

```
1 using System;
2 using System.Data;
3 using System.Configuration;
4 using System.Collections;
5 using System.Web;
6 using System.Web.Security;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9 using System.Web.UI.WebControls.WebParts;
10 using System.Web.UI.HtmlControls;
11
12 public partial class PeopleEdit : System.Web.UI.Page
13 {
14
15     protected void Page_Load(object sender, EventArgs e)
16     {
17
18         //Use the logged in user as the default.
19         if (Session["UserID"] != null)
20         {
21             GetPerson.SelectParameters["UserID"].DefaultValue = Session["UserID"].
22             ToString();
23
24             myTabs.SetActiveTab(myTabs.TAB_INDEX_EDIT);
25
26             myTabs.UrlEdit = "peopleEdit.aspx?userID=" + Request.QueryString["UserID"];
27
28             myNav.UrlDemographics = "peopleEdit.aspx?userID=" + Request.QueryString[
29             "UserID"];
30             myNav.UrlPassword = "peopleEdit.aspx?mode=pass&userID=" + Request.QueryString
31             ["UserID"];
32             myNav.UrlPrivileges = "peopleEdit.aspx?mode=priv&userID=" + Request.
33             QueryString["UserID"];
34
35             switch (Request.QueryString["mode"])
36             {
37                 case "pass":
38                     EditView.ActiveViewIndex = 1;
39                     myNav.SetActiveTab(myNav.TAB_INDEX_PASSWORD);
40                     ToolTitle.Text = "Change Password";
41                     break;
42                 case "priv":
43                     EditView.ActiveViewIndex = 2;
44                     myNav.SetActiveTab(myNav.TAB_INDEX_PRIVILEGES);
45                     //select user's role memberships
46                     RolesRB.DataBound += this.RolesListLoaded;
47                     RolesCB.DataBound += this.RolesListLoaded;
48                     ToolTitle.Text = "Edit Privileges";
49                     break;
50                 default:
51                     //set active view
52                     EditView.ActiveViewIndex = 0;
53                     //get ready to set user's skills
54                     SkillsOne.DataBound += this.SkillsListLoaded;
55                     SkillsTwo.DataBound += this.SkillsListLoaded;
56                     //get ready to set user's security clearances
57                     SecurityClearancesOne.DataBound += this.SecurityClearancesLoaded;
58                     SecurityClearancesTwo.DataBound += this.SecurityClearancesLoaded;
59                     if (Request.QueryString["UserID"] == "0")
60                     {
61                         ToolTitle.Text = "New Person";
62                         myNav.SetActiveTab(myNav.TAB_INDEX_NEW);
63                     }
64             }
65         }
66     }
67 }
```

```
1 using System;
2 using System.Data;
3 using System.Configuration;
4 using System.Collections;
5 using System.Web;
6 using System.Web.Security;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9 using System.Web.UI.WebControls.WebParts;
10 using System.Web.UI.HtmlControls;
11
12 public partial class UserControls_usrCtl_ProposalTemplate : System.Web.UI.UserControl
13 {
14     protected void Page_Load(object sender, EventArgs e)
15     {
16         tsbdDataSetTableAdapters.tbl_templatesTableAdapter templatesTableAdapter = new
17         tsbdDataSetTableAdapters.tbl_templatesTableAdapter();
18         tsbdDataSet.tbl_templatesDataTable templatesDataTable;
19
20         templatesDataTable = templatesTableAdapter.GetData();
21         GVPropTemplateList.EnableViewState = false;
22
23         int RecordCount = templatesDataTable.Count;
24
25         if (ddlExamplePropRecords.SelectedValue == "All")
26         {
27             GVPropTemplateList.PageSize = RecordCount;
28             GVPropTemplateList.Sort("TemplateName", SortDirection.Ascending);
29             GVPropTemplateList.DataBind();
30         }
31         else
32         {
33             GVPropTemplateList.PageSize = Convert.ToInt32(ddlExamplePropRecords.
34             SelectedValue);
35             GVPropTemplateList.DataBind();
36         }
37
38         if (txtBoxSearch.Text == "")
39         {
40             GVPropTemplateList.DataSourceID = "objDSPropTemplateGV";
41             GVPropTemplateList.DataBind();
42         }
43     }
44
45     protected void EditPTCloseDVLnkBtn_Click(object sender, EventArgs e)
46     {
47         mvPropTemplates.ActiveViewIndex = 0;
48     }
49
50     protected void InsertPTCloseDVLnkBtn_Click(object sender, EventArgs e)
51     {
52         mvPropTemplates.ActiveViewIndex = 0;
53     }
54
55     protected void GVPropTemplateList_RowDataBound(object sender, GridViewRowEventArgs
56     e)
57     {
58         if (e.Row.RowType == DataControlRowType.DataRow)
59         {
60             //Create the link to the records word document
61         }
62     }
63 }
64
```

```
65         HyperLink _DocURL = (HyperLink)e.Row.FindControl("ItemDocLink");
66         string _curLinkDoc = _DocURL.Text;
67         string _serverName = "http://" + Server.HtmlEncode(Request.UserHostName); //
        / +":49170"; //port number
68         string _serverDir = Server.HtmlEncode(Request.ApplicationPath);
69         _curLinkDoc = _serverName + _serverDir + "/DATA/" + _curLinkDoc;
70         _DocURL.NavigateUrl = _curLinkDoc;
71         _DocURL.Target = "_blank";
72
73     }
74 }
75
76 protected void btnSearch_Click(object sender, EventArgs e)
77 {
78     tsbdDataSetTableAdapters.tbl_templatesTableAdapter templatesTableAdapter = new
tsbdDataSetTableAdapters.tbl_templatesTableAdapter();
79     tsbdDataSet.tbl_templatesDataTable templatesDataTable;
80
81     templatesDataTable = templatesTableAdapter.GetDataBySearch(txtBoxSearch.Text);
82
83
84     GVPropTemplateList.DataSourceID = "objDSGVPTSearch";
85
86     GVPropTemplateList.DataBind();
87
88 }
89 protected void GVPropTemplateList_DataBound(object sender, EventArgs e)
90 {
91     tsbdDataSetTableAdapters.tbl_templatesTableAdapter templatesTableAdapter = new
tsbdDataSetTableAdapters.tbl_templatesTableAdapter();
92     tsbdDataSet.tbl_templatesDataTable templatesDataTable;
93
94
95     templatesDataTable = templatesTableAdapter.GetData();
96
97     int RecordsCount = templatesDataTable.Count;
98     lblRecordCount.Text = RecordsCount.ToString() + " Record(s). ";
99
100     int PageCount = GVPropTemplateList.PageCount;
101     int PageCurrent = GVPropTemplateList.PageIndex + 1;
102     lblPageCount.Text = "Displaying Page: " + PageCurrent.ToString() + " of " +
PageCount.ToString();
103 }
104 }
105
```

```
1 using System;
2 using System.Data;
3 using System.Configuration;
4 using System.Collections;
5 using System.Web;
6 using System.Web.Security;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9 using System.Web.UI.WebControls.WebParts;
10 using System.Web.UI.HtmlControls;
11 using System.IO;
12
13 public partial class UserControls_usrCtl_ProposalLibrary : System.Web.UI.UserControl
14 {
15     protected void Page_Load(object sender, EventArgs e)
16     {
17         tsbdDataSetTableAdapters.tbl_PastPerformanceTableAdapter
18         PastPerformanceTableAdapter = new tsbdDataSetTableAdapters.
19         tbl_PastPerformanceTableAdapter();
20         tsbdDataSet.tbl_PastPerformanceDataTable PastPerformanceDataTable;
21
22         PastPerformanceDataTable = PastPerformanceTableAdapter.GetPastPerformanceData
23         ();
24         gvPastPerform.EnableViewState = false;
25
26         int RecordCount = PastPerformanceDataTable.Count;
27
28         if (ddlPastPerformRecordCount.SelectedValue == "All")
29         {
30             gvPastPerform.PageSize = RecordCount;
31             gvPastPerform.Sort("PPTitle", SortDirection.Ascending);
32             gvPastPerform.DataBind();
33         }
34         else
35         {
36             gvPastPerform.PageSize = Convert.ToInt32(ddlPastPerformRecordCount.
37             SelectedValue);
38             gvPastPerform.DataBind();
39         }
40
41         if (txtBoxSearch.Text == "")
42         {
43             gvPastPerform.DataSourceID = "objDSPastPerforme_gv";
44             gvPastPerform.DataBind();
45         }
46
47     }
48
49     protected void ItemSelectPastPerformImgBtn_Click(object sender,
50     ImageClickEventArgs e)
51     {
52         mvPastPerform.ActiveViewIndex = 1;
53         dvPastPerform.ChangeMode(DetailsViewMode.Edit);
54         dvPastPerform.Caption = "<h1>Edit Past Performance</h1>";
55     }
56
57     protected void dvPastPerform_ItemUpdated(object sender,
58     DetailsViewUpdatedEventArgs e)
59     {
60         //Rebind the control and change the view back to gridview.
61     }
```

```
62     dvPastPerform.DataBind();
63     mvPastPerform.ActiveViewIndex = 0;
64     gvPastPerform.DataBind();
65 }
66
67 protected void EditPPCancelDVLinkButton_Click(object sender, EventArgs e)
68 {
69     mvPastPerform.ActiveViewIndex = 0;
70 }
71
72 protected void dvPastPerform_ItemInserted(object sender,
73     DetailsViewInsertedEventArgs e)
74 {
75     mvPastPerform.ActiveViewIndex = 0;
76     gvPastPerform.DataBind();
77 }
78 protected void EditPPUpdateDVLinkButton_Click(object sender, EventArgs e)
79 {
80     try
81     {
82         FileUpload _Upload = (FileUpload)dvPastPerform.FindControl(
83             "EditPPFileUpload");
84         TextBox _FileLocation = (TextBox)dvPastPerform.FindControl(
85             "EditPPDescriptionDVTextBox");
86
87         if (_Upload.HasFile)
88         {
89             string extension = Path.GetExtension(_Upload.FileName);
90             if ((extension == ".doc") || (extension == ".txt") || (extension == ".
91                 rtf"))
92             {
93                 _Upload.SaveAs(Server.MapPath("DATA\\pastperformance\\docs\\") +
94                     _Upload.FileName);
95                 _FileLocation.Text = "pastperformance/docs/" + _Upload.FileName.
96                 ToString();
97             }
98             else
99             {
100                 lblPastPerformStatus.Text = "Not a valid txt file.";
101             }
102         }
103         catch (Exception ex)
104         {
105             lblPastPerformStatus.Text = ex.Message;
106         }
107         //pastperformance/docs/DSS_Criminal_History.doc
108     }
109
110 protected void gvPastPerform_RowCommand(object sender, GridViewCommandEventArgs e)
111 {
112     if (e.CommandName == "DocOpen")
113     {
114         //int _RowID = Convert.ToInt32(e.CommandArgument);
115         //DataKey _curDataKey = gvPastPerform.DataKeys[_RowID];
116         //string _curDoc = _curDataKey.Values["Description"].ToString();
117         string _curDoc = Server.MapPath("DATA\\") + e.CommandArgument.ToString();
118
119         if (File.Exists(_curDoc))
120         {
121             Response.ContentType = "application/octet-stream";
122             Response.AddHeader("Content-Disposition", "attachment; filename=" +
123                 _curDoc);
124             Response.Flush();
125         }
126     }
127 }
```

```
122         Response.WriteFile(_curDoc);
123         Response.End();
124     }
125 }
126 }
127
128 protected void ItemDescripImgBtn_Click(object sender, ImageClickEventArgs e)
129 {
130     //Get the file
131
132 }
133
134 protected void gvPastPerform_RowDataBound(object sender, GridViewRowEventArgs e)
135 {
136     if (e.Row.RowType == DataControlRowType.DataRow)
137     {
138         //Create the link to the records word document
139         HyperLink _DocURL = (HyperLink)e.Row.FindControl("ItemDocLink");
140         string _curLinkDoc = _DocURL.Text;
141         string _serverName = "http://" + Server.HtmlEncode(Request.UserHostName); //
142         //+":49170"; //port number
143         string _serverDir = Server.HtmlEncode(Request.ApplicationPath);
144         _curLinkDoc = _serverName + _serverDir + "/DATA/" + _curLinkDoc;
145         _DocURL.NavigateUrl = _curLinkDoc;
146         _DocURL.Target = "_blank";
147
148         //Get the assinged Group Name for the ID that is being read
149         tsbdDataSetTableAdapters.tbl_GroupsTableAdapter GroupTableAdapter = new
150         tsbdDataSetTableAdapters.tbl_GroupsTableAdapter();
151         tsbdDataSet.tbl_GroupsDataTable GroupDataTable;
152
153         Label _groupID = (Label)e.Row.FindControl("ItemGroupLabel");
154         Label _groupName = (Label)e.Row.FindControl("ItemGroupNameLabel");
155
156         if (_groupID.Text != "")
157         {
158             int GroupID = Convert.ToInt32(_groupID.Text);
159             GroupDataTable = GroupTableAdapter.GetData();
160
161             //Get the data row by group ID
162             DataRow _groupRow = GroupDataTable.FindBygroupID(GroupID);
163
164             _groupName.Text = _groupRow["groupName"].ToString();
165         }
166     }
167 }
168
169 protected void EditSubmitCalImgBut_Click(object sender, ImageClickEventArgs e)
170 {
171     if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
172     {
173         Calendar _submitCal = (Calendar)dvPastPerform.FindControl(
174         "EditSubmitCalendar");
175         //WebCalendarView _submitCal = (WebCalendarView)dvPastPerform.FindControl
176         ("EditSubmitCalendar");
177         if (_submitCal.Visible == false)
178         {
179             _submitCal.Visible = true;
180         }
181         else if (_submitCal.Visible == true)
182         {
183             _submitCal.Visible = false;
184         }
185     }
186 }
187
188 protected void EditStartCalImgBut_Click(object sender, ImageClickEventArgs e)
```

```
185     {
186         if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
187         {
188             Calendar _startCal = (Calendar)dvPastPerform.FindControl(
189                 "EditStartCalendar");
190
191             if (_startCal.Visible == false)
192             {
193                 _startCal.Visible = true;
194             }
195             else if (_startCal.Visible == true)
196             {
197                 _startCal.Visible = false;
198             }
199         }
200     }
201     protected void EditCompleteCalImgBut_Click(object sender, ImageClickEventArgs e)
202     { //
203
204         if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
205         {
206             Calendar _completeCal = (Calendar)dvPastPerform.FindControl(
207                 "EditCompleteCalendar");
208
209             if (_completeCal.Visible == false)
210             {
211                 _completeCal.Visible = true;
212             }
213             else if (_completeCal.Visible == true)
214             {
215                 _completeCal.Visible = false;
216             }
217         }
218     }
219     protected void EditSubmitCalendar_SelectionChanged(object sender, EventArgs e)
220     {
221         if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
222         {
223             TextBox _submitYearMonth = (TextBox)dvPastPerform.FindControl(
224                 "EditPPSubmissionDateDVTextBox");
225             Calendar _submitCal = (Calendar)dvPastPerform.FindControl(
226                 "EditSubmitCalendar");
227             _submitYearMonth.Text = _submitCal.SelectedDate.ToShortDateString();
228         }
229     }
230     protected void EditStartCalendar_SelectionChanged(object sender, EventArgs e)
231     {
232         if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
233         {
234             TextBox _startDate = (TextBox)dvPastPerform.FindControl(
235                 "EditPPStartDateDVTextBox");
236             Calendar _startCal = (Calendar)dvPastPerform.FindControl(
237                 "EditStartCalendar");
238             _startDate.Text = _startCal.SelectedDate.ToShortDateString();
239         }
240     }
241     protected void EditCompleteCalendar_SelectionChanged(object sender, EventArgs e)
242     {
243         if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
244         {
245             TextBox _completeDate = (TextBox)dvPastPerform.FindControl(
246                 "EditPPCompleteDateTextBox");
247             Calendar _completeCal = (Calendar)dvPastPerform.FindControl(
```

```
"EditCompleteCalendar");
245     _completeDate.Text = _completeCal.SelectedDate.ToShortDateString();
246 }
247 }
248
249 protected void EditGroupsDDL_SelectedIndexChanged(object sender, EventArgs e)
250 {
251     if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
252     {
253         TextBox _groupID = (TextBox)dvPastPerform.FindControl(
254             "EditPPGroupIDDVTextBox");
255         DropDownList _ddlGroup = (DropDownList)dvPastPerform.FindControl(
256             "EditGroupsDDL");
257         _groupID.Text = _ddlGroup.SelectedValue.ToString();
258     }
259 }
260
261 protected void EditGroupsDDL_DataBound(object sender, EventArgs e)
262 {
263     if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
264     {
265         TextBox _groupID = (TextBox)dvPastPerform.FindControl(
266             "EditPPGroupIDDVTextBox");
267         DropDownList _ddlGroup = (DropDownList)dvPastPerform.FindControl(
268             "EditGroupsDDL");
269
270         if (_groupID.Text != "")
271         {
272             ListItem item = _ddlGroup.Items.FindByValue(_groupID.Text);
273             _ddlGroup.SelectedItem.Selected = false;
274             item.Selected = true;
275         }
276         else
277         {
278             _ddlGroup.Items.Insert(0, new ListItem("-Select Group-", "0", true));
279         }
280     }
281 }
282
283 protected void EditSectorsDDL_SelectedIndexChanged(object sender, EventArgs e)
284 {
285     if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
286     {
287         TextBox _sectorID = (TextBox)dvPastPerform.FindControl(
288             "EditPPSectorIDDVTextBox");
289         DropDownList _ddlSector = (DropDownList)dvPastPerform.FindControl(
290             "EditSectorsDDL");
291         _sectorID.Text = _ddlSector.SelectedValue.ToString();
292     }
293 }
294
295 protected void EditSectorsDDL_DataBound(object sender, EventArgs e)
296 {
297     if (dvPastPerform.CurrentMode == DetailsViewMode.Edit)
298     {
299         TextBox _sectorID = (TextBox)dvPastPerform.FindControl(
300             "EditPPSectorIDDVTextBox");
301         DropDownList _ddlSector = (DropDownList)dvPastPerform.FindControl(
302             "EditSectorsDDL");
303
304         if (_sectorID.Text != "")
305         {
306             ListItem item = _ddlSector.Items.FindByValue(_sectorID.Text);
307             _ddlSector.SelectedItem.Selected = false;
308             item.Selected = true;
309         }
310     }
311 }
```

```
303         else
304         {
305             _ddlSector.Items.Insert(0, new ListItem("-Select Sector-", "0", true));
306         }
307     }
308 }
309
310 protected void imgButAddRecord_Click(object sender, ImageClickEventArgs e)
311 {
312     mvPastPerform.ActiveViewIndex = 1;
313     dvPastPerform.ChangeMode(DetailsViewMode.Insert);
314     dvPastPerform.Caption = "<h1>Add New Past Performance</h1>";
315 }
316
317 protected void InsertPPCancelDVLinkButton_Click(object sender, EventArgs e)
318 {
319     mvPastPerform.ActiveViewIndex = 0;
320 }
321
322 protected void InsertSubmitCalImgBut_Click(object sender, ImageClickEventArgs e)
323 {
324     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
325     {
326         Calendar _submitCal = (Calendar)dvPastPerform.FindControl(
327             "InsertSubmitCalendar");
328         if (_submitCal.Visible == false)
329         {
330             _submitCal.Visible = true;
331         }
332         else if (_submitCal.Visible == true)
333         {
334             _submitCal.Visible = false;
335         }
336     }
337 }
338
339 protected void InsertSubmitCalendar_SelectionChanged(object sender, EventArgs e)
340 {
341     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
342     {
343         TextBox _submitDate = (TextBox)dvPastPerform.FindControl(
344             "InsertPPSubmissionDateDVTextBox");
345         Calendar _submitCal = (Calendar)dvPastPerform.FindControl(
346             "InsertSubmitCalendar");
347         _submitDate.Text = _submitCal.SelectedDate.ToShortDateString();
348     }
349 }
350
351 protected void InsertStartCalImgBut_Click(object sender, ImageClickEventArgs e)
352 {
353     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
354     {
355         Calendar _startCal = (Calendar)dvPastPerform.FindControl(
356             "InsertStartCalendar");
357         if (_startCal.Visible == false) _startCal.Visible = true;
358         else if (_startCal.Visible == true) _startCal.Visible = false;
359     }
360 }
361
362 protected void InsertStartCalendar_SelectionChanged(object sender, EventArgs e)
363 {
364     TextBox _startDate = (TextBox)dvPastPerform.FindControl(
365         "InsertPPStartDateDVTextBox");
366     Calendar _startCal = (Calendar)dvPastPerform.FindControl("InsertStartCalendar");
367 }
```

```
363     _startDate.Text = _startCal.SelectedDate.ToShortDateString();
364
365 }
366
367 protected void InsertCompleteCalImgBut_Click(object sender, ImageClickEventArgs e)
368 {
369     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
370     {
371         Calendar _completeCal = (Calendar)dvPastPerform.FindControl(
372             "InsertCompleteCalendar");
373         if (_completeCal.Visible == false) _completeCal.Visible = true;
374         else if (_completeCal.Visible == true) _completeCal.Visible = false;
375     }
376
377 protected void InsertCompleteCalendar_SelectionChanged(object sender, EventArgs e)
378 {
379     TextBox _completeDate = (TextBox)dvPastPerform.FindControl(
380         "InsertPPCompleteDateTextBox");
381     Calendar _completeCal = (Calendar)dvPastPerform.FindControl(
382         "InsertCompleteCalendar");
383     _completeDate.Text = _completeCal.SelectedDate.ToShortDateString();
384
385 protected void InsertSectorsDDL_SelectedIndexChanged(object sender, EventArgs e)
386 {
387     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
388     {
389         TextBox _sectorID = (TextBox)dvPastPerform.FindControl(
390             "InsertPPSectorIDDVTextBox");
391         DropDownList _ddlSector = (DropDownList)dvPastPerform.FindControl(
392             "InsertSectorsDDL");
393         _sectorID.Text = _ddlSector.SelectedValue.ToString();
394
395 protected void InsertSectorsDDL_DataBound(object sender, EventArgs e)
396 {
397     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
398     {
399         TextBox _sectorID = (TextBox)dvPastPerform.FindControl(
400             "InsertPPSectorIDDVTextBox");
401         DropDownList _ddlSector = (DropDownList)dvPastPerform.FindControl(
402             "InsertSectorsDDL");
403
404         if (_sectorID.Text != "")
405         {
406             ListItem item = _ddlSector.Items.FindByValue(_sectorID.Text);
407             _ddlSector.SelectedItem.Selected = false;
408             item.Selected = true;
409         }
410         else
411         {
412             _ddlSector.Items.Insert(0, new ListItem("-Select Sector-", "0", true));
413         }
414
415 protected void InsertGroupsDDL_SelectedIndexChanged(object sender, EventArgs e)
416 {
417     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
418     {
419         TextBox _groupID = (TextBox)dvPastPerform.FindControl(
420             "InsertPPGroupIDDVTextBox");
421         DropDownList _ddlGroup = (DropDownList)dvPastPerform.FindControl(
422             "InsertGroupsDDL");
```

```
420     _groupID.Text = _ddlGroup.SelectedValue.ToString();
421     }
422 }
423
424 protected void InsertGroupsDDL_DataBound(object sender, EventArgs e)
425 {
426     if (dvPastPerform.CurrentMode == DetailsViewMode.Insert)
427     {
428         TextBox _groupID = (TextBox)dvPastPerform.FindControl(
429             "InsertPPGroupIDDVTextBox");
430         DropDownList _ddlGroup = (DropDownList)dvPastPerform.FindControl(
431             "InsertGroupsDDL");
432         if (_groupID.Text != "")
433         {
434             ListItem item = _ddlGroup.Items.FindByValue(_groupID.Text);
435             _ddlGroup.SelectedItem.Selected = false;
436             item.Selected = true;
437         }
438         else
439         {
440             _ddlGroup.Items.Insert(0, new ListItem("-Select Group-", "0", true));
441         }
442     }
443 }
444
445 protected void InsertPPInsertDVLLinkButton_Click(object sender, EventArgs e)
446 {
447     try
448     {
449         FileUpload _Upload = (FileUpload)dvPastPerform.FindControl(
450             "InsertPPFileUpload");
451         TextBox _FileLocation = (TextBox)dvPastPerform.FindControl(
452             "InsertPPDescriptionDVTextBox");
453         if (_Upload.HasFile)
454         {
455             string extension = Path.GetExtension(_Upload.FileName);
456             if ((extension == ".doc") || (extension == ".txt") || (extension == ".rtf"))
457             {
458                 _Upload.SaveAs(Server.MapPath("DATA\\pastperformance\\docs\\") +
459                     _Upload.FileName);
460                 _FileLocation.Text = "pastperformance/docs/" + _Upload.FileName.
461                 ToString();
462             }
463             else
464             {
465                 lblPastPerformStatus.Text = "Not a valid txt file.";
466             }
467         }
468         catch (Exception ex)
469         {
470             lblPastPerformStatus.Text = ex.Message;
471         }
472     }
473 }
474
475 protected void gvPastPerform_DataBound(object sender, EventArgs e)
476 {
477     tsbdDataSetTableAdapters.tbl_PastPerformanceTableAdapter
478     PastPerformanceTableAdapter = new tsbdDataSetTableAdapters.
479     tbl_PastPerformanceTableAdapter();
480     tsbdDataSet.tbl_PastPerformanceDataTable PastPerformanceDataTable;
```

```
478
479     PastPerformanceDataTable = PastPerformanceTableAdapter.GetPastPerformanceData
    ();
480
481     int RecordsCount = PastPerformanceDataTable.Count;
482     lblRecordCount.Text = RecordsCount.ToString() + " Record(s). ";
483
484     int PageCount = gvPastPerform.PageCount;
485     int PageCurrent = gvPastPerform.PageIndex + 1;
486     lblPageCount.Text = "Displaying Page: " + PageCurrent.ToString() + " of " +
    PageCount.ToString();
487 }
488 protected void btnSearch_Click(object sender, EventArgs e)
489 {
490     tsbdDataSetTableAdapters.tbl_PastPerformanceTableAdapter
    PastPerformanceTableAdapter = new tsbdDataSetTableAdapters.
    tbl_PastPerformanceTableAdapter();
491     tsbdDataSet.tbl_PastPerformanceDataTable PastPerformanceTable;
492
493     PastPerformanceTable = PastPerformanceTableAdapter.GetDataBySearch
    (txtBoxSearch.Text);
494
495
496
497
498     gvPastPerform.DataSourceID="objDSPastPermeSearch_gv";
499     //objDSPastPerme_gv.SelectParameters["GetDataBySearch"];
500
501     gvPastPerform.DataBind();
502
503
504
505
506 }
507 }
508 }
509
```

```
1 using System;
2 using System.Data;
3 using System.Configuration;
4 using System.Collections;
5 using System.Web;
6 using System.Web.Security;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9 using System.Web.UI.WebControls.WebParts;
10 using System.Web.UI.HtmlControls;
11 using System.IO;
12
13 public partial class UserControls_usrCtl_ExampleProposal : System.Web.UI.UserControl
14 {
15     protected void Page_Load(object sender, EventArgs e)
16     {
17         tsbdDataSetTableAdapters.tbl_examplesTableAdapter examplesTableAdapter = new
18         tsbdDataSetTableAdapters.tbl_examplesTableAdapter();
19         tsbdDataSet.tbl_examplesDataTable examplesDataTable;
20
21         examplesDataTable = examplesTableAdapter.GetDataExamplePropData();
22         gvExampleProp.EnableViewState = false;
23
24         int RecordCount = examplesDataTable.Count;
25
26         if (ddlExamplePropRecords.SelectedValue == "All")
27         {
28             gvExampleProp.PageSize = RecordCount;
29             gvExampleProp.Sort("fromProposalName", SortDirection.Ascending);
30             gvExampleProp.DataBind();
31         }
32         else
33         {
34
35             gvExampleProp.PageSize = Convert.ToInt32(ddlExamplePropRecords.
36 SelectedValue);
37             gvExampleProp.DataBind();
38         }
39
40         if (txtBoxSearch.Text == "")
41         {
42             gvExampleProp.DataSourceID = "objDSEExamplePropGV";
43             gvExampleProp.DataBind();
44         }
45     }
46 }
47
48 protected void gvExampleProp_RowDataBound(object sender, GridViewRowEventArgs e)
49 {
50     if (e.Row.RowType == DataControlRowType.DataRow)
51     {
52         //Create the link to the records word document
53         HyperLink _DocURL = (HyperLink)e.Row.FindControl("ItemDocLink");
54         string _curLinkDoc = _DocURL.Text;
55         string _serverName = "http://" + Server.HtmlEncode(Request.UserHostName);
56 // +":49170"; //port number
57         string _serverDir = Server.HtmlEncode(Request.ApplicationPath);
58         _curLinkDoc = _serverName + _serverDir + "/DATA/" + _curLinkDoc;
59         _DocURL.NavigateUrl = _curLinkDoc;
60         _DocURL.Target = "_blank";
61     }
62 }
63
64 protected void EditEPCloseDVLnkBtn_Click(object sender, EventArgs e)
```

```
65     {
66         mvExampleProp.ActiveViewIndex = 0;
67     }
68 }
69
70 protected void InsertEPCloseDVLnkBtn_Click(object sender, EventArgs e)
71 {
72     mvExampleProp.ActiveViewIndex = 0;
73 }
74
75 protected void imgButAddNewRecord_Click(object sender, ImageClickEventArgs e)
76 {
77     mvExampleProp.ActiveViewIndex = 1;
78     dvExampleProp.ChangeMode(DetailsViewMode.Insert);
79     dvExampleProp.Caption = "<h1>Add New Example Proposal</h1>";
80 }
81
82 protected void ItemEPSelectGVImgBtn_Click(object sender, ImageClickEventArgs e)
83 {
84     mvExampleProp.ActiveViewIndex = 1;
85     dvExampleProp.ChangeMode(DetailsViewMode.Edit);
86     dvExampleProp.Caption = "<h1>Edit Example Proposal </h1>";
87 }
88
89 protected void EditEPUpdateDVLnkBtn_Click(object sender, EventArgs e)
90 {
91     try
92     {
93         FileUpload _Upload = (FileUpload)dvExampleProp.FindControl(
94             "EditEPExampleDVFileUpload");
95         TextBox _FileLocation = (TextBox)dvExampleProp.FindControl(
96             "EditEPExampleDVTTextBox");
97
98         if (_Upload.HasFile)
99         {
100             string extension = Path.GetExtension(_Upload.FileName);
101             if ((extension == ".doc") || (extension == ".txt") || (extension == ".rtf"))
102             {
103                 _Upload.SaveAs(Server.MapPath("DATA\\examples\\docs\\") + _Upload.
104                 FileName);
105                 _FileLocation.Text = "examples/docs/" + _Upload.FileName.ToString
106                 ();
107             }
108             else
109             {
110                 lblExamplePropStatus.Text = "Not a valid txt file.";
111             }
112         }
113     }
114     catch (Exception ex)
115     {
116         lblExamplePropStatus.Text = ex.Message;
117     }
118     mvExampleProp.ActiveViewIndex = 0;
119 }
120
121 protected void dvExampleProp_ItemUpdated(object sender,
122     DetailsViewUpdatedEventArgs e)
123 {
124     dvExampleProp.DataBind();
125     mvExampleProp.ActiveViewIndex = 0;
126     gvExampleProp.DataBind();
127 }
```

```
126     }
127
128
129
130     protected void InsertEPInsertDVLnkBtn_Click(object sender, EventArgs e)
131     {
132         try
133         {
134             FileUpload _Upload = (FileUpload)dvExampleProp.FindControl(
135                 "InsertEPExampleDVFileUpload");
136             TextBox _FileLocation = (TextBox)dvExampleProp.FindControl(
137                 "InsertEPExampleDVTextBox");
138
139             if (_Upload.HasFile)
140             {
141                 string extension = Path.GetExtension(_Upload.FileName);
142                 if ((extension == ".doc") || (extension == ".txt") || (extension == ".rtf"))
143                 {
144                     _Upload.SaveAs(Server.MapPath("DATA\\examples\\docs\\") + _Upload.
145                         FileName);
146                     _FileLocation.Text = "examples/docs/" + _Upload.FileName.ToString
147                         ();
148                 }
149                 else
150                 {
151                     lblExamplePropStatus.Text = "Not a valid txt file.";
152                 }
153             }
154             catch (Exception ex)
155             {
156                 lblExamplePropStatus.Text = ex.Message;
157             }
158         }
159     }
160     protected void dvExampleProp_ItemInserted(object sender,
161         DetailsViewInsertedEventArgs e)
162     {
163         mvExampleProp.ActiveViewIndex = 0;
164         gvExampleProp.DataBind();
165     }
166     protected void gvExampleProp_DataBound(object sender, EventArgs e)
167     {
168         tsbdDataSetTableAdapters.tbl_examplesTableAdapter examplesTableAdapter = new
169             tsbdDataSetTableAdapters.tbl_examplesTableAdapter();
170         tsbdDataSet.tbl_examplesDataTable examplesDataTable;
171
172         examplesDataTable = examplesTableAdapter.GetDataExamplePropData();
173
174         int RecordsCount = examplesDataTable.Count;
175         lblRecordCount.Text = RecordsCount.ToString() + " Record(s). ";
176
177         int PageCount = gvExampleProp.PageCount;
178         int PageCurrent = gvExampleProp.PageIndex + 1;
179         lblPageCount.Text = "Displaying Page: " + PageCurrent.ToString() + " of " +
180             PageCount.ToString();
181     }
182     protected void btnSearch_Click(object sender, EventArgs e)
183     {
184         tsbdDataSetTableAdapters.tbl_examplesTableAdapter examplesTableAdapter = new
185             tsbdDataSetTableAdapters.tbl_examplesTableAdapter();
186         tsbdDataSet.tbl_examplesDataTable examplesDataTable;
187
188         examplesDataTable = examplesTableAdapter.GetDataBySearch(txtBoxSearch.Text);
189     }
190 }
```

```
184
185
186     gvExampleProp.DataSourceID = "objDSEExamplePropSearchGV";
187     gvExampleProp.DataBind();
188
189 }
190
191 }
192
```

VisuMAX © 2008

```
1 using System;
2 using System.Data;
3 using System.Configuration;
4 using System.Collections;
5 using System.Web;
6 using System.Web.Security;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9 using System.Web.UI.WebControls.WebParts;
10 using System.Web.UI.HtmlControls;
11
12 public partial class UserControls_ursCtl_AcronymList : System.Web.UI.UserControl
13 {
14     protected void Page_Load(object sender, EventArgs e)
15     {
16         try
17         {
18             tsbdDataSetTableAdapters.tbl_AcronymsTableAdapter AcronymsTableAdapter =
19             new tsbdDataSetTableAdapters.tbl_AcronymsTableAdapter();
20             tsbdDataSet.tbl_AcronymsDataTable AcronymsDataTable;
21             AcronymsDataTable = AcronymsTableAdapter.GetAcronymData();
22             gvAcronyms.EnableViewState = false;
23
24             int RecordsCount = AcronymsDataTable.Count;
25
26             CurrentFilter.Value = AlphaFilter1.Filter;
27
28             if (ddlAcronymRecordCount.SelectedValue == "All")
29             {
30                 gvAcronyms.PageSize = RecordsCount;
31                 gvAcronyms.Sort("Acronym", SortDirection.Ascending);
32                 gvAcronyms.DataBind();
33             }
34             else
35             {
36                 gvAcronyms.PageSize = Convert.ToInt32(ddlAcronymRecordCount.
37                 SelectedValue);
38                 gvAcronyms.DataBind();
39             }
40
41         }
42         catch (Exception ex)
43         {
44             lblAcronymListStatus.Text = ex.Message;
45         }
46     }
47
48     public void FilterChanged(object sender, EventArgs e)
49     {
50         if (txtBoxSearch.Text == "")
51         {
52             CurrentFilter.Value = AlphaFilter1.Filter;
53         }
54     }
55
56     protected void imgBtnAddNewAcronym_Click(object sender, ImageClickEventArgs e)
57     {
58         try
59         {
60             mvProposalCenter.ActiveViewIndex = 1;
61             dvAcronyms.ChangeMode(DetailsViewMode.Insert);
62             dvAcronyms.Caption = "<h1>Add Acronym</h1>";
63         }
64         catch (Exception ex)
65         {
```

```
66         lblAcronymEditStatus.Text = ex.Message;
67     }
68 }
69
70 protected void GVAcronyms_DataBound(object sender, EventArgs e)
71 {
72
73     try
74     {
75         tsbdDataSetTableAdapters.tbl_AcronymsTableAdapter AcronymsTableAdapter =
76         new tsbdDataSetTableAdapters.tbl_AcronymsTableAdapter();
77         tsbdDataSet.tbl_AcronymsDataTable AcronymsDataTable;
78         AcronymsDataTable = AcronymsTableAdapter.GetDataBySearchLetter
79         (CurrentFilter.Value);
80
81         int RecordsCount = AcronymsDataTable.Count;
82         lblRecordCount.Text = RecordsCount.ToString() + " Record(s).";
83
84         int PageCount = gvAcronyms.PageCount;
85         int PageCurrent = gvAcronyms.PageIndex + 1;
86         lblPageCount.Text = "Displaying Page: " + PageCurrent.ToString() + " of "
87         + PageCount.ToString();
88     }
89     catch (Exception ex)
90     {
91         lblAcronymListStatus.Text = ex.Message;
92     }
93 }
94
95 protected void GVAcronyms_RowCommand(object sender, GridViewCommandEventArgs e)
96 {
97     //GridView gv = (GridView)sender;
98     //int id_file = (int)gv.DataKeys[gv.SelectedIndex].Values[0];
99
100    try
101    {
102        switch (e.CommandName)
103        {
104            case "Select":
105                mvProposalCenter.ActiveViewIndex = 1;
106                dvAcronyms.ChangeMode(DetailsViewMode.Edit);
107                dvAcronyms.Caption = "<h1>Edit Acronym</h1>";
108                break;
109        }
110    }
111    catch (Exception ex)
112    {
113        lblAcronymEditStatus.Text = ex.Message;
114    }
115 }
116
117 protected void GVAcronyms_SelectedIndexChanged(object sender, EventArgs e)
118 {
119     //Get the Acronym ID of the selected field
120     //lblAcronymListID.Text = gvAcronyms.SelectedValue.ToString();
121 }
122
123 protected void dvAcronyms_ItemUpdated(object sender, DetailsViewUpdatedEventArgs
124 e)
125 {
126     mvProposalCenter.ActiveViewIndex = 0;
127     gvAcronyms.DataBind();
128 }
```

```
129     protected void dvAcronyms_ItemInserted(object sender, DetailsViewInsertedEventArgs e)
130     {
131         mvProposalCenter.ActiveViewIndex = 0;
132         gvAcronyms.DataBind();
133     }
134
135     protected void lnkBtnClose_Click(object sender, EventArgs e)
136     {
137         mvProposalCenter.ActiveViewIndex = 0;
138     }
139
140     protected void lnkBtnClose_Command(object sender, CommandEventArgs e)
141     {
142         if (e.CommandName == "Close") mvProposalCenter.ActiveViewIndex = 0;
143     }
144 }
145
146 protected void btnSearch_Click(object sender, EventArgs e)
147 {
148     tsbdDataSetTableAdapters.tbl_AcronymsTableAdapter AcronymsTableAdapter = new
149     tsbdDataSetTableAdapters.tbl_AcronymsTableAdapter();
150     tsbdDataSet.tbl_AcronymsDataTable AcronymsDataTable;
151     //Call our search
152     AcronymsDataTable = AcronymsTableAdapter.GetDataBySearch(txtBoxSearch.Text,
153     AlphaFilter1.Current.ToString());
154     //Rebind grid with results
155     gvAcronyms.DataBind();
156
157     //Set the current filter to null to display search
158     CurrentFilter.Value = null;
159 }
160 }
161 }
162
```

```
1 using System;
2 using System.Data;
3 using System.Configuration;
4 using System.Collections;
5 using System.Web;
6 using System.Web.Security;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9 using System.Web.UI.WebControls.WebParts;
10 using System.Web.UI.HtmlControls;
11
12 public partial class PeopleList : System.Web.UI.Page
13 {
14     protected void Page_Load(object sender, EventArgs e)
15     {
16         GridView1.PageSize = Convert.ToInt32(DisplayCount.SelectedValue);
17         GridView1.PagerSettings.Visible = false;
18         GridView1.EnableViewState = false;
19
20         //if they can't edit user's hide the control
21         GridView1.Columns[0].Visible = Security.IsAllowed("PEOPLE_EDIT", Session[
22             "UserID"]);
23
24         //make our alphaFilters filter value available to the gridview's container
25         (lame)
26         CurrentFilter.Value = AlphaFilter1.Filter;
27
28         //hook our pager control up to the gridview
29         Pager.Grid = GridView1;
30
31         //override the default newUser url
32         myNav.UrlNew = "PeopleEdit.aspx?userID=0";
33         if (Request.QueryString["userCatID"] == "2")
34         {
35             myTabs.SetActiveTab(myTabs.TAB_INDEX_LISTSUPPORT);
36         }
37
38         //set default grid sorting
39         if (GridView1.SortExpression == "")
40         {
41             GridView1.Sort("lastName", SortDirection.Ascending);
42         }
43
44         //get notified when somebody selects a gridview item
45         GridView1.RowCommand += this.RowCommandHandler;
46
47         //set search filter, if necessary
48         PeopleTableData.Selecting += this.setSearchParameter;
49
50         //update records count when data is loaded
51         PeopleTableData.Selected += this.GridBound;
52
53         //update the page count when grid is loaded;
54         GridView1.DataBound += this.PageChanged;
55     }
56
57     protected void GridBound(object sender, ObjectDataSourceStatusEventArgs e)
58     {
59         //get the total records
60         int totalRecords = ((DataTable)e.ReturnValue).Rows.Count;
61
62         if (totalRecords == 1)
63         {
64             RecordCount.Text = totalRecords.ToString() + " Record";
65         }
66         else
```

```
66     {
67         RecordCount.Text = totalRecords.ToString() + " Records";
68     }
69 }
70
71 protected void PageChanged(object sender, EventArgs e)
72 {
73     int totalPages = GridView1.PageCount;
74     int page = GridView1.PageIndex + 1;
75
76     pageCount.Text = "Page " + page.ToString() + " of " + totalPages.ToString();
77 }
78
79 protected void RowCommandHandler(object sender, GridViewCommandEventArgs e)
80 {
81     // ignore pager and sort commands
82     if (e.CommandName == "Page" || e.CommandName == "Sort") { return; }
83
84     try
85     {
86         // select the row in question
87         GridView G = (GridView)sender;
88         G.SelectedIndex = Convert.ToInt32(e.CommandArgument);
89         // The identity for the row that was clicked
90         int ID = Convert.ToInt32(G.SelectedValue);
91
92         switch (e.CommandName)
93         {
94             case "EditUser":
95                 Response.Redirect("PeopleEdit.aspx?userID=" + ID, true);
96                 break;
97
98             case "ViewUser":
99                 Response.Redirect("PeopleView.aspx?userID=" + ID, true);
100                break;
101
102                default:
103                    break;
104            }
105
106            Trace.Warn("Row ID:" + ID + ", Command: " + e.CommandName);
107        }
108        catch (Exception ex)
109        {
110            //Trace.Write("Warning", ex.Message);
111        }
112    }
113
114
115    public void FilterChanged(object sender, EventArgs e)
116    {
117        CurrentFilter.Value = AlphaFilter1.Filter;
118        searchTerms.Text = "";
119    }
120
121    public void setSearchParameter(object sender, ObjectDataSourceSelectingEventArgs e)
122    {
123        if (searchTerms.Text.Trim() != "")
124        {
125            e.InputParameters["SearchFilter"] = searchTerms.Text.Trim();
126            SearchTermLiteral.Text = searchTerms.Text.Trim();
127            SearchResultsText.Visible = true;
128        }
129    }
130    public void clearSearchResults(object sender, EventArgs e)
131    {
```

```
132     searchTerms.Text = "";  
133 }  
134  
135 }  
136
```

VisuMAX © 2008

```
64         else
65         {
66             myNav.SetActiveTab(myNav.TAB_INDEX_DEMOGRAPHICS);
67         }
68         break;
69     }
70
71     //load up the data
72     if (!IsPostBack)
73     {
74         GetPerson.Selected += this.LoadForm;
75         GetPerson.Select();
76     }
77 }
78
79 protected void RolesListLoaded(object sender, EventArgs e)
80 {
81     TSBDPeopleDataSet.tbl_RolesDataTable RD = new TSBDPeopleDataSetTableAdapters.
tbl_RolesTableAdapter().GetDataByUserID(Convert.ToInt32(UserIDHiddenField.Value));
82     TSBDPeopleDataSet.tbl_RolesDataTable Roles = new
TSBDPeopleDataSetTableAdapters.tbl_RolesTableAdapter().GetData();
83     ArrayList UserRoles = new ArrayList();
84     Hashtable RoleDescriptions = new Hashtable();
85
86     //get all the role descriptions for tooltips
87     foreach (TSBDPeopleDataSet.tbl_RolesRow R in Roles.Rows)
88     {
89         RoleDescriptions.Add(R.roleID.ToString(), R.roleDescription);
90     }
91
92
93     //get the user's roles
94     foreach (TSBDPeopleDataSet.tbl_RolesRow R in RD.Rows)
95     {
96         UserRoles.Add(R.roleID.ToString());
97     }
98
99     //select the roles that the user belongs to
100    ListControl L = (ListControl)sender;
101    foreach (ListItem I in L.Items)
102    {
103
104        if (UserRoles.Contains(I.Value))
105        {
106            I.Selected = true;
107        }
108
109        I.Attributes["class"] = "tooltip";
110        if (I.Value.Trim() != "" && RoleDescriptions.ContainsKey(I.Value))
111        {
112            String tip = RoleDescriptions[I.Value].ToString();
113            I.Attributes["title"] = "Description::" + tip;
114        }
115        else
116        {
117            I.Attributes["title"] = "Description::Sets the role to none.";
118        }
119    }
120 }
121
122
123
124
125 protected void SecurityClearancesLoaded(object sender, EventArgs e)
126 {
127
128     if (UserIDHiddenField.Value != "")
```

```
129     {
130         //make a list of the user's security clearances
131         TSBDPeopleDataSet.tbl_SecurityClearancesDataTable SD = new
TSBDPeopleDataSetTableAdapters.tbl_SecurityClearancesTableAdapter().
GetDataByUserID(Convert.ToInt32(UserIDHiddenField.Value));
132         ArrayList UserClearances = new ArrayList();
133
134         foreach (TSBDPeopleDataSet.tbl_SecurityClearancesRow R in SD.Rows)
135         {
136             UserClearances.Add(R.securityClearanceID.ToString());
137         }
138
139         ListControl DDL = (ListControl)sender;
140
141         //select a list item if it exists in the UserClearances array
142         foreach (ListItem I in DDL.Items)
143         {
144             if (UserClearances.Contains(I.Value))
145             {
146                 DDL.SelectedValue = I.Value;
147             }
148         }
149     }
150 }
151
152
153
154 protected void SkillsListLoaded(object sender, EventArgs e)
155 {
156     if (UserIDHiddenField.Value != "")
157     {
158         //make a quick list of the user's skills
159         TSBDPeopleDataSet.tbl_SkillsDataTable SD = new
TSBDPeopleDataSetTableAdapters.tbl_SkillsTableAdapter().GetUserSkillsByUserID
(Convert.ToInt32(UserIDHiddenField.Value));
160         ArrayList UsersSkills = new ArrayList();
161
162         foreach (TSBDPeopleDataSet.tbl_SkillsRow R in SD.Rows)
163         {
164             UsersSkills.Add(R.skillID.ToString());
165         }
166
167         CheckBoxList CBL = (CheckBoxList)sender;
168         //check each checkbox item to see if it should be checked, based on the
usersSkills array list
169         foreach (ListItem I in CBL.Items)
170         {
171             if (UsersSkills.Contains(I.Value))
172             {
173                 I.Selected = true;
174             }
175             else
176             {
177                 I.Selected = false;
178             }
179         }
180     }
181 }
182
183
184
185 protected void LoadForm(object sender, ObjectDataSourceStatusEventArgs e)
186 {
187     tsbdDataSet.tbl_PeopleDataTable PD = (tsbdDataSet.tbl_PeopleDataTable)e.
ReturnValue;
188
189     //new user default values
```

```

190     if (Request.QueryString["UserID"] == "0")
191     {
192         FTETextBox.Text = "1.00";
193     }
194
195     if (PD.Rows.Count == 1)
196     {
197         //populate common fields
198
199         //Stash the UserID in the form, in a hidden field
200         UserIDHiddenField.Value = PD[0].userID.ToString();
201         ProfileName.Text = PD[0].firstName + " " + PD[0].lastName;
202
203         //figure out when they last logged in
204         TSBDDPeopleDataSet.tbl_Audit_LogonDataTable AD = new
TSBDDPeopleDataSetTableAdapters.tbl_Audit_LogonTableAdapter().getUserLoginTime(PD
[0].userID);
205
206         //then load the fields needed for the active form/mode
207         switch (Request.QueryString["mode"])
208         {
209             case "pass":
210
211                 if (!PD[0].IsloginNull()) { LoginLabel.Text = PD[0].login; }
212                 if (!PD[0].IspasswordNull()) { PasswordLabel.Text = "* * * * *"; }
213
214                 if (!PD[0].IsloginExpiresNull()) { LoginExpiresTB.Text = PD[0].
loginExpires.ToShortDateString(); }
215                 if (!PD[0].IsloginDisabledNull()) { LoginDisabledCB.Checked = PD
[0].loginDisabled; }
216
217                 if (PD[0].IspasswordNull() || PD[0].password.Trim() == "")
218                 {
219                     PasswordRequiredValidator.Enabled = true;
220                 }
221                 else
222                 {
223                     PasswordRequiredValidator.Enabled = false;
224                 }
225                 //If the logon audit data contains some info, update the Last
Login label
226                 if (AD.Rows.Count > 0)
227                 {
228                     LastLoginLabel.Text = AD[0].timestamp.ToString("MMMM dd, yyyy
h:mm tt 'Mountain Time'");
229                 }
230                 break;
231
232             case "priv":
233
234                 //all loading is currently handled in the control databound events
235                 break;
236
237             default:
238                 firstNameTextBox.Text = PD[0].firstName;
239                 if (!PD[0].IsmiddleNameNull()) { middleNameTextBox.Text = PD[0].
middleName; }
240                 lastNameTextBox.Text = PD[0].lastName;
241
242                 if (!PD[0].IspreferredNameNull()) { preferredNameTextBox.Text = PD
[0].preferredName; }
243
244                 if (!PD[0].IsroomLocationNull()) { roomLocationTextBox.Text = PD
[0].roomLocation; }
245                 if (!PD[0].IsaddressNull()) { addressTextBox.Text = PD[0].address; }

```

```
    }
247         if (!PD[0].IscityNull()) { cityTextBox.Text = PD[0].city; }
248         if (!PD[0].IsstateNull()) { stateTextBox.Text = PD[0].state; }
249         if (!PD[0].IszipNull()) { zipTextBox.Text = PD[0].zip; }
250
251         if (!PD[0].IsofficePhoneNull()) { officePhoneTextBox.Text = PD[0].
officePhone; }
252         if (!PD[0].IscellPhoneNull()) { cellPhoneTextBox.Text = PD[0].
cellPhone; }
253         if (!PD[0].IseMailPrimaryNull()) { eMailPrimaryTextBox.Text = PD
[0].eMailPrimary; }
254         if (!PD[0].IseMailSecondaryNull()) { eMailSecondaryTextBox.Text =
PD[0].eMailSecondary; }
255
256         if (!PD[0].IsemployeeNumberNull()) { employeeNumberTextBox.Text =
PD[0].employeeNumber; }
257         if (!PD[0].IsemploymentStatusIDNull()) { EmploymentStatusDropDown.
SelectedValue = PD[0].employmentStatusID.ToString(); }
258         if (!PD[0].IssupervisorNameNull()) { supervisorNameTextBox.Text =
PD[0].supervisorName; }
259
260         if (!PD[0].IsNGIDNull()) { NGIDTextBox.Text = PD[0].NGID; }
261         if (!PD[0].IsstartDateNull()) { startDateTextBox.Text = PD[0].
startDate.ToShortDateString(); }
262
263         if (!PD[0].IsFTENull()) { FTETextBox.Text = Decimal.Round(PD[0].
FTE, 2).ToString(); }
264
265         if (!PD[0].IsorganizationCategoryIDNull()) { OrgCategoryDropDown.
SelectedValue = PD[0].organizationCategoryID.ToString(); }
266         if (!PD[0].IsgroupIDNull()) { GroupIDDropDown.SelectedValue = PD
[0].groupID.ToString(); }
267         if (!PD[0].IslegacySectorIDNull()) { LegacySectorIDDropDown.
SelectedValue = PD[0].legacySectorID.ToString(); }
268
269         if (!PD[0].IsjobTitleIDNull()) { JobTitleDropDown.SelectedValue =
PD[0].jobTitleID.ToString(); }
270         if (!PD[0].IslaborCatIDNull()) { LaborCategoryDropDown.
SelectedValue = PD[0].laborCatID.ToString(); }
271         if (!PD[0].IsuserCatIDNull()) { UserCatIDDropDown.SelectedValue =
PD[0].userCatID.ToString(); }
272
273         if (!PD[0].IsbiographyNull()) { biographyTextBox.Text = PD[0].
biography; }
274         break;
275     }
276 }
277 }
278
279 protected void FormCommandHandler(object sender, CommandEventArgs e)
280 {
281     //Configure data adapters
282     tsbdDataSetTableAdapters.tbl_PeopleTableAdapter peopleTable = new
tsbdDataSetTableAdapters.tbl_PeopleTableAdapter();
283     TSBDPeopleDataSetTableAdapters.tbl_SkillsTableAdapter skillsTable = new
TSBDPeopleDataSetTableAdapters.tbl_SkillsTableAdapter();
284     TSBDPeopleDataSetTableAdapters.tbl_SecurityClearancesTableAdapter
securityClearancesTables = new TSBDPeopleDataSetTableAdapters.
tbl_SecurityClearancesTableAdapter();
285     TSBDPeopleDataSetTableAdapters.MiscQueries miscQueries = new
TSBDPeopleDataSetTableAdapters.MiscQueries();
286     TSBDPeopleDataSetTableAdapters.tbl_RolesTableAdapter rolesTable = new
TSBDPeopleDataSetTableAdapters.tbl_RolesTableAdapter();
287     switch (e.CommandName)
288     {
289
290         case "Save":
```

```
291         // Save the demographics info
292         if (EditView.ActiveViewIndex == 0)
293         {
294             //extract data from form
295             int? UserCatID = Convert.ToInt32(UserCatIDDropDown.SelectedValue);
296             string employeeNumber = employeeNumberTextBox.Text;
297             string NGID = NGIDTextBox.Text;
298             string firstName = firstNameTextBox.Text;
299             string middleName = middleNameTextBox.Text;
300             string lastName = lastNameTextBox.Text;
301             string preferredName = preferredNameTextBox.Text;
302             string eMailPrimary = eMailPrimaryTextBox.Text;
303             string eMailSecondary = eMailSecondaryTextBox.Text;
304             string supervisorName = supervisorNameTextBox.Text;
305             string officePhone = officePhoneTextBox.Text;
306             string cellPhone = cellPhoneTextBox.Text;
307             string roomLocation = roomLocationTextBox.Text;
308             string address = addressTextBox.Text;
309             string city = cityTextBox.Text;
310             string state = stateTextBox.Text;
311             string zip = zipTextBox.Text;
312             string BiographyText = biographyTextBox.Text;
313
314             int? JobTitleID = null;
315             if (JobTitleDropDown.SelectedValue != "") { JobTitleID = Convert.
ToInt32(JobTitleDropDown.SelectedValue); }
316
317             int? EmploymentStatusID = null;
318             if (EmploymentStatusDropDown.SelectedValue != "") {
EmploymentStatusID = Convert.ToInt32(EmploymentStatusDropDown.SelectedValue); }
319
320             decimal? FTE = null;
321             try { FTE = Convert.ToDecimal(FTETextBox.Text); }
322             catch (Exception ex) { /* save FTE as null */ }
323
324             DateTime? startDate = null;
325             try { startDate = DateTime.Parse(startDateTextBox.Text); }
326             catch (Exception ex) { /* save startDate as null */ }
327
328             int? OrgCategoryID = null;
329             if (OrgCategoryDropDown.SelectedValue != "") { OrgCategoryID =
Convert.ToInt32(OrgCategoryDropDown.SelectedValue); }
330
331             int? GroupID = null;
332             if (GroupIDDropDown.SelectedValue != "") { GroupID = Convert.
ToInt32(GroupIDDropDown.SelectedValue); }
333
334             int? LegacySectorID = null;
335             if (LegacySectorIDDropDown.SelectedValue != "") { LegacySectorID =
Convert.ToInt32(LegacySectorIDDropDown.SelectedValue); }
336
337             int? LaborCategoryID = null;
338             if (LaborCategoryDropDown.SelectedValue != "") { LaborCategoryID =
Convert.ToInt32(LaborCategoryDropDown.SelectedValue); }
339
340             int UserID = 0;
341
342             if (UserIDHiddenField.Value != "")
343             {
344                 UserID = Convert.ToInt32(UserIDHiddenField.Value);
345             }
346
347
348             if (UserID == 0)
349             {
350                 //insert new record
351                 Response.Write("Saving new user");
```



```
409     OrgCategoryID, GroupID,                                0, LegacySectorID, officePhone, ✓
410     cellPhone, roomLocation,                             address, city, state, zip,      ✓
411     BiographyText, LaborCategoryID, UserID);
412         //update security clearances
413         //clear current clearances, clarence
414         securityClearancesTables.DeleteUsersSecurityClearances(UserID) ✓
415     ;
416         //add selected clearances
417         if (SecurityClearancesOne.Selected.Trim() != "")
418         {
419             securityClearancesTables.AddUserSecurityClearance(UserID, ✓
420             Convert.ToInt32(SecurityClearancesOne.Selected));
421         }
422         if (SecurityClearancesTwo.Selected.Trim() != "")
423         {
424             securityClearancesTables.AddUserSecurityClearance(UserID, ✓
425             Convert.ToInt32(SecurityClearancesTwo.Selected));
426         }
427
428         //update skills
429         //clear current skills
430         skillsTable.DeleteUsersSkills(UserID);
431         //add selected skills
432         foreach (ListItem skill in SkillsOne.Items)
433         {
434             if (skill.Selected)
435             {
436                 skillsTable.AddUserSkill(UserID, Convert.ToInt32(skill ✓
437                 .Value));
438             }
439         }
440         foreach (ListItem skill in SkillsTwo.Items)
441         {
442             if (skill.Selected)
443             {
444                 skillsTable.AddUserSkill(UserID, Convert.ToInt32(skill ✓
445                 .Value));
446             }
447         }
448     }
449 }
450
451 // Save the password screen info
452 if (EditView.ActiveViewIndex == 1)
453 {
454
455     string P = passwordTextBox.Text.Trim();
456     string PR = passwordRepeatTextBox.Text.Trim();
457
458     int UserID = 0;
459     if (UserIDHiddenField.Value != "")
460     {
461         UserID = Convert.ToInt32(UserIDHiddenField.Value);
462     }
463
464     string Expiry = LoginExpiresTB.Text.Trim();
465     DateTime? expirationDate = null;
466     if (Expiry != "")
467     {
```

```
468         try { expirationDate = DateTime.Parse(Expiry); }
469         catch (Exception ex)
470         {
471             //date was invalid, do not use it
472         }
473     }
474
475     bool? loginDisabled = LoginDisabledCB.Checked;
476
477     //if password was provided and seems OK, go ahead and update it.
478     if (P == PR && UserID != 0 && (P != "" || PR != ""))
479     {
480         miscQueries.UpdatePassword(P, UserID);
481     }
482
483     miscQueries.UpdateLoginInfo(loginDisabled, expirationDate, UserID)
484 ;
485 }
486
487 // Save the privileges info
488 if (EditView.ActiveViewIndex == 2)
489 {
490     int UserID = 0;
491     if (UserIDHiddenField.Value != "")
492     {
493         UserID = Convert.ToInt32(UserIDHiddenField.Value);
494     }
495
496     //update memberships
497     //clear current memberships
498     rolesTable.DeleteUsersMemberships(UserID);
499     foreach (ListItem role in RolesRB.Items)
500     {
501         if (role.Selected)
502         {
503             String roleValue = role.Value;
504             if (roleValue.Trim() != "")
505             {
506                 rolesTable.AddUserRole(UserID, Convert.ToInt32
507                 (roleValue));
508             }
509         }
510     }
511     foreach (ListItem role in RolesCB.Items)
512     {
513         if (role.Selected)
514         {
515             String roleValue = role.Value;
516             if (roleValue.Trim() != "")
517             {
518                 rolesTable.AddUserRole(UserID, Convert.ToInt32
519                 (roleValue));
520             }
521         }
522     }
523
524     //audit changes?
525     break;
526
527     case "Cancel":
528         Response.Write("Cancelling");
529         Response.Redirect("peopleList.aspx");
530         break;
531 }
```

```
532         default:
533             Response.Write("Unknown Command:" + e.CommandName);
534             break;
535     }
536 }
537 }
538
```

VisuMAX © 2008