new work order number, and how to insert the information into the database.

```
Dim Constring As String = "data source=127.0.0.1;initial
catalog=WORF;integrated security=SSPI;persist security
info=False;workstation id=WATASH;packet size=4096"
    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        If (Session("UserID") = "" Or Session("UserMode") = "") Then
            Response.Redirect("../Login/Login.aspx")
        Else
```

```
            Response.Write("<h3><strong><FONT color=blue>Welcome " &
Session("First") & " to the Work Order Request Form
</font></strong></h3>")
        End If

        'Put user code to initialize the page here
        txtEmployeeID.Text = Session("UserID")
        txtDateRequested.Text = DateTime.Today
        txtEstCompDate.Text = DateTime.Today.AddDays(7)
        'Response.Write(txtEstCompDate.Text)

        Try
            Dim strSQL As String
            Dim numRows As Integer
            strSQL = "Select * from tblWorkOrder"
            Dim DBConn As New SqlClient.SqlConnection(Constring)
            Dim myCommand As New SqlDataAdapter(strSQL, DBConn)
            Dim ds As DataSet = New DataSet()
            myCommand.Fill(ds, "tblWorkOrder")
            numRows = ds.Tables("tblWorkOrder").Rows.Count
            ' getting the workorder ID number
            txtWorkOrderID.Text =
(ds.Tables("tblWorkOrder").Rows(numRows - 1)("WorkOrderID")) + 1

        Catch obj As Exception
            lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
        End Try

    End Sub

    Private Sub cmdSubmit_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles cmdSubmit.Click
        Dim WorkOrderNum As Integer
        WorkOrderNum = Convert.ToInt32(txtWorkOrderID.Text)

        If (Session("UserID") = "" Or Session("UserMode") = "") Then
            Response.Redirect("../Login/Login.aspx")
        End If

        If (IsDate(txtReqCompDate.Text) = True) Then
            Try
                Dim adoConn As New SqlClient.SqlConnection(Constring)
                Dim adoCommand As New SqlClient.SqlCommand()
                adoConn.Open()

                With adoCommand
                    .CommandText = "INSERT INTO tblWorkOrder
(WorkOrderID, EmployeeID, BriefDescription, Description, DateRequested,
ReqCompDate, EstCompDate )" & _

"VALUES('" & WorkOrderNum & "','" & _
 txtEmployeeID.Text & "','" & _
 Replace(txtBriefDescription.Text, "'", "''") & "','" & _

Replace(txtDescription.Text, "'", "''") & "','" & _
txtDateRequested.Text & "','" & _
```

```
txtReqCompDate.Text & "','" & _
txtEstCompDate.Text & "')"
                    .CommandType = CommandType.Text
                    .Connection = adoConn
                    .ExecuteNonQuery()
                End With
                adoCommand.Dispose()
                adoConn.Close()
                adoConn.Dispose()

                Response.Redirect("../main.aspx")

            Catch obj As Exception

                lblMessage.Text = (obj.Message & "<br>" & "Call your
EMS administrator")
            End Try

        Else
            lblMessage.Text = "Please enter a date in the form
mm/dd/yyyy => 10/21/2003"
        End If

    End Sub
```

C 3. The Admin Employee Page

The Admin Employee Page is an ASP.NET code snippet. This page enables the Admin to select, insert, update, and delete any of the data in the Employee Table. The employee select dropdown box is dynamic. It uses session variables and index values to obtain the textbox values and also to manipulate the dropdown boxes. It also shows how to use Page.IsPostBack and session flags for the dropdown boxes.

```
Dim Constring As String = "data source=127.0.0.1;initial
catalog=WORF;integrated security=SSPI;persist security
info=False;workstation id=WATASH;packet size=4096"

    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

     If (Session("UserID") = "" Or Session("UserMode") <> "Admin") Then
            Response.Redirect("../Login/Login.aspx")
        End If

        lblMessage.Text = ""
        Dim strSQL As String
        Dim strSQL2 As String
        Dim count As Integer
        Dim count2 As Integer
        strSQL = "Select * from tblEmployee"
```

```
        strSQL2 = "Select * from tblDepartment"
        Try
            If Session("AdminEmployeeFlag") <> 1 Then
' fill drop down boxes only once
                Dim DBConn As New SqlClient.SqlConnection(Constring)
                Dim myCommand As New SqlDataAdapter(strSQL, DBConn)
                Dim ds As DataSet = New DataSet()
                myCommand.Fill(ds, "tblEmployee")
' Fill data set with employee table
                count = ds.Tables("tblEmployee").Rows.Count
                Dim DBConn2 As New SqlClient.SqlConnection(Constring)
                Dim myCommand2 As New SqlDataAdapter(strSQL2, DBConn2)
                Dim ds2 As DataSet = New DataSet()
                myCommand2.Fill(ds2, "tblDepartment")
' Fill data set with department table
                count2 = ds2.Tables("tblDepartment").Rows.Count

                Dim i As Integer
                For i = 0 To count - 1
'Populating Drop Down Account Box  Employee ID  FirstName  Last Name
DropDownAccount.Items.Add(ds.Tables("tblEmployee").Rows(i)("EmployeeID"
) & " - " & ds.Tables("tblEmployee").Rows(i)("FirstName") & " " &
ds.Tables("tblEmployee").Rows(i)("LastName"))
                Next i

                Dim j As Integer
                For j = 0 To count2
'Populating Drop Down Department Box Department Work Location
DropDownDept.Items.Add(ds2.Tables("tblDepartment").Rows(j)("Department"
) & "-" & ds2.Tables("tblDepartment").Rows(j)("WorkLocation"))
                Next j
                Session("AdminEmployeeFlag") = 1
            End If
        Catch obj As Exception
            lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
        End Try

    End Sub

    Private Sub cmdInsert_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles cmdInsert.Click
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If

        Dim intDepartment As Integer
        Dim valMode As String
        valMode = DropDownMode.SelectedItem.Value
        intDepartment = DropDownDept.SelectedIndex

        Try
            If (txtPassword.Text <> "" And txtEmployeeID.Text =
txtUserID.Text) Then
                Dim adoConn As New SqlClient.SqlConnection(Constring)
                Dim adoCommand As New SqlClient.SqlCommand()
```

```
                adoConn.Open()

                With adoCommand
                    .CommandText = "INSERT INTO tblEmployee
(EmployeeID, DepartmentID, LastName, FirstName, MailDrop, Phone," & _
                                                                "UserID,
Password1, UserMode, EMail) VALUES('" & txtEmployeeID.Text & "'," & _
intDepartment + 1 & ",'" & _
txtLastName.Text & "','" & _
txtFirstName.Text & "','" & _
txtMailDrop.Text & "','" & _
txtPhone.Text & "','" & _
txtUserID.Text & "','" & _
txtPassword.Text & "','" & _
valMode & "','" & _
txtEMail.Text & "')"
                    .CommandType = CommandType.Text
                    .Connection = adoConn
                    .ExecuteNonQuery()
                End With
                adoCommand.Dispose()
                adoConn.Close()
                adoConn.Dispose()

                Dim strSQL2 As String
                Dim count2 As Integer
                strSQL2 = "Select * from tblEmployee"
                DropDownAccount.Items.Clear()

                Dim DBConn2 As New SqlClient.SqlConnection(Constring)
                Dim myCommand2 As New SqlDataAdapter(strSQL2, DBConn2)
                Dim ds2 As DataSet = New DataSet()
                myCommand2.Fill(ds2, "tblEmployee")
' Fill data set with employee table
                count2 = ds2.Tables("tblEmployee").Rows.Count
                Dim i As Integer
                For i = 0 To count2 - 1
'Populating Drop Down Account Box  Employee ID  FirstName Last Name
DropDownAccount.Items.Add(ds2.Tables("tblEmployee").Rows(i)("EmployeeID
") & " - " & ds2.Tables("tblEmployee").Rows(i)("FirstName") & " " &
ds2.Tables("tblEmployee").Rows(i)("LastName"))
                Next i
                txtEmployeeID.Text = ""
                txtMailDrop.Text = ""
                txtFirstName.Text = ""
                txtLastName.Text = ""
                txtPhone.Text = ""
                txtUserID.Text = ""
                txtPassword.Text = ""
                txtEMail.Text = ""
                lblMessage.Text = ""
                Session("AdminEmployeeFlag") = 1
                cmdInsert.Visible = False
                cmdUpdate.Visible = True
                cmdDelete.Visible = True

            Else
```

```
                lblMessage.Text = "Please make EmployeeID equal UserID
or add password (no blanks)."
            End If

        Catch obj As Exception
            lblMessage.Text = (obj.Message & "<br>" & "Try creating a
unique ID or Call your EMS administrator")
        End Try
    End Sub

    Private Sub cmdUpdate_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles cmdUpdate.Click
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If

        Dim intDepartment As Integer
        Dim valMode As String
        valMode = DropDownMode.SelectedItem.Value
        intDepartment = DropDownDept.SelectedIndex

        Dim strSQL As String

        Try
            If (txtPassword.Text <> "" And txtEmployeeID.Text =
txtUserID.Text And txtEmployeeID.Text = Session("EmployeeID")) Then
                Dim adoConn As New SqlClient.SqlConnection(Constring)
                Dim adoCommand As New SqlClient.SqlCommand()
                adoConn.Open()

                With adoCommand
                    .CommandText = "UPDATE tblEmployee Set EmployeeID =
'" & txtEmployeeID.Text & _
"', DepartmentID = " & intDepartment + 1 & _
", LastName = '" & txtLastName.Text & _
"', FirstName = '" & txtFirstName.Text & _
"', MailDrop = '" & txtMailDrop.Text & _
"', Phone = '" & txtPhone.Text & _
"', UserID = '" & txtUserID.Text & _
"', Password1 = '" & txtPassword.Text & _
"', UserMode = '" & valMode & _
"', EMail = '" & txtEMail.Text & _
"' where EmployeeID = '" & Session("EmployeeID") & "'"
                    .CommandType = CommandType.Text
                    .Connection = adoConn
                    .ExecuteNonQuery()
                End With
                adoCommand.Dispose()
                adoConn.Close()
                adoConn.Dispose()

                Dim strSQL2 As String
                Dim count2 As Integer
                strSQL2 = "Select * from tblEmployee"
                DropDownAccount.Items.Clear()
```

```
                Dim DBConn2 As New SqlClient.SqlConnection(Constring)
                Dim myCommand2 As New SqlDataAdapter(strSQL2, DBConn2)
                Dim ds2 As DataSet = New DataSet()
                myCommand2.Fill(ds2, "tblEmployee")
' Fill data set with employee table
                count2 = ds2.Tables("tblEmployee").Rows.Count
                Dim i As Integer
                For i = 0 To count2 - 1
'Populating Drop Down Account Box Employee ID FirstName  Last Name
DropDownAccount.Items.Add(ds2.Tables("tblEmployee").Rows(i)("EmployeeID
") & " - " & ds2.Tables("tblEmployee").Rows(i)("FirstName") & " " &
ds2.Tables("tblEmployee").Rows(i)("LastName"))
                Next i
                Session("AdminEmployeeFlag") = 1
                DropDownAccount.SelectedIndex =
Session("indexvalAccount")

            Else
                lblMessage.Text = "EmployeeID = UserID (don't change) /
Do an insert to create a new ID."
            End If
        Catch obj As Exception

            lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
        End Try
    End Sub

    Private Sub cmdDelete_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles cmdDelete.Click
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If

        Try
            Dim adoConn As New SqlClient.SqlConnection(Constring)
            Dim adoCommand As New SqlClient.SqlCommand()
            adoConn.Open()

            With adoCommand
                .CommandText = "Delete from tblEmployee where
EmployeeID = '" & Session("EmployeeID") & "'"
                .CommandType = CommandType.Text
                .Connection = adoConn
                .ExecuteNonQuery()
            End With
            adoCommand.Dispose()
            adoConn.Close()
            adoConn.Dispose()

            Dim strSQL2 As String
            Dim count2 As Integer
            strSQL2 = "Select * from tblEmployee"
            DropDownAccount.Items.Clear()

            Dim DBConn2 As New SqlClient.SqlConnection(Constring)
```

```
            Dim myCommand2 As New SqlDataAdapter(strSQL2, DBConn2)
            Dim ds2 As DataSet = New DataSet()
            myCommand2.Fill(ds2, "tblEmployee")
' Fill data set with employee table
            count2 = ds2.Tables("tblEmployee").Rows.Count
            Dim i As Integer
            For i = 0 To count2 - 1
'Populating Drop Down Account Box Employee ID FirstName  Last Name
DropDownAccount.Items.Add(ds2.Tables("tblEmployee").Rows(i)("EmployeeID
") & " - " & ds2.Tables("tblEmployee").Rows(i)("FirstName") & " " &
ds2.Tables("tblEmployee").Rows(i)("LastName"))
            Next i
            txtEmployeeID.Text = ""
            txtMailDrop.Text = ""
            txtFirstName.Text = ""
            txtLastName.Text = ""
            txtPhone.Text = ""
            txtUserID.Text = ""
            txtPassword.Text = ""
            txtEMail.Text = ""
            lblMessage.Text = ""
            Session("AdminEmployeeFlag") = 1

        Catch obj As Exception
            lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
        End Try
    End Sub

    Private Sub cmdNewEmployee_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmdNewEmployee.Click
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If

        txtEmployeeID.Text = ""
        txtMailDrop.Text = ""
        txtFirstName.Text = ""
        txtLastName.Text = ""
        txtPhone.Text = ""
        txtUserID.Text = ""
        txtPassword.Text = ""
        txtEMail.Text = ""
        cmdInsert.Visible = True
        cmdUpdate.Visible = False
        cmdDelete.Visible = False
        lblMessage.Text = ""
    End Sub

    Private Sub cmdAdminMain_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles cmdAdminMain.Click
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If
        Response.Redirect("Admin.aspx")
```

```
    End Sub

    Private Sub DropDownAccount_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
DropDownAccount.SelectedIndexChanged
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If

        Dim strSQL As String
        Dim indexVal As Integer
        Dim indexVal2 As Integer
        Dim valMode As String
        Dim indexVal3 As Integer 'for valMode
        strSQL = "Select * from tblEmployee"
        indexVal = DropDownAccount.SelectedIndex
' Get index from selected employee
        Session("indexvalAccount") = indexVal ' used in update

        Try
            Dim DBConn As New SqlClient.SqlConnection(Constring)
            Dim myCommand As New SqlDataAdapter(strSQL, DBConn)
            Dim ds As DataSet = New DataSet()
            myCommand.Fill(ds, "tblEmployee")
' Fill data set with employee table

            txtEmployeeID.Text =
ds.Tables("tblEmployee").Rows(indexVal)("EmployeeID")
' You need this session varible to update your IDs in update
' This is because you lost the original ID when you modified the new ID
            Session("EmployeeID") = txtEmployeeID.Text
            txtMailDrop.Text =
ds.Tables("tblEmployee").Rows(indexVal)("MailDrop")
            txtFirstName.Text =
ds.Tables("tblEmployee").Rows(indexVal)("FirstName")
            txtLastName.Text =
ds.Tables("tblEmployee").Rows(indexVal)("LastName")

'Department - get integer value from employee table and set drop down box
'The -1 is because the values in the drop down start at zero
            indexVal2 =
ds.Tables("tblEmployee").Rows(indexVal)("DepartmentID")
            DropDownDept.SelectedIndex = indexVal2 - 1
            txtPhone.Text =
ds.Tables("tblEmployee").Rows(indexVal)("Phone")
            txtUserID.Text =
ds.Tables("tblEmployee").Rows(indexVal)("UserID")
            txtPassword.Text =
ds.Tables("tblEmployee").Rows(indexVal)("Password1")
            txtEMail.Text =
ds.Tables("tblEmployee").Rows(indexVal)("EMail")
            valMode =
ds.Tables("tblEmployee").Rows(indexVal)("UserMode")
            'Filling UserMode drop down
            If (valMode = "User") Then
                indexVal3 = 0
```

```
            Else
                indexVal3 = 1
            End If
            'Setting drop down boxes
            DropDownMode.SelectedIndex = indexVal3
            cmdInsert.Visible = False
            cmdUpdate.Visible = True
            cmdDelete.Visible = True

        Catch obj As Exception

            lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
        End Try
    End Sub

    Private Sub DropDownAccount_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles DropDownAccount.Load
        If (Session("UserID") = "" Or Session("UserMode") <> "Admin")
Then
            Response.Redirect("../Login/Login.aspx")
        End If

        If Not Page.IsPostBack Then
            Dim strSQL As String
            Dim indexVal As Integer
            Dim indexVal2 As Integer
            Dim valMode As String
            Dim indexVal3 As Integer 'for valMode
            strSQL = "Select * from tblEmployee"
            indexVal = DropDownAccount.SelectedIndex
' Get index from selected employee
            Session("indexvalAccount") = indexVal ' used in update

            Try
                Dim DBConn As New SqlClient.SqlConnection(Constring)
                Dim myCommand As New SqlDataAdapter(strSQL, DBConn)
                Dim ds As DataSet = New DataSet()
                myCommand.Fill(ds, "tblEmployee")
' Fill data set with employee table
                txtEmployeeID.Text =
ds.Tables("tblEmployee").Rows(indexVal)("EmployeeID")
' You need this session varible to update your IDs in update
' This is because you lost the original ID when you modified the new ID
                Session("EmployeeID") = txtEmployeeID.Text
                txtMailDrop.Text =
ds.Tables("tblEmployee").Rows(indexVal)("MailDrop")
                txtFirstName.Text =
ds.Tables("tblEmployee").Rows(indexVal)("FirstName")
                txtLastName.Text =
ds.Tables("tblEmployee").Rows(indexVal)("LastName")
'Department - get integer value from employee table and set drop down box
'The -1 is because the values in the drop down start at zero
                indexVal2 =
ds.Tables("tblEmployee").Rows(indexVal)("DepartmentID")
                DropDownDept.SelectedIndex = indexVal2 - 1
```

```
                txtPhone.Text =
ds.Tables("tblEmployee").Rows(indexVal)("Phone")
                txtUserID.Text =
ds.Tables("tblEmployee").Rows(indexVal)("UserID")
                txtPassword.Text =
ds.Tables("tblEmployee").Rows(indexVal)("Password1")
                txtEMail.Text =
ds.Tables("tblEmployee").Rows(indexVal)("EMail")
                valMode =
ds.Tables("tblEmployee").Rows(indexVal)("UserMode")
                'Filling UserMode drop down
                If (valMode = "User") Then
                    indexVal3 = 0
                Else
                    indexVal3 = 1
                End If
                'Setting drop down boxes
                DropDownMode.SelectedIndex = indexVal3
                cmdInsert.Visible = False
                cmdUpdate.Visible = True
                cmdDelete.Visible = True

            Catch obj As Exception
          lblMessage.Text = (obj.Message & "<br>" & "Call your EMS
administrator")
            End Try
        End If

    End Sub
```

Notes

A. If expenses are an issue, the .NET Framework SDK(software development kit) is a free download (100Mb) which can run the .aspx pages (12). The developer of the application could borrow the VS.NET developer tool to create the application or the developer could use a line editor like notepad. This would drive the cost of the application to a minimum. If you use a line editor like notepad, remember, you don't have all of the bells and whistles of the .NET editor, which make troubleshooting the application easier.

References

1. Baker, Edward. Senior System Analyst, Lockheed Martin.
   Personal interview. October 25, 2002.

2. Capanet's Helper. "TechMail Web Page".
   http://www.capa.net/manuals/TechMail/outlook.htm. October 07, 2002.

3. Frazee, Wayne. "Where Cost Cutting meets Company Expansion
   Web Page". http://www.epinions.com/. November 09, 2002.

4. Geonetta, Sam. Professor, Information Engineering, College of Applied Science.
   Personal interview. October 17, 2002.

5. Jennings, Roger. ".NET Magazine Web Page"
   http://www.fawcette.com/dotnetmag/2002_01/online/online_eprods/
   sql_rjennings01_10/default.asp. November 13, 2002.

6. Martin, Mindy. "Automating Microsoft Outlook 98". Informant Communications
   Group, Inc . August 21, 1998.

7. McMahon, Russ. Assistant Professor, Information Engineering, College of Applied
   Science. Personal interview. October 17, 2002.

8. Microsoft. "MSDN Library Web Page".
   http://msdn.microsoft.com/library/default.asp. October 28, 2002.

9. Microsoft. "Microsoft Project Web Page".
   http://www.microsoft.com/office/project/default.asp . November 09, 2002.

10. Middle Tennessee State University. "Facilities Services
    Web Page". http://www.mtsu.edu/~facserv/. October 27, 2002.

11. Miles, Stephanie. "Linux closing in on Microsoft market share, study says
    Web Page". http://news.com.com/2100-1001-243527.html. November 11, 2002.

12. Payne, Chris. Sams Teach Yourself ASP.NET in 21 Days. Indiana:
    Sam's Publishing, 2001.

13. Prabhakar, Annu. Assistant Professor, Information Engineering, College of Applied
    Science. Personal interview. October 17, 2002.

14. Schlemmer, Bob. Associate Professor, Information Engineering, College of Applied
    Science. Personal interview. October 23, 2002.

15. University of Notre Dame. "Office of Information Technology
    Web Page". http://www.nd.edu/~ndoit/apptech/request.html. October 27, 2002.