Extra 16-2 Create a Product Maintenance application

In this exercise, you’ll create an application that lets you maintain the products in the Products table.



Create and configure the ListView control

1. Open the XEx16ProductMaint application in your exercises\_extra directory. This application contains the starting page with the validation summary and label controls for error messages, along with the database, image, and style sheets used by the page.
2. Add a ListView control to the page below the div element that contains the validation summary and label controls. Create a data source for the control that contains all of the rows from the Products table, sorted by the ProductID column. Be sure to generate Insert, Update, and Delete statements, but don’t use optimistic concurrency.
3. Configure the ListView control so it uses flow layout and provides for deleting, editing, inserting, and paging with a next/previous pager.
4. Modify the DataPager control so only one product is displayed at a time. Then, run the application to see how it looks, and test it to make sure it works correctly.

Format the ListView with Bootstrap CSS classes

1. In Source view, delete the AlternatingItem, EmptyData, and SelectedItem templates.
2. Assign the row CSS class to the first div element in the Layout template, assign the text-center CSS class to the second div element (the one that contains the data pager), and assign the btn and btn-default CSS classes to the buttons in the data pager using the ButtonCssClass property.

Unless otherwise noted, apply steps 7 through 17 to all three templates.

1. Find the span element that contains the generated text and controls, and change it to a div element. Then, assign the col-sm-6 CSS class to this element.
2. Add a Bootstrap list group with two list items inside this div element. If necessary, use the second example in figure 10-10 for guidance.
3. Assign the “bg-halloween” CSS class to the div element for the first list item, and add header text to it like the text shown above.
4. Add a table to the div element for the second list item, and assign the table CSS class to the table. The table should contain nine rows and two columns in the Item template, but nine rows and three columns in the other templates.
5. Move the product items to the table rows. Put the text in the first column, the corresponding label or text box in the second column, and delete the line break. Assign the form-control CSS class to the text boxes.
6. Modify the last row so it has one table cell that spans all the columns (two or three, depending on the template). Then, move the template buttons to this cell, delete the line breaks, and assign the btn and btn-default classes to these buttons.
7. Run the application to see how it looks now.

Adjust the template controls and add validation

1. Make the text boxes for the long description columns multiline. In addition, apply currency format to the unit price column, but only in the Item template.
2. Add another data source to the page below the data source for the ListView control. Configure this data source to retrieve the CategoryID and LongName columns from the Categories table sorted by the LongName column.
3. Replace the text boxes for the Category columns in the EditItem and InsertItem templates with drop-down lists. Bind the lists by setting the DataSourceID attribute to the ID of the data source you just created, the DataTextField attribute to LongName, the DataValueField attribute to CategoryID, and the SelectedValue attribute to a Bind expression. Finally, assign the form-control CSS class to the drop-down list.
4. Add the appropriate validators for the text boxes in the third table cell of the EditItem and InsertItem templates. These validators should display an asterisk to the right of the text box and an error message in the validation summary control. Then, associate the template’s Update or Insert button with the correct validation group.

Add the C# code and test the application

1. Add the C# code that’s required to detect database errors when a row is updated, deleted, or inserted.
2. Run the application to see how it looks, and test it to be sure it works.