```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.ComponentModel;
 4  using System.Data;
 5  using System.Drawing;
 6  using System.IO;
 7  using System.Linq;
 8  using System.Text;
 9  using System.Threading.Tasks;
10  using System.Windows.Forms;
11
12  namespace ProductPerformance
13  {
14      public partial class frmObservations : Form
15      {
16          private string _selectedPerson = string.Empty;
17          private string _selectedProductType = string.Empty;
18          private string _selectedProduct = string.Empty;
19          private string _selectedDate = string.Empty;
20          private string _selectedContinuous = "False";
21          private string _selectedSize = "250ml";
22          private string _selectedObservation1 = "Very poor";
23          private string _selectedObservation2 = "None";
24          private string _selectedObservation3 = "None";
25
26          private Observation _currentObservation = new Observation();
27
28          private List<string> _dataState = new List<string>();
29          private bool _dataGood = false;
30
31
32          public frmObservations()
33          {
34              InitializeComponent();
35
36              //-- Populate the combo box with Persons
37              List<string> people = PersonDB.GetAllPersonShortData();
38              foreach (var value in people)
39              {
40                  cboPerson.Items.Add(value);
41              }
42
43              //-- Populate the ProductType combo
44              List<string> types = Lists.GetProductTypes();
45              foreach (var value in types)
46              {
47                  cboProductType.Items.Add(value);
48              }
49
50              //-- Populate the product sizes list
51              List<string> sizes = Lists.ProductSizes();
52              foreach (var value in sizes)
53              {
54                  cboSize.Items.Add(value);
55              }
56
```

```
57              //-- Populate the Observation1 options
58              List<string> obs1List = Lists.ObservationOneOptions();
59              foreach (var value in obs1List)
60              {
61                  cboObservation1.Items.Add(value);
62              }
63
64              //-- Populate Observations 2 and 3
65              List<string> obsOther = Lists.ObservationOtherOptions();
66              foreach (var value in obsOther)
67              {
68                  cboObservation2.Items.Add(value);
69                  cboObservation3.Items.Add(value);
70              }
71
72              //-- Date
73              _selectedDate = this.dtpFirstUse.Value.ToString("dd/MM/yyyy");
74
75              //-- Continuous use
76              _selectedContinuous = "Not continuous use";
77
78              //-- Set control values to default
79              ResetFormControls();
80
81              LoadObservationsListBox();
82          }
83
84          private void ResetFormControls()
85          {
86              cboPerson.SelectedIndex = -1;
87              cboProductType.SelectedIndex = -1;
88              cboProduct.SelectedIndex = -1;
89              cboSize.SelectedIndex = -1;
90              lblOtherSize.Visible = false;
91              txtOtherSize.Text = String.Empty;
92              txtOtherSize.Visible = false;
93              dtpFirstUse.Value = DateTime.Now;
94              chkContinuousUse.Checked = false;
95              cboObservation1.SelectedIndex = -1;
96              txtOtherObservation1.Text = String.Empty;
97              txtOtherObservation1.Visible = false;
98              cboObservation2.SelectedIndex = -1;
99              txtOtherObservation2.Text = String.Empty;
100             txtOtherObservation2.Visible = false;
101             cboObservation3.SelectedIndex = -1;
102             txtOtherObservation3.Text = String.Empty;
103             txtOtherObservation3.Visible = false;
104         }
105
106
107
108         private void exitToolStripMenuItem_Click(object sender, EventArgs e)
109         {
110             Utility.CloseApplication();
111         }
112
```

```
113         private void Clear_Click(object sender, EventArgs e)
114         {
115             ResetFormControls();
116         }
117
118
119         private void ClearObservationsListBox()
120         {
121             this.lbObservations.Items.Clear();
122         }
123
124         private void LoadObservationsListBox()
125         {
126             using (StreamReader reader = new StreamReader(@"D:
                \observations.txt"))
127             {
128                 while (true)
129                 {
130                     string line = reader.ReadLine();
131                     if (line == null)
132                     {
133                         break;
134                     }
135
136                     string[] fields = line.Split(',');
137                     Observation observation = new Observation()
138                     {
139                         Person = fields[0],
140                         ProductType = fields[1],
141                         Product = fields[2],
142                         ProductSize = fields[3]
143
144                     };
145
146                     string listItem = observation.ObservationDataShort();
147
148                     this.lbObservations.Items.Add(listItem);
149
150                 }
151             }
152         }
153
154         private void btnSummary_Click(object sender, EventArgs e)
155         {
156             var formSummary = new frmSummary();
157             formSummary.Show();
158             this.Hide();
159         }
160
161         // -- Select products and fill combo based on ProductType
162         private void cboProductType_DropDownClosed(object sender, EventArgs e)
163         {
164             //-- Track product type selected
165             string productType = cboProductType.GetItemText
                (this.cboProductType.SelectedItem);
166             _selectedProductType = productType.Trim();
```

```
167
168             cboProduct.SelectedIndex = -1;
169             cboProduct.Items.Clear();
170
171             List<string> products = ProductCatalogueDB.ProductsByType
                  (_selectedProductType);
172
173             foreach (var value in products)
174             {
175                 cboProduct.Items.Add(value);
176             }
177         }
178
179         // -- Get 'Other' value selected and show text box
180         private void cboSize_DropDownClosed(object sender, EventArgs e)
181         {
182             string size = cboSize.GetItemText(cboSize.SelectedItem);
183             _selectedSize = size.Trim();
184
185             if (size == "Other")
186             {
187                 txtOtherSize.Visible = true;
188                 lblOtherSize.Visible = true;
189             }
190             else
191             {
192                 lblOtherSize.Visible = false;
193                 txtOtherSize.Visible = false;
194             }
195         }
196
197         private void cboPerson_DropDownClosed(object sender, EventArgs e)
198         {
199             var person = cboPerson.GetItemText(cboPerson.SelectedItem);
200             _selectedPerson = person.Trim();
201         }
202
203         private void cboProduct_DropDownClosed(object sender, EventArgs e)
204         {
205             string product = cboProduct.GetItemText
                  (this.cboProduct.SelectedItem);
206             _selectedProduct = product.Trim();
207         }
208
209         // -- Update module wide variable if selection changed
210         private void dtpFirstUse_ValueChanged(object sender, EventArgs e)
211         {
212             _selectedDate = this.dtpFirstUse.Value.ToString("dd/MM/yyyy");
213         }
214
215         private void chkContinuousUse_CheckStateChanged(object sender,
              EventArgs e)
216         {
217             if (chkContinuousUse.Checked)
218             {
219                 _selectedContinuous = "In continuous Use";
```

```
220             }
221             else
222             {
223                 _selectedContinuous = "Not continuous use";
224 
225             }
226         }
227 
228 
229 
230         // -- Observation combos - deal with other being selected item
231 
232         private void cboObservation1_DropDownClosed(object sender, EventArgs
              e)
233         {
234             string observationOne = cboObservation1.GetItemText
                  (this.cboObservation1.SelectedItem);
235             _selectedObservation1 = observationOne.Trim();
236 
237             if (_selectedObservation1 == "Other")
238             {
239                 txtOtherObservation1.Visible = true;
240             }
241             else
242             {
243                 txtOtherObservation1.Visible = false;
244             }
245         }
246 
247 
248         private void cboObservation2_DropDownClosed_1(object sender, EventArgs
              e)
249         {
250             string observationTwo = cboObservation2.GetItemText
                  (this.cboObservation2.SelectedItem);
251             _selectedObservation2 = observationTwo.Trim();
252 
253             if (_selectedObservation2 == "Other")
254             {
255                 txtOtherObservation2.Visible = true;
256             }
257             else
258             {
259                 txtOtherObservation2.Visible = false;
260             }
261         }
262 
263         private void cboObservation3_DropDownClosed_1(object sender, EventArgs
              e)
264         {
265             string observationThree = cboObservation3.GetItemText
                  (this.cboObservation3.SelectedItem);
266             _selectedObservation3 = observationThree.Trim();
267 
268             if (_selectedObservation3 == "Other")
269             {
```

```
270                 txtOtherObservation3.Visible = true;
271             }
272             else
273             {
274                 txtOtherObservation3.Visible = false;
275             }
276         }
277 
278 
279         // -- Method to harvest all observation data
280         // -- If 'Other is and option, must get values from the control
281         // -- else get from module wide variables previously set when
282         // -- options were changed
283         private void HarvestObservationData()
284         {
285             _currentObservation = new Observation();
286 
287             _dataState.Clear();
288 
289             //-- Person
290             if (_selectedPerson == string.Empty)
291             {
292                 //MessageBox.Show("You must select a person");
293                 _dataState.Add("Bad");
294             }
295             else
296             {
297                 _currentObservation.Person = _selectedPerson;
298                 _dataState.Add("Good");
299             }
300 
301             //-- ProductType
302             if (_selectedProductType == String.Empty)
303             {
304                 //MessageBox.Show("You must select a type");
305                 _dataState.Add("Bad");
306             }
307             else
308             {
309                 _currentObservation.ProductType = _selectedProductType;
310                 _dataState.Add("Good");
311             }
312 
313             //-- Product
314             if (_selectedProduct == String.Empty)
315             {
316                 _dataState.Add("Bad");
317             }
318             else
319             {
320                 _currentObservation.Product = _selectedProduct;
321                 _dataState.Add("Good");
322                 ;
323             }
324 
325             //-- Size
```

```
326             if (_selectedSize == "Other")
327             {
328                 if (txtOtherSize.Text != "")
329                 {
330                     _selectedSize = txtOtherSize.Text;
331                 }
332             }
333 
334             _currentObservation.ProductSize = _selectedSize;
335 
336             //-- First use. Should always have a value
337             _currentObservation.FirstUse = _selectedDate;
338 
339             //-- Continuous
340             _currentObservation.Continuous = _selectedContinuous;
341 
342             //-- Observation1
343             if (_selectedObservation1 == "Other")
344             {
345                 _currentObservation.Observation1 = txtOtherObservation1.Text;
346             }
347             else
348             {
349                 _currentObservation.Observation1 = _selectedObservation1;
350             }
351 
352             //-- Observation2
353             if (_selectedObservation2 == "Other")
354             {
355                 _currentObservation.Observation2 = txtOtherObservation2.Text;
356             }
357             else
358             {
359                 _currentObservation.Observation2 = _selectedObservation2;
360             }
361 
362             //-- Observation3
363             if (_selectedObservation3 == "Other")
364             {
365                 _currentObservation.Observation3 = txtOtherObservation3.Text;
366             }
367             else
368             {
369                 _currentObservation.Observation3 = _selectedObservation3;
370             }
371         }
372 
373 
374         // --- Write the observation to the database
375         private void btnAddObservation_Click(object sender, EventArgs e)
376         {
377             //-- Populate class with form data
378             HarvestObservationData();
379             //-- Try the write
380             if (ObservationDB.WriteObservation(_currentObservation))
381             {
```

```
382
383                 MessageBox.Show("Observation written to file");
384             }
385
386             ClearObservationsListBox();
387             LoadObservationsListBox();
388
389             //_currentObservation = null;
390             ResetFormControls();
391         }
392     }
393 }
394
```