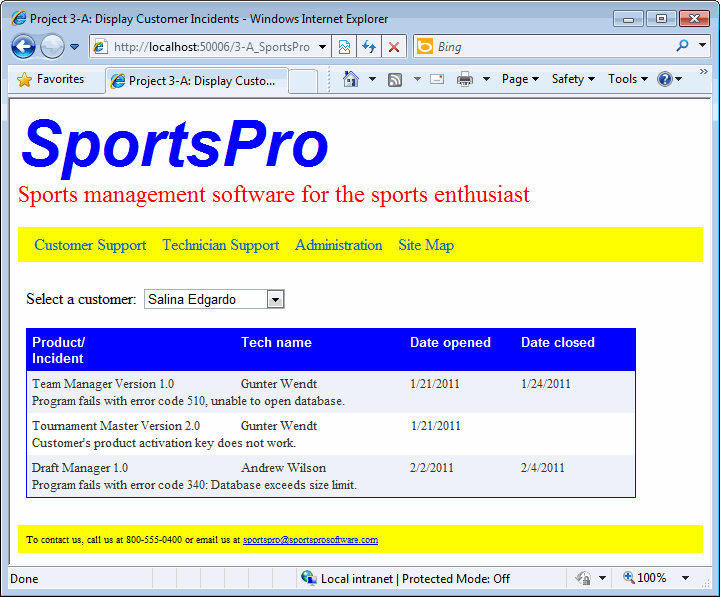
Display customer incidents using a DataList

* Repair any errors from Cases 1 & 2. Make a copy of the project named SportsPro2—LastName, name it SportsPro3—LastName.
* Create a page named **CustomerIncidentDisplay.aspx** that displays the incidents for a selected customer in a **DataList**. This page will require two SQL data sources. (*Required reading: chapter 13*)

The design of the *Customer Incident Display* page



Operation

* When the user selects a customer from the drop-down list, all the incidents for that customer are displayed in the DataList.
* A message should be displayed in the DataList if no incident exists for the selected customer.

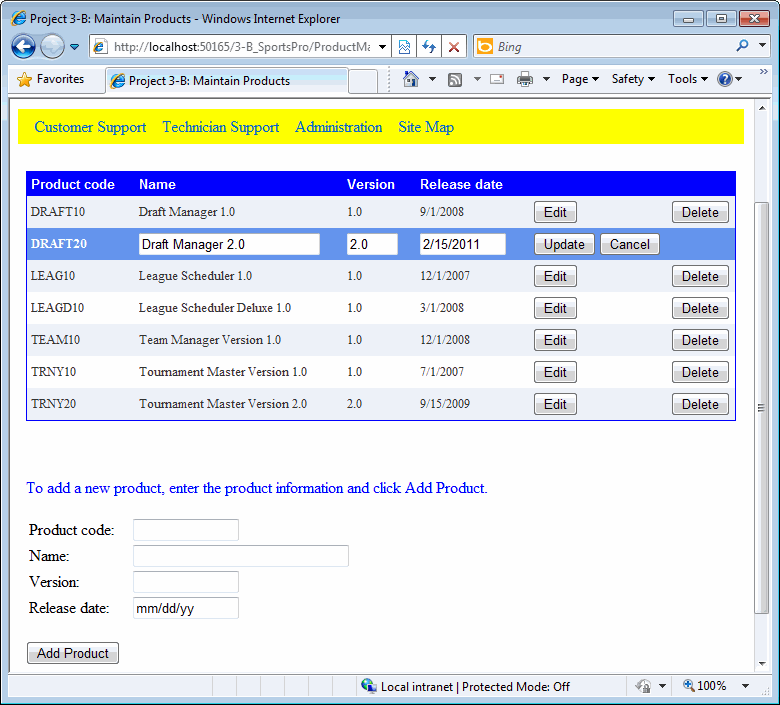
Specifications

* The Customers drop-down list is bound to a SQL Server Data Source that retrieves the CustomerID and Name columns from the Customers table and sorts the rows by the Name column. Save the connection string for this Data Source in the web.config file.
* The SQL Data Source for the DataList joins data from three tables (Incidents, Technicians, and Products) and includes only the incidents for the selected customer. Name it **dsDataList**.
* The Item **template** for the DataList control should contain a table with two rows. The first row includes column titles. The second row should contain a single cell that spans all four columns. To do that, research the colspan attribute of the td element.
* Use CSS3 to format the **DataList** any way you like, as long as it can be considered professional quality.

Maintain products using a GridView

Create a page named **ProductMaintenance.aspx** that lets a user modify and delete existing products using a **GridView** control. This page will also let the user add new products by entering the product information into Text Boxes. (*Required reading: chapters 13 and 14*)

The design of the *Product Maintenance* page



Operation

* When the user clicks the **Edit** button for a product, the product data is displayed in edit mode. The user can modify the data and click the **Update** button to update the database. The user can also click the **Cancel** button to cancel the edit operation.
* When the user clicks the **Delete** button for a product, the product is deleted from the database. Add a confirmation popup that warns the user that the product is about to be deleted.
* When the user enters the data for a new product into the Text Boxes below the GridView and clicks the **Add Product** button, that product is added to the database and displayed in the GridView.

Specifications

* The SQL data source for this project should retrieve all the columns and rows from the Products table, and it should sort the rows by the ProductCode column. It should generate Insert, Update, and Delete statements and use optimistic concurrency (aka, CompareAllValues). Name it **dsGridView**.
* The products should be displayed in a **GridView** control that allows editing all the columns except for the ProductCode column.
* Validate the data the user enters into the grid to be sure that a product name, version, and release date are provided and that the date is in a valid format. To do that, you’ll need to use templates. Display a red asterisk next to any field that has an error, and display the error messages in a Validation Summary control below the grid.
* Validate the data the user enters for a new product to be sure that a value is entered for each field and that the date is in a valid format. Display an appropriate error messages to the right of any fields in error.
* When a product is updated or deleted, display an error message at the top of the page if an **exception** occurs or if the operation could not be performed.
* When a product is added, display an error message at the top of the page if an **exception** occurs. Otherwise, initialize the Text Boxes to prepare for the next entry.
* Be sure to set the **EnableViewState** property of the **Label** you use to display error messages to False so it will be reset to its initial value each time the page is loaded instead of refilling with previous data.

Style and formatting

* You can format the GridView any way you like. Your formatting will be graded on a scale of (0-10) based on such things as using images instead of button; fonts choice (serif fonts shouldn’t be used on web pages); color choices; using controls in the EditTemplates to restrict users to valid choices; page position; using box shadows. Be sure all content displays well on your background image, you can change your background image if needed.