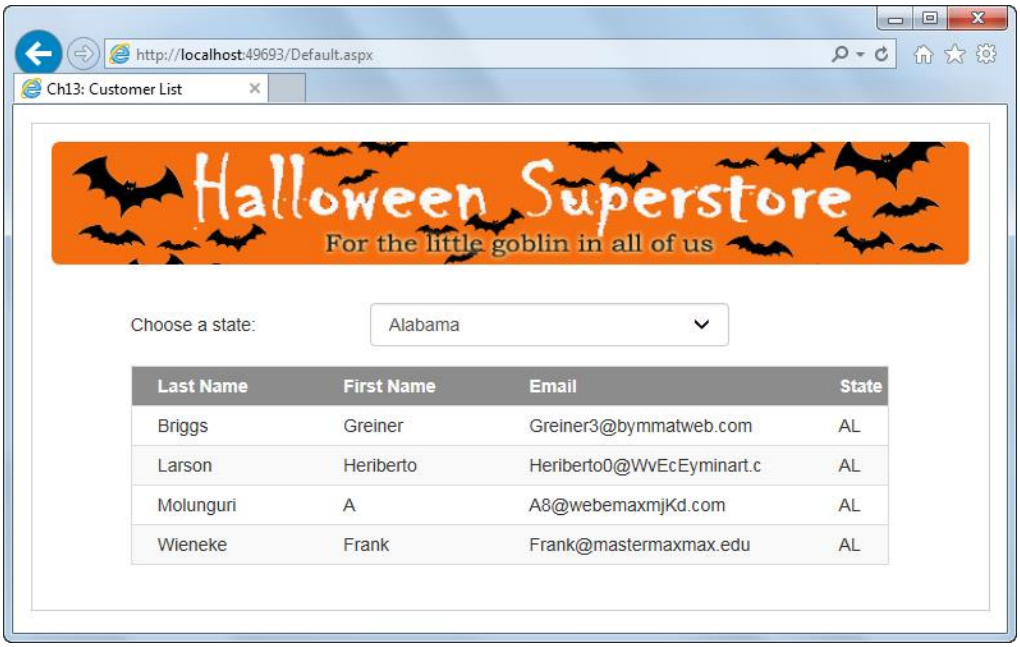
Extra 13-2 Create an application that displays customers by state

In this exercise, you'll create an application that lets the user select a state from a drop-down list and then displays the customers in that state in a data list.



Create the drop-down list and its data source

1. Open the XEx13CustomerList application in your exercises_extra directory. This application contains the starting page and the database, image, and style sheets used by the page.
2. Open the Default.aspx file in Source View, and add a drop-down list inside the div that indicates it should contain the state drop-down list. After that, you can continue working in Source View or switch to Design View.
3. Configure the data source for the drop-down list so it includes the StateCode and StateName columns from the States tables, sorted by state name. Be sure to store the connection in the Web.config file.
4. Bind the drop-down list to the data source so the state name is displayed in the list and the state code is returned by the SelectedValue property of the list.
5. Change the name of the drop-down list to ddlState and set its AutoPostBack property to True. Finally, set its CssClass property to form-control.
6. Run the application and review the data in the drop-down. Select a few states to make sure the page posts back to the server. Stop the application when you're done.

Create another data source and bind it to a data list

7. In Source View, add a DataList control inside the div that indicates it should contain the customer data list. Then, change its name to dlCustomers. After that, you can continue in Source View or switch to Design View. You should know, though, that it can be easier to work with Bootstrap CSS in Source View.

8. Configure the data source for the data list so it uses the connection string that's in the Web.config file and so it selects the LastName, FirstName, Email, and State columns from the Customers table, sorted by last name and then by first name. The data source should select only the customers in the state that's selected from the drop-down list.
9. Run the application and review the data in the data list and the default layout. Select a few states to make sure that the customer list is working right. Stop the application when you're done.

Modify the templates for the data list

10. Add a Header template, and add text that identifies each column. You can type this by hand or copy the text from the Item template.
11. In the Item template, remove the text that identifies the column names. When you're done, this template should contain only the data-bound ASP.NET label controls that are generated by default.
12. Run the application, review the data list, and see that the formatting is worse than before. Stop the application when you're done.

Format the data list with Bootstrap and a custom CSS class

13. In Source View, assign these Bootstrap CSS classes to the DataList control: table, table-bordered, table-striped, and table-condensed.
14. In the Header template, wrap the column names inside span tags. Then, assign the col-xs-3 class to the span tags for first and last name, assign the col-xs-5 calss to the span tags for the email, and assign the col-xs-1 class to the span tag for the state. Finally, delete any other HTML, such as br tags, that might be in the template.
15. In the Item template, assign a Bootstrap column class to each Label control so it matches the column class of the corresponding span tag in the Header template. Then, just as you did with the Header template, delete any other HTML.
16. Add a HeaderStyle element to the DataList control, and an assign the bg-halloween class to it. If you're curious, you can review this custom CSS class in the site.css file.
17. Run the application to make sure it's formatted properly.