

Spring MVC Tutorial – Contact List Application

Step 07: Client-Side View Functionality – Home Screen

Copyright © 2016 The Learning House, Inc.

All rights reserved. No part of these materials may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of The Learning House. For permission requests, write to The Learning House, addressed "Attention: Permissions Coordinator," at the address below.

The Learning House

427 S 4th Street #300

Louisville KY 40202

Step 07: Client-Side View Functionality – Home Screen

Overview

In this step of the tutorial, we will start to develop the client-side functionality of the Home screen. In some cases, this version will use canned data created in JavaScript files and in other cases, it will use placeholder data. We will implement the following features and associated JavaScript code:

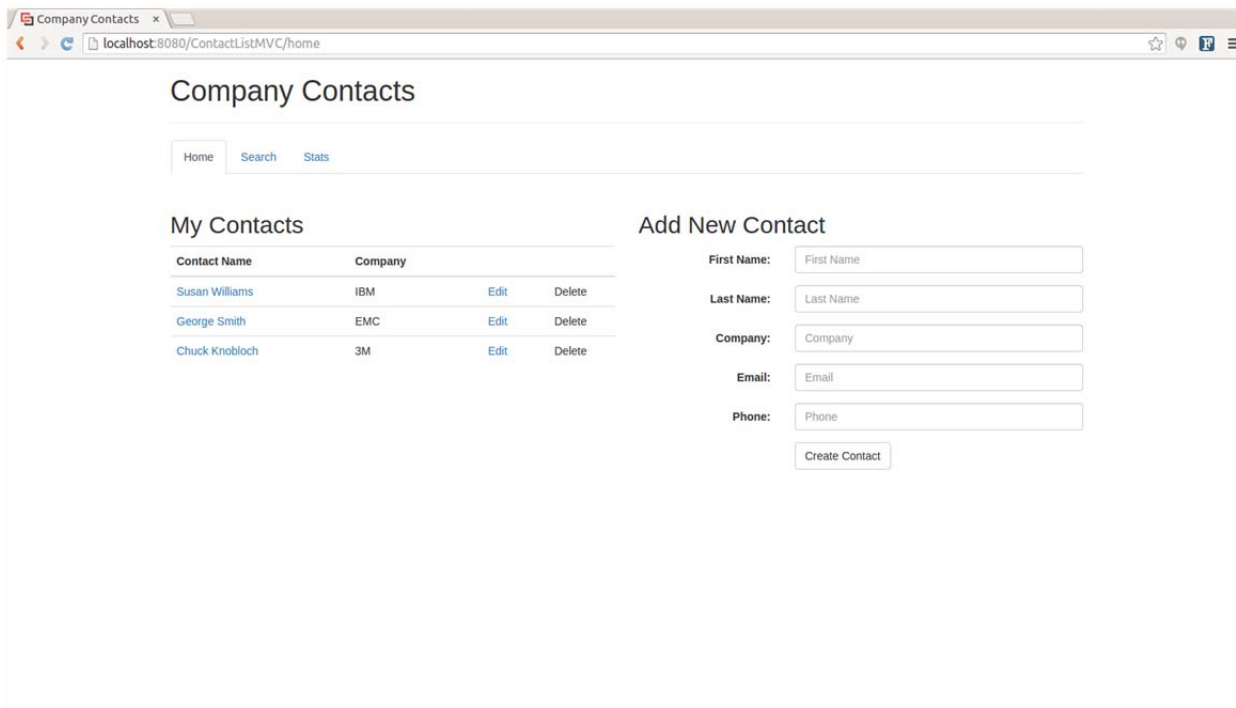
1. Main page layout — summary table of contact information and the New Contact form
2. Contact Details modal dialog
3. Edit Contact modal dialog

Main Layout

As outlined in Step 00, the main layout of the Home screen consists of two main components:

1. Summary table of contact information
2. New Contact form

We will take advantage of the Bootstrap grid layout system to place these components on the screen. When we're finished, the screen will look like this:



The screenshot shows a web browser window titled "Company Contacts" with the URL "localhost:8080/ContactListMVC/home". The page has a navigation bar with "Home", "Search", and "Stats" links. The main content area is divided into two columns. The left column, titled "My Contacts", contains a table with contact information. The right column, titled "Add New Contact", contains a form with fields for First Name, Last Name, Company, Email, and Phone, along with a "Create Contact" button.

Contact Name	Company		
Susan Williams	IBM	Edit	Delete
George Smith	EMC	Edit	Delete
Chuck Knobloch	3M	Edit	Delete

Add New Contact

First Name:

Last Name:

Company:

Email:

Phone:

Summary Table and New Contact Form

Our first step will be to add the Summary table and the New Contact form. As mentioned above, we'll use the Bootstrap grid layout to put these page elements side by side, just beneath the navigation bar. Initially, our table will be empty — we'll add some data in just a bit. For now, modify your **home.jsp** file so that it looks like this:

Notes:

1. We're adding a row div to our container, which will hold the Summary table and the New Contact form.
2. Add a col div to hold the Summary table. Note that we're just filling in the headers and all content rows will be dynamically generated.
3. This tbody element will contain all of the dynamically-generated content rows.
4. Add a col div to hold the New Contact form.
5. Include references to the jQuery and Bootstrap JavaScript files.

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="s" uri="http://www.springframework.org/tags" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <title>Company Contacts</title>
    <!-- Bootstrap core CSS -->
    <link href="${pageContext.request.contextPath}/css/bootstrap.min.css" rel="stylesheet">

    <!-- SWC Icon -->
    <link rel="shortcut icon" href="${pageContext.request.contextPath}/img/icon.png">
  </head>
  <body>
    <div class="container">
      <h1>Company Contacts</h1>
      <hr/>
      <div class="navbar">
        <ul class="nav nav-tabs">
          <li role="presentation" class="active">
            <a href="${pageContext.request.contextPath}/home">Home</a>
          </li>
          <li role="presentation">
            <a href="${pageContext.request.contextPath}/search">Search</a>
          </li>
          <li role="presentation">
            <a href="${pageContext.request.contextPath}/stats">Stats</a>
          </li>
        </ul>
      </div>
    </div>
  </body>
</html>
```

```

<!--
    #1: Start Changes
    Add a row to our container - this will hold the summary table and the new
    contact form.
-->
<div class="row">
    <!--
        #2: Add a col to hold the summary table - have it take up half the row
    -->
    <div class="col-md-6">
        <h2>My Contacts</h2>
        <table id="contactTable" class="table table-hover">
            <tr>
                <th width="40%">Contact Name</th>
                <th width="30%">Company</th>
                <th width="15%"></th>
                <th width="15%"></th>
            </tr>
            <!--
                #3: This holds the list of contacts - we will add rows dynamically
                using jQuery
            -->
            <tbody id="contentRows"></tbody>
        </table>
    </div> <!-- End col div -->
    <!--
        #4: Add col to hold the new contact form - have it take up the other
        half of the row
    -->
    <div class="col-md-6">
        <h2>Add New Contact</h2>
        <form class="form-horizontal" role="form">
            <div class="form-group">
                <label for="add-first-name" class="col-md-4 control-label">
                    First Name:
                </label>
                <div class="col-md-8">
                    <input type="text"
                        class="form-control"
                        id="add-first-name"
                        placeholder="First Name"/>
                </div>
            </div>
            <div class="form-group">
                <label for="add-last-name" class="col-md-4 control-label">
                    Last Name:
                </label>
                <div class="col-md-8">
                    <input type="text"
                        class="form-control"
                        id="add-last-name"
                        placeholder="Last Name"/>
                </div>
            </div>
        </form>
    </div>

```

```

<div class="form-group">
  <label for="add-company" class="col-md-4 control-label">
    Company:
  </label>
  <div class="col-md-8">
    <input type="text"
      class="form-control"
      id="add-company"
      placeholder="Company" />
  </div>
</div>
<div class="form-group">
  <label for="add-email" class="col-md-4 control-label">Email:</label>
  <div class="col-md-8">
    <input type="email"
      class="form-control"
      id="add-email"
      placeholder="Email" />
  </div>
</div>
<div class="form-group">
  <label for="add-phone" class="col-md-4 control-label">Phone:</label>
  <div class="col-md-8">
    <input type="tel"
      class="form-control"
      id="add-phone"
      placeholder="Phone" />
  </div>
</div>
<div class="form-group">
  <div class="col-md-offset-4 col-md-8">
    <button type="submit"
      id="add-button"
      class="btn btn-default">
      Create Contact
    </button>
  </div>
</div>
</form>
</div> <!-- End col div -->

</div> <!-- End row div -->

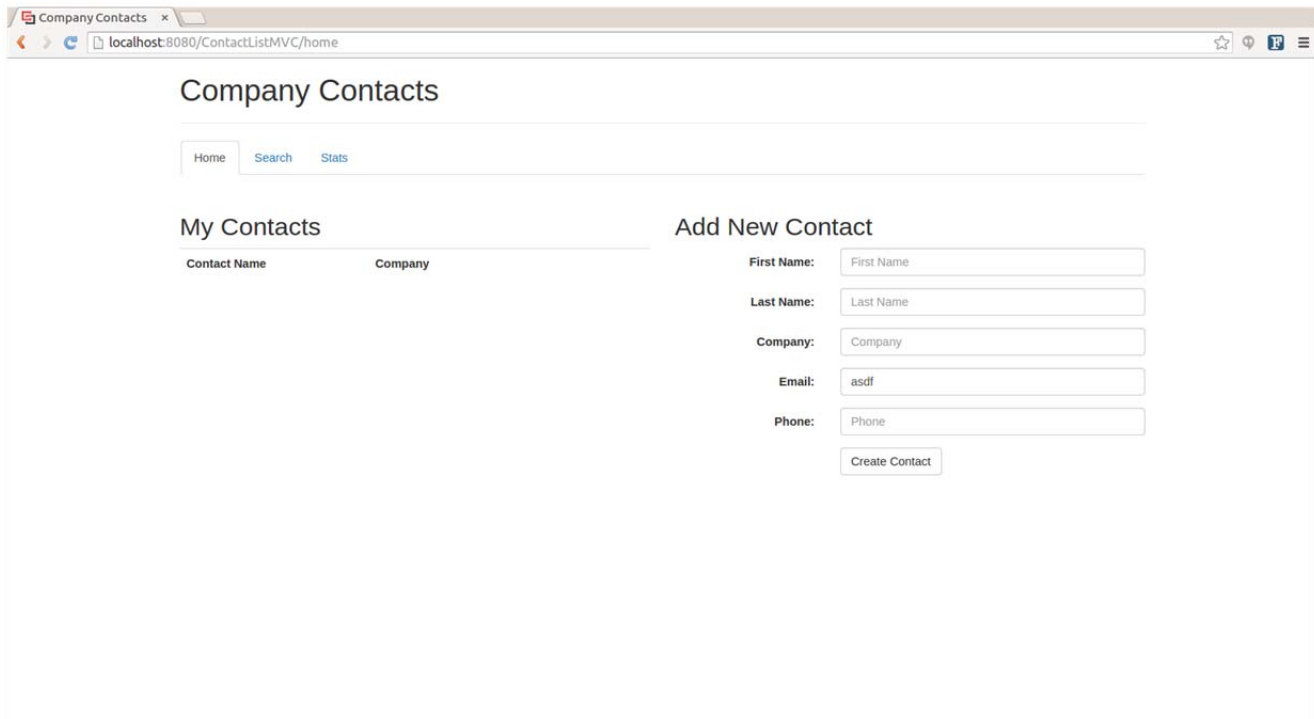
</div>

<!-- #5: Placed at the end of the document so the pages load faster -->
<script src="{pageContext.request.contextPath}/js/jquery-1.11.1.min.js"></script>
<script src="{pageContext.request.contextPath}/js/bootstrap.min.js"></script>

</body>
</html>

```

After these changes, your Home screen should look like this:



Rendering Summary Contact Data

Now that we have the structure in place, we need to put some data into our table. Rather than simply hardcoding the data in HTML, we are going to create JSON contact objects containing canned test data which will be rendered to the table via JavaScript when the page loads. Later on, we'll replace this canned test data with JSON objects retrieved via an Ajax call to the server.

The next step in the process is the creation of a JavaScript file to hold our data and rendering logic. Create a file called **contactList.js** in the **js** folder. Initially, our file will contain:

1. Canned JSON contact test data
2. A function called `loadContacts` that renders the test data into our Summary table
3. A function called `clearContactTable` that removes all rows from the Summary table (this function is called by `loadContacts` before it re-renders the table)
4. A document-ready function that calls the `loadContacts` function when the page loads

Add the following code to your **contactList.js** file:

```

// Document ready function
$(document).ready(function () {
    loadContacts();
});

//=====
// FUNCTIONS
//=====

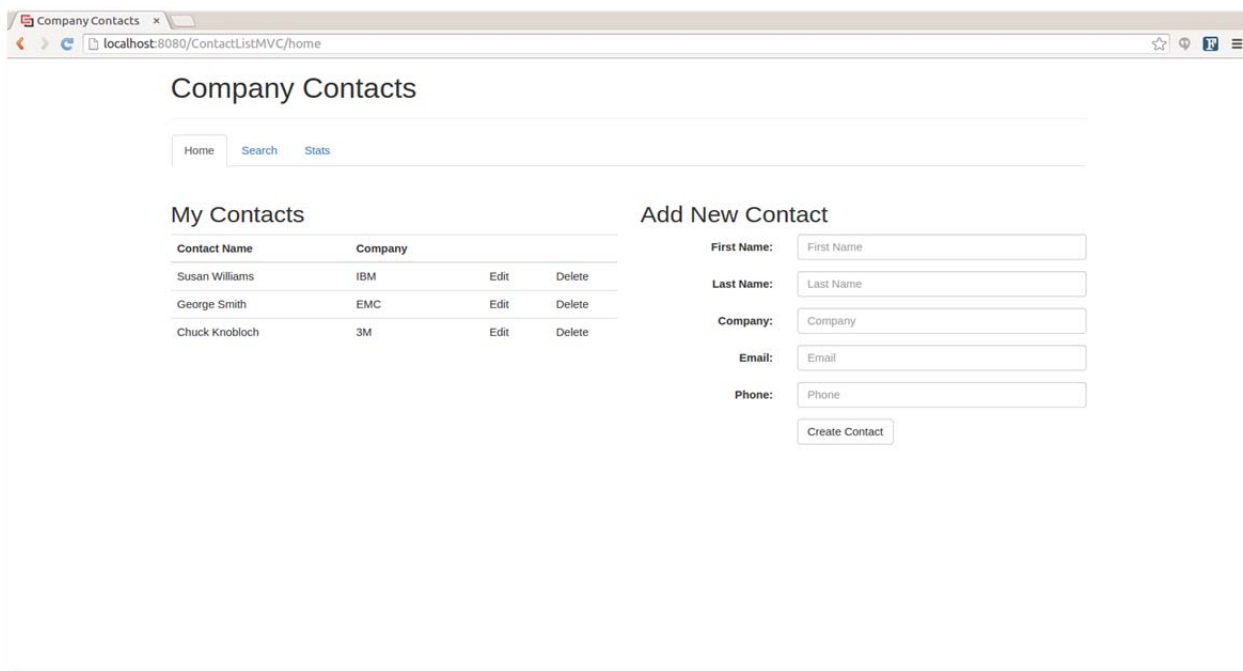
// Load contacts into the summary table
function loadContacts() {
    // Clear the previous list
    clearContactTable();
    // Grab the tbody element that will hold the new list of contacts
    var cTable = $('#contentRows');
    // Iterate through each of the JSON objects in the test contact data
    // and render to the summary table
    $.each(testContactData, function (index, contact) {
        cTable.append($('

```


Now, we need to add a reference our new JavaScript file (**contactList.js**) in **home.jsp** so that our new logic will run on page load. Make sure the reference to **contactList.js** comes after the reference to the jQuery JavaScript file. The reference block should now look like this:

```
<!-- Placed at the end of the document so the pages load faster -->
<script src="${pageContext.request.contextPath}/js/jquery-1.11.1.min.js"></script>
<script src="${pageContext.request.contextPath}/js/bootstrap.min.js"></script>
<script src="${pageContext.request.contextPath}/js/contactList.js"></script>
```

If all went well, your Home screen should now look like this:



Viewing and Editing Contacts

Now, we will add the functionality that allows us to view and/or edit detailed information about a particular contact. Per our design requirement, the Contact Details and the Edit Contact forms will be displayed as Bootstrap modal dialogs. In this step, we are going to put in the plumbing that shows the Details dialog when the name of the contact is clicked and the Edit dialog when the Edit link is clicked. For now, both these dialogs will contain static dummy data — in a later step, we'll add code that populates these dialogs with live data from the server.

Details on the workings of Bootstrap modals can be found here: <http://getbootstrap.com/javascript/#modals>. We will be controlling the modals via data attributes rather than JavaScript and we will be taking advantage of some of the custom events when loading live data into these components. See <http://getbootstrap.com/javascript/#modals-usage> for further details.

Contact Details

First, we'll add the ability to show the Contact Details modal. We'll do this in three steps:

1. Create the HTML for the modal
2. Create the JavaScript that loads data into the modal
3. Modify the code that renders contact information into the Summary table so that clicking on the rendered contact name will cause the Contact Details modal to display

Add the code on the next page to your **home.jsp** file. Insert it just above the script tags that include the jQuery, Bootstrap, and Contact List JavaScript files.

```

<!-- Details Modal -->
<!--
    The aria- attributes are for Accessible Rich Internet Applications standards -
    their purpose is to make web applications more accessible to people with disabilities.
-->
<div class="modal fade" id="detailsModal" tabindex="-1" role="dialog"
    aria-labelledby="detailsModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal">
                    <span aria-hidden="true">&times;</span>
                    <span class="sr-only">Close</span>
                </button>
                <h4 class="modal-title" id="detailsModalLabel">Contact Details</h4>
            </div>
            <div class="modal-body">
                <h3 id="contact-id"></h3>
                <table class="table table-bordered">
                    <tr>
                        <th>First Name:</th>
                        <td id="contact-firstName"></td>
                    </tr>
                    <tr>
                        <th>Last Name:</th>
                        <td id="contact-lastName"></td>
                    </tr>
                    <tr>
                        <th>Company:</th>
                        <td id="contact-company"></td>
                    </tr>
                    <tr>
                        <th>Phone:</th>
                        <td id="contact-phone"></td>
                    </tr>
                    <tr>
                        <th>Email:</th>
                        <td id="contact-email"></td>
                    </tr>
                </table>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-default" data-dismiss="modal">
                    Close
                </button>
            </div>
        </div>
    </div>
</div>

```

Next, modify your **contactList.js** file so that it contains the following:

Notes:

1. This code runs when the modal is shown. It will load the data from `dummyDetailsContact` into the modal for display.
2. This is the dummy data for contact details.

```
// Document ready function
$(document).ready(function () {
    loadContacts();
});

//=====
// FUNCTIONS
//=====

// Load contacts into the summary table
function loadContacts() {
    // Clear the previous list
    clearContactTable();
    // Grab the tbody element that will hold the new list of contacts
    var cTable = $('#contentRows');
    // Iterate through each of the JSON objects in the test contact data
    // and render to the summary table
    $.each(testContactData, function (index, contact) {
        cTable.append($('|  |  |  |  |
| --- | --- | --- | --- |
|'
            .append($(' ').text(contact.firstName + ' ' + contact.lastName))             .append($(' ').text(contact.company))             .append($(' ').text('Edit'))             .append($(' ').text('Delete'))         ));     }); }  // Clear all content rows from the summary table function clearContactTable() {     $('#contentRows').empty(); } | | | |

```

```

// #1 - NEW CODE
// This code runs in response to the show.bs.modal event - it gets the correct
// contact data and renders it to the dialog
$('#detailsModal').on('show.bs.modal', function (event) {
    // Get the element that triggered this event - in our case it is a contact
    // name link in the summary table. This link has an attribute that contains
    // the contactId for the given contact. We'll use that to retrieve the
    // contact's details.
    var element = $(event.relatedTarget);
    // grab the contact id
    var contactId = element.data('contact-id');

    // PLACEHOLDER: Eventually we'll make an ajax call to the server to get the
    //                  details for this contact but for now we'll load the dummy
    //                  data
    var modal = $(this);
    modal.find('#contact-id').text(dummyDetailsContact.contactId);
    modal.find('#contact-firstName').text(dummyDetailsContact.firstName);
    modal.find('#contact-lastName').text(dummyDetailsContact.lastName);
    modal.find('#contact-company').text(dummyDetailsContact.company);
    modal.find('#contact-phone').text(dummyDetailsContact.phone);
    modal.find('#contact-email').text(dummyDetailsContact.email);

});

// TEST DATA
var testContactData = [
    {
        contactId: 1,
        firstName: "Susan",
        lastName: "Williams",
        company: "IBM",
        email: "swilliams@ibm.com",
        phone: "555-1212"},
    {
        contactId: 2,
        firstName: "George",
        lastName: "Smith",
        company: "EMC",
        email: "smith@emc.com",
        phone: "555-1234"},
    {
        contactId: 3,
        firstName: "Chuck",
        lastName: "Knobloch",
        company: "3M",
        email: "chuck@3m.com",
        phone: "555-5656"}
];

```

```
// #2: NEW CODE
var dummyDetailsContact =
{
  contactId: 42,
  firstName: "Kent",
  lastName: "Hrbek",
  company: "3M",
  email: "kent@3m.com",
  phone: "444-6798"
};
```

Okay, we're almost there... now we must change the way that our code renders the contact name in the Contact Summary table. We want the first name and last name of the contact to have the following characteristics:

1. It should render as a link so the user knows to click on it.
2. When clicked, the link should bring up the Contact Details modal.
3. Each link must have an attribute that contains the id for the given contact. We'll need this in the future so that we can call the server to get the contact details.

All of this will be accomplished by modifying the loadContacts function in **contactList.js**. Now, change your loadContacts function so it look like this:

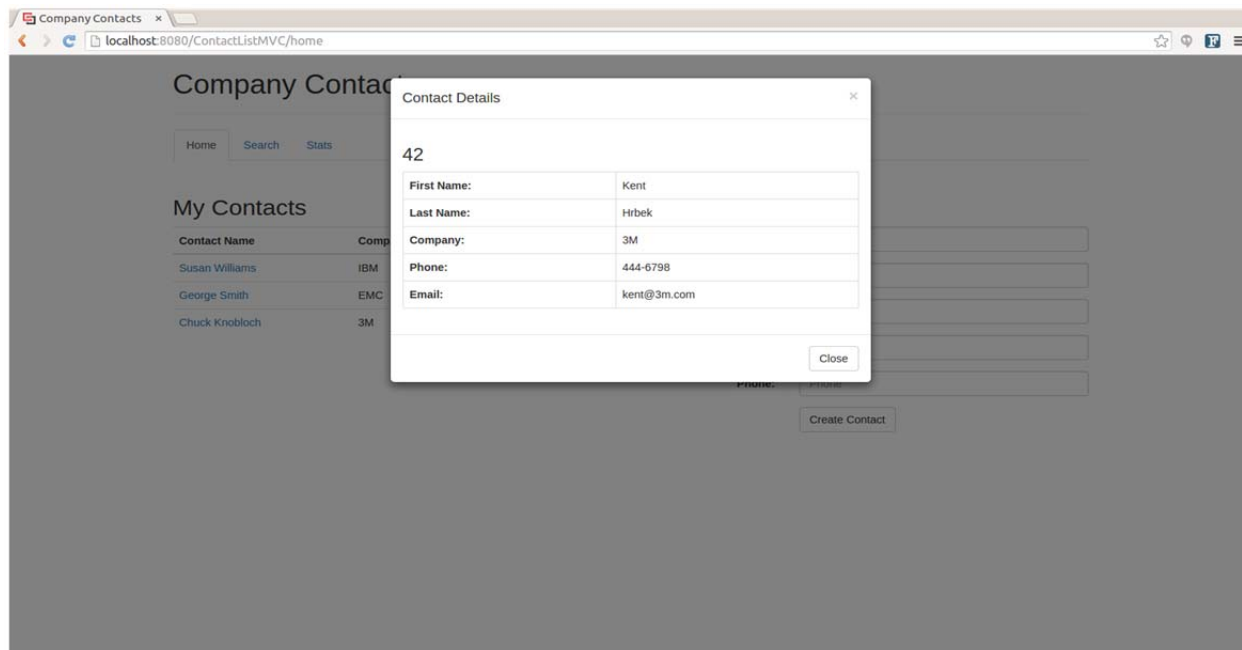
```
// Load contacts into the summary table
function loadContacts() {
  // Clear the previous list
  clearContactTable();
  // Grab the tbody element that will hold the new list of contacts
  var cTable = $('#contentRows');
  // Iterate through each of the JSON objects in the test contact data
  // and render to the summary table
  $.each(testContactData, function (index, contact) {
    cTable.append($('

```

The only code that changed (marked in bold) is in the way we are rendering the first <td> of the row. Instead of simply putting text in the cell, we're building a link and then adding three attributes to the link. The resulting HTML should look something like this:

```
<a data-contact-id="2" data-toggle="modal" data-target="#detailsModal">George Smith</a>
```

Now, when you click on the link, the Contact Details modal should appear:



Edit Contact Form

The approach we'll take for the Edit Contact form will be very similar to the approach we just took for the Contact Details modal. The two main differences are that we want the Edit modal to appear when the Edit link is clicked and that this modal will contain a form instead of just text.

Insert the following code into **home.jsp** just below the Contact Details modal code inserted in the previous step:

```
<!-- Edit Modal -->
<div class="modal fade" id="editModal" tabindex="-1" role="dialog"
    aria-labelledby="detailsModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal">
                    <span aria-hidden="true">&times;</span>
                <span class="sr-only">Close</span></button>
                <h4 class="modal-title" id="detailsModalLabel">Edit Contact</h4>
            </div>
            <div class="modal-body">
                <h3 id="contact-id"></h3>
                <form class="form-horizontal" role="form">
                    <div class="form-group">
                        <label for="edit-first-name" class="col-md-4 control-label">
                            First Name:
                        </label>
                        <div class="col-md-8">
                            <input type="text" class="form-control" id="edit-first-name"
                                placeholder="First Name">
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="edit-last-name" class="col-md-4 control-label">
                            Last Name:
                        </label>
                        <div class="col-md-8">
                            <input type="text" class="form-control" id="edit-last-name"
                                placeholder="Last Name">
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="edit-company" class="col-md-4 control-label">
                            Company:
                        </label>
                        <div class="col-md-8">
                            <input type="text" class="form-control" id="edit-company"
                                placeholder="Company">
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>
</div>
```



```

        <div class="form-group">
            <label for="edit-email" class="col-md-4 control-label">
                Email:
            </label>
            <div class="col-md-8">
                <input type="email" class="form-control" id="edit-email"
                    placeholder="Email">
            </div>
        </div>
        <div class="form-group">
            <label for="edit-phone" class="col-md-4 control-label">
                Phone:
            </label>
            <div class="col-md-8">
                <input type="tel" class="form-control" id="edit-phone"
                    placeholder="Phone">
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-offset-4 col-md-8">
                <button type="submit" id="edit-button" class="btn btn-default"
                    data-dismiss="modal">
                    Edit Contact
                </button>
                <button type="button" class="btn btn-default"
                    data-dismiss="modal">
                    Cancel
                </button>
                <input type="hidden" id="edit-contact-id">
            </div>
        </div>
    </form>
</div>
</div>
</div>
</div>
</div>

```

We also have to make modifications to **contactList.js**. Add the following JSON object to the end of the file (we'll use this to fill the edit form values):

```

var dummyEditContact =
{
    contactId: 52,
    firstName: "Kirby",
    lastName: "Puckett",
    company: "Sun",
    email: "kirby@sun.com",
    phone: "123-5599"
};

```

Next, we'll add code to render the Edit text as a link and to show the Edit Contact modal when the Edit link is clicked. Change the loadContacts function so that it looks like this:

```
// Load contacts into the summary table
function loadContacts() {
    // Clear the previous list
    clearContactTable();
    // Grab the tbody element that will hold the new list of contacts
    var cTable = $('#contentRows');
    // Iterate through each of the JSON objects in the test contact data
    // and render to the summary table
    $.each(testContactData, function (index, contact) {
        cTable.append($('
```

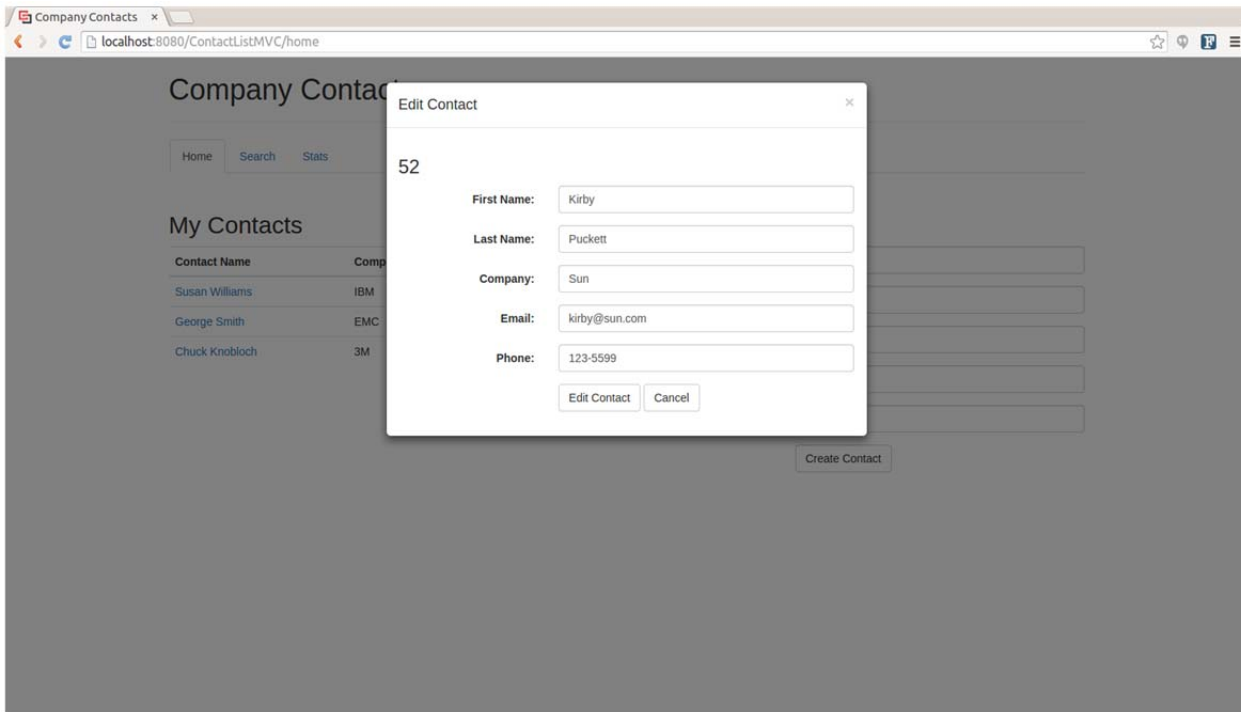
The only code that changed here (marked in bold) is the code that renders the Edit link; we change it from plain text to a link and we add some attributes. The rendered Edit link should look similar to this:

```
<a data-contact-id="2" data-toggle="modal" data-target="#editModal">Edit</a>
```

Finally, we need to add code that runs when the Edit modal is shown. Like the code related to the Details modal, this code will render contact data into the modal when it opens. Add the following function to your **contactList.js** file (put it just after the code for the Details modal):

```
// This code runs in response to the show.bs.modal event - it gets the correct
// contact data and renders it to the dialog
$('#editModal').on('show.bs.modal', function (event) {
    // Get the element that triggered this event - in our case it is a contact
    // name link in the summary table. This link has an attribute that contains
    // the contactId for the given contact. We'll use that to retrieve the
    // contact's details.
    var element = $(event.relatedTarget);
    // Grab the contact id
    var contactId = element.data('contact-id');
    // PLACEHOLDER: Eventually we'll make an ajax call to the server to get the
    //                 details for this contact but for now we'll load the dummy
    //                 data
    var modal = $(this);
    modal.find('#contact-id').text(dummyEditContact.contactId);
    modal.find('#edit-first-name').val(dummyEditContact.firstName);
    modal.find('#edit-last-name').val(dummyEditContact.lastName);
    modal.find('#edit-company').val(dummyEditContact.company);
    modal.find('#edit-phone').val(dummyEditContact.phone);
    modal.find('#edit-email').val(dummyEditContact.email);
});
```

After all of this code is in place, the Edit modal should pop up when you click the Edit link:



Wrap-up

That does it for this step. We did a lot in this session:

1. Created the initial layout of the Home screen
2. Created code that dynamically renders the Contact Summary table on page load
3. Created the Contact Details and Edit Contact modal windows
4. Modified the code that renders the Contact Summary table so that the contact names and the Edit text are rendered as links. We also added attributes to the links so that they open the correct modal when clicked.
5. Created JavaScript code that renders contact values in the Details and Edit modals when these modals are displayed

In the next step, we'll set up the initial Search screen.