

Developing a Survey Application

By Bipin Joshi

LANGUAGES: VB.NET
ASP.NET VERSIONS: 2.0

In Part 2 of this series we developed web form for managing questions, choices and participants. In this part we will make provision for sending survey invitations. Further, we will develop an important web form of the application that displays survey questions and choices to the end user. Finally, we will develop an administrative web form for generating survey statistics.

Sending Survey Invitations

Recollect that in the ManageSurvey.aspx web form we added a LinkButton titled “Send Invitations”. When the administrator clicks on this LinkButton an invitation should be send to all the participants of the survey. Let’s code this functionality now.

Sending emails require a “from” address which we store in the <appSettings> section of the web.config file. Figure 30 shows the relevant markup from the web.config file.

```
<appSettings>
<add key="webmasteremail" value="someone@somedomain.com"/>
</appSettings>
```

Figure 30: Using <appSettings> section to store sender’s email address

We store a key called “webmasteremail” along with its value. We will use this key further while sending email invitations. We also need to configure the SMTP host and its credentials in the web.config file. Figure 31 shows this markup:

```
<system.net>
<mailSettings>
<smtp>
<network host="localhost" defaultCredentials="true"/>
</smtp>
</mailSettings>
</system.net>
```

Figure 31: Configuring SMTP host

The <system.net> section defines the SMTP configuration settings. The <network> tag specifies the SMTP host name or IP address and we set it to “localhost”. The defaultCredentials tag specifies whether to use the default credentials for authentication with the SMTP host.

Now open ManageSurvey.aspx and go in the ItemCommand event handler of the DetailsView. Add the code shown in Figure 32 in the event handler.

```
Protected Sub DetailsView1_ItemCommand(ByVal sender As
Object, ByVal e As
System.Web.UI.WebControls.DetailsViewCommandEventArgs)
Handles DetailsView1.ItemCommand
If e.CommandName = "Invite" Then
Dim cnn As New
SqlConnection(ConfigurationManager.ConnectionStrings("Conne
ctionString").ConnectionString)
Dim cmd As New SqlCommand("select * from surveyparticipants
where surveyid=@id", cnn)
Dim p1 As New SqlParameter("@id",
DetailsView1.SelectedValue)
cmd.Parameters.Add(p1)
Dim da As New SqlDataAdapter
da.SelectCommand = cmd
Dim ds As New DataSet
da.Fill(ds, "participants")
Dim client As New SmtpClient()
For Each row As DataRow In ds.Tables("participants").Rows
client.Send(ConfigurationManager.AppSettings
("webmasteremail"), row("email"),
"Invitation to participate in our survey",
"Please take the survey at the following URL :" & vbCrLf &
http://" & Request.Url.Host & "/survey.aspx?id=" &
DetailsView1.SelectedValue)
Next
Label3.Text = "Invitations sent successfully!"
End If
End Sub
```

Figure 32: Sending survey invitations

Recollect that we have set the CommandName property of the “Send Invitations” LinkButton to Invite. The code checks the CommandName property of DetailsViewCommandEventArgs and if it is “Invite” fetches all the participants for that survey in a DataSet. The code then iterates through all the fetched records and sends an email to each participant. The email contains the URL where the participant can take the survey. In order to send the email we created an instance of System.Net.Mail.SmtpClient class. The SmtpClient class has a method called Send() that takes four parameters – From, Recipients, Subject and Body. The “from” address is retrieved from the <appSettings> section using AppSettings collection of the ConfigurationManager class. The complete URL to Survey.aspx is formed with the help of Request.Url.Host property and currently

selected survey ID. Once all the emails are sent a success message is displayed in a Label control.

Taking Survey

After receiving the invitations the end users are supposed to take the survey at the specified URL. The survey questions and possible choices are displayed on Survey.aspx web form. This web form receives the survey ID as a query string parameter and renders the corresponding questions.

In order to develop Survey.aspx, add a new web form in the root folder of the web site and name it as Survey.aspx. The web form uses a DataList control to display the questions and the choices. Drag and drop a DataList control and an SQL data source control on the web form. Configure the SQL data source control to select all the records from the SurveyQuestions table matching the SurveyID passed via query string. Then set the DataSourceID property of the DataList control to SqlDataSource1. Design the ItemTemplate of DataList as shown in Figure 33.

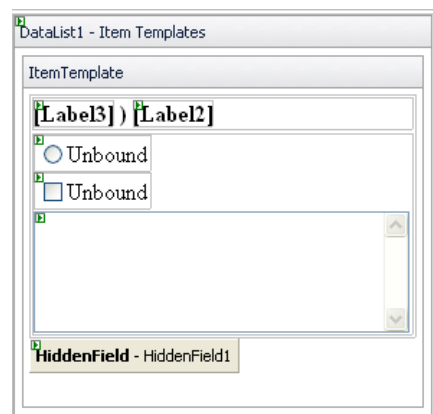


Figure 33: Displaying survey questions and choices

Using “DataBindings” dialog data bind Label3 to display QuestionID and data bind Label2 to display the question. Below the Labels there is an instance of RadioButtonList, CheckBoxList and TextBox respectively. Depending on the AnswerType of the question only one of them will be displayed. Finally, there is a HiddenField that is bound with the AnswerType property. The FooterTemplate of the DataList simply contains a Button titled “Submit”.

Figure 34 shows the complete markup of Survey.aspx.

```
<%@ Page Language="VB" AutoEventWireup="false"
CodeFile="Survey.aspx.vb" Inherits="_Default" %>
```

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
<title>Untitled Page</title>
</head>
<body>
<form id="form1" runat="server">
<div>
<table style="width: 100%">
<tr>
<td>
<asp:HyperLink ID="HyperLink1" runat="server"
ImageUrl="~/Images/logo.gif"></asp:HyperLink></td>
</tr>
<tr>
<td>
<asp:DataList ID="DataList1" runat="server"
DataSourceID="SqlDataSource1" Width="100%">
<ItemTemplate>
<table style="width: 100%">
<tr>
<td style="height: 21px">
<asp:Label ID="Label3" runat="server" Font-Bold="True"
Text='<%# Eval("QuestionID") %>'></asp:Label>
<strong></strong>
<asp:Label ID="Label2" runat="server" Font-Bold="True"
Text='<%# Eval("Question") %>'></asp:Label></td>
</tr>
<tr>
<td>
<asp:RadioButtonList ID="RadioButtonList1" runat="server">
</asp:RadioButtonList>
<asp:CheckBoxList ID="CheckBoxList1" runat="server">
</asp:CheckBoxList>
<asp:TextBox ID="TextBox1" runat="server" Columns="30"
Font-Bold="False" Rows="5" TextMode="MultiLine">
</asp:TextBox>
</td>
</tr>
</table>
<asp:HiddenField ID="HiddenField1" runat="server"
Value='<%# Eval("AnswerType") %>' />
</ItemTemplate>
<HeaderTemplate>

```

```

<asp:Label ID="Label4" runat="server" Font-Bold="True"
Text="Please answer the following survey questions
:"></asp:Label>
</HeaderTemplate>
<FooterTemplate>
<asp:Button ID="Button1" runat="server" Text="Submit"
OnClick="Button1_Click" />
</FooterTemplate>
</asp:DataList></td>
</tr>
<tr>
<td align="center">
<asp:Label ID="Label1" runat="server" Font-Bold="True"
Text="Copyright (C) 2006. All rights
reserved."></asp:Label></td>
</tr>
</table>
</div>
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<%= ConnectionStrings:ConnectionString
%>"
SelectCommand="SELECT * FROM [SurveyQuestions] WHERE
([SurveyID] = @SurveyID)">
<SelectParameters>
<asp:QueryStringParameter Name="SurveyID"
QueryStringField="id" Type="Int32" />
</SelectParameters>
</asp:SqlDataSource>
</form>
</body>
</html>

```

Figure 34: Markup of Survey.aspx

We need to display the choices in a RadioButtonList or CheckBoxList or TextBox depending on the AnswerType of a question. This task is handled in the ItemDataBound event of the DataList. Add the code shown in Figure 35 in the ItemDataBound event handler of the DataList.

```

Protected Sub DataList1_ItemDataBound(ByVal sender As
Object, ByVal e As
System.Web.UI.WebControls.DataListItemEventArgs) Handles
DataList1.ItemDataBound
If e.Item.ItemType = ListItemType.Item Or e.Item.ItemType =
ListItemType.AlternatingItem Then
Dim anstype As HiddenField =
e.Item.FindControl("HiddenField1")
Dim questionid As Label = e.Item.FindControl("Label3")

```

```

Dim rbl As RadioButtonList =
e.Item.FindControl("RadioButtonList1")
Dim cbl As CheckBoxList =
e.Item.FindControl("CheckBoxList1")
Dim txt As TextBox = e.Item.FindControl("TextBox1")
Dim ds As DataSet = GetDataSet(questionid.Text)
Select Case anstype.Value
Case "S"
    rbl.Visible = True
    cbl.Visible = False
    txt.Visible = False
    rbl.DataSource = ds
    rbl.DataTextField = "Choice"
    rbl.DataValueField = "ChoiceID"
    rbl.DataBind()
Case "M"
    rbl.Visible = False
    cbl.Visible = True
    txt.Visible = False
    cbl.DataSource = ds
    cbl.DataTextField = "Choice"
    cbl.DataValueField = "ChoiceID"
    cbl.DataBind()
Case "T"
    rbl.Visible = False
    cbl.Visible = False
    txt.Visible = True
End Select
End If
End Sub

```

Figure 35: Displaying choices

The event handler first check the ItemType property of the item being data bound. If the ItemType is ListItemType.Item or ListItemType.AlternatingItem only then the rest of the code is executed. Inside the “If” condition we get a reference to all the controls of the ItemTemplate (i.e. QuestionID Label, RadioButtonList, CheckBoxList, TextBox and HiddenField) using the FindControl() method. Then we fill a DataSet with all the choices for the current QuestionID using a helper function called GetDataSet (discussed later). Depending on the AnswerType (single choice or multiple choices) this DataSet is bound with the RadioButtonList or CheckBoxList respectively. The remaining controls are kept hidden from the end user.

The GetDataSet() helper function accepts a QuestionID as parameter and returns a DataSet with all the choices for that question. Figure 36 shows the complete code for this function.

```

Private Function GetDataSet(ByVal id As Integer) As DataSet
Dim cnn As New
SqlConnection(ConfigurationManager.ConnectionStrings
("ConnectionString").ConnectionString)
Dim cmd As New SqlCommand("select * from surveychoices
where questionid=@id", cnn)
Dim p1 As New SqlParameter("@id", id)
cmd.Parameters.Add(p1)
Dim da As New SqlDataAdapter
da.SelectCommand = cmd
Dim ds As New DataSet
da.Fill(ds, "choices")
Return ds
End Function

```

Figure 36: The GetDataSet() helper function

Figure 37 shows a sample run of Survey.aspx. Note how RadioButtonList, CheckBoxList and TextBox is displayed depending on the AnswerType.

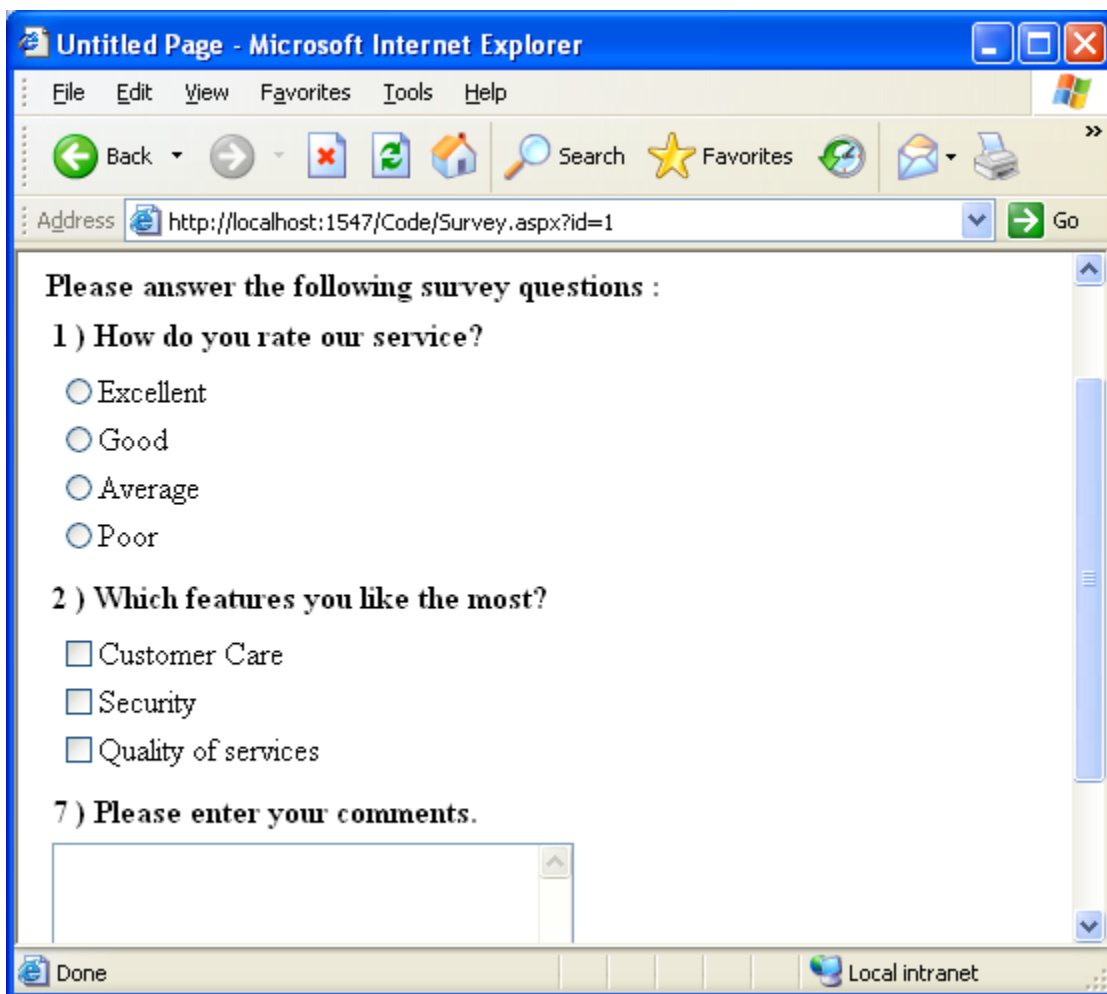


Figure 37: Sample run of Survey.aspx

When the user fills the survey and clicks on the Submit button the answers should be stored in SurveyAnswers table for later reference. This is achieved by handling the Click event of the Submit button. Add the code shown in Figure 38 in the Click event handler of the Submit button.

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As System.EventArgs)
    For Each item As DataListItem In DataList1.Items
        If item.ItemType = ListItemType.Item Or item.ItemType = ListItemType.AlternatingItem Then
            Dim questionid As Integer
            Dim choiceid As Integer = 0
            Dim choicetext As String = ""
            questionid = CType(item.FindControl("Label3"), Label).Text
            Dim anstype As HiddenField = item.FindControl("HiddenField1")
            Select Case anstype.Value
                Case "S"
                    Dim rbl As RadioButtonList = item.FindControl("RadioButtonList1")
                    choiceid = rbl.SelectedValue
                    SaveAnswer(questionid, choiceid, "")
                Case "M"
                    Dim cbl As CheckBoxList = item.FindControl("CheckBoxList1")
                    For i As Integer = 0 To cbl.Items.Count - 1
                        If cbl.Items(i).Selected Then
                            choiceid = cbl.Items(i).Value
                            SaveAnswer(questionid, choiceid, "")
                        End If
                    Next
                Case "T"
                    Dim txt As TextBox = item.FindControl("TextBox1")
                    choicetext = txt.Text
                    SaveAnswer(questionid, 0, choicetext)
            End Select
        End If
    Next
End Sub
```

Figure 38: Saving survey answers

The code iterates through all the Items and AlternatingItems of the DataList. With each iteration QuestionID and AnswerType is retrieved using the FindControl() method as before. If the AnswerType is "S" the SelectedValue of the RadioButtonList is retrieved. This answer is stored in SurveyAnswers table using

a helper function called `SaveAnswer` (discussed later). If the `AnswerType` is “M” the code iterates through the `CheckBoxList` and each selected option is saved in the table using `SaveAnswer` helper function. Finally, if the `AnswerType` is “T” the Text entered in the `TextBox` is saved to the database using `SaveAnswer` helper function.

Figure 39 shows the `SaveAnswer()` function.

```
Private Sub SaveAnswer(ByVal qid As Integer, ByVal cid As Integer, ByVal ct As String)
    Dim cnn As New SqlConnection(ConfigurationManager.ConnectionStrings("ConnectionString").ConnectionString)
    Dim cmd As New SqlCommand("insert into surveyanswers (QuestionID,ChoiceID,ChoiceText) values(@qid,@cid,@ct)", cnn)
    Dim p1 As New SqlParameter("@qid", qid)
    Dim p2 As New SqlParameter("@cid", IIf(cid = 0, DBNull.Value, cid))
    Dim p3 As New SqlParameter("@ct", IIf(ct = "", DBNull.Value, ct))
    cmd.Parameters.Add(p1)
    cmd.Parameters.Add(p2)
    cmd.Parameters.Add(p3)
    cnn.Open()
    cmd.ExecuteNonQuery()
    cnn.Close()
End Sub
```

Figure 39: The `SaveAnswer()` helper function.

The function accepts `QuestionID`, `ChoiceID` and `ChoiceText` as parameters. Inside it prepares an `INSERT` statement with required parameters. Note how the `@cid` and `@ct` parameters are created. If the `cid` method parameter is 0 then the `@cid` parameter inserts `NULL` value in the `ChoiceID` column. Similarly if the `ct` method parameter is empty string then `@ct` parameter inserts `NULL` value in the `ChoiceText` column.

Viewing Survey Statistics

Once the survey campaign has run and participants have submitted their answers the administrator would like to see the statistics of the survey responses. In order to provide such functionality add a new web form in the Admin folder called `SurveyStats.aspx`. Figure 40 shows a sample run of this web form.

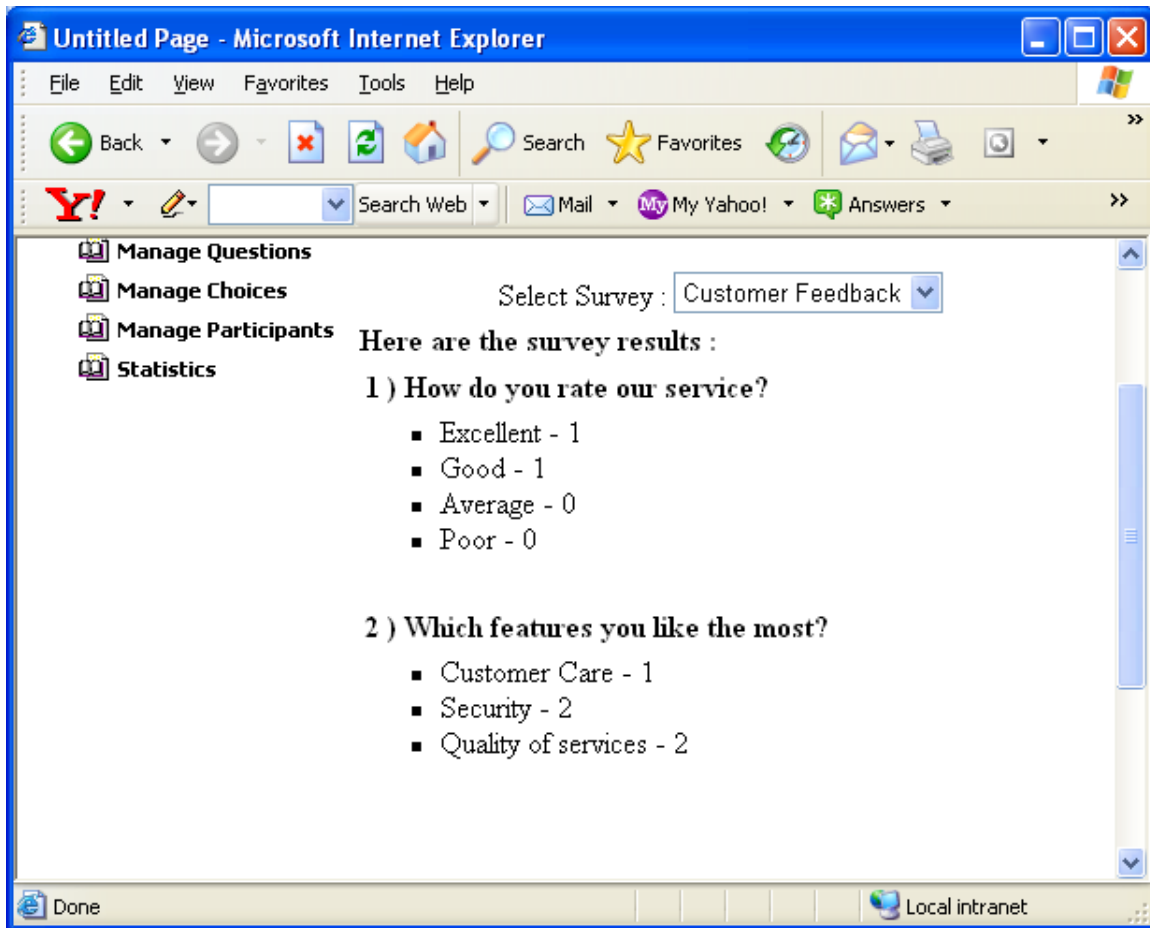


Figure 40: Showing survey statistics

The web form consists of a DropDownList listing all the surveys. Selecting a survey shows all its objective questions, their choices and votes for each choice. In order to design this web form, drag and drop two SQL data source controls, a DropDownList and a DataList on the web form. Configure one of the SQL data source controls to select all the records from Survey table where as configure the other to select all records from SurveyQuestions table with AnswerType equal to "S" or "M" and matching the selected SurveyID from the DropDownList. Bind the DropDownList to the first SQL data source control and DataList to the other SQL data source control. Design the ItemTemplate of the DataList as shown in Figure 41.

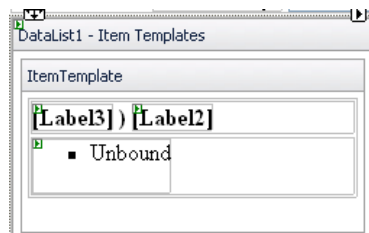


Figure 41: ItemTemplate to display survey statistics

Using “DataBindings” dialog bind Label3 with QuestionID column and Label2 with Question column. The BulletedList displays all the choices along with the votes for each choice. Figure 42 shows the complete markup of SurveyStats.aspx.

```
<%@ Page Language="VB"
MasterPageFile="~/Admin/AdminMasterPage.master"
AutoEventWireup="false" CodeFile="SurveyStats.aspx.vb"
Inherits="Admin_SurveyStats" title="Untitled Page" %>
<asp:Content ID="Content1"
ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
<asp:Label ID="Label2" runat="server" Text="<h3>Survey
Statistics</h3>"></asp:Label><br />
<table style="width: 100%">
<tr>
<td align="center">
<asp:Label ID="Label1" runat="server" Text="Select Survey
:"></asp:Label>
<asp:DropDownList ID="DropDownList1" runat="server"
AutoPostBack="True" DataSourceID="SqlDataSource1"
DataTextField="Title" DataValueField="SurveyID">
</asp:DropDownList></td>
</tr>
<tr>
<td>
<asp:DataList ID="DataList1" runat="server"
DataSourceID="SqlDataSource2" Width="100%">
<ItemTemplate>
<table style="width: 100%">
<tr>
<td style="height: 21px">
<asp:Label ID="Label3" runat="server" Font-Bold="True"
Text='<%# Eval("QuestionID") %>'></asp:Label>
<strong></strong>
<asp:Label ID="Label2" runat="server" Font-Bold="True"
Text='<%# Eval("Question") %>'></asp:Label></td>
</tr>
<tr>
<td>
<asp:BulletedList ID="BulletedList1" runat="server"
BulletStyle="Square">
</asp:BulletedList>
</td>
</tr>
</table>
</ItemTemplate>
<HeaderTemplate>
```

```

<asp:Label ID="Label4" runat="server" Font-Bold="True"
Text="Here are the survey results :"></asp:Label>
</HeaderTemplate>
</asp:DataList></td>
</tr>
</table>
<br />
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<%= $ConnectionStrings:ConnectionString
%>"
SelectCommand="SELECT [SurveyID], [Title] FROM
[Survey]"></asp:SqlDataSource>
<asp:SqlDataSource ID="SqlDataSource2" runat="server"
ConnectionString="<%= $ConnectionStrings:ConnectionString
%>"
SelectCommand="SELECT * FROM [SurveyQuestions] WHERE
(([SurveyID] = @SurveyID) AND ([AnswerType] <>
@AnswerType))">
<SelectParameters>
<asp:ControlParameter ControlID="DropDownList1"
Name="SurveyID" PropertyName="SelectedValue"
Type="Int32" />
<asp:Parameter DefaultValue="T" Name="AnswerType"
Type="String" />
</SelectParameters>
</asp:SqlDataSource>
</asp:Content>

```

Figure 42: Markup of SurveyStats.aspx

In order to display number of votes for all the choices we need to handle ItemDataBound event of the DataList. Write the code shown in Figure 43 in the ItemDataBound event of the DataList.

```

Protected Sub DataList1_ItemDataBound(ByVal sender As
Object, ByVal e As
System.Web.UI.WebControls.DataListItemEventArgs) Handles
DataList1.ItemDataBound
If e.Item.ItemType = ListItemType.Item Or e.Item.ItemType =
ListItemType.AlternatingItem Then
Dim questionid As Integer
questionid = CInt(CType(e.Item.FindControl("Label3"),
Label).Text)
Dim cnn As New
SqlConnection(ConfigurationManager.ConnectionStrings("Conne
ctionString").ConnectionString)

```

```

Dim cmd As New SqlCommand("select * from surveychoices
where questionid=@qid", cnn)
Dim p1 As New SqlParameter("@qid", questionid)
cmd.Parameters.Add(p1)
Dim da As New SqlDataAdapter
da.SelectCommand = cmd
Dim ds As New DataSet
da.Fill(ds, "choices")
cnn.Open()
For Each row As DataRow In ds.Tables("choices").Rows
Dim cmdAns As New SqlCommand("select count(*) from
surveyanswers where choiceid=@cid", cnn)
Dim pCid As New SqlParameter("@cid", row("choiceid"))
cmdAns.Parameters.Add(pCid)
Dim count As Integer = cmdAns.ExecuteScalar()
row("Choice") = row("Choice") & " - " & count
Next
cnn.Close()
Dim list As BulletedList =
e.Item.FindControl("BulletedList1")
list.DataSource = ds
list.DataTextField = "Choice"
list.DataBind()
End If
End Sub

```

Figure 43: Displaying choices along with number of votes

We retrieve the QuestionID using FindControl() method. Then we fetch all the possible choices for that question into a DataSet. We iterate through all the choices and for each choice select total number of records from the SurveyAnswers table. The count is simply appended to the Choice column. Finally, the resultant DataSet is bound with the BulletedList control.

Securing Administrative Pages

In this last section we will secure the administrative pages that we created earlier so that only the administrator(s) can access them. To begin with open the web.config file and add the markup as shown in Figure 44.

```

<authentication mode="Forms">
<forms loginUrl="login.aspx"></forms>
</authentication>
<authorization>
<allow users="*" />
</authorization>
...
<location path="admin">

```

```
<system.web>
<authorization>
<deny users="?" />
</authorization>
</system.web>
</location>
```

Figure 44: Enabling Forms authentication

Here, we set the authentication mode to “Forms”. We also set the loginUrl attribute of the <forms> tag to Login.aspx. The web forms in the root folder (Login.aspx and Survey.aspx) can be accessed by all so we configure the <authorization> tag to allow all the users (*). The web forms in the Admin folder needs to be protected and hence we add <location> tag for that folder and configure <authorization> tag to deny anonymous users (?).

Now, create a user called Administrator (or admin) using Web Site Administration Tool (Figure 45).

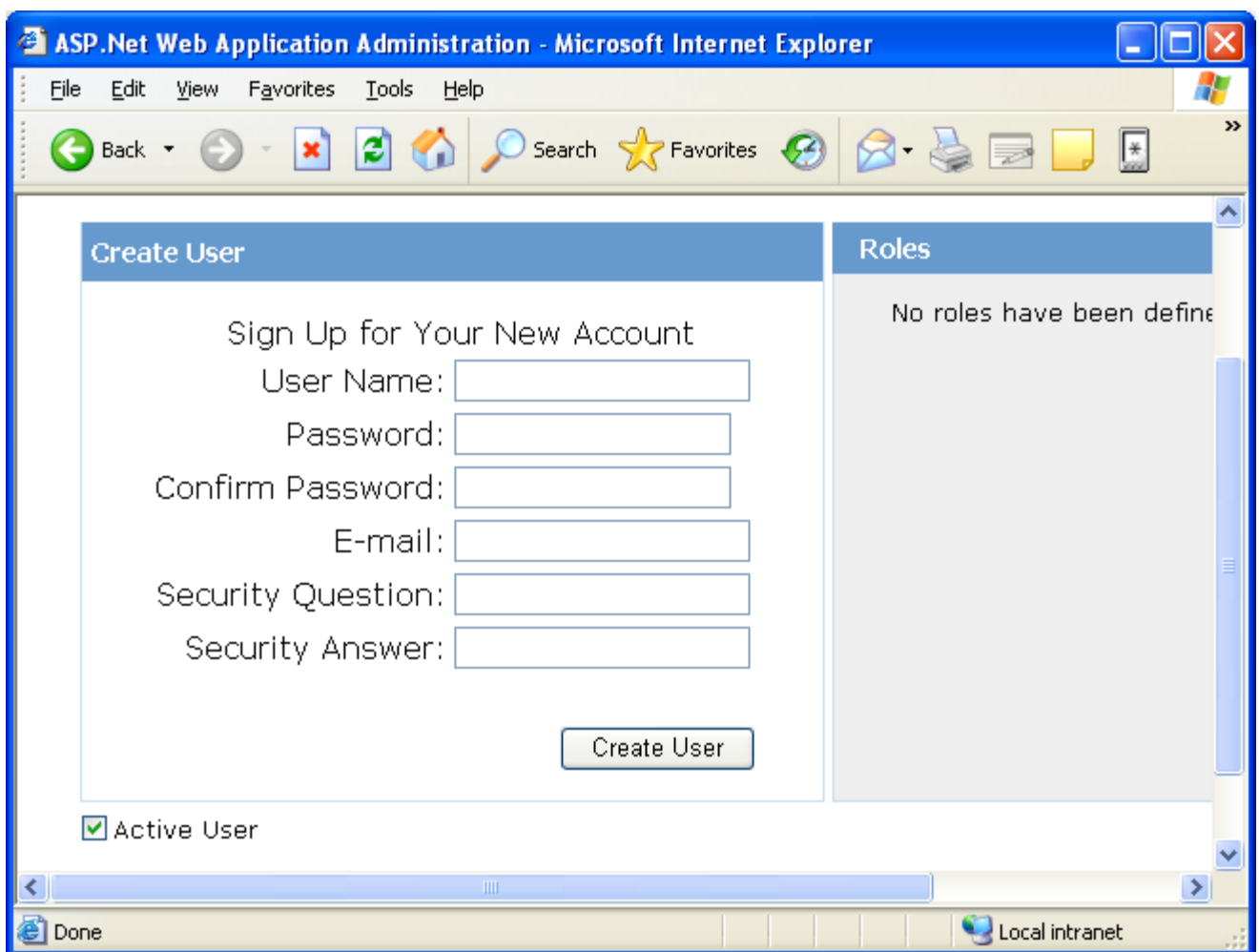


Figure 45: Creating User using Web Site Administration Tool

Note that since we have not configured any specific membership provider, the membership information will be stored in the ASPNETDB.MDF file under App_Data folder.

Next, add a web form called Login.aspx and drag and drop a Login control on it. That's it! The site is now protected from anonymous users. Any attempt to access administrative pages directly will redirect the user to the login page as shown in Figure 46.

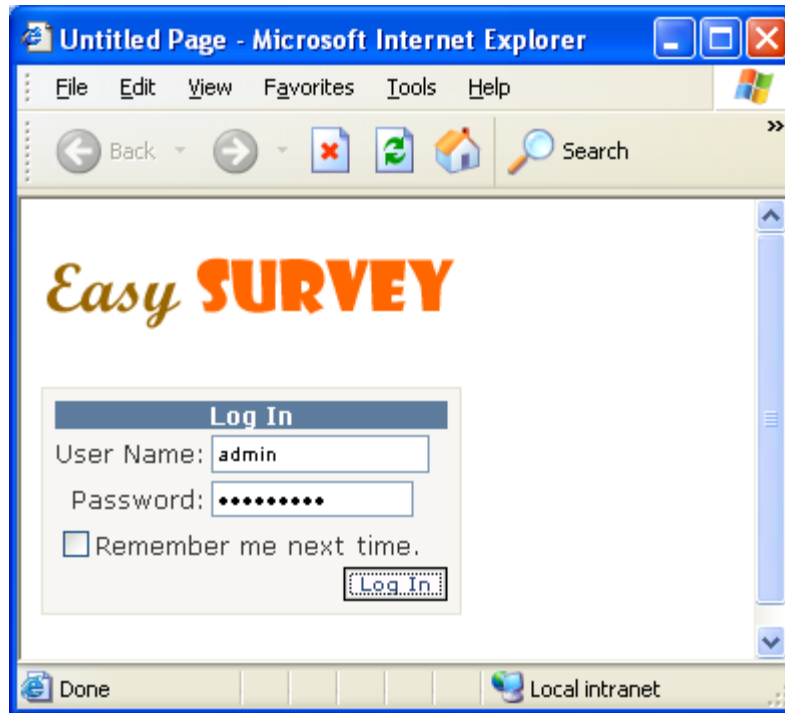


Figure 46: Login.aspx in action

Conclusion

In this final we completed the survey application by adding invitation feature and web forms for display survey questions and their choices. We also developed an administrative web form for displaying survey statistics. As a security feature we implemented forms based authentication that protects the administrative pages.

Some suggestions for extending and enhancing the survey application are given below:

- You can add functionality to track survey participants
- You can add functionality such that a participant can participate in the survey only once
- The survey statistics can be graphical and can include more details
- You may consider automating the job of sending invitations to the participants

- You can add some exception handling code and custom error pages especially for Survey.aspx
- You may want to implement role based security so that several levels of users can use the survey data in different way

***Bipin Joshi** is the founder and owner of BinaryIntellect Consulting (www.binaryintellect.com) where he conducts professional training programs on .NET technologies. He is the author and publisher of Developer's Guide to ASP.NET 2.0 and co-author of three WROX press books on .NET 1.x. He runs two community web sites dedicated to .NET articles and code samples - www.dotnetbips.com, and www.binaryintellect.net. He is a Microsoft MVP, member of ASPInsiders, MCAD and MCT. When away from computers he spends time practicing Yoga and meditation. You can contact him via his blog at www.bipinjoshi.com where he writes about life, Yoga and spirituality*